ESCUELA SUPERIOR POLITECNICA DEL LITORAL

Escuela de Diseño y Comunicación Visual

DIFUSIÓN DE LOS GÉNEROS, CARACTERÍSTICAS, PLATAFORMAS, HERRAMIENTAS DE DESARROLLO QUE SE USAN EN LA INDUSTRIA DE LOS VIDEO-JUEGOS

Previo a la obtención del Título de

Magíster en Comunicación Pública de la Ciencia y la Tecnología

Presentado por

YAMIL EDINSON LAMBERT SARANGO

Guayaquil - Ecuador

2012

DEDICATORIA

A Dios,
Norma Sarango Intriago (Madre),
Galo Lambert Borja (Padre)
Yenny Celi Campoverde (Esposa)
Kevin Lambert Celi (Hijo)
Por estar siempre a mi lado, ser guía invaluable, fuente de inspiración y a quienes les debo todo.

Yamil Edinson Lambert Sarango

AGRADECIMIENTO

A Fausto Jácome López,
Ruth Matovelle Pastenes,
Internet y fuentes bibliográficas disponibles.
Por su guía y sobre todo por creer en este proyecto

Yamil Edinson Lambert Sarango

TRIBUNAL DE GRADO

Msig. Luis Rodriguez Vélez
DIRECTOR (E) DE LA ESCUELA

Mae. Fausto Jácome López
DIRECTOR DE TESIS

Mae. Víctor Muñoz Chachapoya
VOCAL PRINCIPAL

DECLARACIÓN EXPRESA

La responsabilidad del contenido de este Proyecto de Grado, corresponde exclusivamente a su autor; y el patrimonio intelectual del mismo a la Escuela Superior Politécnica del Litoral.

Yamil Edinson Lambert Sarango

Índice General

Capitulo 3. Hardware usado en la industria de los videojuegos

Capitulo 4. Software en el desarrollo de Videojuegos

Índice de Cuadros

Índice de Figuras

Capítulo 1

Generalidades

1.1 PLANTEAMIENTO DEL PROBLEMA

Los video-juegos son programas informáticos creados para el entretenimiento en general además de la interacción de una o varias personas, desde las etapas de niñez del ser humano, estos viven los momentos de jugar con instrumentos clásicos llamados juguetes, pero esta área ha evolucionado gracias a la tecnología informática que la ha llevado a nuevos niveles de entretenimiento y forma.

En Ecuador los video-juegos son mayoritariamente medios de entretenimiento, ya sean en versiones para computadores o consolas de video-juegos, destacando a las consolas Wii de la empresa Nintendo, Sony con PlayStation 1,2 y 3, y Microsoft con Xbox 360, como las más importantes; en el área de desarrollo de software, **Ecuador se ha destacado con algunos sistemas de tipo comercial y bancarios; además de la exportación de los mismos; es decir a manera de maquila de software[1] para empresas extranjeras, pero no en la creación y desarrollo de video-juegos por el desconocimiento que se requiere a nivel de hardware y software** para la creación de estos, **no se conoce las tecnologías involucradas en esta industria**, como lenguajes de programación y conceptos apropiados en su desarrollo, motores 2D y 3D que ayudan a disminuir los tiempos de desarrollo.

Los programadores en un porcentaje extremadamente bajo se dedican a desarrollar y crear juegos, y los pocos que existen lo hacen como pasatiempo, es decir sin esperar llevar sus creaciones muy básicas y comercializarlas, es decir; **la industria local en esta área es inexistente**; incluso otro factor es que **no existen ofertas académicas serias para que estos programadores se adentren en este nuevo mundo, y aprendan los conceptos fundamentales.**

[1] Soporte lógico de una computadora digital; comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas.

Hacer un video-juego dista mucho de un sistema comercial, por ejemplo de los sistemas contables, puntos de ventas entre otros; que han sido tradicionales, pero sus bases conceptuales y prácticas les servirán para incursionar en el desarrollo de un video-juego; sobre todo para los programadores de dichos sistemas.

El área multimedia[2] es una de las que ha sido explotada en cierta medida; con proyectos educativos para enseñar a los niños o personas discapacitadas, aquí los programadores locales han incursionado y explotado a medida pequeños desarrollos de entretenimiento, juegos incorporados en el mismo DVD multimedia, por lo que con una **buena capacitación a estos programadores en el desarrollo de video-juegos tanto en conceptos, arquitectura y herramientas estos tendrán la capacidad de crearlos** y sus productos pueden estar a la altura de grandes empresas que se dedican a crear juegos de videos con estándares y mucha tecnología de software para un hardware especifico.

Preguntas a resolver:

- ¿Que son los video-juegos y su historia?
- ¿Cómo ha evolucionado esta industria a nivel de hardware para consolas de sobremesa, consolas portátiles y PC?
- ¿Cómo ha evolucionado esta industria a nivel de Software con las herramientas de desarrollo hasta los motores 3D más populares y sus características?

1.2 JUSTIFICACIONES

Es importante difundir esta temática localmente, sobre todo al área de profesionales informáticos sean estos Analistas de Sistemas, Ingenieros en Sistemas o Licenciados en Sistemas de Información, porque estos ya conocen el desarrollo de software y las etapas del ciclo de vida de un sistema informático.

Comenzar a fomentar y divulgar que esta industria en muy rentable, a nivel internacional, que se puede convertir en nuevas plazas de trabajo locales para los informáticos Ecuatorianos.

Además de dejar los cimientos para una futura carrera de desarrollo de video-juegos a nivel local y profesional; incluso para los profesionales en ramas de Informática y Diseño Gráfico que puedan incursionar en esta industria.

[2] Objeto o sistema que utiliza múltiples medios de expresión (fisicos o digitales) para presentar o comunicar información. Usando texto e imágenes, animación, sonido y video.

Fomentar la creación de pequeñas productoras de juegos, con una primera visión local pero a larga plazo expandirse internacionalmente, primero se podrá comenzar con profesionales a manera de maquila, es decir que estos sean mano de obra para productoras extranjeras ya posicionadas en este mercado.

Perfeccionar a programadores que demandan las productoras de multimedio que necesitan que ellos tengan conocimientos en el tema de video-juegos.

Esto pretende generar un impacto social en la sociedad ecuatoriana para que esta no sea solo un consumidor sino productor de video-juegos; los beneficiarios son desde los desarrolladores locales, las empresas que incursionen en ello, hasta los usuarios que recibirán productos de calidad y entretenidos; incluso convirtiéndose en consumidores de las creaciones locales.

Además las universidades se podrán preparar en nuevas ofertas académicas para lanzar profesionales en esta área y con ello crear un nuevo nicho de mercado laboral.

1.3 OBJETIVOS GENERALES Y ESPECÍFICOS

Difundir y dar a conocer el hardware donde se ejecuta un video-juego, y el software para desarrollarlos, plataformas, mercados, características, géneros, clasificaciones pioneras y entornos de desarrollo que nos permiten crear juegos para computadoras, empezando con los conceptos para los **del tipo 2D y 3D**, que en la historia han tenido mucho éxito, tales como los juegos de "**Tetris**", "**Mario Bros**", "**Pacman**", "**Gusano comelón**", "**Doom 1, 2 y 3**", "**Quake1,2**", "**Unreal Tournament**", "**Descent 1,2,3**" entre otros, que en los actuales momentos existen versiones que aprovechan la tecnología actual conocidos como REMAKE[3], con ellos los programadores de sistemas podrán comprender y enrumbar sus proyectos en esta nueva área, y que localmente en el país se explote este tipo de desarrollo.

Objetivos Específicos

1. Explicar la evolución histórica de los video-juegos tanto en aspectos conceptuales como tecnológicos incluyendo su mercado, características, géneros y clasificación.

[3] Término en inglés que identifica las producciones audiovisuales que reproducen fidedignamente la trama, personajes, ambientación y prácticamente cualquier detalle de una obra anterior.

2. Explicar el hardware que se usa para ejecutar un video-juego desde las plataformas, consolas de video-juegos por generaciones, consolas portátiles y PC.
3. Explicar el software para el desarrollo de video-juegos tanto en formato 2D y 3D, mundos virtuales y los motores 3D más populares.
4. Explicar la tecnología requerida para desarrollar un video-juego a través del análisis de los más exitosos video-juegos.

1.4 METODOLOGÍA DE LA INVESTIGACIÓN

El método que se aplicará es el de análisis y síntesis porque se requiere recopilar información tanto de tipo cuantitativo como cualitativo. Es decir que en un momento de la investigación se indagará por los recursos bibliográficos disponibles sobre video-juegos en idioma inglés y español.

En un segundo momento la investigación indagará sobre el hardware usado para la ejecución de los video-juegos; sean para máquinas recreativas, PC, consolas de sobremesa y portátiles; incluyendo temas que tienen más alta aceptación o se presentan como los más adecuados para el desarrollo del mismo

Este análisis supone necesariamente un procesamiento de la información que sintetice de forma adecuada las tendencias más altas tanto en hardware y software usado en el desarrollo de video-juegos.

- Análisis y revisión bibliográficos, para recopilar información de libros, textos y documentos relacionados y más apropiados con la problemática de investigación.
- Lectura crítica, para determinar los contenidos teóricos y metodológicos que permitirán fundamentar de forma adecuada la investigación y sus instrumentos respectivos.
- Consulta de fuentes secundarias, en caso de que sea necesario incrementar la información en las diferentes etapas y capítulos de la investigación.
- Procesamiento estadístico. Para comparar la variación en las demandas bibliográficas por parte de los diferentes actores involucrados.

1.5 ESTRUCTURA DE LA TESIS

1.5.1 CAPITULO 2

En este apartado se documenta la base teórica de esta investigación empezando desde la historia de los video-juegos, la guerra fría, los primeros video-juegos, la primera videoconsola, la segunda guerra mundial y Japón como puntos importantes en cuanto a su aporte al nacimiento de esta industria; además se describe el nacimiento de los primeros video-juegos como SpaceWar, Tennis for two, PONG, Space Invaders, Pac-man, el aporte de la empresa ATARI, hasta la crisis de 1983, luego se analiza el mercado de los video-juegos, su naturaleza, y características; se explicara los distintos géneros, además de la clasificación de contenidos usando la ESRB (Entertainment Software Rating Board por sus siglas en inglés) y la PEGI (Pan European Game Information por sus siglas en inglés).

1.5.2 CAPITULO 3

En este capítulo se documenta el uso y evolución del hardware en el desarrollo de video-juegos, tomando en cuenta las plataformas, máquinas recreativas o Arcades, haciendo énfasis en las consolas y sus generaciones mencionando cada una de ellas con sus aspectos técnicos como procesador, memoria, video, sonido, formato, controles y accesorios; las computadoras personales o PC y las generaciones de las consolas portátiles mencionando sus características de hardware más importante.

1.5.3 CAPITULO 4

En este sección se documenta el uso del software en el desarrollo de video-juegos, tomando en cuenta su arquitectura, sea esta dirigida por datos o componentes, los juegos en 2 y 3 dimensiones; sus técnicas, con sus tecnología en APIs para video sean estas OPENGL o DIRECT3D; se analizará los lenguajes de script con sus ventajas y desventajas; además de los principales motores 3D existente en la industria de los video-juegos.

1.5.4 CAPITULO 5

En esta parte se documenta como se desarrolló un controlador en tercera persona para el personaje principal del video-juego “Corsarios del Pacifico” llamado “Santiago”, se analiza y documenta la cámara en este tipo de video-juegos; se describen los parámetros de entrada que recibe y la explicación general del código fuente propuesto

con el objetivo de demostrar la facilidad en el desarrollo usando el lenguaje script del motor 3D del software Unity3D.

1.5.5 INDICE DE TESIS

PORTADA

DEDICATORIA

AGRADECIMIENTOS

RESUMEN

PRESENTACIÓN

ÍNDICE

Capítulo 2

Base teórica de los Video-juegos

2.1 ANTECEDENTES

Las computadoras, tal y como los conocemos, no tienen más allá de medio siglo de historia. La idea de un dispositivo así se debe a Charles Babbage[4], profesor inglés que acompañado de su alumna predilecta Ada Byron [5] se lanzó a un proyecto definitivamente visionario para aquella época, el año de 1830 la máquina analítica para ejecutar programas de tabulación o computación que nunca llegó a realizarse de manera operativa; después de un siglo y la necesidad militar de hacer cálculos matemáticos para trayectorias balísticas impulsan la creación de una máquina basada en las ideas de Babbage. Fueron los estadounidenses los que presentaron una máquina enorme llamada Mark I, diseñado por Howard Aiken para la IBM. Al Mark le seguiría, en 1946, el ENIAC de los ingenieros Mauchly y Eckert, con sus 18.000 válvulas de vacío y sus cientos de kilómetros de cableado[6], es decir que la base tanto en hardware como en software de los video-juegos actuales se remonta a la historia de las computadoras.

Figura 2.1 Mark I

Fuente: http://www.computersciencelab.com/ComputerHistory/HtmlHelp/Images2/

[4] Se le considera como una de las primeras personas en concebir la idea de lo que hoy llamaríamos una computadora, por lo que se le considera como "El Padre de la Computación"
[5] Considerada como la primera programadora, desde que escribió la manipulación de los símbolos, de acuerdo a las normas para una máquina de Charles Babbage que aún no había sido construida.
[6] DIEZ, E. 2004. La diferencia sexual en el análisis de los video-juegos. Madrid. CIDE Instituto de la Mujer. 468 p.

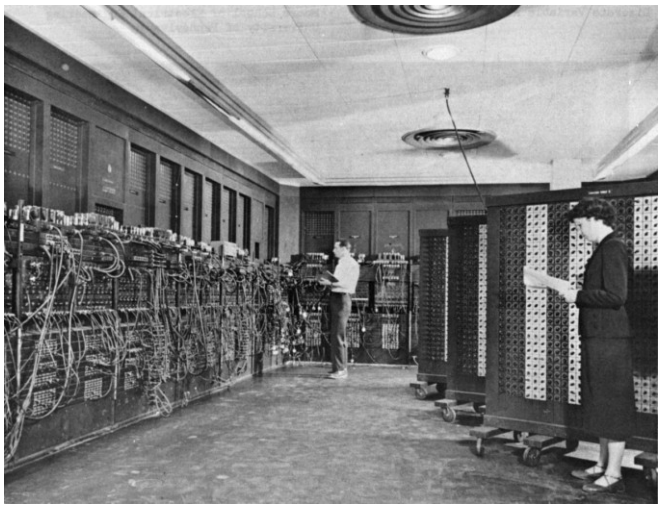

Figura 2.2 Computadora ENIAC

Fuente: http://www.eui.upm.es/sites/default/files/museo/

Por definición general un "video-juego" o también llamado "juego de vídeo" es un software (programa informático) creado para el entretenimiento, se basa en la interacción o jugabilidad entre una o varias personas denominados usuarios o jugadores, un controlador o mando; y un aparato electrónico que ejecuta dicho video-juego; este dispositivo electrónico puede ser una computadora, videoconsola, máquina arcade[7], dispositivo handheld como un teléfono móvil; los cuales son conocidos como "plataformas". El término "video" en la palabra "video-juego" se refiere a un visualizador de gráficos rasterizados.[8]

Los video-juegos son el futuro del entretenimiento, una industria de millones de dólares que ha sumergido al mundo en su reino digital.

2.2 HISTORIA DE LOS VIDEO-JUEGOS

A continuación los siguientes apartados están basados entre opiniones de grandes personajes junto con hechos que marcaron la historia de los video-juegos, las opiniones fueron tomadas del documental de televisión "La era del Videogame" dirigido por Curran Robert en el año 2007.

[7] Término genérico de las máquinas recreativas de video-juegos disponibles en lugares públicos de diversión

8 "Television gaming apparatus and method". United States Patents. http://www.freepatentsonline.com/3659285.html Consultado el 21-02-2012.

Para Nolan Bushnell[9], "*estos no son solamente arte sino también cultura, son creativos; y que en muchos aspectos cuentan con una estructura bastante más compleja que una buena película".* (CURRAN, 2007)

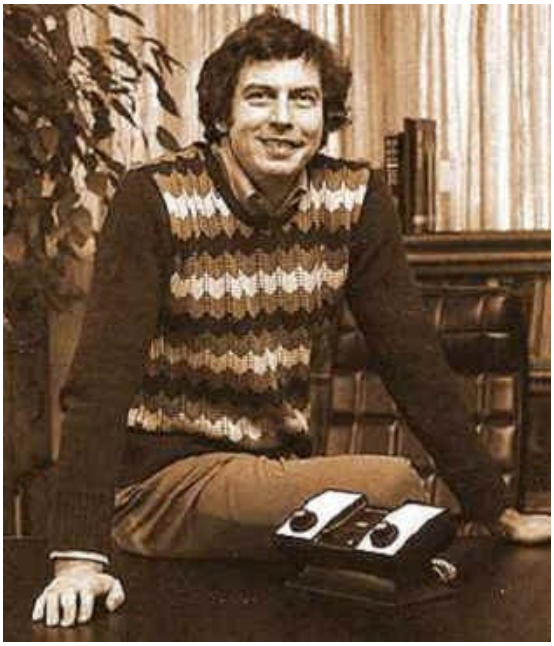

Figura 2.3 Nolan Bushnell

Fuente: http://www.retrogames.cl/imagenes/historia/

John Romero Co-creador del mítico juego Doom[10], "*es que se debe tomar en cuenta que los procesadores o CPU actuales son muy rápidos, las tarjetas de videos junto a la velocidad hacen que estos sean más intensos, las pantallas son también mas grandes así que uno se siente como si estuviese dentro".*[11] (CURRAN, 2007)

[9] Nolan Bushnell es el fundador de Atari y es junto a Ralph Baer uno de los pioneros de la industria de los video-juegos.
[10] *Doom* significa *maldición*.
[11] Basado en "La era del Videogame" documental de televisión, 2007. Curran Robert.

Figura 2.4 John Romero Co-creador del mítico juego Doom

Fuente: http://gamingsquid.com/wp-content/uploads/2012/04/

Ralph Baer[12] creador del Magnavox Odyssey[13], "*producir un video-juego hoy en día es como producir una película, porque se necesitan muchas personas y dinero*". (CURRAN, 2007)

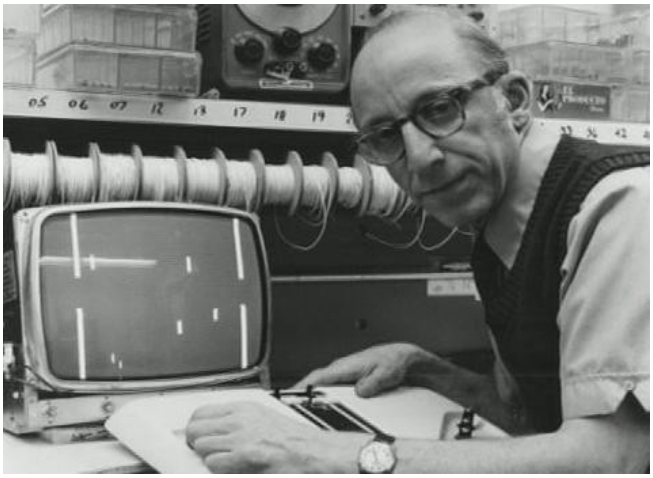

Figura 2.5 Ralph Baer

Fuente: http://3.bp.blogspot.com/-NPdD4MqqiPI/T16HMCLX5EI/AAAAAAAAAOc/6CD_TNU6F3Q/s640/

Desde el inicio los video-juegos han sido un reflejo de su época. Ernesto Fernández de Nintendo España opina que, "*el papel de los video-juegos en nuestra vida*

[12] Apodado "el padre de los video-juegos caseros.
[13] Comercializada por la filial de Philips en Estados Unidos, la *Magnavox Odyssey* es la primera videoconsola de la historia.

es puramente entretenimiento desde nuestra perspectiva humana, que compite con otras formas de ocio como la del cine, la televisión o la lectura". (CURRAN, 2007)

2.2.1 LA GUERRA FRÍA

En el mundo actual estar conectado resulta esencial incluso fuera del mundo del video-juego pulsar un botón forma parte de la rutina del siglo XXI, sin embargo hace tan solo 50 años los ciudadanos del mundo no querían saber de un botón en concreto "el botón rojo", porque según ED Halter autor de "FROM SUN TZU TO XBOX", *"la imagen de presionar un botón que venía a la mente no era de un video-juego sino que era el botón para acabar con el mundo".* (CURRAN, 2007)

Según la opinión de Alekseéi Pázhitnov creador del juego Tetris[14], *en ese tiempo se era consciente del miedo que había en Rusia, ya que en el colegio y la universidad se daban clases especiales de lo que se denominaba "Defensa Civil" sobre cómo había que reaccionar en caso de destrucción masiva.* (CURRAN, 2007)

Figura 2.6 Alekséi Pázhitnov creador del juego Tetris

Fuente: http://media.giantbomb.com/uploads/3/33308/

Con las tropas soviéticas de más de cinco millones de soldados; los implicados se tambaleaban al borde de la conocida "GUERRA FRIA" y la tecnología surgía en sus inicios desatando el pánico colectivo, esto fue el resultado de las dilatadas tensiones diplomáticas entre el mundo comunista y el liberal, se denominó FRIA porque nunca se llegó a un conflicto abierto.

[14] Video-juego de puzzle originalmente diseñado y programado por Alekséi Pázhitnov en la Unión Soviética.

Para Henry Jenkins director de Comparative Media Studies – MIT, *la GUERRA FRIA resultó ser una batalla de simulación y la compara con los juegos en que estos también las usan, ya que se emplea para recrear procesos del mundo real, para predecir resultados, muy parecido al video-juego BATTLE SHIELD*. (CURRAN, 2007)

La guerra fría solo supuso un simulacro con el cual empezar, no existe una guerra de referencia solo existen las guerras del futuro, el miedo a la guerra a lo que pueda pasar. Los simulacros y los comienzos de la informática en la guerra fría, reflejaban los miedos que eran imposibles disiparlos, ya que el jugador no era capaz de controlar sus armas.

Según BUSHENELL, *a la gente le gusta los mundos predecibles donde pasan cosas predecibles o al menos tener la sensación de que se pueden controlar o que cuentan con los medios para controlarlas y eso representa un grado de satisfacción ya que el mundo es normalmente incontrolable y desalentador*. (CURRAN, 2007)

Pronto aparecería el primer juego de computador gracias a la tecnología de la guerra fría y el mundo virtual o real nunca volvería a ser el mismo. En los años cincuenta la guerra fría abría impulsado el desarrollo de la tecnología de la informática para los simulacros de combate y lanzamiento de misiles.

A medida que los preparativos para la guerra fría se intensificaban un joven físico que trabajaba en un laboratorio nuclear cambiaba para siempre la idea de pulsar un botón. En 1958 William Higinbotham que había trabajado en la primera bomba atómica convirtió dos líneas normales y corrientes; y una pelotita, es la primera experiencia interactiva con un computador llamada “TENIS PARA DOS” en inglés “Tennis for Two” siendo este el primer video-juego de la historia.

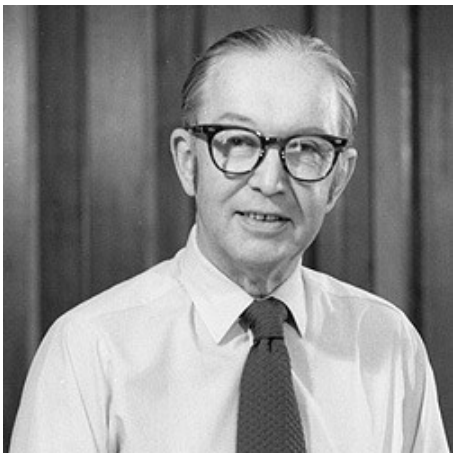

Figura 2.7 William Higinbotham

Fuente: http://2.bp.blogspot.com/_GJHAezShK6I/TSPpcrtun4I/AAAAAAAAUCk/fM4ZAMgD3Q8/s400/

Según Paul Steed director creativo de ATARI organizaban concursos los fines de semana para ver a quien se le ocurría la cosa más ingeniosa; ya que William apareció con esta idea que transformaba un osciloscopio en un partido de tenis.

Para Henry Jenkins fue la manera que tenía *un adolecente que trabajaba en un proyecto militar para expresar su rebeldía, coger la pieza más cara del equipo en aquel entonces y usarla para jugar, de repente se abría un mar de posibilidades para las computadoras*, pero según avanzaba la guerra también lo hacían los juegos y los jugadores. (CURRAN, 2007)

Sin duda el viaje del cosmonauta Yuri Gagarin fue el acontecimiento del año en 1961, cuando se lo puso en órbita alrededor de la tierra; Yuri Gagarin daba los primeros pasos en el espacio exterior en nombre de la Unión Soviética, pero los EEUU no andaban lejos y las expresiones de su presidente en aquel entonces John F. Kennedy lo demostraba cuando dijo en un discurso "hemos decidido ir a la luna en este momento no porque sea fácil, si no porque es difícil".

La guerra fría se convirtió en una carrera espacial, lo que tuvo gran influencia en Steve Ruesell un programador del M.I.T., él dio el siguiente paso en los video-juegos con el universo virtual de "SPACEWAR", la carrera espacial estaba siempre en las noticias, las naves espaciales despegando y los transbordadores explotando, no había escapatoria, además unas de sus influencias fue el libro "Triplanetary"[15], que resumía en

[15] Novela que resume todos los aciertos y todos los fallos de la ciencia ficción americana escrita por E.E. Doc Smith.

un grupo de relatos de ciencia ficción la del héroe que atraviesa la galaxia e inventa nuevas tecnologías mientras lo persiguen las fuerzas del mal, pero antes de enfrentarse a las fuerzas del mal Steve Ruesell tuvo que perfeccionar el gigantesco computador PDP-1[16] (del inglés Programmed Data Processor-1) del campus.

Figura 2.8 Steve Ruesell

Fuente: http://neoshock.files.wordpress.com/2011/05/

El PDP-1 era considerado el primer computador con categoría de computador personal que se podía encender con un interruptor, y se lo consideraba un computador pequeño con el tamaño de una refrigeradora, costaba cerca de unos $90.000 dólares en su época y servía para lo mismo que una máquina de escribir y una calculadora; los programas eran caros, así que era una máquina de escribir con calculadora muy cara, para Aaron Ruby coautor del juego SmartBomb, la PDP-1 no hacía nada espectacular, resultaron tan decepcionante que no se les prestó atención, pero este se podía mejorar en muchos aspectos. (CURRAN, 2007)

2.2.2 LOS JUEGOS "SPACEWAR" Y "TENNIS FOR TWO"

Según Steve Ruesell para el juego SPACEWAR, comenzó a imaginar que podía poner un par de naves espaciales que se disparaban torpedos mutuamente, pero hasta ahí el juego sonaba un poco aburrido y para ello le agrego que las cosas explosionen.

Si "Tennis for Two" era una inocente distracción para evadirse de la guerra fría el juego de Steve Ruesell era un claro reflejo de la carrera espacial y del miedo mundial a la guerra.

[16] Primer computador en serie PDP de la Digital Equipment, producida por primera vez en 1960, fue también el hardware original donde se jugó el primer video-juego computarizado de la historia.

Para Paul Steed director creativo de ATARI ésta representaba lo que se vivía, la sensación de la paranoia soviética, del comunismo que venía por uno, es decir todo eso se fijo en la conciencia colectiva de la sociedad y reflejaba las cosas que tenían ¿vamos a entrar en guerra? (CURRAN, 2007)

Sin duda este juego supuso un gran avance y un adelanto en su momento e influenció en Nolan Bushnell fundador de ATARI, *fue el juego más popular durante un par de años más que nada porque era el único.* (CURRAN, 2007)

Los códigos para jugar se fueron pasando de programador en programador y todas las instituciones que tenían un PDP-1 tenían inevitablemente una versión del SPACEWAR, el código fuente estaba abierto a todos y se les dada una copia a todos aquellos que la querían o la buscaban por parte del mismo Steve Ruesell, más que todo a los jóvenes de la época que formaban parte de un grupo que hoy llamaríamos los primeros piratas informáticos.

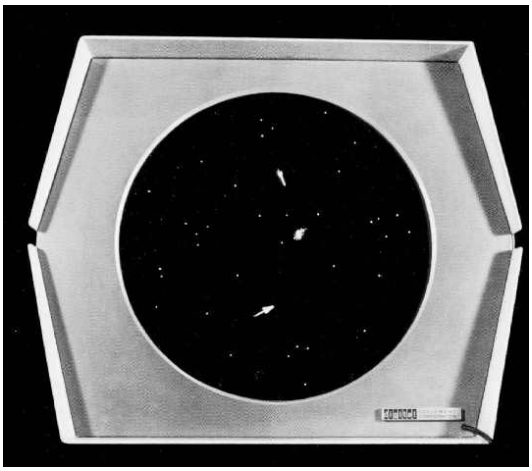

Figura 2.9 SPACEWAR original

Fuente: http://bnbgaming.com/wp-content/uploads/2011/02/spacewar2.jpg

Según Steve Ruesell era más exploración que rebelión, habían muchas cosas nuevas que hacer y qué ver y tenían una especie de mecano o juego de construcción nuevo, con muchísimas piezas diferentes refiriéndose al PDP-1.

2.2.3 CAJA DE MANDO O JOYSTICK

Una de las cosas que construyó Steve Ruesell fue el prototipo de lo que se convertiría en uno de los componentes imprescindibles de los video-juegos “el mando”, los primeros mandos eran simples botones de la consola, hasta que uno de los chicos del

club de maquetas de trenes construyó una caja de mando o joystick[17] con los repuestos de un teléfono. (CURRAN, 2007)

Figura 2.10 Steve Ruesell usando el mando para SPACEWAR

Fuente: http://pc.gamersmafia.com/storage/column/0/spacewar.jpg

Steve Ruesell también fue el primer programador que llevó la destrucción a la pantalla, una de las frases que él suele decir "yo desaté la maldición de los video-juegos en el mundo".

2.2.4 LA TELEVISION

Junto con la guerra fría los conflictos de Corea, Berlín y Vietnam en los años 60 amenazaba con sumergir el mundo en el caos y ahí estaba la televisión para mostrar los ecos de la guerra, pero no había forma de cambiar la realidad.

Para Doug Rushkoff[18] autor del juego "PLAYING THE FUTURE" la televisión empezó a mostrar imágenes de desesperación, mostraba los efectos del napalm[19] o gasolina gelatinosa**,** niños y niñas vietnamitas escapando de las bombas, la televisión era una máquina de sufrimiento y no daba la oportunidad de ajustarla. (CURRAN, 2007)

Para Al Alcorn programador del juego PONG los años 70 la televisión era un medio pasivo y frio, y es que la televisión se convirtió en el incansable emisario de malas noticias que un hombre y una idea cambiarían para siempre, para Al Alcorn de

[17] Palanca de mando o joystick (del inglés joy, alegría, y stick, palo) es un dispositivo de control de dos o tres ejes que se usa desde una computadora o consola.
[18] Escritor, columnista y profesor de cultura virtual, estadounidense, en la Universidad de Nueva York.
[19] Combustible que produce una combustión más duradera que la de la gasolina simple.

repente un video-juego se convirtió en un medio cercano, porque permitía cambiar la realidad que mostraba. (CURRAN, 2007)

Figura 2.11 Al Alcorn programador del juego PONG

Fuente: http://wildgames.es/wp-content/uploads/2011/01/al-alcorn.jpg

2.2.5 LA PRIMERA VIDEOCONSOLA

Ralph Baer conocido como el Thomas Edison de los video-juego, creó la primera videoconsola conocida como la "MAGNAVOX ODYSSEY", según Ralph Bear la mayor diferencia entre los demás y él; es que *se dio cuenta que habían cerca de 40 millones de televisores que solo servían para sintonizar dos o tres canales en aquella época.* Por aquel entonces los comerciales en la televisión que hacían alusión a la empresa MAGNAVOX decían que la misma había adaptado un simulador de juegos electrónicos que convertirá su televisor en una zona de juegos. (CURRAN, 2007)

Para Kirk Erwing creador del juego "JFK RELOADED" comentó que por aquella época *debió ser increíble, tenerlo en casa, en la televisión y jugar video-juegos, esto era el futuro.* (CURRAN, 2007)

Antes de que Ralph Bear le cambie la cara al televisor[20], había vivido la realidad de la guerra, porque escapó de la Alemania Nazi antes del holocausto y más tarde fue reclutado para luchar en la segunda guerra mundial con los EEUU, fue enviado a Normandía en Francia para realizar interrogatorios rodeado de la mecánica de la guerra, lo que influenció en gran medida a Ralph Bear y a la historia de los video-juegos.

[20] Aparato electrónico destinado a la recepción y reproducción de señales de televisión. Usualmente consta de una pantalla y mandos o controles.

Para apartar su mente de la guerra, Ralph Bear había despertado sus ganas innatas de interactuar con la tecnología, una habilidad que le llevaría a Sanders Associates en 1958 donde permaneció hasta haberse jubilado, en el año de 1987.

En aquella época trabajaban para defensa y Ralph Bear era un joven ingeniero con experiencia en tecnología de televisión y fue contratado para desarrollar tecnología militar durante la guerra fría, pero pronto empezó a trabajar en la "Brawnbox" o caja marrón, el prototipo de la primera consola familiar la "MAGNAVOX ODYSSEY".

Figura 2.12 Personas jugando con la MAGNAVOX ODYSSEY

Fuente: http://beegamer.es/wp-content/uploads/2010/08/MagnavoxOdyssey_thumb.jpg

La caja tenía siete juegos diferentes y llevaba muchos accesorios diferentes e idearon un juego fascinante que funcionó, aun hoy divierte y no es más que simbología en la pantalla, Ralph Bear cambió para siempre la forma de ver la televisión, pero como los pioneros que le precedieron toda la diversión y los juegos precisaron altas sumas de dinero.

Si se tiene un empleo muy serio y muy técnico se busca un poco de diversión en el trabajo y así es como surgieron los video-juegos, alguien jugueteando con un osciloscopio se imagina un partido de tenis; para Peer Schneider vicepresidente de ING.COM Ralph Bear trabajaba en un campo muy serio y algunas veces se sentaba y pasaba un buen rato con toda esa tecnología que le rodeaba en vez de usarla solo para destruir. (CURRAN, 2007)

El sector de los video-juegos siempre ha estado muy ligado al ejército como cualquier otro sector tecnológico, fueron los militares quienes investigaron con computadores en la segunda guerra mundial para calcular la trayectoria de los misiles,

así que desde el principio de la historia de los video-juegos siempre ha estado muy ligado al campo militar.

2.2.6 LOS AÑOS 60 Y 70, MOVIMIENTO CONTRACULTURAL

A finales de los años 60 y comienzos de los 70 el movimiento contracultural generó una nueva rama en los video-juegos y Nolan Bushnell fundador de ATARI fue uno de los primeros en verlo, en plena era de acuario se iba a crear un nuevo orden mundial basado en el amor, la armonía y estas formaban el espíritu de la empresa ATARI por aquellos tiempos. (CURRAN, 2007)

Figura 2.13 Logo de la empresa ATARI

Fuente: http://movilzone.org/files/2009/07/atari_logo.jpg

Para Al Arcon programador del juego PONG[21], se debía a que eran jóvenes y a la idea de vivir la vida, siempre les rondaba la idea de poder morir en cualquier momento; así que se habían mentalizado el poder disfrutar al máximo, no se podía posponer nada, intentar hacer cosas divertidas formaban parte de su cultura. (CURRAN, 2007)

2.2.7 EL NEGOCIO DE LOS VIDEO-JUEGOS

Cuando la gente discute sobre quien creó los video-juegos hay muchísimos argumentos, pero nadie cuestiona quien creó el negocio de los video-juegos y ese fue Nolan Bushnell, porque él fue el primero en darse cuenta que con ello se podía ganar dinero. (CURRAN, 2007)

[21] Video-juego de la primera generación de videoconsolas publicado por Atari, creado por Nolan Bushnell y lanzado el 29 de noviembre de 1972. Pong está basado en el deporte de tenis de mesa (o ping pong).

2.2.8 EL JUEGO DE PONG

Nolan Bushnell recluto a su antiguo compañero Al Arcon para programar el primer juego de salón recreativo, le encargo su primer proyecto donde tenía que construir un sencillo juego de ping-pong con una pelota que se moviese para marcar, mintiéndole que tenía un contrato con la empresa General Electric para que piense que no era solo una prueba lo que ayudo a que el prototipo del juego PONG se terminara, incluso Nolan Bushnell le sugirió que debía llevar sonido como gente aplaudiendo, gente que silben y abucheen cuando se pierde, pero Al Arcon no tenía idea de cómo hacerlo incluso ya se había pasado del presupuesto, ya llevaba demasiados chips; así que solo le acopló un pequeño radio casete en el circuito, buscó sonidos más sencillos que ocupaban menos de medio chip y luego se lo mostro a Bushnell el cual quedó alucinado. (CURRAN, 2007)

Figura 2.14 Juego PONG original para ARCADE

Fuente: http://pongmuseum.com/history/_picts/atari/pong_cabbig-prev.jpg

Según Nolan Bushnell luego le cambiaron un detalle al juego sin importancia, cuando la bola golpeaba en distintos segmentos de la barra el ángulo cambiaba de repente y se convertía en un juego emocionante, así que ese pequeño cambio fue como pasar de la noche al día.

Cuando ATARI lanzo el juego PONG en el año de 1972 nació la industria del video-juego, así que PONG fue el comienzo de todo, fue el primer juego que era accesible a todos aquellos que no trabajaban en un laboratorio.

Al Arcon escribió unas breves instrucciones para el juego PONG en la máquina que decían:

- Paso 1 inserte una moneda.
- Paso 2 el saque es automático.
- Paso 3 procure no perder la bola para ganar puntos.

Y eso era todo lo necesario, porque el juego era muy sencillo.

Para Doug Rushkoff *todos aquellos que jugaron al PONG se acordaran de la primera partida como recuerdan otros acontecimientos de la época porque se tenía lo que se denominaba "el poder del pixel"*. (CURRAN, 2007)

Otras de las rarezas del PONG es que estaba pensado para dos jugadores así que hacían falta dos personas para jugarlo no se podía solo. PONG era un juego para enfrentarte con los amigos, una nueva forma de batalla y demostrar quién era el mejor.

Cosa curiosa es que su salida coincidió con el principio de la liberación femenina así que era bastante normal que una chica recién liberada entrase en un bar y dijese -*hacemos una partida al PONG, invito yo*. Para Nolan Bushnell las mujeres coordinan los músculos pequeños mejor que los hombres así que en general las mujeres jugaban al PONG mejor que los hombres. Era la primera vez que las chicas podían ganar a los hombres en el mismo terreno de juego y de hecho las mujeres comenzaron a ir a los bares y comenzar a apostar. (CURRAN, 2007)

Un enfoque más interesante es el de Henry Jenkins cuando afirma que el PONG llega en un momento en que los EEUU vivía una época muy seria de su historia, no solo por la guerra de Vietnam, sino también por los movimientos en pro de los derechos civiles, en cambio los años 70 parecían ser más frívolos y se comparaban con los 60 que habían sido muy intensos, Jenkins aclara que *no piensa que los video-juegos creasen ese sentido de frivolidad, era un síntoma de la tendencia que tenía la gente al recluirse en sus hogares, ha tener compromisos sociales, ha sentirse un poco más serios*.

Higinbotham, Ruesell, Baer y Bushnell fueron los pioneros que conscientemente o no canalizaron el miedo colectivo hacia sus creaciones virtuales, era obvio que el mundo lo estaba deseando y que los video-juegos eran las armas perfectas de distracción masiva para todo el planeta.

2.2.9 LA SEGUNDA GUERRA MUNDIAL Y JAPON

Los primeros video-juegos nacieron como un subproducto de la guerra fría pero en el lejano oriente los primeros juegos japoneses surgieron como consecuencia de un conflicto directo, la segunda guerra mundial, las bombas atómicas de Hiroshima y Nagasaki habían asolado el país, pero las fuerzas aliadas trabajaban estrechamente con Japón en la reconstrucción.

La tecnología se mostró rápidamente como la clave del éxito japonés, el gobierno hacia grandes esfuerzos por encaminar la educación hacia la tecnología, la juventud japonesa estaba altamente alfabetizada en materia tecnológica en ese momento.

Japón encontró un aliado en la electrónica, imaginando lo que el resto del mundo estaría haciendo en ese aspecto, perfeccionándolo, produciéndolo en masa y más barato.

Pero las viejas heridas aún no se habían cerrado, para Heather Chaplin coautora de SmartBomb*, si nos fijamos en la cultura japonesa tras la segunda guerra mundial veremos cómo se refleja la experiencia de la bomba atómica en muchas de sus obras.* (CURRAN, 2007)

Para Henry Jenkins, es la época en la que triunfan "Godzilla" y las películas de grandes monstruos de goma, y si combinamos el contenido cultural de Godzilla con el poder tecnológico del computador tenemos algo que se parece mucho a la fascinación que producen los video-juegos. (CURRAN, 2007)

2.2.10 EL JUEGO SPACE INVADERS

Japón estaba a la cabeza de la industria electrónica, pero en 1978 Tomohiro Nishikado colocó a Japón al frente de la industria de los video-juegos gracias una interminable ola de invasores digitales, creando el juego "Space Invaders" o invasores del espacio.

Según el mismo Tomohiro Nishikado creador de Space Invaders, en un principio pensó que los invasores podrían ser humanos, pero la idea de un humano disparando a otro humano se tachó de agresiva; así que pensó que usando extraterrestres en su lugar sería más adecuado. (CURRAN, 2007)

Para N´gai Croal editor general de NewsWeek, cuando se ve el juego, *no es tan difícil establecer un paralelismo entre las naves extraterrestres y los aviones sobrevolando Japón, es decir son claramente armas de destrucción masiva cayendo desde el cielo.* (CURRAN, 2007)

Para Nic Kelman autor de Video Game Art, lo más interesante de Space Invaders es que *cuenta con uno de los primeros escenarios virtuales*, y según Kirk Ewing en realidad cuando se juega se está sufriendo una invasión y uno está defendiendo la humanidad en primera línea. (CURRAN, 2007)

Space Invaders cambio algo en las mentes de los japoneses porque millones de adolescentes jugaban y malgastaban el escaso dinero del país, la manera tan rápida en la que se extendió solo puede producirse en un lugar como Japón.

Figura 2.15 Juego Space Invaders

Fuente: http://www.videogamesblogger.com/wp-content/uploads/2009/03/space-invaders-arcade-coming-to-wii-virtual-console.jpg

Para los sonidos según Tomohiro Nishikado habían grabado los latidos del corazón, para crear el efecto sonoro en el video-juego, el sonido grave del suelo temblando cuando se acercaban, aunque hoy en día se pueden crear sonidos mejores hay que tomar en cuenta que en aquella época se tenían unos dispositivos y una técnica muy limitada y no se pudo hacerlo mejor, pero quedó bastante bien. (CURRAN, 2007)

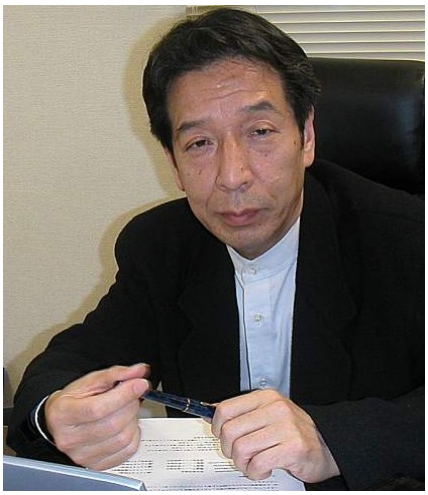

Figura 2.16 Tomohiro Nishikado

Fuente: http://cdn4.spong.com/news/s/p/spaceinvad161084l/_-Space-Invaders-Creator-Tomohiro-Nishikado-Unveils-Nintendo-DS-Title-_.jpg

Para Tommy Tallarico compositor de música de video-juegos, *si se piensa en la música y el sonido de Space Invaders eran solo 4 notas*, y lo que hicieron fue sincronizar el ritmo, y según se iban acercando los invasores la música era cada vez más rápida; así lograban conseguir una pequeña sensación de pánico mientras se jugaba. (CURRAN, 2007)

Para aprovechar el apogeo los propietarios de las tiendas retiraban la mercancía por las noches he instalaban recreativas de Space Invaders, la invasión de este video-juego señalaba un vacío generacional en la cultura japonesa. Muchas madres no dejaban que sus hijos lo jueguen, se convirtió en motivo de alarma social desde que los chicos comenzaron a robar a sus padres y a trucar las máquinas recreativas, era un problema serio frente a la educación de los niños.

Para Peer Schneider de IGN.COM, piensa que la llegada de tanta tecnología de golpe produjo una fisura entre generaciones, y la generación mayor que pensaba que lo había asimilado todo, que había entendido lo que pasaba con la industria de la automoción y la televisión sintió que la dejaban atrás, porque no podía entender cómo podía ser tan popular un juego tan tonto. (CURRAN, 2007)

Se comenzaron a aprobar normas sobre donde se suponía que no podía haber cierto número de chicos a la vez como en tiendas o salones recreativos, se creó una auténtica cultura a su alrededor.

Para Stuart Moulder consultor de video-juegos, *resultaba un poco triste que inevitablemente el juego acababa con la derrota del jugador, aunque en aquel momento no se fue consciente de ello*, pero de algún modo refleja la paranoia de la guerra fría porque al fin y acabo la guerra es un juego derrotista, siempre se pierde. (CURRAN, 2007)

2.2.11 EL JUEGO DE PAC-MAN

Space Invaders fue la expresión digital de un periodo oscuro, pero Japón siempre ha sabido dar un aspecto dulce gracias a sus encantadores personajes cuando llegó el juego de Pac-Man, con muchos colores, este es el ejemplo perfecto que define la rareza japonesa.

Según su creador Toru Iwatani, *los personajes de PAC-MAN son adorables aunque en los países occidentales no los llamarían así, sino infantiles*. (CURRAN, 2007)

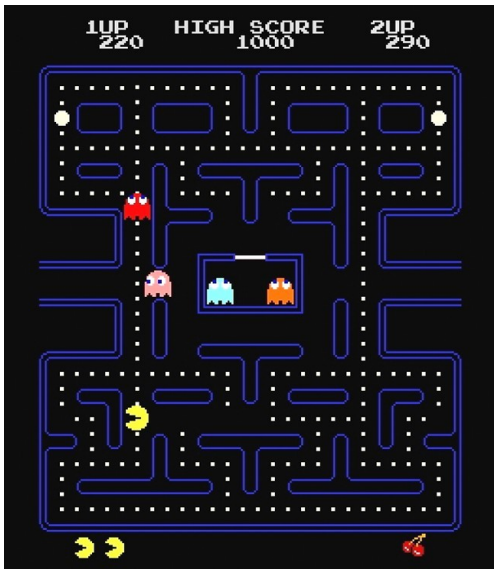

Figura 2.17 Juego PAC-MAN

Fuente: http://4.bp.blogspot.com/_cZsSbIqoiuY/TUhgBlpzrDI/AAAAAAAABBY/VY0fgLZJ5cE/s1600/pac-man.jpg

Para Henry Jenkins, *su cultura tiene un nivel de tolerancia para la cursilería, más alto que en las culturas occidentales*, que se lo ha visto a lo largo del siglo XX, refleja el nivel de consumo de las chicas japonesas. (CURRAN, 2007)

Toru Iwatani era un joven diseñador que trabajaba en una empresa de juegos japonesa NAMCO, antes de crear la estrella de la historia de los video-juegos, Pac-Man

se lanzó en 1980, pero la atmosfera en los salones recreativos eran un tanto salvaje, porque solo había juegos violentos como los de disparar y era un terreno reservado a los chicos.

Figura 2.18 Toru Iwatani creador del juego PAC-MAN

Fuente: http://images.wikia.com/pacman/images/7/78/390_11749_PHOTO_-_ToruIwatani.jpg

Toru Iwatani, quería llevar a las chicas y a las parejas a ese terreno, así que centro el juego en las chicas, pensó en muchos tópicos desde la moda hasta los chicos, pero como las chicas siempre toman postre pase lo que pase, decidieron escoger "comer" como palabra clave, comenta que estaban a punto de comer y pidieron pizza, cogió un trozo de ella y viendo como quedó la pizza sin él, tenía a Pac-man allí mismo, y decidieron desarrollarlo de esa manera. (CURRAN, 2007)

Para Victor Sanchez editor de plástico.tv, Lo más importante de Pac-Man o conocido también como el Comecocos es que *fue el primer personaje de la historia de los video-juegos, supuso un salto muy importante para la industria, los desarrolladores empezaron abrir nuevas puertas para crear nuevos personajes, nuevas figuras para que los jugadores puedan sentirse identificados*. (CURRAN, 2007)

También cambio mucho el panorama, mientras que muchos de los primeros juegos se basaban en invasores, acción y disparos, Pac-Man hacia algo diferente se trataba de comer, es el mejor ejemplo de cómo se empieza con algo pequeño y luego se hace tan grande que nadie sabe cómo detenerlo.

Todos lo jugaban hombres, mujeres, niños y familias, parecía que este juego se había apropiado de sus vidas; para Peer Schneider Pac-Man supone el comienzo real de

la era del "Merchandising"[22] porque se coge una idea que ha surgido para un video-juego y de repente se convierte en un producto completamente distinto como música, peluches, muñecos, dibujos animados, incluso algunos se les ponían piernas y nacía la Sra. Pac-Man.

Para su creador Pac-man fue querido y aceptado durante muchos años y sus ventas nunca disminuyeron, incluso con todo el tiempo que ha pasado actualmente Pac-Man aun sigue divirtiendo.

A principios de los años 80 ya estaba claro que los video-juegos y las formas de vida pixeladas habían sido creadas para quedarse, porque son una forma de arte expresiva y representativa, como afirma Henry Jenkins estos representan nuestras creencias, actitudes o preguntas que nos hacemos en algún momento de nuestras vidas, llegaran hacer doctrinas sociales muy ricas, pero el reto radica en cómo interpretarlas porque a diferencia de una película en la que el director establece un discurso unitario un video-juegos está sujeto a cambios que dependen de las elecciones de los jugadores.

2.2.12 LA EMPRESA ATARI

Surge un nuevo mundo que no solo afecta a los jugadores sino también a sus creadores, por ejemplo ATARI fue el arquetipo de empresa del Silicon Valley[23] de los años 70, definiendo todo un estilo de vida, como no llevar traje y corbata y seguir haciendo negocios serios y dinero con un enfoque menos estricto, todo se basaba en los méritos.

También fue la época del dinero fácil y de los grandes negocios, ATARI siguió su curso cuando Bushnell la vendió a la "Warner Communications" en 1976 (ahora Time Warner) en un estimado de $28 a $32 millones de dólares, él dejó la división en 1979 mientras era propiedad de Warner, pero según iba pasando el tiempo los nuevos dueños se iban sintiendo incómodos con la extraña manera de llevar la empresa; con las fiestas y lo divertido que les parecía ganar dinero, pero incluso con la nueva dirección los cerebros de ATARI seguían avanzando su tecnología y empezaron a desarrollar la nueva consola portátil la ATARI VCS "Video Computer System" por sus siglas en inglés.

[22]Conjunto de técnicas encaminadas a poner los productos a disposición del consumidor, obteniendo una rentabilidad a la inversión hecha en el establecimiento.
[23] Silicon Valley o Valle del Silicio es el nombre que recibe la zona sur del área de la Bahía de San Francisco, en el norte de California, (Estados Unidos).

2.2.13 LA ATARI 2600

Para Al Arcon habían creado un chip para el juego PONG de recreativas; que solo funcionaba con ese juego, para crear otro juego nuevo había que crear otro chip diferente; y desarrollarlo llevaba unos 9 meses de trabajo de ingeniería, por lo que puso al equipo de Grass Valley[24] a crear en 3 meses el prototipo de lo que sería la ATARI VCS.

La VCS se llamó ATARI 2600 y tras obtener la licencia del space invader de la empresa Taito en 1980, se puso primero en la lista de regalos de navidad, porque era como tener el salón recreativo en casa[25].

ATARI gano mucho dinero con cada uno de sus juegos, pero el nuevo ambiente corporativo de la empresa estaba acabando lentamente con los principios en los que se había fundado, porque se respiraba mucha tensión entre la directiva de ATARI; sobre todo tras la adquisición por parte de Warner y los programadores, porque les pagaban entre $12.000 y $15.000 dólares al año y eran ellos los que creaban los juegos, a medida que estos se hacían más rentables se empezó a notar entre los diseñadores cierto malestar por no obtener un porcentaje. (CURRAN, 2007)

Poco después, cuatro de los chicos se separaron de ATARI y formaron su propia empresa llamada ACTIVISION y empezaron a vender sus cartuchos que eran bastantes buenos y eso supuso el fin del monopolio para ATARI.

Posteriormente Warner despidió a Bushnell de ATARI en una reunión de la junta directiva, a pesar de que no hay duda de que la vida de Bushnell mejoro en lo económico se vio apartado del negocio que el mismo puso en marcha.

Aun se creía que ATARI era el único pez gordo del océano capitalista que mantenía el flujo de beneficios pero llegó la marea baja de los años 80 y pensaron que nunca acabaría.

Para Bushnell el éxito de ATARI era tan sencillo que las personas que la dirigían no sintieron que hubiese necesidad alguna de ir con cuidado, que no era necesario preocuparse, los juegos perdieron creatividad y se tornaron aburridos, el juego del que

[24] El nombre fue en honor al bar Grass Valley en California donde Bushnell instaló la primera máquina recreativa con el juego PONG.
[25] Ver http://www.atarimuseum.com/videogames/consoles/2600menu/2600menu.htm

hablaba todo el mundo era el de una versión de la película "ET"[26], que era tan mala, incluso criticada por el mismo Al Arcon, y todo se debió porque el director de la Warner había cerrado un trato con Steven Spielberg en que se pagaron cerca de siete millones de dólares, el juego se hizo barato y rápido, con el tiempo justo para producirlo y sacarlo al mercado, pero la gente que se agolpó para comprarlo enseguida se dieron cuenta que lo que les encantaba en el cine era un fiasco en el juego. (CURRAN, 2007)

2.2.14 LA CRISIS DE 1983 PARA LOS VIDEO-JUEGOS

Se hicieron demasiados cartuchos del juego de "ET" que fue imposible venderlos, según Al Arcon se deshicieron de ellos cavando una fosa para enterrarlos, pero se corrió la voz, entonces la cubrieron con cemento y lo negaron diciendo que los cartuchos eran defectuosos lo que era cierto desde el punto de vista de la creatividad; porque no se vendieron, así que las ventas de los video-juegos ATARI sufrieron una caída drástica, 1983 fue el año de la gran quiebra, porque si se hablaba con la gente que hacia video-juegos a finales de los 80, pensaban que "era tan sencillo como lanzar un juego y esperar a que lloviese el dinero", y como todo el mundo quería entrar en el negocio había mucha competencia, muchas empresas, los video-juegos estaban por todas partes, pero habían muchos, tan malos que la gente acabo dándole la espalda al medio. (CURRAN, 2007)

2.3 EL MERCADO DE LOS VIDEO-JUEGOS

Para José Martí Parreño, para darse cuenta de la importancia de la industria y el mercado de los video-juegos dentro del sector del ocio y el entretenimiento basta con saber que en los últimos años supera con creces el ingreso en taquilla del mercado cinematográfico (tanto en los EE.UU como en otros países). El mercado de los video-juegos generó un volumen de negocio en 2010, solo en los EE.UU., 25.100 millones de dólares (según fuente NPD Group, (ESA[27], 2011:11). (Martí, 2010)

[26] E.T., el extraterrestre es una película Estadounidense de ciencia ficción de 1982, coproducida y dirigida por Steven Spielberg.

[27] La ESA (Entertaiment Software Association) es la antigua IDSA (Interactive Digital Software Association), y representa a los principales editores de video-juegos (Atari, Eidos, Electronic Arts, Ubisoft, Nintendo of America, Sega of America, etc.) en los EE.UU.

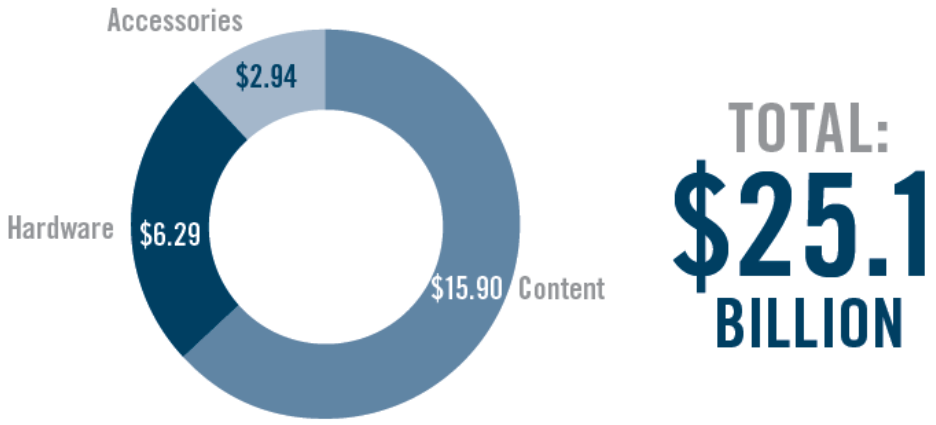

Figura 2.19 Gasto total de los consumidores en la industria de los video-juegos 2010

Fuente: ESA, 2011

En este país las ventas de video-juegos alcanzaron los 15.900 millones durante 2010 (frente a los 11.700 millones de dólares de 2008) con 298.3 millones de unidades vendidas (ESA, 2011:10). (Martí, 2010)

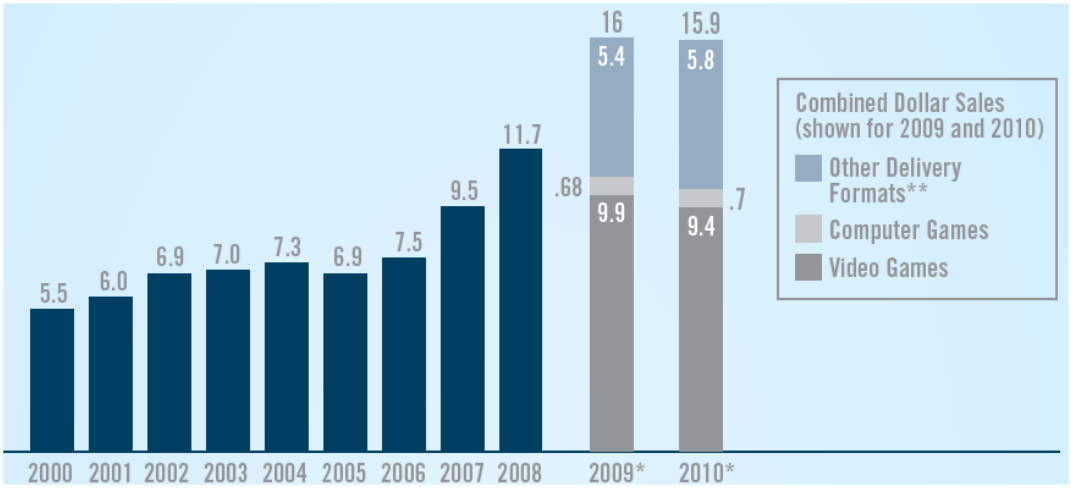

Figura 2.20 Evolución de los ingresos por venta de Video-juegos en EE.UU (en miles de millones)

Fuente: ESA, 2011

A nivel europeo, las cifras de los nueve mercados más importantes alcanzaron los 13.000 millones de euros en 2007 (7.300 millones de euros correspondientes a la venta de software y 5.700 millones de euros a la venta de hardware) (Nielsen Games, 2008). En Reino Unido (tercer mercado en importancia a nivel mundial por detrás de

Estados Unidos y Japón) el volumen de ventas alcanzó los 2.300 millones de euros en 2007[28].

Según José Martí Parreño los importantes crecimientos del sector de los video-juegos (hasta hace apenas unos años cercanos al 25% anual) se debe en parte a que la industria ha sabido captar nuevos segmentos de mercado que hasta ahora le eran totalmente inaccesibles. Queda ya muy lejos la imagen de los consumidores de video-juegos como adolecentes varones fanáticos de la informática. En la actualidad, los video-juegos son consumidos virtualmente por cualquier segmento de edad de ambos sexos (Fattah y Paul, 2002). El hecho de que en los EE.UU. las mujeres mayores de 18 años representen el 37% de la población total video-jugadora en 2010 (superando al grupo de varones de 17 años o menores, que representa el 13% de esta población) o que en 2011 el 29% de la población norteamericana mayor de 50 años jugara a los video-juegos (frente a tan solo el 9% en 1999)[29] da buena prueba de esta diversificación y segmentación de sus públicos. (Martí, 2010)

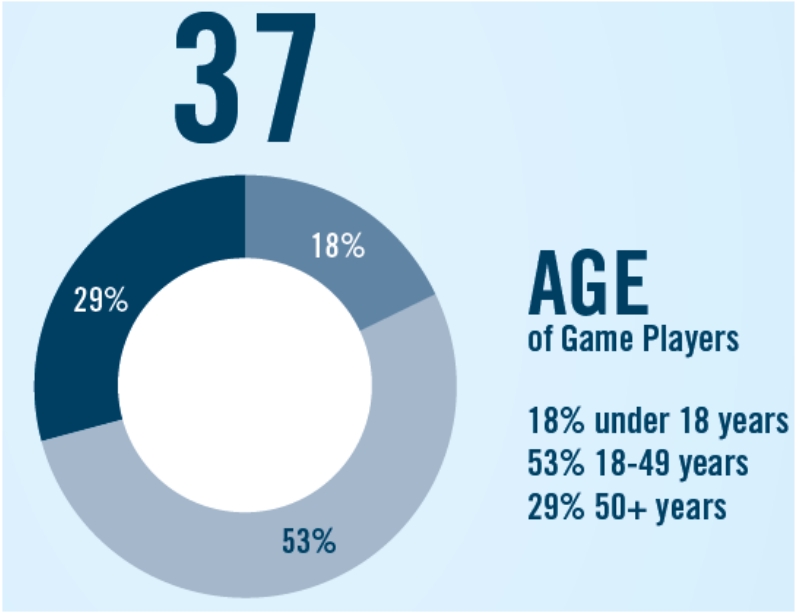

Figura 2.21 Promedio de video-jugadores por edad 2011

Fuente: ESA, 2011

En los últimos años también se han producido importantes cambios de consumo en relación a la edad de los video-jugadores ya que la edad media de los compradores

[28] Con España ocupando el cuarto puesto (700 millones de euros), tras el propio Reino Unido (2.300 millones de euros), Francia (1.600 millones de euros) y Alemania (1.400 millones de euros) (Nielsen Games, 2008)
[29] <http://www.theesa.com>

más frecuentes de video-juegos en los EE.UU. es de 37 años (ESA,2011:2). Este dato indica que la media de edad va creciendo puesto que los video-jugadores que empezaron a jugar en su adolescencia o juventud continúan jugando en la edad adulta (de hecho, los video-jugadores adultos llevan jugando una media de 13 años a los video-juegos). No menos importante resulta el hecho de que el 65% de los hogares norteamericanos se juega a los video-juegos[30]. (Martí, 2010)

En España, la edad media del video-jugador activo es de 26 años (mientras que el 28% de los españoles con edades comprendidas entre los 16 y los 49 años se consideran jugadores activos, cifra que aumenta hasta el 43% si consideramos solo a los menores de 30 (Nielsen Games,2008).

2.4 NATURALEZA DE LOS VIDEO-JUEGOS

Tanto este como el siguiente apartado tienen como único objetico ofrecer un marco conceptual de los video-juegos, definiéndolos y contextualizándolos con respecto a medios como el cine o la televisión, y analizando las características principales que los conforman como un "medio" diferenciado de los demás. Porque autores como Aarseth (2001) o Hunicke, LeBlanc y Zubeck (2004), han afirmado que los video-juegos no son un medio. Es cierto que, a falta de encontrar otro calificativo mejor, más que un medio, los video-juegos parecen ser una compleja amalgama que hibrida lenguajes, soportes y tecnologías diversas, cuya definición quizá sería más oportuno buscar en el campo de los géneros de entretenimiento (Jenkins, 2000) o de los géneros culturales (Aarseth, 1998;2001). (Martí, 2010)

El punto de partida teórico del estudio de los video-juegos es el de los juegos (Hizinga, 1972; Avedon y Sutton-Smith, 1971; Callois, 1986). En este marco teórico encontramos conceptos como el de *campo de juegos* (lugar en el que se desarrolla el *mundo de juego* y lo diferencia del *mundo real*), reglas y objetivos, etc. Que son perfectamente aplicables al estudio de los video-juegos. De hecho la diferenciación realizada por Callois (1986) entre paidéia y ludus[31] y su taxonomía de los juegos[32] ha sido un punto de partida frecuente a la hora de analizar los video-juegos.

[30] <http://www.thesa.com>
[31] Muy simplificadamente, los primeros hacen referencia a los juegos más espontáneos de los niños mientras que los segundos corresponden a juegos con reglas más complejas.
[32] Callois contempla cuatro categorías básicas: agón (juegos de combate), alea (juegos de azar), mimicry (juegos de imitación), e illinx (juegos de velocidad).

El antecedente inmediato de los video-juegos son los juegos electromecánicos surgidos y popularizados como atracciones de feria. En estos juegos encontramos, como en los video-juegos, la mediación "tecnológica" ausente en el resto de juegos (en los que por supuesto existen herramientas y *materiales* con los que se juega pero, por decirlo de alguna manera, se carece de una *interfaz* para ser jugados que no sea nuestro propio cuerpo y los materiales utilizados: cuerdas, pelotas, etc.). (Martí, 2010)

El salto a los video-juegos proviene precisamente de una interfaz gráfica tecnológica que simula el espacio de juego en una pantalla (*video*-juegos): ya sea el osciloscopio utilizado por Higinbotham[33], la pantalla del computador o la del televisor. La singularidad de los video-juegos parece radicar en que son jugados a través de una pantalla. Esta pantalla permite el desarrollo de un juego tecnológicamente mediano no solo a través de esa interfaz gráfica sino mediante el uso de otros periféricos (teclado, joystick, etc.) A través de los cuales se interactúa con el video-juego.

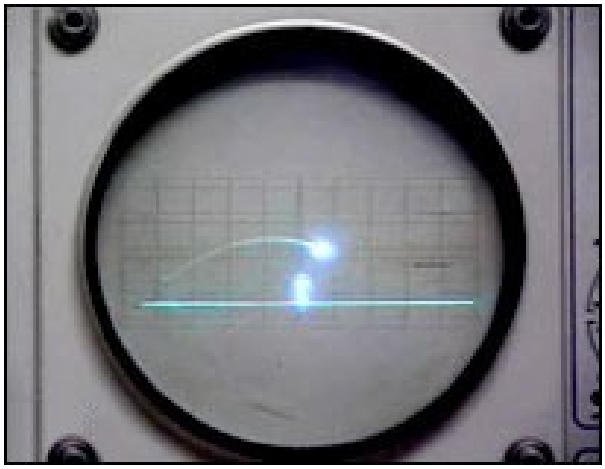

Figura 2.22 *Tennis for two* desarrollado por Willian Higinbotham en 1958

Fuente: http://upload.wikimedia.org/wikipedia/commons/a/a9/

Precisamente esta condición tecnológica y su habilidad para anidar como "huésped" en diversas pantallas, como la del televisor o la de los computadores personales, ha provocado que a menudo se haya tratado de estudiar los video-juegos desde los diversos campos que estudian los medios de comunicación de masas. Por ello se les ha intentado aplicar también muy a menudo las diversas teorías que desde la literatura, el drama, el teatro o los estudios cinematográficos tratan de analizar las

[33] Willian Higinbotham creó en 1958 lo que se considera el primer "video-juegos" de la historia: una simulación de una partida de tenis (llamada tennis for two) que se podía jugar en un osciloscopio. En ella se inspiró Nolan Bushnell para desarrollar Pong (1972).

diversas narrativas y lenguajes utilizados en estos medios (a pesar de que autores como Aarseth (2001), Eskelinen (2001a) o Frasca (1998) han rechazado esta colonización de otros campos de estudio reivindicando un campo de estudio propio para los video-juegos). (Martí, 2010)

Es evidente que los video-juegos comparten características con medios como el cine o la televisión. El lenguaje audiovisual utilizado por estos medios, y también por los video-juegos, es una de ellas. Muchos de los géneros de video-juegos comparten también con estos medios las narrativas utilizadas, hasta el punto que en muchas ocasiones las escenas cinemática[34] que aparecen en un video-juego nos pueden recordar a una película y algunas escenas de películas (sobre todo las de acción) nos pueden recordar a un video-juego. De hecho, no son pocos los autores que han examinado las similitudes entre video-juegos y el cine (Crawford, 1982; Nelson, 1990; Wolf, 2001). También se ha intentado analizar los video-juegos desde la dramaturgia (Laurel, 1991). Sin embargo son cada vez más los autores que destacan las importantes diferencias que presentan los video-juegos en especial con las narrativas utilizadas en medios como el cine o la televisión. (Martí, 2010)

En que radica pues la especificidad de los video-juegos? Cuál es la principal característica diferenciadora de los video-juegos? Algunos autores dirán que la interactividad (Rouse, 2004); otros que la espacialidad (Aarseth, 1998); otros que la temporalidad (Eskelinen, 2001b). La mayoría coincidiría en que su principal características es que "son jugados" y aunque esto pueda parecer una tecnología (como si dijéramos que la característica de una película es que "es vista"), el concepto de jugabilidad es un aspecto crucial y ontológico al propio concepto de video-juego. Tal y como ha sido descrito por Eskelinen (2001a) al comparar los video-juegos con las narrativas entendidas como "sistemas de huecos" (Sternberg, 1978), estos huecos necesitan acción por parte del usuario para ser "encontrados, cerrados y negociados.", es decir la principal característica de los video-juegos parece ser que demandan ser jugados por sus consumidores para poder desarrollarse (a diferencia de una película que puede ser proyectada hasta su término sin que nadie la vea). Tal y como se ha afirmado "los

[34] Las secuencias o escenas cinemáticas (cut-scene) son fragmentos narrativos insertados en los video-juegos (como la secuencia de introducción al video-juego o las escenas que enlazan el paso a otra prueba o nivel del video-juego).

jugadores no se implican en experiencias de juego ya hechas sino que toman parte de forma activa en construir estas experiencias" (Ermi y Mayra, 2005:2). (Martí, 2010)

Ahora bien, nos puede ayudar esto a dar una definición de video-juegos? Tras analizar algunas definiciones propuestas por los diversos teóricos y profesionales de los video-juegos, Juul [35] propone una definición en la que esta acción de jugar es contemplada, al definir un video-juegos como un "sistema basado en reglas con una respuesta variable y cuantificable en el que se le asignan diferentes valores a las diferentes efectos, en el que el jugador realiza un esfuerzo por influir en estos efectos, tiene un apego emocional a los efectos y las consecuencias de dicha actividad son opcionales y negociables" (Juul, 2003). (Martí, 2010)

Esta compleja y vasta definición da buena prueba de la dificultad a la hora de definir un género que debe satisfacer numerosos parámetros. En todo caso, la mayoría de las definiciones que se pueden encontrar harán hincapié en que un video-juego: a) es un "Sistema" (hoy día se podría decir "interactivo y digital"); b) obedece a unas reglas; c) tiene unos objetivos; d) plantea un desafío al video-jugador; y e) desencadena interacciones (entre el propio video-jugador y el sistema así como, en determinados géneros, con otros video-jugadores). (Martí, 2010)

2.5 CARACTERÍSTICAS DE LOS VIDEO-JUEGOS

Realizar un análisis de las características de los video-juegos puede parecer particularmente complicado cuando se ha llegado a afirmar que "la característica que mejor define a los juegos de computador parece ser que no tienen una características que los defina". No hay un "lenguaje de los juegos de computador" para ser descubierto y reconstruido puesto que los juegos presentan una rica diversidad. Entre el Tetris[36], un Furby[37] y los juegos online masivos multijugador 3D, como Everquest[38], hay unas diferencias cognitivas y materiales muy grandes y poco en común (Aarseth, 2001).

[35] http://www.jesperjuul.net/text/gameplayerworld/ Fecha de consulta: 30/abril/2012
[36] Video-juego de puzle originalmente diseñado y programado por Alekséi Pázhitnov en la Unión Soviética. Él juego deriva su nombre del prefijo numérico griego tetra- (todas las piezas del juego, conocidas como Tetrominós que contienen cuatro segmentos) y del tenis, el deporte favorito de Pázhitnov.
[37] Juguete electrónico fabricado por Tiger Electronics (subsidiaria de Hasbro). La atracción que ofrece es su aparente «inteligencia» y capacidad de aprendizaje al mejorar sus facultades para hablar.
[38] Famoso MMORPG (del inglés de massively multiplayer online role-playing game) que fue lanzado en marzo de 1999, es un video-juegos de rol que permite a miles de jugadores introducirse en un mundo virtual de forma simultánea a través de internet e interactuar entre ellos.

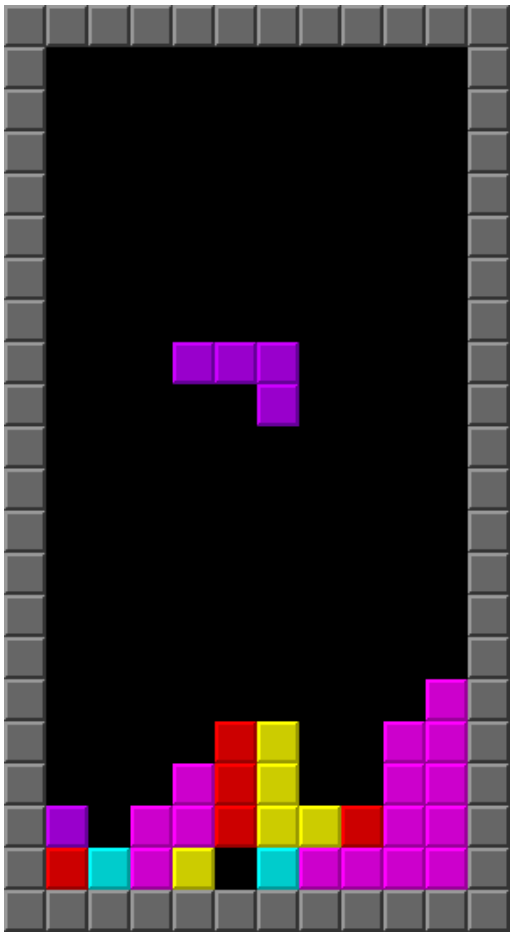

Figura 2.23 Tetris clásico

Fuente: http://es.wikipedia.org/wiki/Archivo:Emacs_Tetris_vector_based_detail.svg

Figura 2.24 Muñeco de Furby clásico

Fuente: http://1.bp.blogspot.com/-F23GVIlz7mE/TvJCs-YnTbI/AAAAAAAAkj8/4EBRoJnse7s/s640/

Así pues aunque pueda parecer realmente complicado encontrar características que definan un género de entretenimiento que, como se ha dicho, resulta complejo en cuanto a la variedad de lenguajes que utiliza, las plataformas en las que se juega, etc. Este apartado pretende, al menos, esbozar algunas de las características más comúnmente aceptadas entorno a los video-juegos.

2.5.1 INTERACTIVIDAD

Según Martí, "No cabe la menor duda de que si preguntamos a alguien que haya jugado alguna vez a un video-juego afirmará que la interactividad es uno de los elementos diferenciadores a lo hora de experimentar dicho contenido de otro tipo de contenido (por ejemplo, una película o cualquier contenido televisivo). Frente a la linealidad de estos otros contenidos, los video-juegos demandan de forma continua la interacción del video-jugador para actualizar lo que ocurre en la pantalla. Sin embargo, la interactividad es un concepto complejo que ha sido definido desde diferentes campos, adoptando distintas perspectivas (MacMillan y Hwan, 2002). Esta situación ha llevado a algunos académicos incluso a reprochar que "la interactividad es un término ampliamente utilizado con una intuición que resulta atractiva, pero es un concepto todavía por definir" (Rafaeli, 1988:110 cfr. En Wu, 2005). En todo caso, prácticamente todas las definiciones "enfatizan la importancia de la interacción entre el usuario y un sistema" (Sundar y Kim, 2005)". (Martí, 2010)

Steuer define la interactividad como "la capacidad de los usuarios para participar y modificar la forma y el contenido de un entorno mediado en tiempo real" (Steuer, 1992:84). En un video-juego de carreras, por ejemplo, "acelerar" o "girar" el volante a izquierda o derecha forma parte de esa interactividad que permite actuar sobre el entorno virtual del video-juego. Para Crawford (1994) la interactividad en los video-juegos esta filosóficamente relacionada con la dicotomía entre la libre voluntad y el determinismo. Partiendo de estos dos enfoques se puede desarrollar dos tipos de interactividad en el diseño de video-juegos:

a) El diseño de datos intensivos (que da como resultado un diseño determinista, en que el video-juego tiene que seguir una "trama" y un desarrollo férreamente determinados).
b) El diseño intensivos de procesos (en los que el diseñador se limita a fijar los procesos que libremente se pueden desarrollar en el video-juego: tanto en las

relaciones del video-jugador con el sistema como entre los diferentes subprocesos del sistema).

Según Crawford (1987), esta dicotomía da como resultado tres niveles principales de interactividad en los video-juegos. El nivel más bajo es la interacción con los datos (para Crawford la mayoría de video-juegos de aventuras en los que se ven implicados laberintos y objetos). El nivel medio es el de la interacción con los procesos (básicamente juegos de simulación al estilo Sims[39]). El nivel más alto de interactividad es la interacción con el libre albedrío (a través de modelos que permitan este tipo de interacciones). En este último tipo de video-juegos se trata de recrear un mundo simulado en el que son posibles todo tipo de interacciones entre los objetos y personajes que lo pueblan. (Martí, 2010)

2.5.2 ENTRETENIMIENTO

Se sabe que el entretenimiento es uno de los determinantes fundamentales de las motivaciones del video-jugador para jugar a video-juegos. Aunque este no es el único factor de motivación y los video-jugadores pueden jugar por muchas otras razones (desde evadirse de la realidad hasta fantasear con los mundos virtuales o simplemente entrar en contacto con otros video-jugadores) no cabe la menor duda de que muchos de los video-jugadores juegan para entretenerse. De hecho, se ha detectado que jugar a los video-juegos es la actividad más entretenida, muy superior a la televisión (Rodgers, 2002) y se ha afirmado que el entretenimiento es el estado emocional final que los jugadores esperan experimentar como consecuencia de jugar (Bartle cfr. En Ermi y Máyrá, 2005). La gente juega a los video-juegos porque se divierte con ellos y, de hecho, en el momento en que ya no le divierten (se "aburren" de él) dejan de jugar. Los retos que plantean al video-jugador, la amplia variedad de géneros existentes y un fuerte componente competitivo que anima a la superación (ya sea personal o frente a otros video-jugadores) representan un escenario idóneo para desarrollarse como actividad lúdica.

Son muchos los factores que pueden afectar a las percepciones de los video-jugadores a la hora de calificar un video-juego como "entretenido" (desde factores individuales, como la preferencia por el género de video-juego, hasta la novedad al

[39] Los Sims es un video-juego de simulación social y estrategia creado por el diseñador de videojuegos Will Wright, desarrollado por Maxis y publicado por Electronic Arts en el año 2000.

jugarlo por primera vez, etc.) pero uno de los factores que más influyen (y que puede ser controlado por los diseñadores) es el de la jugabilidad. (Martí, 2010)

2.5.3 JUGABILIDAD

La jugabilidad o playability por su término en inglés, ha sido definida como un factor "dependiente de la compilación total de todas las acciones del jugador o resolución de los resultados de la activación de los sistemas y subsistemas del juego" (King y Fawcett, "Game Theory: The Myth of Reality"). Es decir, básicamente, la jugabilidad tiene como objetivo relacionar todas las acciones, reacciones e interacciones tanto del video-jugador con el video-juego como entre los propios sistemas y subsistemas programados en el video-juego.

En el libro de Martí menciona a "Jarvinen, Helio y Mayra (2002) definen la jugabilidad desde dos puntos de vista: a) desde el diseño y b) desde la evaluación (comparando en este sentido la jugabilidad con la usabilidad de los entornos interactivos). Desde el punto de vista del diseño la definen como "las directrices que indican cómo implementar los elementos necesarios (como las reglas) para dar nacimiento a una especie de tiempo de juego o un entretenimiento social deseado". Desde el punto de vista de la evaluación (usabilidad) la definen como "una colección de criterios con los que evaluar el tiempo de juego o interacción de un producto" (Jarvinen, Helio y Mayra, 2002: 17). Es decir, la jugabilidad nos permite tanto desarrollar el video-juego como evaluar el producto final en función de cómo es percibido al ser jugado por los video-jugadores. De esta manera, una "buena jugabilidad" nos indicara que el video-juego "funciona" bien. Por ejemplo, nos indicará si los tiempos de respuesta a las acciones del video-jugador son correctos (ni demasiado rápidos ni demasiado lentos); si el nivel de dificultad en cada momento es el adecuado (y esta "equilibrado": es decir, que el video-juego no resulta ni demasiado fácil ni imposible de jugar); si los movimientos de los objetos, personas, etc., y sus actualizaciones en la pantalla y en las interacciones del sistema son correctos y fluidos, etc." (Martí, 2010)

Jarvinen, Helio y Mayra (2002) definen cuatro tipos de jugabilidad:

- La jugabilidad funcional: está relacionada con las variables que afectan el "gameplay"[40]. Entre estas variables se encuentran los mecanismos de control

[40] El gameplay es un término de difícil traducción al español pero responde básicamente a lo que se podría denominar "experiencia de juego".

(teclas de juego, por ejemplo); las interfaces (mandos y otros periféricos) a través de las cuales el video-jugador interactúan con el video-juego.

- La jugabilidad estructural: está relacionada con la "estética" del video-juego y entre sus principales variables se encuentran las reglas y estados del juego (diferentes cambios desencadenados por los patrones que dictan las reglas de juego.
- La jugabilidad audiovisual: está relacionada básicamente con la representación ofrecida por el video-juego. Entre las principales variables se encuentran la dimensionalidad (perspectiva isométrica, 2D o 3D); el punto de vista en primera persona, también denominada visión subjetiva, frente a la vista en tercera persona); y el nivel de realismo de la representación (del foto-realismo al abstraccionismo).
- La jugabilidad social: principalmente relacionada con los aspectos comunicativos y socioculturales que puede desencadenar el video-juego (desde los chats en los video-juegos online multiusuario hasta las comunidades virtuales de video-jugadores congregadas en tomo a un video-juego online).

2.5.4 SIMULACIÓN Y VIRTUALIDAD

Cuando se habla de simulación se habla, básicamente, de una representación de la realidad. No obstante se trata de una representación "a medida" cuyo objetivo es que nos permita interaccionar y relacionamos con lo representado según nuestros intereses. Es decir, muchas de las simulaciones en las que podemos estar pensando (simuladores de vuelo, por ejemplo) nos permiten "recrear" lo real partiendo de una manipulación de dicha realidad. En el caso de las simulaciones de conducción de vehículos, por ejemplo, nos permite eliminar riesgos de la experiencia real (como un posible accidente). Esto es, en la simulación se puede simular dicho accidente pero no sufrir sus consecuencias reales. En este sentido el concepto de simulación está íntimamente relacionado con el de virtualidad. Virtualidad como una parte diferente de la realidad. (Martí, 2010)

No cabe duda de que los espacios generados en las pantallas del computador son un entorno idóneo para la simulación. Y también lo son los video-juegos jugados a través de cualquiera de las múltiples pantallas en las que se pueden jugar. Ya se trate de la simulación de un terreno de juego (por ejemplo, un campo de fútbol) o de un universo de seres imaginarios, la simulación forma parte inalienable de los video-juegos. Una simulación que en el caso de los video-juegos ha ido siempre de la mano de nuevos

avances tecnológicos que permitieran presentar dicha simulación lo más verosímil y próxima a la realidad. De hecho Molesworth (2006), parafraseando a Baudrillard (1970, 1998), afirma que el placer que se obtiene de la simulación es resultado tanto de la distancia que lo mantiene de lo real como la alusión a lo real. (Martí, 2010)

En este sentido las simulaciones y virtualizaciones que ofrecen los video-juegos suelen incluir escenarios o lugares, objetos y también productos reales con el fin de representar lo más fidedigno posible a la realidad. De hecho, una de las justificaciones clásicas del emplazamiento de marcas y productos en video-juegos es el realismo que aportan (Nelson, 2002).

2.5.5 INMERSIÓN

Otra de las características fundamentales a la hora de experimentar un video-juego a la inmersión. La inmersión ha sido definida como "la sensación de estar rodeado por completo por otra realidad (…) que requiere por completo toda nuestra atención, todo nuestro aparato perceptivo" (Murray, 1997, citada en Ermi y Máyrá, 2005). Desde este punto de vista, los video-juegos no son el único género capaz de generar la inmersión del consumidor: cualquier narrativa de ficción (por ejemplo un libro o una película) es capaz de absorber nuestra atención hasta el punto de hacemos sentir inmersos en ese mundo de ficción. Sin embargo, en el marco de los medios de comunicación, la inmersión se aborda desde la desintermediación de dichos medios. Es decir, se considera la inmersión como un estado en el que el medio no es percibido. Dicho de otra forma, nos sentimos inmersos en una película cuando nos olvidamos de que estamos viendo una película a través de una pantalla y un proyector (o la televisión) y tan sólo atendemos a lo que está sucediendo ante nuestros ojos. Desde este punto de vista, el concepto de inmersión está muy relacionado con el de presencia. De hecho, "la inmersión y la presencia no son muy diferentes, y de hecho se utilizan a menudo como sinónimos" (Ermi y Mayra, 2005:4). Por ejemplo, Steuer define la presencia como "el grado en el que alguien se siente presente en un entorno mediado más que en su entorno físico inmediato" (Steuer, l992:79) y Lee como "un estado psicológico en el que la virtualidad de la experiencia no es percibida". Es decir, experimentamos un sentimiento de presencia (nos encontramos inmerso en un entorno mediado) cuando nuestras percepciones de lo que ocurre ante nosotros a través de este medio son similares a lo que experimentaríamos si no mediara ningún medio (como si lo "viviéramos" en la realidad). (Martí, 2010)

La inmersión se ha considerado pues tradicionalmente como uno de los factores que facilita la "conexión" con las experiencias mediadas. Ermi y Máyrá (2005), utilizando el marco teórico de Pine y Gihnore (1999) sobre los tipos de experiencias, otorgan a la inmersión un papel determinante en los video-juegos, al definir la experiencia de juego dentro de las "experiencias escapistas, en las que además de la participación activa, la inmersión juega un papel central" (Ermi y Máyrá, 2005:4). (Martí, 2010)

2.5.6 MULTIPLATAFORMA

La multiplataforma es una de esas características que acrecienta las dudas acerca de la consideración de los video-juegos como medio. El hecho de que no se pueda hablar de unas características específicas del medio de consumo sino de las características propias de cada uno de los dispositivos (computador, videoconsola, televisión, teléfono móvil, etc.) a través de los cuales se juega el video-juego refuerza su consideración más bien como género de entretenimiento audiovisual que como medio.

Los video-juegos son capaces de utilizar como "huésped" múltiples plataformas que ofrecen diversas características que pueden ser aprovechadas tanto a nivel de diseño (por ejemplo, afectando a la jugabilidad) como a la hora de desarrollar acciones de marketing a través de ellos. Por ejemplo, un video-juego jugado mediante un teléfono móvil puede aprovechar la tecnología bluetooth[41] del dispositivo para permitir diversas modalidades de juego entre los jugadores que se encuentran dentro del radio de acción del dispositivo. Un video-juego jugado mediante un computador con conexión a Internet puede contener enlaces a sitios de comercio electrónico de los productos emplazado en el video-juego, etc. Cada plataforma impone asimismo unas limitaciones técnicas como consecuencia sus características propias. Estas características pueden ser tanto de hardware como de software. Los teléfonos móviles, por ejemplo, presentan ciertas limitaciones a la hora de interactuar con su teclado por lo que la interfaz de juego debe ser especialmente estudiada. En el caso de los video-juegos online la velocidad de conexión a Internet obliga a considerar el tipo de resolución, gráficos y animaciones porque estos se tienen que renderizar en la pantalla; tomando en consideración que el escaso ancho de banda no provoque parones o saltos en el juego". (Martí, 2010)

[41] Especificación industrial para Redes Inalámbricas de Área Personal (WPAN) que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en la banda ISM de los 2,4 GHz.

2.6 GÉNEROS DE VIDEO-JUEGOS

El estudio de los video-juegos en lo referente a su taxonomía presenta tales dificultades que, a pesar de que se han llevado a cabo diversos intentos, actualmente se carece de una taxonomía general de los video-juegos. Son tantas las variables que pueden ser susceptibles de intervenir a la hora de realizar esta taxonomización (desde el número de video-jugadores participantes hasta las habilidades cognitivas necesaria para jugar a cada video-juego o las reglas de juego empleadas) que hasta ahora ninguna de las propuestas realizadas parece contar con un consenso general. Wolf (2001) basa su clasificación en el tipo de interactividad que ofrece el video-juego. Estallo (1995) utiliza una doble clasificación: psicologista en un primer nivel (centrada en las habilidades y recursos psicológicos utilizados por el video-jugador) y temática en un segundo nivel. Crawford (1982) divide los video-juegos en dos grandes categorías: a) juegos de habilidad y acción (que enfatizan las habilidades perceptivas y motoras) y b) juegos de estrategia (que enfatizan el esfuerzo cognitivo). Dentro de ambas categorías Crawford realiza una posterior subdivisión que enmarca a los diversos video-juegos atendiendo a diferentes categorías. Así por ejemplo, Pacman[42] (Namco, 1980) queda emplazado en la categoría de Laberintos y Pong o Breakout en la de Raquetas. (Martí, 2010) (Barrios, 2009)

[42] Conocido también popularmente como “comecocos”.

	Tipo	Ejemplo
Juegos de Acción	De Combate	Space Invaders, Asteroids
	De Laberinto	Pac-man
	De Deporte	
	De Raqueta	Pong, Breakout
	De Carrera	Dog Daze
	Miscelánea	Donkey Kong, Panic
Juegos de Habilidad	Aventuras	Adventure
	D & D (Dungeons & Dragons)	Temple of Apshai
	Juegos de Guerra	Eastern Front 1941
	De azar	
	Educacionales	Hangman
	Interpersonales	

Cuadro 2-1 Clasificación de Crawford (1982)

Fuente: (Martí, 2010, p. 29)

Arcade	Plataformas	(Prince of Persia)		
	Laberintos	(3d-Wolfestein)		
	Deportivos	(Winter Games)		
	Dispara y Olvida	(Space invaders)		
Simuladores	Instrumentales	(Flight Simulator)		
	Situacionales		Simuladores de Dios	Mitológicos (Populus)
				Socio-Económicos (Sim City)
				Bio-Ecológicos (Sim Earth)
			Simuladores deportivos	(Link 386 Pro)
Estrategias	Aventuras graficas (Monkey Island)			
	Juegos de rol (Eye of the Beholder)			
	War-Games (Perfect General)			
Juegos de mesa	Trivial pursuit			

Cuadro 2-2 Clasificación de Estallo (1995)

Fuente: (Martí, 2010, p. 30)

La clasificación que se ofrece es una revisión de la propuesta por Martí (2002). Dicha clasificación parte de los intentos de Crawford (1982) y Estallo (1995) por ofrecer una clasificación general de los video-juegos. Siendo plenamente consciente de todas sus limitaciones se presenta en el mismo sentido que la de los hermanos Le Diberder (1993), con un fin más "didáctico" que taxonómico. (Martí, 2010)

La clasificación propuesta comprende tres parámetros de clasificación.

El primer parámetro se basa en los mecanismos psicológicos desarrollados, siguiendo la línea utilizada por Crawford y Estallo. Este parámetro clasifica los video-juegos en tres grandes grupos según el tipo de habilidades cognitivas que el video-juego le exige al jugador. El primer grupo (Arcades) requiere habilidades meramente viso-motrices[43]. El segundo grupo (estratégicas/narrativos) comprende aquellos video-juegos que exigen unos mecanismos cognitivos más complejos El tercer grupo (aventuras de acción) se basa en la utilización conjunta de estos dos tipos de habilidades.

El segundo parámetro define las características morfológicas del mundo que se representa en el video-juego (características que determinan en buena parte tanto la mecánica como las reglas del video-juego). Por ejemplo, el hecho de que un tipo de video-juegos dentro de la categoría de los arcade tenga como espacio de juego un laberinto (pensemos en Pacman) determina el subtipo laberinto.

El tercer parámetro se corresponde con los aspectos contextuales del video-juego (hasta cierto punto resulta una variable temática) que van a determinar buena parte de la imagen gráfica, la estética del video-juego, etc. Por ejemplo, la imaginería utilizada y el tipo de contexto narrativo de las historias de fantasía mitológica propia de los video-juegos RPG (Rol Playíng Games), como Ultima, diferenciaría a este subtipo de otros video-juegos de la categoría estratégicos/narrativos (por ejemplo, de los juegos de guerra como Sun Tzu's Art of War). (Martí, 2010)

[43] La coordinación visomotriz implica coordinación entre lo que ve el ojo y lo que se le ordena a la mano que haga.

2.6.1 ARCADES

En un principio parecería un término poco correcto ya que la denominación de arcade proviene de un tipo de plataforma concreta, la de las "máquinas recreativas" (conocidas también en España y buena parte de Latinoamérica como "maquinitas"), y hoy en día este tipo de video-juegos, que se jugaban antes solamente en estas máquinas, se puede jugar tanto en consolas como en computadores, televisión y telefonía móvil. No obstante es un término de uso común totalmente generalizado y aceptado para este tipo de video-juegos que encontraremos en cualquier clasificación de las revistas especializadas. Se suele reservar esta categoría para definir aquellos video-juegos de la época dorada de los arcades clásicos o similares; video-juegos como Pong (Nolan Bushnell, 1972), Space Invaders (Toshihiro Nishikado, 1978), Asteroids (Ed Logg, 1979), etc. En este contexto se utiliza el término para designar todos los video-juegos que se fueron incorporando posteriormente a estas salas y que incluirá a los juegos de plataforma, deportivos, simuladores, etc. Todos estos video-juegos no sólo tienen como nexo común estar centrados en las habilidades viso-motrices de los video-jugadores sino que la plataforma en la que surgieron (las máquinas recreativas) condicionaron de manera fundamental las características de este tipo de video-juegos. Por ejemplo, el hecho de que se debiera favorecer la rotación de los jugadores (por intereses económicos obvios de los dueños de las salas recreativas) exigía que fueran partidas con tiempos de juego cortos así como unas reglas sencillas y un funcionamiento intuitivo y rápido de aprender. Estallo (1995) caracteriza este género de video-juegos bajo los siguientes parámetros:

- Ritmo rápido de juego.
- Tiempos de reacción complejos.
- Ausencia de componente estratégico (a lo sumo estrategias de ensayo-error).
- Atención focalizada.

A estas características se les podría sumar:

- Reglas sencillas e intuitivas.
- Alcance de la experiencia de juego (gameplay) incluso con tiempos de juego muy cortos.
- Estructuración de las partidas en niveles (de dificultad creciente).

2.6.1.1 PADDLES

Representan fundamentalmente dos tipos de video-juegos: a) aquellos video-juegos en los que el objetivo del video-jugador es interceptar un objeto, b) aquellos video-juegos en los que el objetivo del video-jugador es devolver un objeto. El ejemplo prototipo de paddle de intercepción sería el del video-juego en el que un objeto cae desde la parte superior de la pantalla y el jugador debe mover su pala-recogedor, que se encuentra en la parte inferior, de izquierda a derecha con el fin de recogerlo antes de que caiga al suelo. El otro tipo, el de devolución, está basado en el concepto de devolver un objeto, generalmente una pelota o un objeto que la simula, como si se tratara de un juego de tenis o ping-pong. El video-juego prototípico sería el conocidísimo Pong (Nolan Bushnell, 1972) que inspiró otro de los video-juegos más conocidos y populares, Breakout (Steve Jobs y Steve Wozniak, 1976). (Martí, 2010)

2.6.1.2 LABERINTOS

En este género de video-juegos el laberinto casi siempre viene representado con una visión cenital desde el punto de vista del video-jugador. El video-juego se desarrolla en una red de pasillos interconectados en los que el video-jugador debe conseguir una serie de objetos que le otorgarán puntos, a la vez que va esquivando o eliminando enemigos. El ejemplo prototípico es Pacman. Muchos otros video-juegos clásicos como Tank (1974) o los modernos FPS (First Person Shooters, también denominados 3D-Shooters) están basados en la mecánica de los video-juegos de laberinto. (Martí, 2010)

2.6.1.3 SHOOT'EM'UP

Estallo (1995) denomina este subgénero de video-juegos Dispara y olvida haciendo descender el término de un video-juego homónimo de los años 90 y los define de la siguiente manera: En esta categoría incluiremos aquellos juegos que se caracterizan por una acción trepidante, que se constituye en el eje fundamental del juego. Los escenarios se mantienen constantes y habitualmente se modifican tras haber eliminado un número suficiente de enemigos. Su desarrollo es lineal, de modo que una vez superado un escenario ya no se vuelve a él (al contrario que en los juegos de laberinto)." (Estallo, 1995: 138).

Crawford (1982) los define como "juegos de combate" que presentan una confrontación directa y violenta. El jugador humano debe disparar y destruir a los chicos malos controlados por el computador. El desafío es posicionarse correctamente

para evitar ser alcanzado por el enemigo mientras le disparamos." (Crawford, 1982:26). Cita como ejemplos de este subgénero video-juegos míticos como SpaceWar! (1961), StarRiders (1984), Space Invaders (1978), Battlezone (1980) y Red Baron (1990), entre otros, observando que mientras que en algunos video-juegos el punto de vista del jugador es en tercera persona en otros lo es en primera persona. Según la clasificación presentada por José Martí Parreño (2010), a este último tipo (es decir, visión en primera persona o subjetiva) se les consideraría en la categoría de simuladores.

2.6.1.4 SIMULADORES

Este subgénero de video-juego implica una dificultad añadida a la hora de definir sus características, debido a las propias connotaciones semánticas del propio término que lo define. De hecho, tal y como se ha visto, todos los video-juegos son "simuladores". Sin embargo, es cierto que es un término históricamente establecido para una categoría específica de video-juegos. Históricamente se ha utilizado el término para designar a un tipo de video-juegos heredero directo de los simuladores militares y que tiene como objetivo que el video-jugador tenga la sensación de encontrarse pilotando o conduciendo un vehículo. (Martí, 2010)

La percepción de inmersión es por tanto prioritaria, por lo que el tipo de visión que adopta el video-jugador resulta una característica fundamental: debe ser una visión en primera persona (visión subjetiva), que le dé la sensación de ir a los mandos del vehículo. Los dos grandes grupos en los que se podría subdividir este género vendrían determinados por el tipo de vehículo pilotado. Así, tendríamos simuladores militares (que incluirían todo tipo de vehículos desde aviones, helicópteros, tanques, submarinos, etc.) e incluirían también a los simuladores de vehículos fantásticos o imaginarios (naves espaciales por ejemplo) y, por otra parte, los simuladores civiles, principalmente automovilísticos como coches y motocicletas, aunque también existen de aviones (no militares) y otros vehículos como grúas o motos acuáticas.

2.6.1.5 FPS (3D SHOOTERS)

La característica principal de estos video-juegos es ofrecer al video-jugador una visión subjetiva de una serie de blancos u objetivos (enemigos) a los que se debe eliminar. El sitio web especializado en video-juegos Game Research equipara los FPS (First-Person-Shooters) a los 3D Shooters a los que define como video-juegos de acción en los que la acción se ve a través de los ojos del protagonista y en el que los gráficos son tridimensionales (y a menudo construidos mediante polígonos).

Este subgénero que híbrida elementos del género de laberinto y en muchas ocasiones incorpora imaginería y temática de los juegos de rol, está emparentado directamente con atracciones offline como los salones cyber-laser, en los que los participantes se arman con dispositivos y pistolas láser que permiten recrear enfrentamientos entre dos equipos que recorren un espacio laberíntico (y también están relacionados aunque más indirectamente con actividades lúdicas como el paintball). Los ejemplos prototípicos de este subgénero de video-juegos son Doom y Quake, video-juegos que sentaron las bases del género (aunque Wolfstein 3D fue el primero de ellos). (Martí, 2010)

2.6.1.6 DEPORTIVOS

Los video-juegos deportivos surgen con la finalidad de simular en el espacio virtual de las máquinas recreativas, computadores, etc., las reglas y objetivos de los juegos basados en algún deporte (fútbol, golf, tenis, baloncesto, etc.). El video-juego Football (Stubbens, Albaugh y Rains, 1978) es considerado el primer video-juego deportivo de la historia. Fue pionero en cuanto a que permitió que el campo de juego fuera más grande de lo que aparecía simplemente en la pantalla, a través del desplazamiento o scrolling[44] y también fue el primero en introducir como interfaz una rueda de desplazamiento o trackball[45].

La evolución del género ha propiciado el paso de un subgénero con las características propias de la categoría arcade a un género híbrido en el que cada vez han ido adquiriendo más importancia los elementos estratégicos. Por ejemplo, en los video-juegos de fútbol actuales como el FIFA (Electronic Arts) se pueden plantear tácticas de juego, componer las alineaciones con unos jugadores de unas determinadas características, etc. Este tipo de video-juegos incluso ha ido incorporando características de los denominados simuladores sociales y de gestión, al estilo Sim City (Electronic Arts, 1989), permitiendo que el video-jugador llegue a asumir los roles de presidente del equipo, y que de su buena gestión económica dependa mejorar el rendimiento del equipo (comprando, en función de la solvencia económica, jugadores virtuales para el

[44] Un ejemplo en los video-juegos es el llamado "scroll horizontal" cuando la acción se desarrolla horizontalmente como en Sonic o Super Mario Bros y juegos de "scroll vertical" cuando se desarrolla verticalmente como Commando o Aero Fighters. También podemos encontrarnos con juegos de scroll multidireccional como juegos de fútbol o hockey con vista aérea.

[45] Dispositivo apuntador estacionario compuesto por una bola incrustada en un receptáculo que contiene sensores que detectan la rotación de la bola en dos ejes —como si fuera un ratón de computadora boca arriba, pero con la bola sobresaliendo más.

equipo, etc.). En los últimos años, el género ha experimentado una nueva evolución (y un crecimiento espectacular) de la mano de plataformas como la Wii de Nintendo acercando al género más a las características de lo que se podría denominar simuladores deportivos o *action sport* (video-juegos de acción deportiva en los que el video-jugador adopta el rol de jugador de tenis, etc.). En este último género de video-juegos, al video-jugador se le exige una participación física activa que trata de simular el ejercicio físico real que realizaría al practicar dicho deporte (como en el caso del Wii Sports de Nintendo. (Martí, 2010)

2.6.1.7 BEAT'EM'UP

Son los denominados video-juegos "de lucha". Básicamente son video-juegos de lucha o combates uno a uno generalmente relacionados con técnicas de lucha oriental como el karate o el taekwondo. También en esta categoría entrarían todos los video-juegos de boxeo o lucha libre americana, por ejemplo. El primer video-juego de la historia que se considera de lucha, Warrior (1979), presentaba unas características muy diferentes a los video-juegos de lucha actuales. Se trataba de un duelo a espada en el que los espadachines eran vistos desde un plano cenital. Sería Karate Champ (Data East, 1984) quien introduciría la visión lateral característica de este tipo de video-juegos sentando las bases de un subgénero" que relanzó la popularidad de los video-juegos arcad hasta niveles insospechados (con una serie de títulos, como Street Fighter (Capcom, 1990) y Mortal Kombat (Midway, 1992) cuya enorme popularidad provocó que posteriormente fueran llevados al cine). Con las innovaciones tecnológicas y la llegada de los motores 3D en los video-juegos se produjo la irrupción de los denominados 3D Fighters (como el exitoso Wrtual Fighter (Sega, 1993) que volvieron a relanzar el género, y cuyas características (tridimensionalidad de los combatientes, rotaciones de 360° en el espacio de juego, etc.) los diferenciaban técnicamente de los beat'em'up clásicos. (Martí, 2010)

2.6.1.8 PLATAFORMAS

El video-juego con el que nació el género fue sin duda alguna Donkey Kong (Nintendo, 1981), aunque el primer video-juego considerado de plataformas fue el Space Panic. Crawford (1982) cataloga este género en un apartado de miscelánea refiriéndose a él como "(...) un juego que parece vagamente como un juego de carreras con obstáculos inteligentes." (Crawford, 1982:30). Donkey Kong marca la pauta mediante la cual se desarrollará posteriormente este subgénero de video-juegos: la no

necesaria confrontación directa con "enemigos" (a los que puedes saltar o esquivar) la aparición de una serie de plataformas sobre las que se desplaza el jugador (que en el caso concreto de Donkey Kong estaban unidas por escaleras y a las que en otros video-juegos, el jugador puede acceder saltando de una a otra, como en el caso de Super Mario Brothers (Nintendo, 1985); la recolección de todo tipo de objetos que aportan puntos extras y bonos al video-jugador; y la estructuración del video-juego en una serie de niveles (entre los que se incluyen niveles ocultos y multitud de puertas que los conectan entre sí). Donkey Kong presentaba una única pantalla estática como escenario (al igual que los primeros beat'em up) en la que los movimientos del video-jugador serpenteaban de izquierda a derecha hasta llegar a la parte superior de la misma desde la parte inferior en la que se encontraba. El siguiente paso, al igual que en los beat'em'up, consistió en dotar a este tipo de video-juegos de un desplazamiento horizontal o scroll horizontal con el que ampliar el campo de acción y dinamizar la velocidad y los movimientos del jugador. (Martí, 2010)

2.6.1.9 PUZZLES EN ACCIÓN

El ejemplo prototípico de rompecabezas o puzzle[46] de acción es el archiconocido video-juego del diseñador ruso Alexei Pajitnov: Tetris (1985). Es el video-juego que inaugura el género y en él se encuentran todas las características que lo definen: altos grados de coordinación viso-motriz; tiempos de respuesta rápidos; visión espacial; incremento en los tiempos de reacción a las pruebas presentadas en las sucesivas pantallas, etc. Kim y Pajitnov (2000) definen los puzzles como entretenidos y resolubles (es decir, tienen una solución que debe encontrar el video-jugador). Pueden estar basados en imágenes, lógica y palabras (así como combinar estos tres elementos) y en el caso de los video-juegos tiene una especial importancia la presión temporal (es decir, el video-jugador tiene un tiempo límite para encontrar la solución al desafío planteado). (Martí, 2010)

2.6.1.10 ADAPTACIONES

Entrarían en este subgénero todos aquellos video-juegos que presentan las características de los video-juegos de arcade (coordinación viso-motriz, etc.) y que han sido "trasvasados" del mundo offline al medio digital. Un ejemplo prototípico sería el juego musical conocido como Simon[47]. Exige memorización y reflejos rápidos y ha sido

[46] Un rompecabezas, un juego en el que hay que armar una figura.
[47] Juego electrónico creado por Ralph Baer en 1978. Tuvo un gran éxito durante los 80. Tiene forma de disco, en una de sus caras se puede ver cuatro cuadrantes, cada uno con un color: verde, rojo, azul y

adaptado a formatos de advergame[48] en numerosas ocasiones. Juegos como el tres en raya, y juegos de memorización como las figuras emparejadas también entrarían en esta categoría. (Martí, 2010)

2.6.2 ESTRATÉGICOS / NARRATIVOS

En esta categoría se enmarcan todos aquellos video-juegos que implican procesos cognitivos más complejos y agrupa tanto a aquellos video-juegos basados en fuertes componentes estratégicos como a los que implican razonamientos deductivos propios de los video-juegos denominados aventuras gráficas (que presentan además argumentos y líneas narrativas complejas). Los dos grandes subgrupos que conforman la categoría son los denominados juegos de estrategia y las aventuras gráficas (que como veremos posteriormente han evolucionado hacia otras formas híbridas más complejas denominadas aventuras de acción). (Martí, 2010)

2.6.2.1 JUEGOS DE ESTRATEGIA

Comprenden una vastísima categoría que abarcaría desde los juegos de rol (los RPG o Rol Playing Game) hasta las últimas generaciones de simuladores sociales al estilo *Sim City*. Sin duda alguna, uno de los subgrupos más importantes de la categoría son los denominados juegos de guerra (con subcategorías como los juegos de estrategia en tiempo real o RTS por sus siglas en inglés) destaca la saga clásica *Age of Empires* (Microsoft, 1997). En esta categoría también se enmarcan video-juegos estratégicos de difícil catalogación como Lemmings (Psygnosis, 1991). Debido a que estos video-juegos incluyen una serie de parámetros que permiten al video-jugador construir o diseñar el mundo virtual en el que se desarrollan muchos de estos video-juegos de estrategia (como Black and White, Lionhead, 2001) son a menudo llamados en inglés ***God games*** (juegos de Dios), haciendo referencia al hecho de que el video-jugador actúa como creador del universo en el que se desarrolla el juego. (Martí, 2010)

2.6.2.2 AVENTURAS GRÁFICAS

Este género surge con Adventure (Crowther, 1972). Se trataba de una recreación interactiva de los mundos fantásticos de Dragones y Mazmorras (o D&D, Dungeons &

---

amarillo en su versión original. Su nombre se debe por el conocido juego tradicional del mismo nombre: Simón dice, de donde se inspira.

[48] El advergaming (del inglés advertising y game) es la práctica de usar video-juegos para publicitar una marca, producto, organización o idea, http://www.exelweiss.com/blog/37/advergaming-cuestiones-basicas/ consultado: 30-Abril-2012.

Dragons) y los mundos relacionados con la imaginería y narrativa tolkieniana[49]. Este subgénero de video-juego posee un fuerte componente narrativo y deductivo centrado en las pistas que va descubriendo el video-jugador. Dentro del género encontramos una gran variedad de tipologías centradas fundamentalmente en dos grandes grupos: las aventuras textuales y las aventuras gráficas (si bien es cierto que hasta cierto punto se podría decir que actualmente las segundas han devorado casi por completo a las primeras). De hecho, primero surgieron las aventuras textuales (como la propia Adventure) y, de la mano de los nuevos desarrollos tecnológicos que permitían la incorporación de imágenes, animaciones, etc., se desarrollaron posteriormente las aventuras gráficas (con títulos emblemáticos como Zork (Infocom, 1979) o Maniac Mansión (LucasArts, 1987). Posteriormente, estas aventuras gráficas fueron evolucionando hasta llegar a lo que actualmente se conoce como narraciones interactivas tipo Myst (Cyan, 1993) (a las que Kim y Pajitnov (2000) enmarcan en los puzzles narrativos). (Martí, 2010)

Este subgénero está muy vinculado al nacimiento del computador personal y a la posibilidad de grabar en un disco las partidas (puesto que éstas se prolongan considerablemente en el tiempo y requieren de muchas sesiones, reanudadas en el último punto en el que se dejó grabada, para poder concluirlas). Entre las características básicas de este subgénero encontramos:

- Procesos cognitivos complejos: procedimientos deductivos e inductivos, etc.
- Ritmo de juegos lentos (el video-jugador debe explorar detalladamente el entorno en busca de pistas y objetos, así como entablar conversaciones con los diversos personajes, etc.).
- Duración de la partida prolongada (semanas e incluso meses) que hace necesario guardar las partidas en un disco (posteriormente en el disco duro del computador).
- Aparición de un inventario de objetos que el jugador va adquiriendo y que le ayudan a resolver los diferentes puzzles.
- Estructuración en pruebas (puzzles) que el jugador debe superar para pasar de nivel.

[49] Etimología: epónimo del literato y lingüista anglo-sudafricano John Ronald Reuel Tolkien, Tolkien sitúa a un enemigo externo como base de la narración y una imperiosa necesidad de sanar una tierra herida por el mal.

- Estructura ramificada de la narrativa que permite en ocasiones resolver ciertos puzzles mediante procedimientos diferentes (e incluso obtener diversos finales).

2.6.2.3 AVENTURAS DE ACCIÓN

Este subgénero se puede considerar un complejo híbrido de los dos subgéneros anteriores al presentar características estratégicas y narrativas de las aventuras gráficas e integrar múltiples subtipos del género de los Arcades; desde juegos de lucha hasta shoot'em'up de plataformas. El ejemplo prototípico sería el de video-juegos como Tomb Raíder (Eidos Interactive, 1996) y todos los subgéneros que han ido evolucionando fundamentalmente en torno a líneas argumentales que se podrían clasificar también temáticamente; la aventura arqueológica (con el ya mencionado Tomb Raider, como máximo exponente), el survival horror (Resident Evil, Capcom, 1996), la aventura policíaca (Max Payne, Gathering of Developers, 2001), la aventura de misterio (Alone in the Dark, Infogrames, 1992), la aventura militar (Metal Gear Solid, Konami, 1987), etc. La estructura narrativa alterna ritmos rápidos y frenéticos, basados en pruebas de coordinación visomotriz con momentos estratégicos e incluso narrativas (incluyendo escenas cinemáticas o cut-scene). El paralelismo existente entre este género de video-juegos y el denominado cine de acción de Hollywood no ha pasado inadvertido para muchos autores. (Martí, 2010).

2.6.3 MUNDOS VIRTUALES

Un mundo virtual es un tipo de comunidad virtual en línea que simula un mundo o entorno artificial inspirado o no en la realidad, en el cual los usuarios pueden interactuar entre sí a través de personajes o avatares[50], y usar objetos o bienes virtuales.

Para ser un mundo virtual, se requiere un mundo en línea persistente, activo y disponible 24 horas al día y todos los días. Mundos virtuales son hechos para que los usuarios vivan e interactúen, generalmente en tiempo real. Los personajes, o avatares, son representados por gráficos en 2D o 3D según el mundo virtual.

Aparecieron mundos virtuales con fines profesionales de aprendizaje (simuladores de vuelo), de enseñanza (MMOLE[51]) o en el entorno médico, pero en la

[50] Representación gráfica, generalmente humana, que se asocia a un usuario para su identificación. Los avatares pueden ser fotografías o dibujos artísticos, y algunas tecnologías permiten el uso de representaciones tridimensionales.

[51] Massively Multilearner Online Learning Environments, en español Entornos de Aprendizaje Online Multiaprendices y Masivos.

actualidad está siendo llevado por las empresas de ocio electrónico, que ven en esta tecnología una nueva era para video-juegos. Aunque no son limitados en video-juegos, muchos de estos mundos virtuales son conocidos como video-juegos masivos en línea o MMO (videojuego multijugador masivo en línea o Massively multiplayer online games por sus siglas en inglés).

TIPOS DE MUNDOS VIRTUALES

Existen varios tipos de mundos virtuales, con formas y objetivos distintos.

- Ocio / Entretenimiento / Social
    - MMORPG (massively multiplayer online role-playing games): Video-juego de rol multijugador masivo en línea
    - MMOFPS (massively multiplayer first-person shooter)
    - Metaverso es un concepto muy parecido a MMORPG: se trata de espacios 3D totalmente inmersivos
    - MMORLG: massively multiplayer online real-life games
    - Juegos sociales (Social Games) han aparecido más recientemente, con el objetivo principal de la interacción entre personajes que ya se conocen o no.
- Educativo
    - MMOLE (Massively Multilearner Online Learning Environments): Entornos de Aprendizaje Online Multiaprendices y Masivos
- Profesional / Comercial / Simuladores
    - Mundos virtuales han tenido éxito muy temprano para simulaciones de vuelo, aplicaciones en la medicina o reproducción de un entorno especial.

Origen y denominación

Según el proyecto Educación Espacios Virtuales 3D[52], los Mundos Virtuales son una combinación de realidad virtual dentro de un entorno de chat, también conocidos como Entornos Virtuales Multi-usuario (MUVEs, Multi-User Virtual Environments).

[52] http://educacionmetaverso.wordpress.com/ Consultado 9/06/2012

El término Mundo Virtual fue utilizado por los creadores del juego Ultima Online, de hecho los Mundos Virtuales nacieron y se desarrollaron inicialmente cómo entornos de juego, y desde un punto de vista técnico, son el producto de la combinación de un entorno gráfico 3D que incorpora sistemas de interacción social basados en chat desarrollados en el Mundo de Dominios Multi-usuario (MUDs[53]). En un MUD, cada usuario toma control de un personaje, encarnación, carácter, etc., computarizado. Se puede caminar alrededor, chatear con otros personajes, explorar y crear salas de conversación, descripciones e ítems.

Ejemplos de Mundos Virtuales en español, SecondLife, Habbo, Sanalika, Smeet, Club Penguin como los más importantes.

[53] Un MUD es un programa de computador sin gráficos, accesible por Telnet, en el cual los usuarios pueden introducirse y explorar.

2.7 CLASIFICACIÓN DE CONTENIDO DE VIDEO-JUEGOS

Un sistema de clasificación de contenido de video-juegos es un sistema usado para la clasificación de video-juegos en grupos idóneos relacionados[54]. La mayoría de estos sistemas están asociados con y/o patrocinados por un gobierno y a veces forman parte del sistema de clasificación de películas local, lo importante es definir el acceso por edad[55]. (Martí, 2010)

País/edad	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17+	Adulto	[3] Notas
ESRB				EC		E				E10+			T				M	AO	
OFLCA	G									PG			M		MA15+			RC	Los videojuegos clasificados RC son prohibidos a la venta, alquiler o exhibición en Australia.
OFLCNZ	G									PG			R13		R15	R16		R18	
BBFC	UC				U					PG			12			15		18	
ELSPA[4]	3+						7+						12+		15+	16+		18+	Usado hasta 2002; reemplazado por los sistemas PEGI o BBFC.
PEGI				3+			7+						12+			16+		18+	En Portugal, algunas clasificaciones difieren del estándar PEGI.
VET																			Usa el sistema PEGI.
USK	All					6							12			16		18	
MJ/DEJUS	L									10		12		14		16		18	
CERO	A												B		C		D	Z	
EOCS/CSA	General														R			18+	Usado principalmente para videojuegos bishōjo.
GRB	A												12			15		18	El sistema KMRB ya no clasifica videojuegos.
TIGRS				Apto para la familia						Contenido adolescente							Contenido adulto		Creado para clasificar videojuegos producidos por desarrolladores independientes.

Figura 2.25 Clasificación de contenido de video-juegos

Fuente: http://es.wikipedia.org/wiki/Sistema_de_clasificaci%C3%B3n_de_contenido_de_video-juegos

La Figura 2.26 muestra el uso de los sistemas de clasificación de contenido de video-juegos en el mundo. Los países pintados con degradados usan más de un sistema.

[54] http://es.wikipedia.org/wiki/Sistema_de_clasificaci%C3%B3n_de_contenido_de_video-juegos
[55] La edad a la cual un individuo alcanza la mayoría de edad varía según el país.

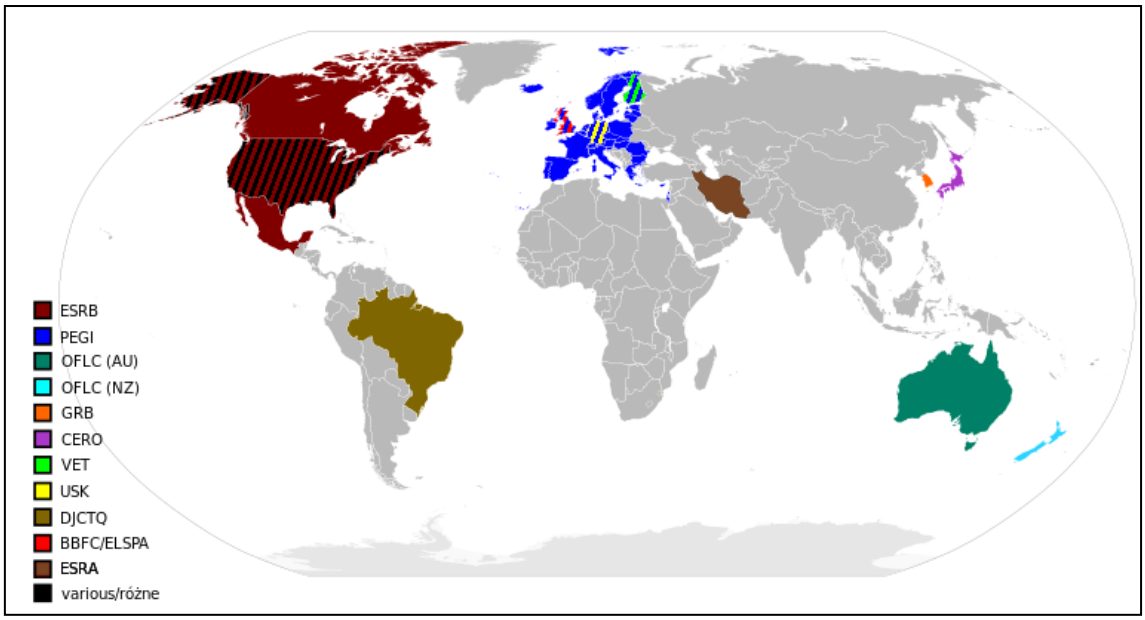

Figura 2.26 Uso de los sistemas de clasificación de contenido de video-juegos

Fuente: http://es.wikipedia.org/wiki/Sistema_de_clasificaci%C3%B3n_de_contenido_de_video-juegos

2.8 ENTERTAINMENT SOFTWARE RATING BOARD (ESRB)

Entertainment Software Rating Board (ESRB) por sus siglas en inglés es un sistema norteamericano para clasificar el contenido de los video-juegos, y asignarle una categoría dependiendo de su contenido[56]. (Entertainment Software Rating Board, 1994)

Fue establecido en 1994 por la Entertainment Software Association (ESA), la anteriormente llamada Interactive Digital Software Association (IDSA).

ESRB realiza de forma independiente clasificaciones, entregando lineamientos y los principios de privacidad para la industria de los video-juegos. Primero clasifica los video-juegos según su contenido de violencia física o verbal y otros elementos como el contenido sexual. Esta clasificación orienta y ayuda a los padres y consumidores a elegir los video-juegos que son correctos para su familia, idea propiciada tras la aparición del video-juego Mortal Kombat[57].

[56] http://www.esrb.org/ratings/faq.jsp Consultado el 30 de enero de 2012.
[57] Comúnmente abreviado MK es una exitosa serie de video-juegos de lucha creada por John Tobias y Ed Boon. Inicialmente desarrollada por la empresa Midway Games, y por Warner Bros. desde 2009, en su momento atrajo mucha atención por lo sangriento y sus gráficos digitalizados.

Los símbolos que usa la ESRB son letras alfabéticas estilizadas que tienen la intención de indicar la etapa para la que es adecuado el juego. La ESRB usa actualmente siete clasificaciones diferentes[58]. (Entertainment Software Rating Board, 1994)

2.8.1 CLASIFICACIONES NORMALES

Clasificación ESRB	Nombre	Año de entrada	Edad adecuada	Descripción de las clasificaciones
	*Early Childhood* (Primera infancia)	1994	De 0 a 5 años y gente con problemas mentales	Material con contenido apto para niños. Los juegos que entran dentro de esta categoría son específicamente desarrollados para niños pequeños y usualmente son de orientación educacional.
	*Everyone* (Todos)	1998	Todas las edades o 6 años en adelante	Abarca temas aptos para todas las edades o de 6 años en adelante. Los títulos comprendidos dentro de esta categoría, pueden contener algo de animación, fantasía o violencia moderada o el uso de insultos suaves.
	*Everyone 10 and up* (Todas las personas mayores de 10 años)	2005	10 años	Material idóneo para edades de 10 y más años. Las obras dentro de esta categoría contienen animaciones, fantasía o violencia media, insultos regulares o sangre en temas sugerentes.

[58] The ESRB Game Ratings & Descriptor Guide en http://www.esrb.org/ratings/ratings_guide.jsp consultado el 31 enero 2012.

	*Teen* (Adolescentes)	1994	13-16 años	Contenido que puede ser adecuado para mayores de 13 a 16 años. Los productos de este género contienen de manera limitada violencia, temas sugerentes, humor crudo, sangre, juegos de azar simulados, o uso de lenguaje fuerte.

Cuadro 2-3 Clasificaciones normales de la ESRB

Fuente: ESRB http://www.esrb.org/ratings/ratings_guide.jsp

2.8.2 CLASIFICACIONES RESTRINGIDAS

Clasificación ESRB	Nombre	Año de entrada	Edad sin restricciones	Descripción de las clasificaciones
	*Mature (17+)* (Edad madura)	1994	17 (Menores deben ir acompañados de un adulto)	Artículos que pueden contener material pertinente para edades de 17 o más años. Las obras de esta categoría muestran violencia, sangre y horror, temas sexuales o insultos. Muchos vendedores aplican políticas sobre no vender juegos con este ordenamiento a menores de edad sin la presencia y aprobación de algún tutor. Desde el año 2003, el símbolo lleva la leyenda "17+" al lado de la palabra *Mature*, a pesar de que estos juegos están destinados a personas mayores de 17 años.
	*Adults Only (18+)* (Sólo adultos)	1994	18 (No se admiten menores)	Contenido solamente para adultos. Los títulos en esta categoría incluyen escenas prolongadas de violencia extrema o temas sexuales y desnudez. La categorización *AO* es tema de controversia por las exageradas restricciones que impone a algunos juegos.

Cuadro 2-4 Clasificaciones restringidas de la ESRB

Fuente: ESRB http://www.esrb.org/ratings/ratings_guide.jsp

2.8.3 CLASIFICACIÓNES SIN RESTRICCIONES

Clasificación ESRB	Nombre	Año de entrada	Edad adecuada	Descripción de las clasificaciones
	*Rating Pending* (Clasificación pendiente)	1994	Ninguna	Aparece en juegos enviados a la ESRB y que esperan una clasificación final. Sin embargo, una vez calificados, toda la publicidad pre-lanzamiento debe contener la calificación oficial del juego puesta por la ESRB. Algunos juegos, dependiendo de la intensidad de su contenido, pueden tener una renuncia de responsabilidad en la que aparece "**Puede contener contenido inapropiado para niños**" para muchos juegos que están clasificados con **T** (Teen), **M** (Mature) y **AO** (Adults Only).

Cuadro 2-5 Clasificaciones sin restricciones de la ESRB

Fuente: ESRB http://www.esrb.org/ratings/ratings_guide.jsp

2.8.4 EN DESUSO

Clasificación ESRB	Nombre	Fecha de uso	Edad adecuada	Descripción de las clasificaciones
	**Kids to Adults** (Desde niños hasta adultos)	1994–1997	6 años	Material con contenido propicio para edades de 6 o más años. Estos títulos eran diseñados para personas de varias edades y gustos. Pueden contener violencia mínima, comicidad o algo de lenguaje crudo. Fue reemplazada por **Everyone** el 1 de enero de 1998.

Cuadro 2-6 Clasificaciones en desuso de la ESRB

Fuente: ESRB http://www.esrb.org/ratings/ratings_guide.jsp

2.8.5 DESCRIPCIONES Y DEFINICIONES

Los descriptores de contenido aparecen en la parte posterior de la caja del juego y en anuncios impresos o sitios web que hagan referencia al juego. La ESRB tiene más de veinte descriptores de contenido, como referencias al alcohol, violencia y desnudez. Abajo, los descriptores de contenido usados en la clasificación:[59]

- Alcohol Reference (*Referencias al Alcohol*)
    - Referencias o imágenes de bebidas alcohólicas.
- Animated Blood (*Sangre animada*)
    - Caricaturas o imágenes pixeleadas de sangre.
- Blood (*Sangre*)
    - Imágenes de sangre realistas.
- Blood and gore (*Sangre y mutilaciones*)
    - Imágenes de sangre o de mutilaciones de partes del cuerpo.
- Cartoon Violence (*Violencia animada*)
    - Acciones violentas que involucran personajes caricaturescos. Puede incluir violencia donde un personaje ha salido y eso después de una acción donde se ha tratado de lastimar.
- Comic Mischief (*Travesuras cómicas*)
    - Escenas gráficas de golpes o de humor vulgar.
- Crude Humor (*Humor crudo*)
    - Humor vulgar, incluyendo humor escatológico.
- Drug References (*Referencias a drogas*)
    - Referencia o imágenes de drogas ilegales.
- Edutainment (*Entretenimiento educativo*)
    - El contenido de producto provee al usuario conocimientos específicos desarrollando o reforzando el aprendizaje por medio de maneras entretenidas. El desarrollo de conocimientos es la parte integral del producto.
- Fantasy Violence (*Violencia de fantasía*)

[59] (Entertainment Software Rating Board, 1994) *www.**esrb**.org*

    - Acciones violentas de naturaleza fantástica, involucrando humanos o criaturas no humanas en situaciones fácilmente distinguidas de la realidad.
- Informational (*Informativo*)
    - Todo el contenido del producto incluye datos, hechos, información de recursos, materiales referentes o instrucciones.
- Intense Violence (*Violencia intensa*)
    - Imágenes gráficas o realistas de conflictos físicos. Puede incluir imágenes extremas o realistas de sangre, mutilaciones, armas de fuego, heridas físicas y la muerte.
- Language (*Lenguaje*)
    - Uso pequeño o moderado de groserías .
- Lyrics (*Letra de canciones*)
    - Referencia menor a groserías, sexualidad, violencia, alcohol o drogas en la música del juego.
- Mature Humor (*Humor maduro*)
    - Bromas y travesuras vulgares o crudas, incluyendo humor escatológico.
- Mild Violence (*Violencia menor*)
    - Escenas de personajes dibujados en situaciones no seguras o violentas.
- Nudity (*Desnudez*)
    - Imágenes de desnudez prolongadas.
- Partial Nudity (*Desnudez parcial*)
    - Desnudez parcial o no muy prolongadas.
- Real Gambling (*Apuestas reales*)
    - El jugador puede apostar dinero real o propiedades.
- Sexual Themes (*Temas sexuales*)
    - Referencias o imágenes sexuales menores o moderadas, puede incluir desnudez parcial.
- Sexual Violence (*Violencia sexual*)
    - Imágenes de violación y otros actos sexuales violentos.
- Simulated Gambling (*Apuestas simulados*)
    - El jugador puede apostar sin perder dinero o propiedades.

- Some adult assistance may be needed (*Se puede necesitar asistencia de un adulto*)
    - Solamente para descripciones en la temprana infancia.
- Strong Language (*Lenguaje fuerte*)
    - Groserías y referencias explicitas al sexo, violencia, alcohol o uso de drogas.
- Strong Lyrics (*Letras de canciones fuertes*)
    - Referencias sexuales, a la violencia, al alcohol o drogas en la música de forma explicita o frecuente.
- Strong Sexual Content (*Contenido sexual fuerte*)
    - Imágenes explicitas o frecuentes de comportamiento sexual, posiblemente incluya desnudez.
- Suggestive Themes **(*Temas sugestivos*)**
    - Referencias o materiales provocativos menores.
- Tobacco Reference (*Referencias al tabaco*)
    - Referencias o imágenes de productos de tabaco.
- Use of Alcohol (*Uso de alcohol*)
    - Consumo de bebidas alcohólicas.
- Use of Drugs (*Uso de drogas*)
    - Consumo o uso de drogas ilegales.
- Use of Tobacco **(*Uso de tabaco*)**
    - Consumo de productos de tabaco.
- Violence (*Violencia*)
    - Escenas que involucran conflictos agresivos.

2.9 PAN EUROPEAN GAME INFORMATION (PEGI)

Pan European Game Information o PEGI es un sistema europeo para clasificar el contenido de los video-juegos y otro tipo de software de entretenimiento. Fue desarrollado por la ISFE y entró en práctica en abril del 2003. El sistema PEGI se aplica en 25 países sin tener relación alguna con la Unión Europea[60]. (Pan European Game Information, 2003). La participación es voluntaria, a discreción del creador. Para obtener las categorizaciones de cualquier software, el desarrollador realiza un cuestionario, el cual es después evaluado por la NICAM para otorgarse finalmente la clasificación. Existen dos formas de clasificación para cualquier software; una de edad sugerida y otra sobre seis descripciones de contenido, tales como el uso de lenguaje indecente, violencia, etc.

2.9.1 CLASIFICACIÓN

La PEGI contempla cinco categorías por edad. En Portugal, la legislación local es conflictiva con algunas de las clasificaciones (3 y 7) y son cambiadas consecuentemente. La clasificación de la PEGI cambió en 2009 cuando los símbolos de las categorías por edad fueron coloreados, mientras que en los de referencia al contenido del juego sumaron la leyenda de su significado[61]. (Pan European Game Information, 2003)

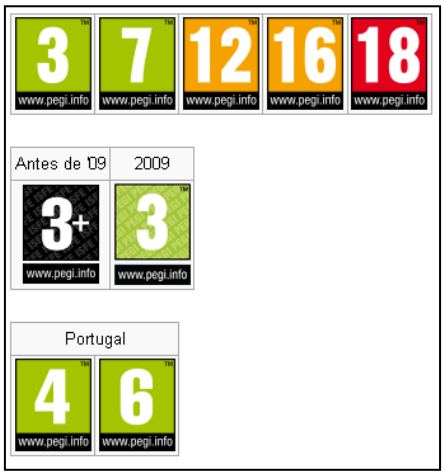

Figura 2.27 Clasificación PEGI por edades

Fuente: http://es.wikipedia.org/wiki/PEGI

[60] http://www.pegi.info/en/index/id/952 consultado el 2 de febrero de 2012.
[61] El sistema de calificación por edades de los video-juegos cambia de imagen, http://www.20minutos.es/noticia/545656/0/clasificacion/edades/video-juegos/ consultado el 2 de febrero de 2012.

2.9.2 DESCRIPTORES DE CONTENIDO

Descripción del logo	Nombre del descriptor	Explicación
Un puño atacando.	Violencia	Puede contener escenas de personas que sufran lesiones o que mueran, a menudo mediante el uso de armas. También pueden contener derramamiento o gotas de sangre.
Un bocadillo con diversos símbolos.	Lenguaje explícito	Puede contener groserías, tacos, insinuaciones sexuales, amenazas, y toda clase de insultos.
Una araña.	Miedo	Puede contener escenas que se consideran demasiado perturbadoras o aterradoras para los más jóvenes o los jugadores emocionalmente vulnerables.
Símbolo Venus y Símbolo Marte	Sexo	Puede contener referencias a atracciones sexuales o relaciones sexuales. También puede contener desnudos y personajes femeninos vestidos con ropa sugestiva.
Una jeringuilla.	Drogas	Puede contener referencias a las drogas ilegales o de una sustancia ficticia que tiene consecuencias paralelos a las drogas ilegales en la vida real (en uso, posesión o venta).
Tres siluetas de personas, una de ellas negra.	Discriminación	Puede contener la violencia o el acoso por motivos de raza, etnia, género o preferencias sexuales.
Dos dados	Apuestas	Puede contener apuestas (con dinero real o ficticio)
Un globo terráqueo conectado a tres computadores.	Online	Contiene un modo de juego online.

Cuadro 2-7 Descriptores de contenido PEGI

Fuente: PEGI http://www.pegi.info

Capítulo 3

Hardware usado en la Industria de los Video-juegos

3.1 INTRODUCCION

Por plataforma se entiende todo aquel hardware capaz de reproducir un video-juego. Históricamente tanto los computadores como las videoconsolas han sido dos de las plataformas más utilizadas para jugar a los video-juegos aunque no se debe olvidar la edad de oro de las máquinas recreativas (años 80) como un período fundamental en el que muchos jóvenes tuvieron su primer contacto con los video-juegos. En la actualidad la versatilidad que ofrece la telefonía móvil augura un papel determinante de esta plataforma.

Cada plataforma posee unas características (e interfaces) que determinan en cierta medida tanto el género como el uso de los video-juegos. No obstante, la convergencia digital nos conduce cada vez más a un espacio online de juego al que se pueda acceder desde diferentes plataformas en función de las necesidades del video-jugador en cada momento. (Martí, 2010)

3.1.1 MÁQUINAS RECREATIVAS

Las máquinas recreativas (conocidas como “maquinitas” en España y buena parte de Latinoamérica), denominadas Arcades o “coin opt machines” en EE.UU., supusieron una auténtica revolución en el ocio juvenil de finales de los años 70 y durante los 80. Títulos emblemáticos como Pacman (1980) o Tetris (1984) se convirtieron en auténticos fenómenos sociales (dando lugar, como el primero, a series de animación y programas de televisión). Como ya se ha comentado al analizar el género de los Arcades, las propias características de esta plataforma condicionaron buena parte de la jugabilidad de los video-juegos desarrollados para correr bajo ella (por ejemplo ritmos rápidos, casi frenéticos, de juego que favorecieran la rotación de los jugadores). La incorporación de un mayor número de jugadores "simultáneos por partida (hasta 4 como en el caso de las Tortugas Ninja Mutantes) fue otra prueba de esta orientación al mercado. (Martí, 2010)

Tras la popularización de la informática doméstica (PCs) y la paulatina adaptación de los títulos arcade a esta plataforma, el mercado de los recreativos llegó prácticamente a desaparecer salvo en centros de ocio especializados ubicados generalmente en centros comerciales (una especialización que llevó a concebir nuevas máquinas más cercanas a los simuladores militares que a máquinas recreativas). (Martí, 2010)

3.1.2 ORDENADOR

La popularización de los computadores personales posibilitó el éxito de géneros como las aventuras gráficas y los video-juegos de rol. En especial gracias a la posibilidad de grabar las partidas en el disco duro del computador se pudieron desarrollar video-juegos con tramas complejas repletos de acertijos y pruebas que el video-jugador debía desentrañar para poder completar un video-juego que podía ocupar durante semanas, meses y años la mente del video-jugador. La posibilidad de interactuar a través del teclado permite que el video-jugador elabore órdenes complejas y acciones tanto con otros personajes del video-juego como con los objetos que va recopilando a lo largo del video-juego. Los años 80 son la época dorada de las aventuras gráficas (o conversacionales) con títulos como Maniac Mansian (Lucas Arts, 1987) y series como King's Quest (Sierra, 1984-1998). Con Populous (Bullfrog, 1989) se inicia la época de los denominados god games (simuladores de mundos virtuales) de los que Sim City (Maxis, 1987) será el gran paradigma. (Martí, 2010)

3.1.3 VIDEOCONSOLAS

Una videoconsola es un pequeño sistema electrónico que está diseñado para ejecutar juegos desarrollados en un computador personal o servidor. Al igual que los computadores personales, pueden adoptar diferentes formas y tamaños; de este modo, pueden ser de sobremesa, es decir, requieren ser conectadas a un televisor para la visualización del video-juego, y a la red eléctrica para su alimentación, en la cual suelen consumir 12 voltios, o bien el dispositivo electrónico videoconsola portátil, que cuenta con una pantalla de visualización integrada y una fuente de alimentación propia (baterías o pilas). (Martí, 2010)

Los video-juegos pueden presentarse en forma de cartuchos de plástico que protegen una placa con chips en los que está almacenado el software, o también en disquete, tarjeta de memorias, disco compactos (como en PlayStation), discos "GOD"

(en el caso de Nintendo GameCube), DVD (como en PlayStation 2, Wii, Xbox y Xbox 360), o Blu-ray (en el caso de la PlayStation 3). Estos dos últimos formatos ópticos de almacenamiento son los que se han impuesto como estándar en las videoconsolas de séptima generación. El formato cartucho se utilizaba básicamente para videoconsolas portátiles o en generaciones pasadas de video-juegos. (Martí, 2010)

Sin duda alguna la videoconsola es la plataforma que más ha evolucionado desde que videoconsolas pioneras como la Odissey de Magnavox (1972) y la Intellivission de Mattel (1979) apostaran por llevar la diversión de los video-juegos a las pantallas de televisión de los hogares norteamericanos. Títulos como Super Mario Brothers (Nintendo, 1985) o The Legend of Zelda (1986) representan un punto de inflexión de una plataforma cuya adaptación constante al mercado ha supuesto desde la aparición de videoconsolas portátiles (como la Gameboy de Nintendo o, ya más recientemente, la PSP de Sony) hasta la última generación de videoconsolas con periféricos cada vez más sorprendentes (como la Wii Fit de Nintendo). En un proceso constante de diferenciación de la plataforma computador, las videoconsolas han ido mutando a entornos prácticamente de realidad artificial (como el Eye Toy de Sony) o a plataformas hibridas que se asemejan a los karaokes (con video-juegos corno Singstar (London Studio, 2004) o Guitar Hero (Harmonix Music Systems, 2005).

Parece que la aparición de una nueva generación de videoconsolas (encabezadas por Nintendo con la Nintendo Wii o con la videoconsola portátil Nintendo DS) está conduciendo a una nueva edad de oro de las videoconsolas. Al menos las cifras en venta así lo auguran ya que durante 2007 el parque de videoconsolas aumentó en Europa un 48% con respecto a 2006 (con España en cuarto lugar a nivel europeo). Asimismo, la venta de video-juegos para videoconsolas en 2007 aumentó un 26% con respecto a 2006 (mientras que los video-juegos para PC disminuyeron un 3%) (ADESE, 2008). El buen momento que parece vivir la plataforma viene también refrendado por otro dato: el 38% de los hogares norteamericanos cuenta con una videoconsola (ESA, 2008).

3.1.4 TELEFONÍA MÓVIL

Los actuales terminales de telefonía móvil permiten a sus usuarios jugar a video-juegos convirtiendo a esta plataforma (gracias a sus características específicas) en una plataforma especialmente interesante desde el punto de vista de los video-juegos publicitarios (advergames). Incluso modelos como el Nokia N-Gage fueron diseñados

específicamente para permitir una mejor experiencia de juego (con una pantalla más grande y teclas especiales para superar las limitaciones del teclado tradicional). El interés de los consumidores por los video-juegos para móviles queda patente por el hecho de que más de una quinta parte de los usuarios de telefonía móvil (22%) se descargan video-juegos para sus móviles (Fundación Orange, 2008). (Martí, 2010)

La posibilidad de descargarse juegos directamente al móvil desde marquesinas interactivas (mediante tecnología bluetooth) o de participar en juegos de rol en escenarios reales, etc., abre nuevas posibilidades a las comunicaciones publicitarias a través de video-juegos para móviles. Uno de los ejemplos más interesantes es la comunidad virtual desarrolla por la empresa surcoreana Amfical Life. Esta comunidad virtual desarrollada para redes móviles 3G permite a los usuarios vivir en la ciudad virtual de su móvil interactuando con otras personas en tiempo real, cooperando con ellas para resolver juegos o pruebas, interactuando con objetos o explorando edificios de la ciudad virtual a través de su móvil. (Martí, 2010)

3.1.5 TELEVISIÓN DIGITAL

La televisión digital también ofrece posibilidades como plataforma de difusión de video-juegos. Las plataformas de pago hace tiempo que ofrecen servicios de video-juegos. Desde un punto de vista publicitario se han desarrollado acciones principalmente de concursos; por ejemplo premio directo mediante publicidad sobrepuesta en la programación o mediante puntos o spots interactivos. El campo de los video-juegos publicitarios (advergames) en la televisión digital está poco explotado y puede convertirse en una de las tendencias de los próximos años debido al creciente interés de productoras y anunciantes por desarrollar contenidos “debranded content” para televisión". (Martí, 2010)

3.2 CONSOLAS

Una videoconsola es un sistema electrónico de entretenimiento para el hogar que ejecuta juegos electrónicos (video-juegos) que están contenidos en cartuchos, discos ópticos, discos magnéticos o tarjetas de memoria. (Martí, 2010)

Los primeros sistemas de videoconsolas fueron diseñados únicamente para jugar video-juegos pero a partir de la sexta generación de videoconsolas han sido incorporadas características importantes como multimedia, internet, tiendas virtuales y servicios en línea.

3.2.1 HISTORIA

En la Industria de los video-juegos, las videoconsolas han sido clasificadas en distintas generaciones. Esta clasificación la determina su tiempo de lanzamiento y la tecnología existente en ese momento. Las empresas fabricantes lanzan una nueva consola en determinado tiempo (que puede variar entre 5 o 6 años). Por otro lado, algunas generaciones están señaladas por un número determinado de bits, los cuales determinan el ancho de bus del procesador, (de la segunda generación hasta la sexta generación). (20minutos.es, 2008) (Indicelatino, 2007)

Las primeras videoconsolas que aparecieron en el mercado llevaban procesador de 8 bits. A partir de la segunda generación algunos fabricantes ya presentaban equipos de 16 bits. A partir de esta cantidad, se fueron realizando las siguientes generaciones de consolas. Una consola de generación superior no tiene que poseer necesariamente un procesador de ancho de bus de datos de más bits, al contrario que la creencia popular que piensa que en cada generación se dobla el número de la anterior, ya que la potencia de un procesador está determinada además de por su ancho de bus por su estructura y velocidad.

En las videoconsolas de reciente generación ya no sólo depende la potencia de la unidad CPU sino también del procesador gráfico GPU que es el procesador encargado del manejo de gráficos en la consola. Cada componente tiene una determinada cantidad de bits y velocidad.

3.2.2 PRIMERA GENERACIÓN 1972 -1977

Si bien los primeros juegos de computadora aparecieron en la década de los 1950, éstos utilizaban pantallas vectoriales, no vídeo analógico. No fue hasta 1972 cuando se lanzó la primera videoconsola de sobremesa por la compañía electrónica Magnavox. La Magnavox Odyssey, fue creada por Ralph Baer, considerado como el padre de los video-juegos. La Odyssey tuvo un moderado éxito, sin embargo, con el lanzamiento del juego arcade Pong de Atari, comenzaron a popularizarse los video-juegos, el público comenzó a mostrar interés ante la nueva industria. En el otoño de 1975, la compañía Magnavox, cede ante la popularidad del Pong, se cancela el proyecto Odyssey, ya que el público sólo jugaba al Pong y Hockey en la Odyssey 100. (Pérez Quintanar, 2004), (20minutos.es, 2008), (Wikipedia, 2012)

Una posterior actualización de la consola Odyssey 100, la 200, llevaba incorporada una pantalla de puntuación, permitía hasta 4 jugadores, y se vendía junto con un tercer juego - Smash. Casi simultáneamente, la cadena de centros comerciales Sears compró los derechos del sistema Atari Pong y lo introdujeron en el mercado de consumo bajo el nombre de Sears-Telegames. Al igual que en el mercado arcade, el mercado pronto fue inundado por consolas clones de Pong y juegos derivados. (Pérez Quintanar, 2004), (20minutos.es, 2008), (Wikipedia, 2012)

La primera generación de videoconsolas empezó en 1972, con el lanzamiento del Magnavox Odyssey, y duró hasta 1977, cuando los fabricantes de consolas tipo "pong" dejaron el mercado en masa para la introducción y el éxito de las consolas basadas en microprocesadores. La idea de una “televisión interactiva” la tuvo el ingeniero e inventor Ralph Baer en 1951, (se le encarga el diseño y creación de equipos de Televisión; una de las ideas que Baer propuso es la creación de sencillos juegos interactivos instalados en el televisor) mientras construía un televisor para Loral Corporation en el barrio del Bronx de Nueva York. Trabajó más profundamente sobre esta idea en 1966, cuando era ingeniero jefe y manager de la División de Diseño de Equipos en Sander Associates, resultando en un video-juego simple para dos jugadores que podía ser mostrado en un televisor estándar. El juego se llamaba Chase y en él dos puntos se movían a lo largo de la pantalla tratando de cazarse el uno al otro. Después de una demostración ante el director de la compañía, Herbert Campman, se consiguieron algunos fondos para el proyecto, que en ese momento se convirtió en “oficial”. En 1967 se llevó ante Bill Harrison, director de Investigación y Desarrollo de la compañía, se construyó un mando en forma de arma de fuego a partir de un rifle de juguete que era dirigido hacia un blanco controlado por otro jugador. (Pérez Quintanar, 2004), (20minutos.es, 2008), (Wikipedia, 2012)

Bill Rusch se unió al proyecto para acelerar el desarrollo y pronto un tercer juego, en esta ocasión de ping pong, fue creado. Con más fondos se crearon más juegos, y Baer tuvo la idea de vender el producto a compañías de Televisión por Cable, que podían transmitir imágenes estáticas como fondos de pantalla para juegos. Un prototipo fue presentado en febrero de 1968 al vicepresidente de Teleprompter Hubert Schlafly, que firmó un acuerdo con Sanders. La industria de la televisión por cable sufría una crisis a finales de los sesenta y principios de los setenta, y la escasez de fondos llevó a explorar otras posibilidades. El desarrollo del hardware y los juegos continuó hasta

resultar finalmente en el prototipo de la “Brown Box”, que tenía dos mandos, un mando en forma de arma de fuego y dieciséis interruptores en la consola que permitían seleccionar distintos juegos. Baer hizo proposiciones a varios manufacturadores de televisores de los Estados Unidos y un acuerdo con Magnavox fue firmado a finales de 1969. El principal cambio realizado a la Brown Box por Magnavox fue la eliminación de los gráficos en color con el fin de reducir costes de manufactura. Fue lanzada en mayo de 1972 como la Magnavox Odyssey. (Pérez Quintanar, 2004), (20minutos.es, 2008), (Wikipedia, 2012)

3.2.2.1 MAGNAVOX ODYSSEY

Comercializada por la filial de Philips en Estados Unidos, la Magnavox Odyssey es la primera videoconsola de la historia. Fue desarrollada por Ralph Baer (apodado “el padre de los video-juegos caseros”) y lanzada en las tiendas norteamericanas a finales del año 1972, convirtiéndose en un éxito de ventas en muy poco tiempo.

Figura 3.1 Magnavox Odyssey

Fuente: http://www.museodelvideojuego.com/wp-content/uploads/2009/06

El prototipo de esta consola, desarrollado en 1968, es conocido como "Brown Box" (Caja Marrón) entre los coleccionistas de video-juegos. La Magnavox Odyssey fue el primer sistema casero de video-juegos. El 27 de enero de 1972, Magnavox comenzó la producción de la máquina, y el sistema fue lanzado en mayo. Se vendieron ese año 100.000 unidades a unos $100 por unidad. El lanzamiento de la Odyssey generó un caso severo de la “locura de Pong”, y compañías por todo el mundo comenzaron a desarrollar sus propias máquinas. La Magnavox Odyssey fue vendida solamente en los almacenes de Magnavox, diciendo además a los clientes que la Odyssey trabajaría

solamente en televisiones de su marca. Una mentira que contribuyó a la cantidad de unidades vendidas. (Wikipedia, 2012)

CARACTERÍSTICAS

- Sus juegos (veintiocho títulos diferentes en total) eran de una sencillez extrema: ping-pong, “tenis de mesa”, voleibol, etc.
- Dado su reducido hardware, carecían de sonido y los jugadores debían memorizar sus puntuaciones.
- En ocasiones eran necesarios algunos dispositivos adicionales para poder ejecutar determinados video-juegos de la plataforma.
- No contenía ninguna unidad central de procesamiento o memoria de acceso aleatorio. La máquina se componía de transistores, resistencias y condensadores.
- La Odyssey utilizaba cartuchos intercambiables para los juegos, traía de origen seis cartuchos de juegos, y un manual de usuario de 36 páginas para los doce juegos ofrecidos por el sistema.

3.2.2.2 PONG

Publicado por Atari, creado por Nolan Bushnell y lanzado el 29 de noviembre de 1972. Pong está basado en el deporte de tenis de mesa (o ping pong). La palabra Pong es una marca registrada por Atari Interactive, mientras que la palabra genérica "pong" es usada para describir el género de video-juegos "bate y bola". La popularidad de Pong dio lugar a una demanda de infracción de patentes y ganada por parte de los fabricantes de Magnavox Odyssey, que poseía un juego similar. (Wikipedia, 2012)

Figura 3.2 PONG de ATARI

Fuente: http://www.museodelvideojuego.com/wp-content/gallery/primera-generacion/

Pong es un juego de deportes en dos dimensiones que simula un tenis de mesa. El jugador controla en el juego una paleta moviéndola verticalmente en la parte izquierda de la pantalla, y puede competir tanto contra un oponente controlado por computadora, como con otro jugador humano que controla una segunda paleta en la parte opuesta. Los jugadores pueden usar las paletas para pegarle a la pelota hacia un lado u otro. El objetivo consiste en que uno de los jugadores consiga más puntos que el oponente al finalizar el juego. Estos puntos se obtienen cuando el jugador adversario falla al devolver la pelota.

Al no tener copyright, porque la circuitería no era reproducible, hubo numerosas máquinas que incluían diversas variaciones del tenis o frontón. Había máquinas que incluían a Pong, o una variante de este, en su colección de juegos. La primera consola que contó con este juego en su colección fue la Atari 2600 que tenía entre sus títulos el juego Pong original. (Wikipedia, 2012)

3.2.2.3 COLECO TELSTAR

Es una consola de video-juegos producida por Coleco. Solo tenía un juego; el Pong. Se empezó a vender a partir de junio de 1976. Originalmente esta consola era un clon de la Atari Pong. Muchas variantes de esta consola fueron lanzadas a la venta en 1978. Después, en 1982, Coleco lanzó la sucesora de esta consola: ColecoVision.

Figura 3.3 Coleco Telstar

Fuente: http://www.old-computers.com/museum/photos/

Hubo cierta polémica por el parecido con la Atari Pong, pero con el tiempo resulto ser muy buena opción ya que entregaba casi la misma jugabilidad a un menor precio lo que la llevó a tener muy buenas ventas. (Wikipedia, 2012)

3.2.3 SEGUNDA GENERACIÓN 1976 -1984

La segunda generación de videoconsolas se inicia a finales de los años 70. Con la llegada del Atari 2600, en 1976 que fue la más popular para la época. Se llamó originalmente Atari VCS y logra un gran éxito, haciendo que la marca Atari fuera sinónimo de video-juegos durante la primera mitad de la década de los 80. El dominio de Atari que venía desde la anterior generación, intentó ser contestado por parte de Colecovision de con el doble de colores y la todopoderosa Intellivision de Mattel que llevaba el primer procesador de 16 bits de la historia de las consolas. Era con diferencia lo más realistas en gráficos y sobre todo sonido que se podía conseguir en las computadoras caseras. Colecovision consiguió unos nada triviales 6 millones de unidades vendidas, e Intelivisión que en su primer año vendió medio millón de unidades se quedó en los 2 millones. No mucho tiempo después sale al mercado Atari 5200 intentando no quedarse rezagada ante Coleco y Mattel, pero la crisis de los video-juegos de principios de los 80 y la llegada de los microordenadores a las casas, hundió las ventas de todos los sistemas, con la única salvedad de Atari 2600 que siguió teniendo cierto tirón hasta la década de 1990. (Pérez Quintanar, 2004) (Wikipedia, 2012)

Otras videoconsolas aparecieron también en esta generación, como la “tv-game 6” de Nintendo, y la SG-1000 de Sega, quienes no fueron muy populares en aquel entonces ya que, el éxito lo tenía Atari, pero tiempo después estas dos (tv-game 6 y SG-1000) tendrían un gran éxito con la llegada de la tercera y cuarta generación. (Wikipedia, 2012)

3.2.3.1 ATARI 2600

La Atari 2600 fue una videoconsola lanzada al mercado en octubre de 1977 bajo el nombre de Atari VCS, convirtiéndose en la primera en tener éxito que utilizaba cartuchos intercambiables. En 1982, tras el lanzamiento de la Atari 5200, adoptó su nombre final basado en el número de catálogo que la identificaba (CX2600). Esta consola fue un gran éxito y logró que durante los años 1980 "Atari" fuese sinónimo de video-juegos. Se vendía acompañada con dos joysticks, un par de controladores tipo paddle y un cartucho de juego. (Wikipedia, 2012)

Figura 3.4 ATARI 2600

Fuente: http://www.feelthebyte.com/wp-content/uploads/2010/06/

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	MOS Technology 6507 a 1,19 MHz	
VIDEO	Procesador de audio y video: TIA.	Resolución de 160 x 190 píxeles, 128 colores en pantalla pero con un máximo de 4 colores por línea
MEMORIA	128 Bytes.	Dentro del chip MOS Technology 6532, algunos juegos pueden añadir hasta 256 octetos integrados en el cartucho
SONIDO	Procesador de audio y video: TIA.	Sonido monoaural de dos vías
SOPORTE	Cartucho.	

Cuadro 3-1 Características técnicas de la consola ATARI 2600

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

ENTRADA: controlada por el chip RIOT incluye:

- Dos puertos DE-9 sin tornillos para conectar joysticks, paddles, trackballs, pedaleras, volantes, teclados numéricos de 12 teclas (0-9, # y *) y cualquier otro tipo de control con funciones adicionales desarrollado por terceras compañías. Este tipo de puerto se convirtió en un estándar de facto para la conexión de joysticks y gamepads, la llamada norma Atari que utilizaron una inmensa

mayoría de equipos de 8 bits en los ochenta, e incluso algunos de 16 bits: Sinclair ZX Spectrum, Amstrad CPC, Commodore 64, MSX, Master System y Mega Drive, entre muchos otros.

- Seis interruptores: Encendido (encendido/apagado), señal de TV (monocromo/color), Nivel de dificultad para cada jugador (A/B), Select y Reset. Exceptuando al interruptor de encendido, los juegos podían (y lo hicieron) asignar otros significados a los demás interruptores. En los modelos que siguieron al original, los interruptores de dificultad fueron reducidos y trasladados a la parte trasera de la consola.

SALIDA: Señal de video y audio para TV a través de un conector RCA (con normas NTSC, PAL o SECAM dependiendo de la región; los cartuchos de juegos pueden ser usados indistintamente entre máquinas NTSC y PAL/SECAM).

3.2.3.2 COLECOVISION

La ColecoVision es una consola de video-juegos de segunda generación lanzada al mercado estadounidense en agosto de 1982 por la empresa Coleco. (Wikipedia, 2012)

Figura 3.5 ColecoVision

Fuente: http://emulators.theoldcomputer.com/images/manufacturers_systems/Coleco/ColecoVision

La ColecoVision ofrecía para su tiempo gráficos y jugabilidad de calidad arcade, la capacidad de jugar con cartuchos de su principal competidora la Atari 2600, y medios para ampliar el hardware del sistema. La ColecoVision se lanza con un catálogo inicial de 12 juegos con otros 10 títulos programados a lo largo de 1982. En total, aproximadamente 100 títulos se lanzaron como cartuchos para la consola entre 1982 y 1984.

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	Zilog corriendo a 3,58 MHz.	Z80A
VIDEO	Texas Instruments.	Resolución de 256x192 pixels, 32 sprites y 16 colores.
MEMORIA	210 bytes.	
SONIDO	Texas Instruments SN76489A.	3 generadores de tonos y 1 generador de ruido, con VRAM: 16 KiB.
SOPORTE	Cartucho.	De 8/16/24/32 KiB

Cuadro 3-2 Características técnicas de la consola ColecoVision

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

SIMILITUDES CON OTRAS PLATAFORMAS

La ColecoVision utiliza la misma CPU y chip de vídeo que los computadores MSX y la Sega SG-1000/SC-3000. Comparte también con las consolas Sega el chip de sonido (incluyendo a la Sega Master System), lo que lo hace igual en capacidades hardware. Los MSX utilizan un chip de sonido diferente, pero con unas capacidades muy similares, el General Instrument AY-3-8910. Debido a ello es muy sencillo portar los juegos entre las tres plataformas.

MÓDULOS DE EXPANSIÓN

Desde su lanzamiento, Coleco ha promocionado la ampliación llamada Expansion Module #1 que permite a la ColecoVision utilizar cartuchos de Atari 2600. Funcionalmente, esto le dio la ColecoVision la mayor biblioteca de software de cualquier consola de su tiempo. Esto trae una demanda de Atari, pero la pierde porque la consola está fabricada con componentes comunes. La decisión del juzgado permite además a Coleco desarrollar su propio clon de la 2600, la Coleco Gemini, que es un clon exacto de la 2600 pero con mandos de joystick/paddle combinados. (Wikipedia, 2012)

Expansion Module #2 es un sistema de conducción que consiste en un volante, pedal de acelerador y el juego incluido Turbo. Este control puede utilizarse además con los juegos Destructor y Dukes Of Hazzard.

Expansion Module #3, la ampliación definitiva, se lanza en el verano de 1983. El Module #3 convierte a la ColecoVision en un completo computador, conocido como el Coleco Adam, que incluye un teclado y unidades digital data pack (DDP, una casete en una carcasa especial que graba a alta velocidad). El Module #3 se concibe originalmente para ser el supermódulo de juegos de la ColecoVision (Super Game Module) que utilizaría game wafers como sistema de almacenamiento. Aunque Coleco presenta una maqueta del SGM en el New York Toy Show de 1983, nunca fue fabricado. Hubo también rumores de que incorporaría una lectora CED de RCA como sistema de almacenamiento masivo de datos.

Coleco crea un prototipo de una cuarta expansión que le permitiría ejecutar cartuchos de la consola Intellivision, pero nunca se lanzó.

Existieron dos controladores de juegos alternativos fabricados por Coleco. El primero fue el Roller Controller, una trackball con un port del juego arcade Slither, un clon de Centipede. El segundo fue el Super Action Controller Set, que recuerdan a un par de guantes de boxeo con joystick de bola, keypad y una rueda con funciones de paddle en la zona superior, y cuatro botones de fuego (uno por dedo) en la empuñadura. Se vendía con el juego Super Action Baseball y posteriormente con loas juegos Super Action Boxing (inspirado en Rocky) y una conversión del arcade Front Line. (Wikipedia, 2012)

3.2.3.3 INTELLIVISION

Es una consola de video-juegos lanzada por Mattel en 1979. El desarrollo de la máquina comenzó en 1978, a menos de un año de la introducción de su principal competidor, el Atari 2600. La palabra “intellivision” es una unión de “intelligent” y “televisión” (televisión inteligente).

Figura 3.6 Intellivision

Fuente: http://www.old-computers.com/museum/photos/

La Intellivision fue desarrollada por Mattel Electronics, una subsidiaria formada expresamente para el desarrollo de juegos electrónicos. La consola fue probada en Fresno, California, en 1979 con un total de cuatro juegos disponibles, y se lanzó en todo Estados Unidos en 1980 con un coste de 299 dólares y un juego de regalo: Las Vegas Poker & Blackjack. A pesar de no ser el primer sistema que se enfrentó a Atari (sistemas hechos por Fairchild Semiconductor, Bally, y Magnavox ya estaban en el mercado), fue la primera consola que significó una seria amenaza al dominio de Atari.

Intellivision fue la primera consola de 16 bits, la primera en usar las redes, a través de la línea telefónica se podían descargar juegos y Intellivision World Series Baseball, lanzado en 1983, fue el primer juego en usar el concepto de mostrar la acción en tres dimensiones a través de "ángulos de cámara", que emulaban los utilizados en las transmisiones de TV. Juegos anteriores siempre usaban una cámara fija o una vista movediza del campo de juego. (Wikipedia, 2012)

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	Microprocesador General Instrument CP1610 a 894.886 kHz, poco menos de 1 MHz	16 bits
VIDEO	General Instruments AY-3-8900 (STIC Standard Television Interface Chip)	Resolución: 160 píxeles de ancho por 196 píxeles de alto, paleta de 16 colores,8 sprites
MEMORIA	1352 bytes.	7168 bytes de ROM
SONIDO	Chip de sonido GIAY-3-8914 con 3 canales.	Cada canal de 8 octavas de sonido más uno de ruido blanco.
SOPORTE	Cartucho.	

Cuadro 3-3 Características técnicas de la consola Intellivision

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

Controladores: Teclado numérico de doce botones (0-9, Clear y Enter), 4 botones de acción localizados a los lados (de los cuales dos eran electrónicamente lo mismo, dando como resultado tres botones de acción diferentes), "disco direccional", capaz de detectar movimiento en 16 direcciones y "overlays", las cuales se ponían encima del teclado para mostrar las funciones específicas de cada juego.

3.2.3.4 ATARI 5200

La Atari 5200 es una consola de video-juegos fabricada por Atari y lanzada al mercado en 1982. No tuvo tanto éxito como su predecesora, la Atari 2600, pero se trataba de una máquina maravillosa para ese momento. El chip de gráficos es mejorado, hasta el punto que es prácticamente igual que los Arcades de la época. (Wikipedia, 2012)

Figura 3.7 Atari 5200

Fuente: http://consollection.de/content/2.consoles/1270.5200

Fue lanzada al mercado en 1982, y fue la sucesora de la Atari 2600. La Atari 5200 ofrecía gráficas mejoradas y varias características que no podían encontrarse en ninguna otra videoconsola de su tiempo, de hecho, si los controladores no hubiesen sido tan frágiles, habría competido en el mercado más tiempo, ya que solo se fabricó desde el año 1982 al 1984, si bien en el 1983 ya estaban preparado la salida de la 7800, por la razón de las quejas que surgieron de todos los que compraron la maquina; no era retro compatible, los mandos no siguen la norma Atari, y al tener potenciómetro y ser de goma los botones, fallaban horriblemente. (Wikipedia, 2012)

Cuando fue lanzada, Atari era una verdadera consola de video-juegos. Después de todo, se trataba de una computadora con 16k en RAM diseñada específicamente para juegos de alta calidad. El corazón del sistema era en esencia un computador Atari 400, el computador de 8 bits más poderoso para el hogar disponible en el momento, y cuyos juegos teóricamente podían portarse rápida y fácilmente entre las dos máquinas.

Aunque el Atari 5200 disfrutó de cierto éxito en sus comienzos, el público nunca llegó a entusiasmarse, y la "Gran Caída De los Video-juegos de 1983" Ayudó a sellar su destino al igual que el del resto de los sistemas de video-juegos caseros. La librería de juegos ronda los 70 títulos. Cabe anotar que en el momento en que fue retirado del mercado, ya había salido el adaptador “cx55” que permitía jugar a los juegos del 2600 en la consola de dos puertos y en las de 4 con modificaciones y que cuando atari la retiro, el Atari 5200 superaba en ventas al Colecovision, en 1984. Cabe anotar que se hizo extremadamente popular puesto que sale en la película Cloak & Dagger, donde el protagonista juega con la 5200. En Europa no llegó a venderse, si bien, el atari 400, 800,

800xl, 600 xl y series xe tienen los mismos gráficos y juegos, ya que son la misma máquina. (Wikipedia, 2012)

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	MOS Technologies 6502C a 1.78 MHz	8 bits
VIDEO	Colores: 256, 16 simultáneamente.	Resolución: 320x192.
MEMORIA	16 Kb.	
SONIDO	4 canales.	
SOPORTE	Cartucho.	

Cuadro 3-4 Características técnicas de la consola Atari 5200

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

CONTROLES

Los controles tienen un pequeño teclado numérico, y dos botones, uno a cada lado. Utiliza un sistema de control analógico, lo que le da un rango de movilidad de 360 grados en contraste con las 4 u 8 posiciones que las demás consolas ofrecían normalmente. La palanca es sensible a la presión, de forma que entre más fuerte se presione hacia la dirección deseada, más rápido se mueve el objeto en pantalla. En el control también podemos encontrar por primera vez en la historia un botón de pausa, para detener temporalmente el juego. A pesar de lo revolucionario de los controles, tenían graves problemas. Por ejemplo, las palancas no se auto-centraban, lo cual hacía muy difícil controlar algunos juegos. Además, estos controles son famosos por estar entre los menos confiables jamás fabricados.

3.2.3.5 SG-1000

SG-1000, fue la primera videoconsola de sobremesa desarrollada por Sega. Fue primero lanzada al mercado en una primera fase de pruebas en Japón en 1981 y finalmente lanzada al mercado japonés en Julio de 1983 por 15.000¥. La consola no consiguió gran éxito en ese mercado. Sin embargo, el sistema se vendió bien en Asia hasta 1985. Fue también comercializada en Australia por John Sands, en Nueva Zelanda

por Grandstand Leisure Limited, y en otros países como Italia, España, y Sudáfrica. La consola en su versión original nunca llegó a Norteamérica. (Wikipedia, 2012)

Figura 3.8 SEGA SG-1000

Fuente: http://blogs.gamefilia.com/files/imce/u657725/

En Julio de 1984, Sega lanzó una versión actualizada de la consola llamada SG-1000 II, que contaba con un conector para una ampliación de teclado opcional. Una versión en forma de computador de esta consola con el teclado integrado fue comercializada bajo el nombre de SC-3000. (Wikipedia, 2012)

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	NEC D780C A 3.5MHz	Clon del Zilog Z80
VIDEO	128 Kbits (16KB)	256x192 (16 colores)
MEMORIA	16 Kbits (2KB)	
SONIDO	Texas Instruments SN76489, 4 canales de sonido monoaural.	3 generadores de sonido, de 4 octavas cada uno, y un generador de ruido blanco
SOPORTE	Cartucho.	

Cuadro 3-5 Características técnicas de la consola SEGA SG-1000

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

3.2.4 TERCERA GENERACIÓN 1983 -1992

Después de que América del Norte se sumiera en el Crash dando como resultado la Crisis del video-juego de 1983, muchas industrias americanas tuvieron que dejar la producción de consolas, este momento lo aprovechó la compañía japonesa Nintendo para hacer lo que sería su jugada maestra: lanzar la Famicom de Nintendo al mercado en 1984, que poco después de su rediseño fue conocida mundialmente como Nintendo Entertainment System. (Pérez Quintanar, 2004) (Wikipedia, 2012)

Dos años más tarde la compañía SEGA lanzaría su primera videoconsola en territorio japonés a la que llamaron Sega Mark III, esta videoconsola se comercializó mundialmente con el nombre de Sega Master System. (Pérez Quintanar, 2004) (Wikipedia, 2012)

Muchas personas creen que esta es la primera generación de consolas, sin embargo las consolas de la generación anterior también usaban 8 bits, no fue hasta el final de esta generación cuando la gente empezó a etiquetar las generaciones con nomenclaturas de bits, en parte esto se debió a que la gente quería diferenciar las consolas de 8 bits y las de 16 bits. Esta generación se conoció como la "era de los 8 bits". (Pérez Quintanar, 2004) (Wikipedia, 2012)

3.2.4.1 NINTENDO ENTERTAINMENT SYSTEM (NES)

Nintendo Entertainment System (conocida también como NES o Nintendo NES) fue una videoconsola de ocho bits perteneciente a la tercera generación en la industria de los video-juegos. Fue lanzada por Nintendo en Norteamérica, Europa y Australia entre 1985 y 1987. En la mayor parte del continente asiático, incluyendo a Japón (donde se comercializó por primera vez en 1983), China, Vietnam, Singapur y Filipinas se la conoció con el nombre de Family Computer abreviado comúnmente como Famicom o simplemente FC). En Corea del Sur se llamó Hyundai Comboy y fue distribuida por Hyundai Electronics, mientras que en regiones como Rusia y el sur de Asia pasó a denominarse Dendy y Tata Famicom, respectivamente. En 1990, la Super Nintendo reemplazó a Nintendo NES en el mercado. (Wikipedia, 2012)

Figura 3.9 Nintendo Entertainment System – NES

Fuente: http://www.serviwebrd.com/rdgamers/images/stories/virtuemart/product/nes-console2.jpg

Está considerada como la videoconsola más exitosa de su época y contribuyó a revitalizar de forma significativa la industria estadounidense de los video-juegos, la cual había sufrido previamente una debacle financiera que ocasionó que varias empresas especializadas quebraranademás de establecer el nivel estándar para consolas posteriores en aspectos primordiales como el diseño de cada juego y planteamiento de mandos. Asimismo, a partir de esta consola Nintendo estableció un modelo de negocios estandarizado en la era contemporánea y referente a la licencia de software para desarrolladores tipo third-party. (Wikipedia, 2012)

LICENCIAS A THIRD-PARTIES

El monopolio que Nintendo alcanzó en el mercado de los video-juegos domésticos le brindó un grado significativo de influencia sobre la industria que llegó a exceder incluso al alcanzado por Atari en su época más exitosa (a principios de los años 1980). No obstante, a diferencia de Atari, la cual nunca apoyó activamente a desarrolladores third-party (recurriendo inclusive a medios legales para obligar a Activision que cesara la producción de juegos para el sistema Atari 2600), Nintendo concibió y sustentó el involucramiento con este tipo desarrolladores bajo estrictos términos. Así, podía instalarse un circuito 10NES en cada consola, y otro adicional en cada cartucho con licencia oficial. Si el chip de la consola no podía detectar a su equivalente en el cartucho, el juego no se iniciaría. Debido a la producción controlada de sus cartuchos, Nintendo era capaz de hacer valer estrictas reglas impuestas a sus desarrolladores third-party. Estos últimos debían firmar asimismo un contrato hecho por Nintendo con el que se comprometían a desarrollar exclusivamente para el sistema.

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	Ricoh 1,79 MHz/1,66 MHz	8 bits, basado en MOS Technology Motorola 6502.
VIDEO	Ricoh 5,37 MHz/5,32 MHz de 2Kb.	Gráficos de 8bits, salida RGB, Paleta: 48 colores y cinco grises, 52 colores en pantalla, 64 Sprites en pantalla de 8x8 o 8x16 pixeles; resolución: 256x240 píxeles.
MEMORIA	2 KB.	con opción de utilizar una expansión si estaba presente en el cartucho
SONIDO	5 canales.	4 generadores de tonos (dos cuadrados, un triángulo, un ruido)
SOPORTE	Cartuchos	60/71 pines.

Cuadro 3-6 Características técnicas de la consola NES

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

3.2.4.2 SEGA MASTER SYSTEM (SMS)

Comercializada en Japón bajo el nombre SEGA Mark III, es una consola de video-juegos de 8 bit basada en cartuchos, que fue producida por Sega para competir con la Nintendo Entertainment System. Aunque estuvo muy por detrás en ventas fuera de Europa y Brasil, la experiencia sentó los cimientos para que Sega continuara con su liderazgo en esos mercados durante la siguiente generación con la Megadrive. (Wikipedia, 2012)

Figura 3.10 Sega Master System

Fuente: http://www.old-computers.com/museum/photos/

El sistema fue rediseñado para venderlo en el mercado de Estados Unidos con el nombre de Sega Master System en junio de 1986, un año después del lanzamiento de la Nintendo Entertainment System. (Wikipedia, 2012)

Aunque más avanzada técnicamente que la Nintendo Entertainment System, no alcanza en EE.UU. el mismo nivel de popularidad. Esto se ha atribuido a varias causas, entre ellas el retraso en el lanzamiento y un número de cartuchos de otras compañías distintas a Sega mucho menor. Esto último debido al contrato que Nintendo obligaba a suscribir a quienes querían licenciar juegos para la NES, con cláusulas de exclusividad o retrasos de varios años entre la aparición de un juego en NES y en otras plataformas. (Wikipedia, 2012)

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	Zilog Z80 3,57MHz/3,54 MHz	8 bits, (NEC D780C, Zilog Z80A o Zilog Z084004)
VIDEO	16 Kb	Custom chip con núcleo Texas Instruments TMS9918, 64 colores, 32 simultáneos, 64 sprites de 8x8, 8x16 o 16x16 por hardware, resolución 248 x 192 y 256 x 192 (modo de las SG1000 /SC3000).
MEMORIA	8 Kb	
SONIDO	Texas Instruments SN76489, 4/9 canales.	3 Sonido de onda cuadrada y 1 de ruido blanco. En los modelos japoneses, un Yamaha YM2413 FM con 9 canales de sonido sintetizado 15 instrumentos + 1 definible por el usuario con control de vibración y modulación de amplitud.
SOPORTE	Cartucho	44 pines Sega 1000, de 50 pines en el resto.

Cuadro 3-7 Características técnicas de la consola Sega Master System

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

3.2.4.3 ATARI 7800

La Atari 7800 (en USA incluye ProSystem en el nombre, lo que significa Profesional System) fue la sucesora de la consola Atari 5200 o supersystem, en un intento de Atari de recuperar su supremacía en el mercado de la videoconsola frente a sus rivales locales Intellivision y Colecovision, y al desembarco japonés de la Sega Master System y Nintendo NES. Fue desarrollada durante el año 1983, y es lo que se supone que debería haber sido la 5200, un sistema retro compatible con la consola Atari 2600, con un chip gráfico mejorado. En todos los aspectos es la mejor máquina de las consolas de la serie mil de atari, si bien el chip de sonido de la 5200 es el pokey (incluido en la serie 8 bits), no es superior al de la 7800, es distinto, creando sonido en varios canales, siendo una mejora del chip de la del 2600, si bien esto posteriormente fue revisado, incluyendo el chip Pokey directamente en algunos cartuchos de juego, que permitiera mayores polifonías. Cabe destacar que el hecho de ser retro compatible, hace que los juegos de la atari 2600 se vean a través del chip Maria, y no del Stella como original de la Atari 2600, esto es perceptible porque los juegos pierden brillo, contrastes y colores, pues el chip retro compatible, hace que se "filtren" los juegos en modo retro compatible. (Wikipedia, 2012)

Figura 3.11 ATARI 7800

Fuente: http://upload.wikimedia.org/wikipedia/commons/8/86/

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	Sally, un MOS 6502C de 1.79 Mhz.	8 bits, cae a 1.19 MHz cuando el chip TIA o el RIOT son accedidos.
VIDEO	Chip Stella para el modo 2600 y el chip MARIA (GCC1702) a 7.16 Mhz.	Acceso DMA, lo que libera el 90% de la CPU, Modo gráfico de 320 x 200, 16 colores de 256 y 160 x 102, 16 colores de 128 (modo 2600).
MEMORIA	4Kb.	4Kb en ROM.
SONIDO	Controlado por el Television Interface Adapter	
SOPORTE	Cartucho.	32 pines, con retro compatibilidad con los cartuchos de la Atari 2600.

Cuadro 3-8 Características técnicas de la consola ATARI 7800

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

3.2.5 CUARTA GENERACIÓN 1988 -1996

La cuarta generación de consolas, comúnmente conocida como la "era de los 16 bits" empezó el 30 de octubre de 1987 cuando la compañía japonesa Nippon Electric Company lanzó al mercado el computador personal o PC Engine, conocida como TurboGrafx-16 en América del Norte. (Wikipedia, 2012)

Aunque NEC se adelantó en esta generación. Las consolas que dominaron fueron Super Nintendo ("Super Famicom" en Japón) y Sega Mega Drive. En esta era los personajes de los juegos que habían impactado al público, se hicieron aún más populares, como Mario Bros. y Sonic the Hedgehog. (Wikipedia, 2012)

Otras compañías también lanzaron consolas al mercado pero a excepción de la consola Neo-Geo las demás compañías no tuvieron éxito. Esta generación vivió un periodo de intensa guerra comercial (principalmente entre SEGA y Nintendo) al cual se denomina comúnmente también como "la época dorada de los video-juegos". (Wikipedia, 2012)

3.2.5.1 SEGA MEGA DRIVE

Es una videoconsola de sobremesa producida por la empresa SEGA, lanzada al mercado en 1988. Esta videoconsola es la sucesora directa de la Sega Master System y compitió contra la SNES de Nintendo, como parte de las videoconsolas de cuarta generación. En Europa la consola llegó en 1990.

Figura 3.12 Sega Mega Drive o Sega Genesis

Fuente: http://jscustom.theoldcomputer.com/images/manufacturers_systems/Sega/Megadrive-Genesis/

En Estados Unidos, México y algunos países de Sudamérica, se comercializó bajo el nombre Sega Genesis y tuvo un amplio mercado. La primera versión de la Mega Drive vio la luz en Japón en octubre de 1988. Fue la sucesora de la Sega Master System y la primera videoconsola de 16 bits reales, siendo la consola más reconocida de Sega. (Wikipedia, 2012)

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	Motorola 68000 7.67MHz/7.61 MHz	16 bits, incluye chip Zilog Z80 de 8 bits para la retro compatibilidad con MasterSystem
VIDEO	VDP (Video Display Processor).	64Kb de memoria, Resolución: 320 x 224, Paleta: 512, a usarse 64 en pantalla, Sprites simultáneos 80/64
MEMORIA	64Kb.	ROM de arranque de 2Kb.
SONIDO	Chip FM Yamaha YM2612	8Kb de Ram, 6 canales FM y 4 canales adicionales con el PSG (Programmable Sound Generator) Texas Instruments SN76489
SOPORTE	Cartuchos.	32 Mbits (4 MB), con excepciones como Super Street Fighter II con 40 Mbits (5 MB), y Pier Solar 64 Mbits (8 MB).

Cuadro 3-9 Características técnicas de la consola Sega Mega Drive

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

PERIFÉRICOS

El Master System Converter era un periférico para Mega Drive que permitía una compatibilidad casi completa con los juegos diseñados para Master System. No obstante, destaca el hecho oculto entonces de que Mega Drive ya contaba internamente de serie con toda la circuitería que la hacía compatible con Master System. El periférico, además de ofrecer una ranura para cartuchos Master System y otra para tarjetas de memoria de la misma consola, lo único que hacía era activar un sensor eléctrico dentro de Mega Drive que activaba la CPU de Master System dentro de la propia Mega Drive. (Wikipedia, 2012)

La Menacer era la pistola que creó Sega para la Mega Drive. Constaba de una mira y una culata desmontables que permitían usarla como pistola o como escopeta, funcionaba de forma inalámbrica vía rayos infrarrojos y se vendía junto con un cartucho de 6 juegos de tiro al blanco.

El Arcade Power Stick de Sega era un joystick de gran tamaño que simulaba a las palancas de mando de las máquinas recreativas. Hubo versiones de 3 y 6 botones con selectores individuales de disparo.auto fuego.

El Sega Multi Tap era un dispositivo que permitía conectar hasta 4 mandos a uno de los puertos de la Mega Drive y utilizarlos simultáneamente en juegos de hasta 4 jugadores, o bien de modo individual activando el mando que quisiéramos en cada caso.

Otros periféricos fueron el Mega Mouse, el Mega Net Modem (el cual permitía jugar online) y un curioso sistema de captura de movimiento llamado Sega Activator Ring que reproducía los movimientos del jugador en el juego.

Además, Mega Drive contó con dos importantes ampliaciones que fueron **Sega Mega-CD y Sega 32X**, siendo la primera un soporte para juegos en formato CD-ROM y la segunda una extensión de 32 bits que permitió a la Mega Drive dar el salto a los juegos en 3D. Ambas funcionaban en conjunción con la Mega Drive, e incluso hubo unos pocos juegos de MegaCD que requerían la 32X. (Wikipedia, 2012)

3.2.5.2 SUPER NINTENDO ENTERTAINMENT SYSTEM (SNES)

Super Nintendo Entertainment System, Super Nintendo, Super NES o SNES (conocida como Super Famicom en Japón) fue la segunda videoconsola de sobremesa de Nintendo y la sucesora de la Nintendo Entertainment System (NES) en América y Europa. Mantuvo una gran rivalidad en todo el mundo con la Sega Mega Drive (o Sega Genesis) durante la era de 16 bits. (Wikipedia, 2012)

Figura 3.13 Super Nintendo Entertainment System o SNES

Fuente: http://watchaplayin.files.wordpress.com/2011/01/

Para la crítica y el público, es una de las mejores y de las más importantes que han aparecido junto con la NES y la PlayStation.

El diseño de la Super Nintendo / Super Famicom fue inusual en su época. Contaba con un procesador relativamente poco potente, ayudado por potentes chips propios para el procesamiento de sonido y video, también llamados chips de apoyo. Esto es común en el hardware de video-juegos de hoy en día, pero fue nuevo para los desarrolladores de juegos, y como primer resultado los juegos de terceras compañías tuvieron poca calidad técnicamente hablando. Los desarrolladores se acostumbraron a su sistema más adelante, así que podían utilizarlo a su máxima capacidad.

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	Procesador WDC W65C816 a 1.79, 2.68 MHz, o 3.58 MHz (variable)	16 Bits.
VIDEO	GPU DSP de 16 bits y MD-7 para efectos 3D.	Paleta: 32,768 Colores con 4096 seleccionables en pantalla según el modo gráfico: ejemplo, 241 en mode 1 o 256 en mode 7, sin contar sub-blending, RAM de texturas y mapas: 128 KB, Resolución: 256x224 a 512x448; 128 sprites en pantalla.
MEMORIA	128Kb.	
SONIDO	Sony SPC700 de 8 bits corriendo a 4.1 MHz, S-SMP de 8 canales (PCM esterio) y S-DSP de 3 canales.	Chip 2A03 de NES 64 KB, descompresión similar al ADPCM, Ram 512Kb.
SOPORTE	Cartucho.	De 2 - 64 Mb.

Cuadro 3-10 Características técnicas de la consola SNES

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

Puerto EXT

La parte inferior de la consola cuenta con una conexión denominada "EXT". En Japón se publicó un periférico llamado Satellaview que utilizaba este puerto de expansión. Se trata de un módem por satélite con el que los jugadores podían competir en línea con otros usuarios gracias a la frecuencia proporcionada por las emisiones de radio por satélite de St.GIGA.

ACCESORIOS

- Un conversor de Game Boy llamado Super Game Boy fue lanzado, para la gente que poseía ambos sistemas, permitía además darle color a los juegos.
- Super Scope una pistola de luz inalámbrica semejante a un bazooka.
- Un ratón de SNES el cual fue creado específicamente para el juego de creatividad llamado Mario Paint, aunque también podía ser usado en el Jurassic Park de Ocean y otros juegos.
- Super Multitap – un adaptador para multi-jugador licenciado por Hudson, similar al accesorio de NES' NES Four Score y NES Satellite. Expande los puertos de controladores de la SNES hasta el número de 16 jugadores simultáneos para aquellos juegos que lo soportasen (no obstante, este gran número requiere que varios Multitaps sean conectados unos a otros). Hubo también un “Super Multitap 2”, pero no funcionaba en consolas PAL y NTSC (EE. UU.).
- Un adaptador para los juegos de EE. UU. (NTSC) para que funcionen en la versión PAL.
- Sufami Turbo - un catridge-adaptador creado por Bandai, que sirve para ejecutar juegos de tamaño Gameboy creados para este adaptador en tierras niponas.
- Satellaview - un satmódem que se añadía a la consola de Nintendo Super Nintendo que fue lanzado únicamente en Japón.

DISPOSITIVOS DE TRUCOS

Dispositivos de trucos de terceros lanzados para la SNES permitían a los jugadores modificar los datos del juego y permitían cosas como vidas infinitas,

energía, etc. Todos los dispositivos de trucos fueron hechos por compañías de terceros y ninguno fue licenciado o apoyado por Nintendo.

- Pro Action Replay 2
- Game Genie

CHIPS DE APOYO

Algunos juegos hicieron usos de chips especiales en el interior de los cartuchos con el fin de aumentar la potencia de la máquina en diferentes aspectos:

- C4: Mejorar el uso de las transparencias y cálculos trigonométricos. Exclusivo en Megaman X2 y X3.
- DSP1: Mejoras en el Modo 7 y cálculo de vectores. Utilizado en Super Mario Kart.
- DSP2: Escalado de gráficos. Aumenta la velocidad de la SNES desde 3.58 a 8Mhz. Utilizado en Dungeon Master.
- DSP3: Descompresión de gráficos. Utilizado en SD Gundam GX
- DSP4: Dibujado de circuitos. Utilizado en Top Gear 3000.
- SA1: Compresor de datos 1:4. Aumenta velocidad de la SNES hasta 10Mhz. Utilizado en Super Mario RPG.
- S-DD1: Compresión de gráficos. Utilizado en Street Fighter Alpha 2 y Exclusivo de Star Ocean.
- SPC7110: Mejora en algoritmos. Usado en 4 juegos:
- Far East of Eden Zero
- Far East of Eden Zero - Shounen Jump no Shou
- Momotarou Dentetsu Happy
- Super Power League 4
- Super FX: Cálculos vectoriales. Aumento velocidad sNES hasta 10.5Mhz. Utilizado en Star Wing (Star Fox).
- Super FX 2: Aumento velocidad SNES hasta 21Mhz. Utilizado en Super Mario World 2:Yoshi Island.

- Seta DSP: Mejoras de sonido. Dibujado de circuitos. Utilizado en F1 Roc 2 (ST010) y Hayazashi Nidan Morita Shougi (ST011 la primera versión y ST018 (o Seta RISC) la segunda)
- OBC-1: Utilizado en Metal Combat
- S-RTC: Utilizado en Daikaijuu Monogatari II

3.2.5.3 PC ENGINE O TURBOGRAFX-16

La PC Engine es una videoconsola que fue lanzada primeramente en Japón por NEC el 30 de octubre de 1987. El sistema fue lanzado en el mercado norteamericano a finales de agosto de 1989 con el nombre de TurboGrafx-16. También fue lanzado en el continente europeo en 1990 pero era muy difícil ver la consola ya que fue lanzado con pocas unidades en el mercado europeo. Salió con el nombre de "Turbografx" (sin incluir el "16" en el nombre que llevaba la americana y la "g" de "grafx" se convirtió a minúscula). (Wikipedia, 2012)

Figura 3.14 PC ENGINE O TURBOGRAFX-16

Fuente: http://upload.wikimedia.org/wikipedia/commons/5/5a/

CARACTERSITICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	MOS Technology 65C02 a 7Mhz.	HuC6280A de 8 bits, un 65SC02 modificado, a 1.79 ó 7.16 MHz (conmutable por software).
VIDEO	Disposición de Procesador Gráfico Dual.	16 bits; codificador de color de vídeo HuC6260 de 16 bits (VCE), y un Controlador de Despliegue de Vídeo (VDC) de 16 bits HuC6270A. Comprendía I/O (Entrada/Salida) Basado en Puerto (Port-based) similar a la familia VDP de TMS99xx; resolución máxima de 512 x 242, Colores 512, Máximo de 512 (256 para los fondos, 256 para sprites); profundidad de 9 bits y 64 sprites simultáneos. RAM Video 64Kb.
MEMORIA	8Kb	
SONIDO	6 canales "PSG", programables.	CPU HuC6280A cada canal tenía una frecuencia de 111,87 kHz
SOPORTE	Hu-Cards y CD-ROMs	1ra en tener un módulo de CD-ROM como expansión

Cuadro 3-11 Características técnicas de la consola TURBOGRAFX-16

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

La PC Engine fue un trabajo correalizado entre el creador de software japonés Hudson Soft (el cual mantiene una división para la creación de chips) y NEC. Hudson estaba buscando ayuda financiera para una consola que ellos habían diseñado y NEC estaba buscando como entrar en el lucrativo mercado de los video-juegos. La PC Engine fue un consola muy pequeña debido principalmente a una arquitectura eficiente de tres chips y a las HuCards, que eran unos cartuchos del tamaño de una tarjeta de crédito, pero algo más anchas.

Fue la primera consola que tenía la opción de un añadido de CD. La Turbografx fue un sistema de 8 bits con un chip gráfico de 16 bits.

3.2.5.4 NEO-GEO

Neo-Geo es el nombre de un sistema de 16 bits basado en cartuchos para Arcades así como videoconsolas para el hogar lanzado en 1990 por la compañía de video-juegos japonesa SNK (actualmente SNK Playmore). La tecnología del sistema ofrecía unos gráficos 2D, y una calidad de sonido muy superiores a la que ofrecían otros sistemas caseros de su época, En un principio el sistema Neo-Geo se creó como plataforma para máquinas recreativas, más tarde también estaba disponible como videoconsola doméstica a un precio demasiado elevado para muchos, dando lugar así a dos versiones del sistema: el sistema arcade, MVS, siglas de Multi Video System (Sistema Multi Video), y el sistema doméstico, AES, siglas de Advanced Entertainment System (Sistema Avanzado de Entretenimiento). (Wikipedia, 2012)

Figura 3.15 Neo-Geo

Fuente: http://www.infoconsolas.com/wp-content/uploads/

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	Motorola 68000 a 12 MHz	Coprocesador: Zilog Z80 a 4 MHz
VIDEO	68 Kb en video.	Resolución: 320x224, Paleta de colores: 65.536, simultáneos en pantalla 4.096, Sprites simultáneos 380 y 3 planos de scroll simultáneos.
MEMORIA	64Kb y 2Kb para el Z80.	Tarjetas de memoria: 8kb externas.
SONIDO	Yamaha YM2610 15 Canales de sonido	7 digitales, 4 de síntesis de FM, 3 PSG, 1 canal de ruido.
SOPORTE	Cartuchos	

Cuadro 3-12 Características técnicas de la consola Neo-Geo

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

OTROS SISTEMAS NEO-GEO

Varias consolas fueron creadas basándose en el mismo hardware que el empleado por los juegos arcade, además de dos portátiles bajo el nombre de Neo Geo Pocket.

Sistemas que salieron al mercado: Neo Geo CD, Neo Geo CDZ, Neo Geo Pocket, Neo Geo Pocket Color, Neo Geo Portable Device.

Así mismo en el año 1997, se creó una nueva placa arcade que sería la sucesora de la MVS, que tenía capacidad para mostrar gráficos en 3D, denominada *Hyper Neo-Geo 64, aunque no tuvo mucha aceptación por el público

3.2.6 QUINTA GENERACIÓN 1993 -2002

La quinta generación de consolas, más conocida como la "era de los 32 bits" aunque ocasionalmente algunas personas se refieren a esta generación como la "era de los 64 bits" puesto que Nintendo lanzaría dos años más tarde un sistema que rompería ese apodo, se trata de la consola Nintendo 64; raramente se le llama también la "era 3D". (Wikipedia, 2012)

Se trata de una generación que supuso el paso de los 2D a los entornos tridimensionales 3D y comenzó en el año 1994 cuando SEGA lanzó su Sega Saturn y Sony su PlayStation, la cual supuso el debut de esta compañía en el mundo de los video-juegos. (Wikipedia, 2012)

Básicamente el mercado estaba dominado por tres consolas, Sega Saturn (1994), PlayStation (1994) y Nintendo 64 (1996). La demografía en las ventas de consolas varió considerablemente, pero estas consolas definieron la guerra de consolas de esta era. La 3DO Interactive Multiplayer, FM Towns Marty, NEC PC-FX, Apple Pippin y la Atari Jaguar fueron también parte de esta era, pero su marketing fue pobre y fallaron a la hora de crear impacto. (Wikipedia, 2012)

3.2.6.1 SEGA SATURN

Es la sexta consola de sobremesa producida por Sega (de ahí su nombre, ya que Saturno es el sexto planeta del Sistema Solar), fue desarrollada para suceder a la Genesis/Mega Drive y competir contra las 3DO de Panasonic, Goldstar (actual LG), etc.; poco más adelante con la PlayStation de Sony y más adelante con la Nintendo 64. Fue lanzada al mercado el 22 de noviembre de 1994 en Japón, y en mayo de 1995 en América. (Wikipedia, 2012)

Figura 3.16 SEGA Saturn

Fuente: http://4.bp.blogspot.com/-cw4YU30z6PQ/TbQueSMAD3I/AAAAAAAABD4/1VYKD4T40NY/s1600/

Sega Saturn llegó a obtener un valioso segundo puesto en la lucha por el mercado de los video-juegos, pero fue perdiendo campo con la aparición de la PlayStation que tenía un amplio catálogo de juegos y era mucho más fácil de programar, dado que la Sega Saturn, al tener dos procesadores de 32 bits en paralelo (Hitachi SH2)

y un tercero de apoyo (Hitachi SH1) añadía una dificultad muy fuerte a los programadores no habituados a este tipo de arquitecturas. (Wikipedia, 2012)

Cabe destacar que ya en esa época, Sega utilizaba un concepto similar al "Doble Núcleo" de los procesadores Intel o AMD actuales, uniendo los procesadores SH2 en paralelo. (Wikipedia, 2012)

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	2 CPU Principal 32bit RISC SH2 a 28.6 MHz, 25MIPS; y 1 de 32bit RISC SH1 a 20 MHz.	32Bits
VIDEO	Memoria 1.5MB	CD Buffer: 500 kb, IPL ROM: 500 Kb; resolución de 320x224, 640x224 ó 720x576 pixels; 24 bits (16,77 millones de colores), Scroll: 5 pantallas (Escalado, Rotación, Ampliación y Reducción); Mapeado de texturas Wire Frame.
MEMORIA	2 MB	
SONIDO	16bit CISC 68EC000 (11.3 MHz), 32 canales PCM y 8 FM	Muestreo de 16 bits, 44,1 kHz máximo con Procesador de señal digital (DSP), RAM 500Kb.
SOPORTE	CD	Unidad de CD: doble velocidad (300 kb/s).

Cuadro 3-13 Características técnicas de la consola SEGA Saturn

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

3.2.6.2 PLAYSTATION

PlayStation (oficialmente abreviada como PS, y de forma no oficial como PSX o PS1), es una videoconsola de sobremesa de 32 bits, lanzada por Sony Computer Entertainment el 3 de diciembre de 1994 en Japón. La consola fue pionera en el empleo del CD-ROM como soporte de almacenamiento para sus juegos, prescindiendo de los cartuchos convencionales que eran usados en la mayoría de videoconsolas de la época. (Wikipedia, 2012)

Figura 3.17 PlayStation

Fuente: http://programbytes48k.files.wordpress.com/2010/12/

PlayStation se considera la videoconsola más exitosa de la quinta generación tanto en ventas, como en popularidad. Además de la original se lanzaron dos versiones más, la Net Yaroze y la PSone. Tuvo gran éxito en emplear el CD-ROM dentro de su hardware, a pesar de que otras compañías ya lo habían empleado, tales como: SEGA (Sega CD), Panasonic (3DO), Phillips (CD-i) y Snk (Neo Geo CD). Dichas compañías tuvieron poco éxito al emplear el CD-ROM como soporte para almacenar juegos. (Wikipedia, 2012)

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	R3000A, de 32 bits RISC a 33 MHz.	32 Bits, LSI Logic Corp, tecnología licenciada de SGI y contiene, en el mismo chip, el "Geometry Transfer Engine" y el "Data Decompression Engine"; Motor de compresión de datos compatible con archivos MPEG-1 y H.261
VIDEO	1 Mb de memoria.	Separada de la CPU, se encarga de procesar toda la información de gráficos en 2D; Paleta de colores: 16,7 millones de colores; Resoluciones: desde 256 x 224 hasta 640 x 480; procesamiento gráfico: 4000 sprites de 8 x 8 píxeles, con scaling y rotación individual
MEMORIA	2 Mb.	ROM del BIOS: 512 kB, soporta tarjetas de memorias estándar (Memory Card) de 128 kB.
SONIDO	Unidad de procesamiento de sonido (SPU) con 24 canales y hasta 44,1 kHz	Efectos digitales como: Modulación Pitch, Cubierta, Enlace, Reverberación digital; puede procesar hasta 512 KB de muestras waveforms. Soporta MIDI, memoria 512kb.
SOPORTE	CD.	CD-ROM: Velocidad de lectura de 2x (300 kB/s); Compatible con el formato CD-XA.

Cuadro 3-14 Características técnicas de la consola PlayStation

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

ACCESORIOS Y PERIFÉRICOS

- Mando analógico: En un principio la PlayStation no contaba con un mando analógico ya que existían muy pocos juegos en 3D. Posteriormente, tras la aparición en el mercado de Nintendo 64 y su mando con joystick central, Sony desarrolló un mando analógico para los juegos en 3D. Se fabricó de un mayor tamaño que el original, fue equipado con dos pads analógicos uno para el movimiento y el otro para observar el entorno del juego, además de la función de vibración en su siguiente versión conocido como DualShock.
- Memory Card: almacenar las partidas de diversos juegos.
- Multitap: Fue creado con el fin de poder jugar más de 2 jugadores a la vez en un solo juego. Contiene un cable que tiene una salida igual a la de los controles y posee 4 entradas de control (A, B, C, D). Va colocado en las mismas ranuras donde se conectan los controles, usualmente sin importar la ranura que sea (1 o 2).

3.2.6.3 NINTENDO 64

La Nintendo 64 fue la Cuarta videoconsola de sobremesa de Nintendo, desarrollada para suceder a la Super Nintendo y para competir con la Saturn de Sega y la PlayStation (PSX) de Sony. Incorpora en su arquitectura un procesador principal de 64 bits. El soporte de almacenamiento de los juegos es en forma de cartuchos, alguno de ellos con memoria interna. El uso de este tipo de almacenamiento le supuso una seria desventaja comercial frente a sus competidores, ya que encarecía los costes de producción lo que aumentaba el precio final, y además era de una capacidad de almacenamiento menor al de un CD-ROM. (Wikipedia, 2012)

Figura 3.18 Nintendo 64 o N64

Fuente: http://1.bp.blogspot.com/-gI7LtofYdjY/TacBKRnicnI/AAAAAAAAAJc/A2HIRdWujsA/s1600/

Técnicamente, la utilización de cartuchos ofrece algunas ventajas frente al formato CD-ROM, siendo ésta la idea que mantenía Nintendo frente a la competencia. El cartucho tiene tiempos de acceso al sistema mucho más cortos, hace posible la inclusión de coprocesadores y otros chips dentro del cartucho para mejorar las capacidades del sistema, logrando ampliar su vida útil; y en un principio parecía que podría ser más económico por no pagar derechos ("regalías" o "royalties") por uso del formato CD-ROM, ni una unidad lectora para el mismo que sería además mucho más delicada que una ranura o slot para cartuchos. (Wikipedia, 2012)

La Nintendo 64 fue innovadora en diversos aspectos. El mando de control fue el primero en incluir unos botones dispuestos en cruz diseñados especialmente para que el usuario tomara el control de aspectos propios de juegos basados en entornos tridimensionales (el control de la perspectiva de juego o cámaras, por ejemplo), los cuales fueron utilizados por primera vez por el juego Super Mario 64. Otra innovación fue el incluir un stick analógico, el cual permite un control de movimiento más preciso. Otro de los aspectos novedosos a destacar fue la función de vibración gracias al llamado Rumble Pak, que consiguió que el mando de Nintendo 64 fuera el primero en vibrar y transmitir sensaciones al jugador. (Wikipedia, 2012)

Los cartuchos tienen una capacidad de hasta 512 Mbit (64 MB) con gran calidad sonora. Sin embargo, eran bastante limitados en comparación con la capacidad de un Compact Disc de 680 MB. (Wikipedia, 2012)

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	RISC CPU R4300i a 93,75 MHz	Arquitectura MIPS 64-bit
VIDEO	RCP: SP (procesador de sonido y gráficos) y DP (procesador de píxeles) incorporados, a 62,5 MHz.	Búfer Z, anti-aliasing (filtro anti-dentado), mapeado de texturas realista: interpolación de mapas MIP tri-lineal filtrada, corrección de perspectiva, mapeado del entorno; Resolución: 256 x 224 ~ 640 x 480 pixeles; paleta de 32 bits, 16,8 millones de colores en pantalla.
MEMORIA	RAMBUS D-RAM 4 Mb (Ampliable a 8 Mbytes mediante Expansion Pack)	
SONIDO	SP (procesador de sonido y gráficos).	Sonido digital.
SOPORTE	Cartucho.	

Cuadro 3-15 Características técnicas de la consola Nintendo 64

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

ACCESORIOS

Conectados a la consola

- Jumper Pak: Incluido en la consola, es un jumper o módulo de continuidad, ya que el BUS de memoria RDRAM transfiere datos del módulo 0 al módulo 1 siendo que el módulo 0 es la memoria interna del sistema y el 1 la bahía del puerto de expansión que requiere el Nintendo 64. Este aditamento hace la función de puente cuando no hay memoria instalada en el módulo 1.
- Expansion Pak: es un módulo de memoria RDRAM de 4MB que incrementa la memoria del sistema a un total de 8MB. Reemplaza al Jumper Pak. Ciertos juegos ofrecen en general una resolución mayor bajo su presencia. Y mejora de gráficos otros, como Donkey Kong 64 o Zelda: The Majora's Mask requieren

este accesorio para funcionar. El Expansion Pak se vendía por separado o junto con algunos juegos.

- Nintendo Disk Drive 64 (solo Japón): permitía leer discos magnéticos especiales de 64MB y conectarse a una línea telefónica para intercambiar datos y descargar juegos.

Conectados a la parte inferior del mando

- Transfer Pak: Permite la conexión de cartuchos de Game Boy y de Game Boy Color para intercambiar datos con los juegos de la Nintendo 64 (personajes, puntuaciones, juegos bloqueados...). Algunos juegos que hicieron uso de este accesorio fueron Pokémon Stadium, Mario Golf 64, Mario Tennis 64, etc.
- Controller Pak: Permite grabar partidas y configuraciones personalizadas. También servía para guardar repeticiones de carreras y partidas de memoria más grandes que el cartucho no puede guardar. Los modelos originales de Nintendo ofrecían una memoria de 256 KB dividida en 123 partes o páginas. Algunos modelos de terceros disponían de más memoria aún.
- Rumble Pak: Fue uno de los primeros accesorios, anunciado en el Shoshinkai Show en noviembre de 1996. Sirve para brindar función de force feedback (vibración) a ciertos juegos, es decir, la capacidad de hacer vibrar el mando de control para aportar viveza y realismo a ciertos eventos del juego, como golpes o ruidos. Cuanto más fuerte sea lo que aparece en el juego, mayor será la vibración. El Rumble Pak fue el primer periférico de la historia de los video-juegos que hacía vibrar un mando y supuso una revolución: la siguiente generación de consolas (Dreamcast, Xbox, Gamecube y PlayStation 2) incorpora tecnologías similares en sus mandos de control, y la tendencia se mantendrá.
- VRU Unit y Micrófono: Este periférico, que se conectaba al puerto de control 4, servía para dar órdenes a Pikachu en el juego interactivo "Hey You, Pikachu", en el cual se le hablaba directamente a Pikachu y este obedecía tus órdenes, mediante el reconocimiento de comandos de voz sólo en idioma inglés y japonés.

3.2.7 SEXTA GENERACIÓN 1998 -2005

La sexta generación de consolas (también llamada "la era de los 128 bits") comienza a finales del siglo XX. La primera consola de esta generación fue la Sega Dreamcast, lanzada en Japón el 27 de noviembre de 1998, siendo la de mayor éxito comercial la Playstation 2 de Sony, que apareció en el año 2000. (Wikipedia, 2012)

Todas las consolas de sobremesa de sexta generación poseen mandos ergonómicos, memorias externas, y, la diferencia más importante, conexiones de internet y red para jugar en línea o en una conexión cerrada. En esta generación que Sony lanza su segunda consola de video-juegos, y sin duda, su consola más popular, por su buen rendimiento y bajo costo, mientras que SEGA sale del sector con su última máquina. (Wikipedia, 2012)

3.2.7.1 SEGA DREAMCAST

La Dreamcast fue la primera consola de esta generación, y la última consola de video-juegos de Sega, también fue la primera en cesar su producción en esta generación. Sega implementó un tipo especial de soporte óptico llamado GD-ROM. Estos discos fueron creados con el fin de evitar la piratería de software, relativamente fácil en las consolas de la generación anterior, ya que coincidió con la salida al mercado de las primeras grabadoras de CD-ROM, sin embargo, este formato fue instantáneamente violado. (Wikipedia, 2012)

En 2001, se suspende la producción de este sistema, Sega se enfoca únicamente al desarrollo de software. Sin embargo, la compañía continuó dando a soporte a las consolas que fueron vendidas y al formato GD-ROM, hasta el 2007. El equipo fue el primero en disponer de un módem de 33.6 Kb, con el cual se podía acceder a Internet y jugar algunos títulos en línea como Phantasy Star Online.

Figura 3.19 Sega Dreamcast

Fuente: http://dagamutenlife.files.wordpress.com/2012/01/

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	SH-4 RISC a 200 MHz.	Unidad FPU vectorial de 128 Bits
VIDEO	PowerVR2DC (PowerVR2 CLX2 modificado) de NEC con una velocidad de reloj de 100 MHz	Produce hasta 7 millones de texturas, iluminado y sombreado de polígonos por segundo, y con 8 Mb de VRAM de 100 MHz de frecuencia, 16,77 millones de colores simultáneos (24 bit).
MEMORIA	16 Mb	
SONIDO	Super Intelligent (Yamaha) (64 canales PCM/ADPCM) con 2 MB de memoria.	Procesador de sonido con CPU RISC ARM de 32-Bit
SOPORTE	Disco óptico.	GD-ROM: velocidad máxima 12x (cuando funciona a una velocidad angular constante) - CAV GD-ROM es un nuevo medio de memoria de alta densidad. Su capacidad máxima era de 1,2 GB

Cuadro 3-16 Características técnicas de la consola Sega Dreamcast

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

Módem extraíble de 56 Kbps (33,6 Kbps en Europa). Un adaptador de banda ancha (básicamente una tarjeta Ethernet propia basada en un chip Realtek) conocido popularmente como BBA (broadband adapter) estaba disponible por separado.

3.2.7.2 PLAYSTATION 2

La PlayStation 2 de Sony continuó el mismo éxito de la PlayStation, y fue la primera videoconsola casera en incluir un reproductor de DVD, que permitía reproducir películas en el sistema. Además existía la posibilidad de poner un disco duro interno, en combinación con el adaptador de red. Al igual que su predecesor, también dispuso de un modelo pequeño que fue lanzado en el 2004 llamado PlayStattion Slim, fué el sistema más vendido en esta generación. (Wikipedia, 2012)

Figura 3.20 PlayStation 2 o PS2

Fuente: http://www.nosologeeks.es/wp-content/uploads/

Actualmente sigue activa con un importante número de usuarios, compitiendo con las consolas de la séptima generación, siendo la única consola que ha logrado tener un ciclo de vida tan largo que compite con la generación posterior a la suya. Una de las principales características distintivas son su procesador central conocido como Emotion Engine y su controlador de Dualshock 2. También el equipo incorpora un lector de DVD y 2 puertos USB 1.0 (algunos controladores utilizan estos puertos). En la versión PlayStation 2 Slim se incorporó un puerto Ethernet para ser utilizado servicio de internet Central Station. El mismo puerto puede ser agregado a las versiones originales de la consola. Algunos juegos incorporan la posibilidad de jugar a través de una red de área local (LAN), lo que permite además jugar por internet a través del sistema gratuito XLink Kai. (Wikipedia, 2012)

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	Emotion Engine (EE) RISC MIPS-IV (R5900) a: 294.912 MHz (299 MHz a partir de la Versión V9).	128 Bit, cuanta con Co-Procesador 1: FPU, Co-Procesador 2: VU0 y Unidad de Procesado Vectorial: VU1.
VIDEO	Sintetizador gráfico a 150 MHz (147.456 MHz).	16 Pixel Pipelines, 2.4 Gigapixels por Segundo (sin texturas), Filtro por Punto, Bilinear, Trilinear, Anisotrópico y Mip-Map, Corrección de Perspectiva de Mapeado de Texturas, Bump Mapping, Environment Mapping 32-bit Color (RGBA), 32-bit Z Buffer, 4MB DRAM Multipuerto Embedida.
MEMORIA	32Mb	RAMBUS DRAM con un ancho de 32 Bits (16 bits en Dual Channel)
SONIDO	48 canales de sonido surround 3D	Frecuencia de salida: hasta 48 kHz (calidad DAT), Compatible con Sonido Multicanal Dolby Digital, Dolby Pro Logic II ,AC3 y DTS.
SOPORTE	Discos CD y DVD	Discos CD-74 (650 Megabytes), CD-80 (700 Megabytes), Discos DVD-5 (4.7 Gigabytes) o DVD-9 (8.5 Gigabytes); DVD-ROM: velocidad aproximada de 4X. CD-ROM: velocidad aproximada de 24X.

Cuadro 3-17 Características técnicas de la consola PlayStation 2

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

INTERFAZ

- A/V Multi Out x1 (conector propietario): Señales de vídeo y audio analógicas.
- Digital Out (Optical) x1 (conector Toslink): Señal de audio óptica digital.
- Puerto de mandos x2 (conector propietario): Conexión de los mandos de PlayStation 2 y PlayStation además de otros periféricos como el receptor de infrarrojos para el mando de DVD.
- Puerto para Memory Card x2 (conector propietario): Conexión para tarjetas de memoria de PlayStation 2 y PlayStation. Posibilidad de usar el cifrado MagicGate en las tarjetas de memoria de PlayStation 2.
- Puerto USB 1.1 x2 (conector USB A con controladora compatible OHCI): Conexión para diversos periféricos, como volantes, micrófonos Singstar, EyeToy, headsets, etc...
- Conexión para el Adaptador de Red y una bahía para alojar un disco duro ATA 3,5".

3.2.7.3 GAMECUBE

La Nintendo GameCube fue la cuarta videoconsola de sobremesa de Nintendo, y el primer sistema de la compañía que prescinde de los cartuchos. Este sistema utiliza un formato de disco similar al DVD, denominado "GOD" (Gamecube Optical Disc), cuyo tamaño es de 8 cm. Este sistema fue un intento de recuperar el mercado perdido por la Nintendo 64 ante la PlayStation. (Wikipedia, 2012)

Figura 3.21 Nintendo GAMECUBE

Fuente: http://niwanetwork.org/wiki/images/thumb/b/bb/Gamecube-console.png/

Su procesador central está basado en un IBM PowerPC (tecnología previa utilizada en computadoras personales y portátiles), y su procesador gráfico desarrollado por ATI. Nintendo, por primera vez, prescinde del cartucho (ROM) como formato de almacenamiento, y adopta un formato óptico propio, el Nintendo Optical Disc. El nombre "GameCube" se debe a que el sistema tiene la forma de un cubo. (Wikipedia, 2012)

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	"Gekko", de la familia PowerPC a 485 MHz.	Fabricado por IBM, Precisión de datos internos: 32 bits (enteros) y 64 bits (punto flotante de doble precisión)
VIDEO	Sistema LSI "Flipper" desarrollado por ATI, a 202,5 Mhz	Color, Z Buffer: Cada uno 24 bits, caché de texturas Incorporado: Aprox. 1MB
MEMORIA	24 Mb.	Memoria auxiliar (utilizada para buffers de sonido y almacenamiento): 16 MB.
SONIDO	Procesador de sonido: DSP 16 bit Especial, a 101,25 MHz; 64 canales ADPCM.	Diseñado por Macronix, Memoria de instrucción: 8 KB RAM + 8 KB ROM. Memoria de datos: 8 KB RAM + 4 KB ROM y Frecuencia de muestreo: 48 KHz.
SOPORTE	Disco óptico, GOD ("GameCube Optical Disc").	Con tecnología de DVD, en un lector óptico fabricado por Matsushita, capacidad de 1,5 o 3 GB (doble capa), lector DVD 5X.

Cuadro 3-18 Características técnicas de la consola GAMECUBE

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

3.2.7.4 XBOX

La Xbox fue la última consola que salió en esta generación y la primera de Microsoft. Se apoyó como lo hizo Sega Dreamcast en el juego online e innovó al proporcionar a la consola un disco duro integrado; utiliza el formato DVD y da la posibilidad de guardar música desde un CD de audio a la consola utilizándolo juegos como GTA San Andreas. Tuvo una buena aceptación, aunque una corta vida. (Wikipedia, 2012)

Figura 3.22 Microsoft XBOX

Fuente: http://images.wikia.com/es.gta/images/4/48/

Sus principales características del mando son sus gatillos analógicos, que pronto serían adoptados por Sony en el Sixasis y el DualShock 3 y el control de su sucesora, la Xbox 360. (Wikipedia, 2012)

Su principal característica es su procesador central basado en el procesador Intel Pentium III. El sistema también incorpora un lector de DVD, un disco duro interno, puerto ethernet y por último el sistema dispone de cuatro conectores para los mandos. (Wikipedia, 2012)

La arquitectura de la Xbox está basada en la arquitectura x86 similar a la de un PC, esto ha facilitado a los desarrolladores adaptar un gran número de títulos de PC para la Xbox, ayudando a ampliar el catálogo de juegos de la consola. (Wikipedia, 2012)

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	Procesador de 32-bit a 733 MHz.	Basado en Pentium III en el modelo Coppermine con sistema Micro-PGA2, arquitectura x86
VIDEO	Chip NV2A ASIC a 233Mhz.	Desarrollado por Nvidia basado en la arquitectura GeForce3 y GeForce4, 120 millones de triángulos (polígonos) por segundo. Resoluciones 480i, 480p, 576i, 576p, 720p, 1080i, 8 Texturas por barrido, compresión de Texturas, AntiAlias a pantalla Completa, BumpMapping, Iluminación Especular.
MEMORIA	64 Mb	Tipo DDR SDRAM a una velocidad de 200 MHz.
SONIDO	Procesador desarrollado por nVidia. AC3/AC97 Sensaura	Codificación Dolby Digital Pro Logic II 5.1.
SOPORTE	Disco óptico, DVD	Disco Duro de 8 GB hasta 2 TB de capacidad. DVD de doble capa de 8.5 GB. DVD de 4,7 GB de capacidad. CD de 700 MB de capacidad.

Cuadro 3-19 Características técnicas de la consola XBOX

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

3.2.8 SÉPTIMA GENERACIÓN 2005 –2011

En la historia de los video-juegos, la séptima generación de videoconsolas es la generación que incluye las consolas lanzadas desde finales de 2005 por Nintendo, Microsoft y Sony. Para consolas de sobremesa, la séptima generación comenzó el 22 de noviembre de 2005 con el lanzamiento de la Xbox 360, de Microsoft y continuó con el lanzamiento de la PlayStation 3, de Sony, el 11 de noviembre de 2006; y de la Wii, de Nintendo, el 19 de noviembre de 2006. Cada nueva consola introdujo un nuevo tipo de avance en la tecnología. La Xbox 360 ofrece juegos renderizados de forma nativa con resoluciones de alta definición (a diferencia de la ampliación de la escala, lo que se podía hacer en un pequeño número de título de sexta generación); PlayStation 3 ofrece, además de juegos de alta definición, la reproducción de contenido multimedia HD a través de su reproductor de Blu-ray Disc; por otro lado, la Wii se centra en la integración de controladores con sensores de movimiento, así como palancas de mando. (Wikipedia, 2012)

Ésta generación de consolas vio el fin de la carrera de bits que había caracterizado a las múltiples generaciones a partir de la tercera generación de 8 bits. Habiendo constatado que 128 bits ofrece un "punto dulce" en términos de precio y rendimiento de compensación, las tres principales consolas de ésta generación utilizaron esta arquitectura, así como la Dreamcast y PlayStation 2 la habían generado en la generación anterior. (Wikipedia, 2012)

Uniéndose junto a Nintendo al mercado de sensor de movimiento, Sony lanzó la PlayStation Move en septiembre de 2010. Dicho accesorio incluye características de juego de detección de movimiento, similar al de Wii. Microsoft igualmente se unió a Sony y Nintendo con su Kinect. A diferencia de los otros dos sistemas ya mencionados, Kinect no utiliza controladores de ningún tipo y hace a los usuarios ser el "controlador". (Wikipedia, 2012)

3.2.8.1 XBOX 360

Fue desarrollada en colaboración con IBM y ATI y lanzada en América del Norte, Japón, Europa y Australia entre 2005 y 2006. Su servicio Xbox Live permite a los jugadores competir en línea y descargar contenidos como juegos arcade, demos, tráilers, programa de televisión y películas. La Xbox 360 es la sucesora directa de la

Xbox, y compite actualmente contra la PlayStation 3 de Sony y la Wii de Nintendo como parte de las videoconsolas de séptima generación. (Wikipedia, 2012)

Figura 3.23 Microsoft XBOX 360

Fuente: http://4.bp.blogspot.com/-Z9sxZvquSXs/T2mvrZZbfQI/AAAAAAAAABk/mlgC71rPs_o/s1600/

Como principales características, están su unidad central de procesamiento basado en un IBM PowerPC y su unidad de procesamiento gráfico que soporta la tecnología de Shaders Unificados. El sistema incorpora un puerto especial para agregar un disco duro externo y es compatible con la mayoría de los aparatos con conector USB gracias a sus puertos USB 2.0. Los accesorios de este sistema pueden ser utilizados en una computadora personal como son los mandos y el volante Xbox 360.

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	Xenon tres núcleo a 3,2 Ghz.	Basado en el diseño de IBM del PowerPC, multithreading simultáneo y 1 MB de caché de nivel 2. Diseñado por IBM, bajo el nombre Xenon.
VIDEO	GPU diseñado por ATI, bajo el nombre ATI Xenos, Radeon R500, y funciona a 500 MHz.	10 MB de EDRAM incorporado. resolución de salida de 720p o 1080i
MEMORIA	512Mb.	Memoria RAM del tipo GDDR3 a 700Mhz, Almacenamiento en Disco duro con capacidades de 20, 60 120 y 250 GB respectivamente.
SONIDO	Más de 256 canales de audio y 320 canales de descompresión independientes , 32 bits para el audio	Soporte de 48 kHz sonido de 16-bits. Los archivos de sonido para los juegos están codificados con el formato XMA audio format.
SOPORTE	Disco Óptico DVD	DVD-Video, DVD-ROM, DVD-R/RW, DVD+R/RW, CD-DA, CD-ROM, CD-R, CD-RW, discos de audio con formato Windows Media Audio, MP3 CD, y JPEG Photo CD; a 12X

Cuadro 3-20 Características técnicas de la consola XBOX 360

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

3.2.8.2 PLAYSTATION 3

Una característica importante que distingue a la PlayStation 3 de sus predecesoras es su servicio unificado de video-juegos en línea, la PlayStation Network, lo que contrasta con la anterior política de Sony de confiar en los desarrolladores de juegos para jugar en línea. Otras características importantes de la consola son sus capacidades sólidas de multimedia, la conectividad con la PlayStation Portable y su principal formato de disco óptico de alta definición Blu-ray Disc, como su principal medio de almacenamiento. La PS3 también da soporte al Blu-ray perfil 2.0, gracias a ello se puede interactuar de manera online con contenidos extras de películas y juegos. (Wikipedia, 2012)

Figura 3.24 Sony PlayStation 3 o PS3

Fuente: http://www.mundotebeos.com/tbos_2011/

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	PowerPC-base Core a 3.2GHz llamado CELL	Desarrollada conjuntamente por Sony Computer Entertainment, Toshiba, e IBM; Cell Broadband Engine Architecture, conformado por 7 núcleos y cada uno se les conoce como "Synergistic Processing Elements" (comúnmente abreviado SPE).
VIDEO	Nvidia, bajo el nombre RSX (basado en la GeForce 7800 GTX) a 550Mhz.	Resoluciones de 480i, 576i SD hasta 1080p HD, además, dispone 256 MB de memoria de vídeo GDDR3 para el RSX.
MEMORIA	256 MB	Memoria XDR, cuenta con una unidad interna de almacenamiento, inicialmente se disponía unidades de disco duro de 20, 40, 60, 80, 160 y 320 GB.
SONIDO	Sonido con Dolby 5.1ch, DTS, LPCM.	
SOPORTE	Disco óptico Blu-Ray	Velocidad de carga es 2x, la unidad soporta diversos formatos de discos: DVD, CD, SACD, entre otros soportes ópticos.

Cuadro 3-21 Características técnicas de la consola PS3

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

El sistema tiene Bluetooth 2.0, Wi-Fi (en los modelos de 40, 60 y 80 GB) Ethernet, USB 2.0 y HDMI 1.3ª.

3.2.8.3 WII

La característica más distintiva de la consola es su mando inalámbrico, el Wii Remote, el cual puede ser usado como un dispositivo de mano con el que se puede apuntar, además de poder detectar movimientos en un plano tridimensional. Otra de sus peculiaridades es el servicio WiiConnect24, que permite recibir mensajes y actualizaciones a través de Internet en el modo de espera.15 Adicionalmente, la consola puede sincronizarse con la portátil Nintendo DS, lo cual permite que Wii aproveche la pantalla táctil de la Nintendo DS como mando alternativo. (Wikipedia, 2012)

Figura 3.25 Nintendo Wii

Fuente:
http://www.tecnoalia.com/WebRoot/StoreES/Shops/eb3665/4E94/951B/0D5E/A04F/CD0D/AC10/1417/32D3/

Es importante resaltar la innovación de su controlador y la tecnología que incorpora en el sistema de juego. (Wikipedia, 2012)

CARACTERÍSTICAS TÉCNICAS

	TÉCNICO	OBSERVACIONES
PROCESADOR	IBM Broadway 729 MHz.	Unidad IBM PowerPC Fabricada con tecnología SOI CMOS de 90nm.
VIDEO	ATI Hollywood 243 MHz.	Chip ATI sistema LSI, tecnología SOI CMOS de 90 nm, resolución hasta 480p. 24Mb de Ram de video.
MEMORIA	64Mb	Memoria flash 512Mb
SONIDO	Estéreo compatible con Dolby Pro Logic II.	Mando con altavoz integrado.
SOPORTE	Disco Óptico Wii Optical Disc	Soporte a Nintendo GameCube Game Disc.

Cuadro 3-22 Características técnicas de la consola Wii

Fuente: (Wikipedia, 2012) Consultado 5/05/2012

3.3 PC (COMPUTADORA PERSONAL)

El mercado de video-juegos para computadoras personales va en descenso, desplazado por las consolas. La plataforma PC fue muy popular en los noventas cuando aparecen los primeros PC multimedia con CD-ROM. Además los juegos para PC ofrecían en sus inicios un estímulo muy fuerte para todo el sector ejecutivo y de oficina que tenía la PC como principal herramienta de trabajo.

En los años 1990 el computador personal o PC ofreció una alternativa a las consolas en cuanto a juegos se refiere, con la aparición de géneros como "aventuras gráficas", juegos de "estrategia" y el más popular en PC y nuevo género de los 90 que fueron géneros que nacieron con la aparición del mouse, los juegos en primera persona que luego ya en la segunda mitad tomaron auge juegos online en primera persona.

El sector de los video-juegos de PC solo se mantiene principalmente por estas razones:

- Es en el que mayor versatilidad hay; un ejemplo es Valve que al poner a disposición de los jugadores los necesario para crear mapas e incluso juegos

aparte, se ha podido crear una enorme comunidad de "moders", y crear juegos tales como el Counter-Strike o el Day Of Defeat.

- Algunos juegos de PC resultan incómodos al jugarlos con un joystick o mando de consola, como los ETR, es decir los juegos de estrategia en tiempo real, tales como los Age of Empires (serie) o los Command & Conquer
- La flexibilidad de hardware al poder incluir en el computador personal o PC tarjetas de video de distinta potencia arreglo a la capacidad económica del usuario o al rendimiento deseado obteniendo mayores resultados en calidad gráfica y generación de objetos 3D que las limitadas consolas.

Los computadores personales o PC fueron la plataforma de salida de los video-juegos y aún permanecen grandes ventas que únicamente salen hoy para PC (los juegos masivos RPG online) como "World of Warcraft", juegos de estrategia y los juegos hechos con tecnología Flash de Adobe para jugar de forma casual, pero es la plataforma "menos" importante en cuanto a ventas y ganancias para las empresas.

El computador, medio donde habían aparecido originalmente los video-juegos, era un lujo al alcance de muy pocos todavía a mediados de la década de 1970. Los primeros computadores personales comenzaron a aparecer en esa época, pero al principio carecían de monitor, y el único dispositivo de visualización eran las lentas impresoras que los usuarios podían conectar a sus máquinas. El éxito de los video-juegos en los salones recreativos estimuló la aparición de los primeros video-juegos programados por los usuarios de estos aparatos, pero las tremendas limitaciones de la primera generación de computadores personales imponían ciertas limitaciones al desarrollo de los programas. La lentitud del sistema dejaba a un lado la posibilidad de crear juegos de acción, y provocó de manera natural la aparición de juegos conversacionales o por turnos. La inmensa mayoría de esos juegos resultaban sumamente crudos, siendo versiones de juegos clásicos como el ahorcado o el juego de los barquitos. Sin embargo, el programa ELIZA de Joseph Weizenbaum -un profesor del Instituto Tecnológico de Massachusetts había impresionado a muchas personas, y entre éstas se encontraba Will Crowther, un programador que trabajaba para Departamento de Defensa estadounidense.

Crowther, un aficionado a los juegos de rol, se había divorciado en 1975 y decidió escribir un juego para poder compartir con sus hijas. Usando un PDP-10

escribió un programa en el que el jugador, usando frases en lenguaje natural, debía salir de unas cuevas y resolver una serie de misterios. El juego gustó no sólo a las hijas de Crowther, sino a muchos otros usuarios del computador, quienes comenzaron a copiarlo a través de ARPAnet para jugar en sus ratos libres a escondidas de los administradores de los sistemas. Sufrió diversas modificaciones (la más importante de las cuales fue llevado a cabo por Don Woods) y en 1976 había logrado una gran popularidad. Adventure no fue el primer programa conversacional (antes se habían lanzado Startrek o Wumpus, por ejemplo), pero sí el más influyente, inaugurando en la práctica un género que dio lugar mucho más tarde a títulos como Ultima, Rogue, o Zork.39 nota 29

Mientras tanto, los nuevos computadores personales como el Altair 8800 o el KIM-1 iban haciendo su aparición en el mercado. Steve Wozniak, tras finalizar su trabajo con el Breakout de Atari, y haber vendido sus primeros Apple I, había decidido construir un computador personal más potente, una máquina que dispondría de gráficos en color, sonido y conectores para los controladores de juego y para los receptores de televisión. Junto con Steve Jobs presentó su idea a Nolan Bushnell, pero éste la rechazó.40 Finalmente, con el apoyo financiero de Mike Markkula consigieron lanzar en 1977 su Apple II, un modelo mucho más avanzado que resultaba perfecto para el diseño de los video-juegos. A pesar la competencia del PET de Commodore y del TRS-80 de Tandy, el nuevo computador se abre un importante hueco en el mercado, inaugurando definitivamente la era de la microinformática. Los primeros juegos para computadores personales no tardaron en aparecer: Microchess para KIM-1 (1976) o Adventureland para TRS-80 (1978) estuvieron entre los primeros, pero muy pronto apareció una "industria casera" de fabricación y venta de video-juegos que no tardaría en dar jugosos frutos. Scott Adams recibió 7 000$ por Adventureland, que era distribuido en bolsas Ziploc con las instrucciones fotocopiadas, Infocom, los creadores de Zork vendieron miles de copias del juego, y en 1980 Ken y Roberta Williams, futuros fundadores de Sierra Entertainment publicaron para Apple II Mystery House, un juego inspirado en Adventure que incluía por primera vez detalles gráficos y que logró vender más de 3 000 copias. Tanktics, en 1977 inauguró el género de los wargames y Alakabeth de Richard Garriot hizo lo propio con los juegos de rol en 1979 (Donovan, 2010, págs. 57-67).

3.4 CONSOLAS PORTATILES

Una videoconsola portátil es un dispositivo electrónico ligero que permite jugar video-juegos y que, a diferencia de una videoconsola clásica, los controles, la pantalla, los altavoces y la alimentación (baterías) están todos integrados en la misma unidad y todo ello con un pequeño tamaño, para poder llevarla y jugar en cualquier lugar o momento. (Indicelatino, 2006)

Es la empresa japonesa Nintendo, la que populariza el video-juego portátil. Primero con los juegos Game & Watch entre 1980 y 1991 que permite jugar a un único juego de bolsillo en cualquier lugar. Segundo con la popular Game Boy que permite jugar en cualquier parte con juegos intercambiables. Por último, Nintendo DS que consigue habituar a jugar en cualquier parte a un público no necesariamente interesado en los video-juegos. La mayoría de las empresas que han intentado sumarse a este mercado han fracasado estrepitosamente. Tan solo Sega Game Gear y Sony Play Station Portable, han conseguido arrebatar una porción de mercado suficiente para generar beneficios y prolongar su vida comercial durante el ciclo normal de una generación. (Indicelatino, 2006)

3.4.1 PRIMERA GENERACIÓN

El primer Video-juego electrónico portátil que aparece en el mercado con su propia pantalla LCD es un minijuego de Mattel llamado Mattel Electronics Auto Race en 1976, al año siguiente 1977 la misma compañía saca al mercado Mattel Electronics Football y poco después en 1978 Coleco saca un juego de similares características llamado Coleco Electronic Quarterback. Esta podría considerarse como la primera generación de video-juegos portátiles. (Indicelatino, 2006)

Figura 3.26 Mattel Electronics Auto Race

Fuente: http://spookyshobbyshop.com/TEMPLATE%20hh%20mattel%20auto%20race%20console%20front%20VG-EX.JPG

Figura 3.27 Coleco Electronic Quarterback

Fuente: http://www.handheldmuseum.com/Coleco/Coleco-ElectronicQuarterback.jpg

3.4.2 SEGUNDA GENERACIÓN

La primera videoconsola portátil con cartuchos intercambiables fue la Microvision de Milton Bradley desarrollada en 1979. Aunque no hay datos oficiales de ventas, tuvo buena acogida en su primer año. Así que si bien no arrasó en ventas si parece que tuvo un discreto éxito. Tuvo 12 juegos en su catálogo, y dejó de fabricarse en 1981. Como curiosidad aparece en la película Viernes 13 II. (Indicelatino, 2006)

Figura 3.28 Microvision de Milton Bradley

Fuente: http://www.handheldmuseum.com/MB/MB-MicrovisionUS.jpg

Hubo otra videoconsola portátil llamada Entex Select a Game que contaba con una pantalla de leds azules y rojos, que llevaban el procesador en el propio cartucho en vez de en la consola.

Figura 3.29 Entex Select a Game

Fuente: http://odstatic.com/pixfans.com/2008/09/selectagamebg.jpg

3.4.3 TERCERA GENERACIÓN

La serie Game & Watch era una línea de aproximadamente 59 juegos electrónicos portátiles hechos por Nintendo y creados por Gunpei Yokoi de 1980 a 1991. Consistían de un solo juego que se podía jugar en pantalla LCD, además de ser un reloj y alarma. Algunos de los títulos del formato Game & Watch fueron Pinball, Donkey Kong, The Legend of Zelda, Mario Bros, Mickey Mouse y Balloon Fight.

Actualmente, conseguir estas consolas portátiles es extremadamente difícil y se consideran piezas históricas y artículos de coleccionista. (Indicelatino, 2006)

Figura 3.30 Game & Watch Donkey Kong

Fuente: http://images.wikia.com/video-juego/images/0/0b/Game_%26_Watch_Donkey_Kong.png

El primer video-juego portátil exitoso, fue la serie de juegos de bolsillo Game & Watch de Nintendo, obra del difunto Gunpei Yokoi. Salieron entre 1980 y 1990 unos 60 juegos, basados en la electrónica de una calculadora. Algunos de los personajes más populares de Nintendo como Mario, Link o Donkey Kong hicieron acto de presencia en esta serie de mono-juegos. (Indicelatino, 2006)

Debido al éxito de estos juegos otras empresas lanzaron propuestas similares. Bandai lanzó en 1982 su serie Bandai LCD Solar Power, cuya principal característica era que no necesitaba pilas ya que funcionaba gracias a un panel solar y además varias incluían la posibilidad de dos paneles distintos para jugar. (Indicelatino, 2006)

Figura 3.31 Bandai LCD Solar Power

Fuente: http://photos.pcpro.co.uk/images/front_picture_library_PC_Pro/dir_308/it_photo_154087_52.jpg

Otra empresa que tuvo fama y éxito con estos juegos portátiles fue Tiger Electronics, que a partir de 1984 lanzó su propia serie de video-juegos de este estilo, aprovechando licencias de famosas películas como Robocop, Batman, etc. o juguetes como G.I.Joe entre otros, y la novedad de incorporar sus juegos a relojes de pulsera. (Indicelatino, 2006). La Epoch Game Pocket Computer anticipa el formato que tendrán las futuras consolas portátiles. Apareció en 1984 en Japón de la mano de la empresa Epoch. Y está justo un paso por detrás de la Nintendo Game Boy, siendo la primera consola con gráficos generados por una LCD sin ayuda de pantallas con el fondo fijo impreso, y con cartuchos intercambiables. Aunque tuvo poco éxito y duró apenas un año, con tan solo 5 juegos y dos en memoria, sus juegos eran parecidos a los de Atari 2600, en versión monocromo. (Indicelatino, 2006)

Figura 3.32 Epoch Game Pocket Computer

Fuente: http://www.museodelvideojuego.com/wp-content/gallery/segunda-generacion/Epoch-Game-Pocket-Computer.jpg

3.4.4 CUARTA GENERACIÓN

La portátil que revolucionó el mercado y que supuso un éxito arrollador en ventas fue la Game Boy el 21 de abril de 1989 en Japón de la mano de Nintendo. En su fase prototipo se llamaba Dot Matrix Game inspirándose en las máquinas Game & Watch. Se alimentaba mediante pilas, y permitía jugar en cualquier lugar y a varios juegos. En 1996 Nintendo sacó una versión reducida de la GameBoy, la versión Pocket, para revitalizar su portátil. Era una versión mucho más pequeña, con menos peso y una pantalla más grande y que además tan sólo usaba 2 pilas en vez de las 4 de antes. (Indicelatino, 2006)

Figura 3.33 Nintendo Game Boy

Fuente: http://imsai8080.files.wordpress.com/2011/03/11.jpg

En el mismo año 1989 fue lanzada la Atari Lynx, una videoconsola portátil de 16 bits producida por Atari. No tuvo éxito debido a su elevado precio y la corta duración de las baterías. También Nec, decidió versionar, su Turbografx, para hacerla de bolsillo, sacando la Turbo Express, que partía con la baza de ser totalmente compatible con su hermana mayor, por lo que no hacía falta crear nuevos juegos. Solo se lanzó en Japón y Estados Unidos, y no llegó a los 2 millones de unidades vendidas. Sus principales problemas eran la autonomía, la mala calidad de su LCD que se rompía con facilidad, y un problema de jugabilidad, los jugos de aventuras, tenían textos pensado para ser jugados en televisores, por lo que eran ilegibles en una pantalla tan pequeña. (Indicelatino, 2006)

Figura 3.34 Atari LYNX

Fuente: http://4.bp.blogspot.com/__mADelK4o78/TRNMW0JF72I/AAAAAAAAHY0/1yKhxNOzU40/s1600/atari-lynx.jpg

Figura 3.35 Turbo Express de NEC

Fuente: http://upload.wikimedia.org/wikipedia/commons/thumb/a/a1/TurboExpress-Front.png/230px-TurboExpress-Front.png

3.4.5 QUINTA GENERACIÓN

Sega intentó suceder a Game Gear, pero sus problemas económicos por los fracaso del Mega CD y Mega 32X, y que la Saturn no acababa de despegar, hicieron que la opción más barata fuese ponerle una pequeña pantalla a una versión reducida de Mega Drive. El aparato en cuestión se llamó Nomad, tenía menos autonomía que Game Gear, y el cartucho de Mega Drive no encajaba del todo bien y se podía salir durante la partida. (Indicelatino, 2006)

Figura 3.36 SEGA Nomad

Fuente: http://www.old-computers.com/museum/photos/sega_nomad_2s.jpg

Tiger Game es una videoconsola portátil creada por Tigers Electronics saliendo a la venta en septiembre de 1997 hasta el 2000. La "game.com", una extraña mezcla entre consola y PDA. Fue distribuida por la compañía Tiger Electronics. Como características especiales disponía de una pantalla táctil, cosa que no ha vuelto a tener ninguna portátil hasta la Nintendo DS, y disponía de unos cuantos accesorios curiosos, como un módem a 14400 bps. (Indicelatino, 2006)

Figura 3.37 Tiger Game - game.com

Fuente: http://www.rubbercat.net/archive/technewsrumorinsider/gamecom1.jpg

La Neo Geo Pocket es una consola portátil de SNK que fue lanzada en Japón en 1998 y en 1999. La escasa aceptación que tuvo este modelo, hizo que al año siguiente SNK optase por lanzar una versión en Color "Neo Geo Pocket Color". En este caso su fracaso fue debido fundamentalmente a la ausencia de promoción y márketing que castigo a esta interesante consola a la relegación. Algunos de sus juegos fueron de notable calidad: Metal Slug, Sonic the Hedgehog Pocket Adventure o SNK vs. Capcom: Match Of The Millennium. (Indicelatino, 2006)

Figura 3.38 Neo Geo Pocket Color

Fuente: http://3.bp.blogspot.com/_2RtAHhhGM9M/TL8aJGKNWMI/AAAAAAAAAVY/_M0gqZa-3Yk/s1600/VideoConsoloNeoGeoPocketGx.jpg

El 23 de octubre de 1998 en Japón, Nintendo lanzó su tercera consola portátil, la Game Boy Color. La novedad de esta nueva videoconsola era el color, tenía 56 colores y un puerto de infrarrojos (para intercambiar más que nada datos entre una consola y otra) y un procesador más. La Game Boy Color tenía compatibilidad con los juegos antiguos de la GameBoy (estos se veían con algo de color) y además, 2 nuevos modos para los juegos, uno de 32 colores (juegos compatibles aún con GameBoy) y otro de 56 (juegos exclusivos GBC). Se actualizaron juegos de la anterior GB, como el Tetris DX y el Zelda DX y se lanzaron más de 230 video-juegos entre marzo de 1998 y noviembre de 2002. (Indicelatino, 2006)

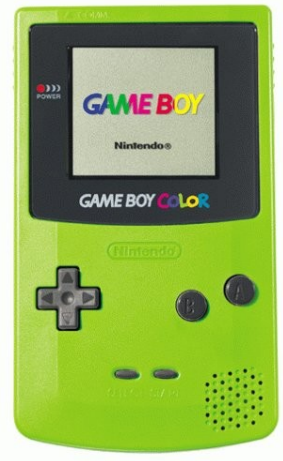

Figura 3.39 Nintendo Game Boy Color

Fuente: http://www.nostalgia90.com/wp-content/uploads/2011/01/GameBoyColor.jpg

3.4.6 SEXTA GENERACIÓN

Nintendo comenzó a fabricar una nueva consola, la Game Boy Advance (GBA), desde noviembre de 2001. Contaba con un procesador ARM propio de 32 bits a 16,7

MHz. El sistema llevaba un procesador Z80 a 2 MHz para dar soporte al software de Game Boy clásica y color, pero sus dos procesadores no podían estar funcionando a la vez debido a diferencias de voltaje y utilización del bus. La pantalla LCD era capaz de mostrar una retícula de 240x160 píxeles en color de 15 bits (32768 colores). (Indicelatino, 2006)

Figura 3.40 Nintendo Game Boy Advance

Fuente: http://images.wikia.com/video-juego/images/6/62/Advance2.jpeg

Esta consola también tuvo varias versiones, la GBA SP y la GB Micro, además de poder plegarse para conseguir un tamaño aún más portátil. Mientras que la GB Micro se hizo aún más pequeña y con una pantalla más nítida y clara. La GBA además de ser compatible con juegos de Game Boy, Game Boy Color tiene una librería propia que llega a los poco más de 800 juegos, entre los que cabe destacar: Mario Kart Super Circuit (Nintendo), Castlevania circle of the Moon(Konami), Super Mario Advance 2 y 3, y Metroid Fusion, Golden Sun II, Zelda ALLTP, Final Fantasy Tactics Avance, Super Mario Advance 4, Wario Ware Inc, Metroid Zero misión entre otros títulos. (Indicelatino, 2006)

Figura 3.41 Nintendo Game Boy Advance SP o GBA SP

Fuente: http://cdn.slashgear.com/wp-content/uploads/2011/01/gameboy_advance_sp-580x435.jpg

La GP32 (GamePark 32) es una consola de video-juegos portátil de 32 bit, producida por la compañía coreana GamePark, que salió en Navidades de 2001. Tiene como principal característica, que es una consola orientada al software libre, siendo la primera de este tipo. Debido a esto, posee un limitado catálogo de juegos comerciales, cercano a la veintena, pero tiene una extensa lista de aplicaciones: desde reproductores de DivX y MP3 a emuladores, incluyendo juegos caseros y otro tipo de aplicaciones más exóticas, como una versión reducida de Linux. (Indicelatino, 2006)

Figura 3.42 GamePark 32 o GP32

Fuente: http://members.fortunecity.es/halbilbo0768/consolas/gamepark_32.jpg

La Tapwave Zodiac es una PDA con sistema operativo Palm OS 5 a la vez que una potente videoconsola portátil fabricada por la compañía norteamericana Tapwave, y lanzada al mercado en octubre de 2003. Hay dos modelos: la Zodiac 1 con 32 MB y la Zodiac 2 con 128 MB. La consola tuvo soporte hasta el 25 de julio de 2005, cuando Tapwave entró en bancarrota y no pudo seguir proporcionando soporte. (Indicelatino, 2006)

Figura 3.43 Tapwave Zodiac

Fuente: http://jscustom.theoldcomputer.com/images/manufacturers_systems/Tapwave/Zodiac/633419zodiac_1.jpg

Nokia N-Gage Apareció en 2003. La idea no era mala, los móviles cada vez tenían más juegos que eran simples y bastante antiguos, pero aún así se iban haciendo

más populares. Así que en Nokia pensaron en juntar en un mismo aparato un móvil de gran calidad con una buena consola portátil. (Indicelatino, 2006)

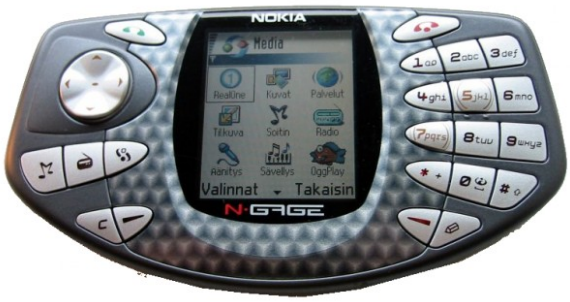

Figura 3.44 Nokia N-Gage

Fuente: http://cdn.uproxx.com/wp-content/uploads/2011/02/Nokia_N-Gage-600x329.png

3.4.7 SÉPTIMA GENERACIÓN

La Nintendo DS tiene dos pantallas que permiten dos puntos de vista de un mismo lugar, acceder a mapas y menús de forma rápida y fácil. Una pantalla táctil que permite interactuar con el entorno del juego, pulsar interruptores con el dedo, quitar el polvo de un libro, escribir con un lápiz, acariciar, rasgar, frotar y otras acciones. La segunda pantalla es un LCD estándar sin características destacables. Dispone además de un sistema de reconocimiento de voz; mediante el micrófono el jugador puede dar órdenes a los juegos, interactuar con los personajes, soplar para hinchar un globo, hacerle el boca a boca a un personaje herido o incluso chatear con alguien que está al otro extremo del mundo mediante la conexión Wi-Fi de Nintendo, una conexión inalámbrica tanto entre consolas como a Internet, siempre que dispongamos de un punto de acceso Wi-Fi con conexión, preferiblemente de banda ancha, que permite jugar online a ciertos juegos. Es compatible con Game Boy Advance. Fue lanzada el 21 de noviembre de 2004. A pesar de un hardware no poco potente, la pantalla táctil permite juegos no tan comunes en consola. (Indicelatino, 2006)

Nintendo DS ha tenido tres revisiones, DS Lite más compacta, y DSi, una revisión más severa, eliminando la ranura para juegos de GBA, incluyendo una cámara de fotos de 0.3mpx, y ranura para tarjetas SD para descargarse juegos, aplicaciones, música y las fotos que hagamos. Se elimina el coprocesdor, y se unifica en un único procesador más potente. Por último una versión de mayor tamaño con pantallas más grandes, llamada DsiXL sale con el fin de recalar en un público de avanzada edad, que tenga problemas de visión y manejabilidad. (Indicelatino, 2006)

Figura 3.45 Nintendo DS

Fuente:
http://2.bp.blogspot.com/_6fN0N2SeNTE/TIA_E99AwCI/AAAAAAAAB9M/Utct0NeOoSo/s1600/nintendo_ds.jpg

La PlayStation Portable o PSP es una videoconsola que fue lanzada al mercado el 12 de diciembre de 2004 en Japón. Tras el éxito de Sony en sobremesa, con la marca playstation. faltaba por saber si Sony tenía algo que decir en el mercado portátil que prácticamente era monopolio de Nintendo. Así que Sony lanza esta consola, dos premisas: Potencia gráfica y capacidad multimedia. Aunque durante el primer año hubo un empate técnico, a mediados del segundo año PSP empezó a quedar por detrás de DS en ventas. No obstante aunque la diferencia se ha ido ampliando con el paso del tiempo. La PSP tiene tres revisiones PSP 2000 y 3000, con mejoras en la pantalla, y más RAM para reducir los tiempos de carga del UMD, y una última versión PSPGO sin UMD. (Indicelatino, 2006)

Figura 3.46 PlayStation Portable o PSP

Fuente: http://img97.imageshack.us/img97/5356/psp3on.jpg

3.4.8 OCTAVA GENERACIÓN

El 15 de junio de 2010 se presenta en el discurso de Nintendo en la Electronic Entertainment Expo la videoconsola 3DS, con la cual se puede jugar a juegos y ver películas en 3D. Además, la nueva pantalla ofrece imágenes estereoscópicas sin necesidad de gafas especiales para disfrutar del efecto 3D, incorpora una pantalla tactil, Wi-Fi, sensor de movimiento con giroscopio de tres ejes y acelerometro de tres ejes. La consola también incorpora la captura de imágenes en 3D gracias a dos cámaras exteriores. Incluye también una consola virtual, para disfrutar los clasicos de Gameboy refrescados en 3D. Aparte de su capacidad para mostrar imágenes en 3D, también incluye juegos de realidad aumentada y se puede disfrutar de películas en 3D. Fue lanzada en febrero de 2011 en Japón y un mes después en América y Europa. (Wikipedia, 2012)

Figura 3.47 Nintendo 3DS

Fuente: http://www.sectorn.net/wp-content/uploads/2011/07/3ds.jpg

PS Vita es el nombre bajo el que se conoce a la sucesora de la PlayStation Portable o PSP. Este nuevo modelo de videoconsola portátil, fue presentado el día 27 de enero de 2011 por la multinacional de origen japonés Sony Computer Entertainment. El sistema presenta una forma ovalada similar al diseño de la primera PSP, con una pantalla de 5 pulgadas OLED con capacidad multitáctil localizada al centro del sistema. Tiene dos sticks análogos, un pad, los cuatro botones clásicos de la playstation, dos botones L y R, un botón Playstation, los botones Start y Select. Por dentro el sistema presenta un procesador ARM Cortex-A9 MPCore de cuatro núcleos, un procesador gráfico SGX543MP4+. El sistema también trae un touchpad trasero, una cámara frontal, otra trasera, micrófono, sistema sensor de movimiento Sixaxis (giroscopio de tres ejes,

acelerómetro de tres ejes), brújula electrónica de tres ejes, GPS, Wi-Fi, 3G, y Bluetooth v2.1+EDR. La nueva PS Vita, será lanzada en dos versiones: una con soporte para 3G, y una versión más barata sin 3G. La version 3G vendrá con aplicaciones pre-cargadas en la consola, que utilizarán el 3G. (Wikipedia, 2012)

Figura 3.48 PS Vita de SONY

Fuente: http://www.planetadejuego.com/uploads/media/images/PSVITA03.jpg

Capítulo 4

Software en el Desarrollo de Video-juegos

El desarrollo de video-juegos es la actividad por la cual se diseña y crea un video-juego, desde el concepto inicial hasta el video-juego en su versión final. Ésta es una actividad multidisciplinaria, que involucra profesionales de la informática, el diseño, el sonido, la actuación, etcétera. El desarrollo de un video-juego generalmente sigue el siguiente proceso: (Barrios, 2009)

- Concepción de la idea del video-juego
- Diseño
- Planificación
- Pruebas
- Producción
- Mantenimiento

El proceso es similar a la creación de software en general, aunque difiere en la gran cantidad de aportes creativos (música, historia, diseño de personajes, niveles, etc) necesarios. El desarrollo también varía en función de la plataforma objetivo (PC, móviles, consolas), el género (estrategia en tiempo real, RPG, aventura gráfica, plataformas, etc) y la forma de visualización (2D y 3D). (Barrios, 2009)

4.1 ARQUITECTURA DE LOS VIDEO-JUEGOS

Durante décadas, los ingenieros del software han creado sus propios diseños empezando de cero, o reutilizando diseños de otros. Hoy en día, esa situación ha cambiado, surgiendo una nueva disciplina conocida como arquitecto de software que utiliza un lenguaje común a todos los ingenieros del software para detallar diseños de alto y bajo nivel (Braude, 2001). (Gómez Martín, 2007)

4.1.1 DEFINICIÓN DE ARQUITECTURA

La definición de lo que se entiende por arquitectura de software no está consensuada pero según Gómez Martin menciona que, “sin embargo algunos autores,

como Braude (2001) indican simplemente que el término *arquitectura* es equivalente a *diseño* al más alto nivel (Braude, 2001, pag. 150). También Eoin Woods habla de decisiones de diseño al decir que una arquitectura software es el conjunto de decisiones de diseño que, si son incorrectas, pueden causar que el proyecto se cancele (Rozanski y Woods, 2005)". (Gómez Martín, 2007)

Además dice que "Shaw y Garlan (1996), no hablan de arquitectura como tal, sino de diseño a nivel de arquitectura, entendiendo éste como el relacionado con cuestiones estructurales, como la organización del sistema, protocolos de comunicación, sincronización y acceso a datos, asignación de funcionalidad a cada elemento, composición de cada uno de estos elementos, distribución física, escalado y rendimiento, etc. Cuando se habla de arquitectura como un diseño de alto nivel, tiene sentido hablar de patrones arquitectónicos (llamados estilos arquitectónicos por Shaw y Garlan (1996), arquitecturas por capas, tuberías y filtros, pizarras, o modelo-vista-controlador (Buschmann et al., 1996)". (Gómez Martín, 2007)

Una definición más apropiada es la expuesta por Bass et al. (2003), en la que definen la arquitectura de un programa o sistema de computador como la estructura o estructuras del sistema, que comprenden los elementos software, las propiedades de esos elementos visibles desde el exterior y las relaciones entre ellos.

Esta forma de entender una arquitectura software es compartida por Rollings y Morris (2004), y también es la que aceptaremos de aquí en adelante, "***una arquitectura software es la estructura de un programa, abarca también el flujo de datos, definiendo las interacciones entre todos los elementos"***. (Gómez Martín, 2007)

4.1.2 ARQUITECTURA DE SOFTWARE EN VIDEO-JUEGOS

De acuerdo a Gómez Martín, la complejidad de los juegos ha llegado a unos límites inimaginables. Hoy por hoy, para conseguir un video-juego de alta calidad, se requiere de un presupuesto capaz de soportar un gran número de personas trabajando durante al menos año y medio. Para corroborarlo, basten los siguientes datos:

- En Noviembre de 1999 salía Unreal Tournament, después de 18 meses de trabajo de 16 personas, un presupuesto de dos millones de dólares y 350.000 líneas de código entre C++ y UnrealScript (Reinhart, 2000).
- El juego *Vampire: the Masquerade* requirió 12 desarrolladores durante 24 meses, casi dos millones de dólares y 366.000 líneas de código (Huebner, 2000).

- Peter Molyneux puso a la venta en Marzo de 2001 su juego Black & White, después de invertir casi seis millones de dólares en 25 desarrolladores durante tres años, que escribieron dos millones de líneas de código (Molyneux, 2001).
- En Junio de 2002 salió a la luz Neverwinter Nights, después de 5 años de desarrollo con un equipo que en algunos momentos llegó a ser de hasta 75 personas (Grieg et al., 2002).
- En 2003, sólo portar el Splinter Cell de la plataforma XBox a PlayStation 2 requirió 76 personas durante cinco meses a tiempo completo (Hao, 2003).
- El desarrollo de juegos *de nueva generación* para las nuevas consolas maneja cifras aún más escalofriantes. Para el Project Gotham Racing 3 de XBox 360, tuvieron que invertir 80.000 libras sólo para comprar servidores en los que almacenar el arte del juego, y las veinte mil fotografías en las que se basaron para modelar cada una de las ciudades que aparecían en él (Gillen, 2005).
- El desarrollo de Call of Duty 2 para XBox 360 requirió 2 años de trabajo de 75 personas, y tuvo un presupuesto de más de 14 millones de dólares (Duffy, 2005).

Gómez Martín nos plantea que al ver estas cifras, queda claro que el desarrollo de video-juegos ha cambiado desde aquellas primeras aplicaciones desarrolladas para consolas, recreativas y computadores de 8 bits. Desde siempre, este tipo de aplicaciones han intentado aprovechar hasta el último recurso del hardware para hacer más llamativo el aspecto final del juego. Estos programas son aplicaciones de tiempo real, donde una respuesta dada tarde puede romper toda la sensación de inmersión del jugador. Es necesario que el número de fotogramas mostrados por segundo sea elevado, para que el entorno presentado se considere interactivo y animado. Es por esto que los programadores deben conocer a la perfección el funcionamiento interno de todos los componentes para no desperdiciar recursos debido a una mala programación. Cuando los recursos disponibles crecen, no decrece el esfuerzo necesario para crear los juegos. Al contrario, lo que ocurre es que aumentan sus pretensiones, lo que hace que la complejidad de programación, generación de modelos y recursos en general, también aumente. Esto conlleva un crecimiento notable tanto en el tiempo de desarrollo de un único título como en el aumento del número de personas del equipo. (Gómez Martín, 2007) (Barrios, 2009)

Desde el punto de vista histórico, durante la década de los 80 y principios de los 90, era habitual que los juegos se siguieran desarrollando en lenguaje ensamblador, a pesar de los grandes avances en los compiladores. La idea generalizad era que el compilador no podía competir con un programador de ensamblador experto, por lo que los desarrollos seguían utilizándolo a pesar de aumentar significativamente la duración del proyecto. Durante la década de los 90 la situación, por fin, fue cambiando, y los estudios comenzaron a utilizar activamente lenguajes de alto nivel, especialmente C debido a sus capacidades de bajo nivel. Entre los compiladores estrella de aquella época hay que destacar, en el mundo del computador personal o PC, Watcom C. Este compilador fue ampliamente utilizado a mediados de los 90, para la programación de juegos para MS-DOS (como Doom). Su éxito radicó en una muy buena calidad del código generado, y en su soporte para la programación en modo protegido del Intel 386. Desgraciadamente, Sybase, la empresa que lo creó, abandonó el proyecto en 1999, ya que su principal mercado era el del mundo del computador personal o PC (en Windows), y no podían competir con Microsoft. (Gómez Martín, 2007). (Gómez Martín, 2007) (Barrios, 2009)

Se puede decir que en aquella época se vivió un gran salto en el proceso de desarrollo, ya que se pasó de programar la práctica totalidad del juego en ensamblador, a utilizar lenguajes de alto nivel. Pensando fríamente, constituyó un gran paso adelante, ya que los programadores tuvieron que rendirse a la evidencia de que era mejor aceptar que el código ejecutado por la máquina durante el juego fuera generado por un compilador, que alargar un proceso de desarrollo debido al uso del tedioso y propenso a errores aunque óptimo ensamblador. (Gómez Martín, 2007)

Lo que se tardó mucho más en aceptar fue el uso de código desarrollado por otros (código externo) en el propio juego. Cada estudio, una y otra vez, programa parte de código que podría haber comprado a desarrolladores externos. Es lo que en el mundo anglosajón se conoce como el síndrome "*no inventado aquí*" o en inglés "*not invented here*", o reticencia a utilizar algo no creado en el propio estudio. (Gómez Martín, 2007)

Esta reticencia se debía principalmente al hecho de pensar que el estudio podía desarrollar ese módulo o funcionalidad de forma más óptima que el componente comprado.

De acuerdo a Gómez Martín, una de las cosas que ayudó a dar el salto, y a que se empezara a aceptar el incluir desarrollos externos en el propio juego fueron la explosión de tarjetas gráficas aceleradoras. Pronto se hizo evidente que los desarrolladores no

podían optar por la programación a bajo nivel de motor gráfico, accediendo directamente a las capacidades de la tarjeta. Esto era debido a que el número de tarjetas disponibles crecía rápidamente. En este contexto surgió Windows 95, que además imponía limitaciones de acceso al hardware. Afortunadamente, también proporcionaba las librerías DirectX y OpenGL. Éstas colocan una barrera de abstracción entre el hardware y la aplicación, de tal forma que el programador no tenía que lidiar con las diferencias entre las distintas tarjetas. Las amplias ventajas de esta aproximación hicieron que los estudios incorporaran estas librerías (externas) en sus juegos rápidamente. (Gómez Martín, 2007)

Sea cual sea la causa, poco a poco los estudios de desarrollo han ido aceptando la inclusión de código externo en forma de librerías o módulo completos comprados a terceros. Son lo que en inglés se conoce como COTS (Component of the shelf). (Gómez Martín, 2007)

4.1.2.1 OPENGL

OpenGL (Open Graphics Library) es una especificación estándar que define una API multilenguaje y multiplataforma para escribir aplicaciones que produzcan gráficos 2D y 3D. La interfaz consiste en más de 250 funciones diferentes que pueden usarse para dibujar escenas tridimensionales complejas a partir de primitivas geométricas simples, tales como puntos, líneas y triángulos. Fue desarrollada originalmente por Silicon Graphics Inc. (SGI) en 19922 y se usa ampliamente en CAD, realidad virtual, representación científica, visualización de información y simulación de vuelo. También se usa en desarrollo de video-juegos, donde compite con Direct3D en plataformas Microsoft Windows. (Barrios, 2009, pp. 40-41, 46-47)

Fundamentalmente OpenGL es una especificación, es decir, un documento que describe un conjunto de funciones y el comportamiento exacto que deben tener. Partiendo de ella, los fabricantes de hardware crean implementaciones, que son bibliotecas de funciones que se ajustan a los requisitos de la especificación, utilizando aceleración hardware cuando es posible. Dichas implementaciones deben superar las pruebas de conformidad para que sus fabricantes puedan calificar su implementación como conforme a OpenGL y para poder usar el logotipo oficial de OpenGL.

Hay implementaciones eficientes de OpenGL para Mac OS, Microsoft Windows, GNU/Linux, varias plataformas Unix y PlayStation 3. Existen también varias

implementaciones en software que permiten ejecutar aplicaciones que dependen de OpenGL sin soporte de aceleración hardware. Es destacable la biblioteca de software libre / código abierto Mesa 3D, una API de gráficos sin aceleración hardware y completamente compatible con OpenGL. Sin embargo, para evitar los costes de la licencia requerida para ser denominada formalmente como una implementación de OpenGL, afirma ser simplemente una API muy similar.

La especificación OpenGL era revisada por el OpenGL Architecture Review Board (ARB), fundado en 1992. El ARB estaba formado por un conjunto de empresas interesadas en la creación de una API consistente y ampliamente disponible. Microsoft, uno de los miembros fundadores, abandonó el proyecto en 2003.

El 21 de septiembre de 2006 se anunció que el control de OpenGL pasaría del ARB al Grupo Khronos. Con ello se intentaba mejorar el marketing de OpenGL y eliminar las barreras entre el desarrollo de OpenGL y OpenGL ES[62]. ARB se convirtió dentro de Khronos en el OpenGL ARB Working Group[63]. El subgrupo de Khronos que gestiona la especificación de OpenGL se denomina OpenGL ARB Working Group.

El gran número de empresas con variados intereses que han pasado tanto por el antiguo ARB como por el grupo actual han hecho de OpenGL una API de propósito general con un amplio rango de posibilidades.

Mark Segal y Kurt Akeley fueron los autores de la especificación original de OpenGL. Chris Frazier fue el editor de la versión 1.1. Jon Leech ha editado las versiones desde 1.2 hasta la presente 3.0.7

OpenGL tiene dos propósitos esenciales:

- Ocultar la complejidad de la interfaz con las diferentes tarjetas gráficas, presentando al programador una API única y uniforme.
- Ocultar las diferentes capacidades de las diversas plataformas hardware, requiriendo que todas las implementaciones soporten la funcionalidad completa de OpenGL (utilizando emulación software si fuese necesario).

[62] OpenGL ES (OpenGL for Embedded Systems) es una variante simplificada de la API gráfica OpenGL diseñada para dispositivos integrados tales como teléfonos móviles, PDAs y consolas de video-juegos. La define y promueve el Grupo Khronos, un consorcio de empresas dedicadas a hardware y software gráfico interesadas en APIs gráficas y multimedia.
[63] http://www.opengl.org/archives/about/arb/ consultado 7/May/2012

El funcionamiento básico de OpenGL consiste en aceptar primitivas tales como puntos, líneas y polígonos, y convertirlas en píxeles. Este proceso es realizado por una pipeline gráfica conocida como Máquina de estados de OpenGL. La mayor parte de los comandos de OpenGL bien emiten primitivas a la pipeline gráfica o bien configuran cómo la pipeline procesa dichas primitivas. Hasta la aparición de la versión 2.0 cada etapa de la pipeline ejecutaba una función prefijada, resultando poco configurable. A partir de la versión 2.0 algunas etapas son programables usando un lenguaje de programación llamado GLSL.

4.1.2.2 DIRECT3D

Direct3D es parte de DirectX (conjunto de bibliotecas para multimedia), propiedad de Microsoft. Consiste en una API para la programación de gráficos 3D. Está disponible tanto en los sistemas Windows de 32 y 64 bits, como para sus consolas Xbox y Xbox 360. (Barrios, 2009, p. 42)

El objetivo de esta API es facilitar el manejo y trazado de entidades gráficas elementales, como líneas, polígonos y texturas, en cualquier aplicación que despliegue gráficos en 3D, así como efectuar de forma transparente transformaciones geométricas sobre dichas entidades. Direct3D provee también una interfaz transparente con el hardware de aceleración gráfica.

Se usa principalmente en aplicaciones donde el rendimiento es fundamental, como los video-juegos, aprovechando el "hardware de aceleración gráfica"[64] disponible en la tarjeta gráfica.

El principal competidor de Direct3D es OpenGL, desarrollado por Silicon Graphics Inc.

Direct3D es uno de los múltiples componentes que contiene la API DirectX de Windows. Se le podría situar al nivel del GDI[65] de Windows, presentando un nivel de abstracción entre una aplicación de gráficos 3D y los drivers de la tarjeta gráfica. Con

[64] En informática, la aceleración por hardware es el uso del hardware para realizar alguna función más rápido de lo que es posible usando software ejecutándose en una CPU de propósito general. Ejemplos de aceleración por hardware incluyen aceleración de la funcionalidad de Bit blit en GPUs (del inglés Graphics Processing Unit).

[65] Graphics Device Interface, uno de los tres componentes o subsistemas de la interfaz de usuario de Microsoft Windows. Trabaja junto con el núcleo y la API de Windows.

arquitectura basada en el COM[66] de Microsoft, la mayor ventaja que presenta Direct3D frente al GDI es que Direct3D se comunica directamente con los drivers de pantalla, consiguiendo mejores resultados en la representación de los gráficos por pantalla que aquel.

Direct3D está compuesto por dos grandes APIs. El modo retenido y el modo inmediato. El modo inmediato da soporte a todas las primitivas de procesamiento 3D que permiten las tarjetas gráficas (luces, materiales, transformaciones, control de profundidad, etc). El modo retenido, construido sobre el anterior, presenta una abstracción de nivel superior ofreciendo funcionalidades pre-construidas de gráficos como jerarquías o animaciones. El modo retenido ofrece muy poca libertad a los desarrolladores, siendo el modo inmediato el que más se usa.

El modo inmediato de Direct3D trabaja fundamentalmente con los llamados dispositivos (devices). Son los encargados de realizar la renderización de la escena. El dispositivo ofrece una interfaz que permite diferentes opciones de renderización. Por ejemplo un dispositivo mono permite la renderización en blanco y negro mientras que un dispositivo RGB permite el uso de colores.

4.1.3 INTRODUCCION A GPGPU

Las unidades de procesamiento gráfico, GPU, se han convertido en una parte integral de los sistemas actuales de computación. El bajo costo y marcado incremento del rendimiento, permitieron su fácil incorporación al mercado. En los últimos años, su evolución implicó un cambio, dejó de ser un procesador gráfico potente para convertirse en un co-procesador apto para el desarrollo de aplicaciones paralelas de propósito general con demanda de anchos de banda de procesamiento y de memoria sustancialmente superiores a los ofrecidos por la CPU.

La rápida adopción de las GPU como computadora paralela de propósito general se vio favorecida por el incremento tanto de sus capacidades como de las facilidades y herramientas de programación. Actualmente la GPU se ha posicionado como una alternativa atractiva a los sistemas tradicionales de computación paralela.

[66] Component Object Model, plataforma de Microsoft para componentes de software introducida por dicha empresa en 1993. Esta plataforma es utilizada para permitir la comunicación entre procesos y la creación dinámica de objetos, en cualquier lenguaje de programación que soporte dicha tecnología.

4.1.3.1 GPU VS CPU

La evolución de los sistemas de computación con multiprocesadores ha seguido dos líneas de desarrollo: las arquitecturas *multicore* (*multi-núcleos*) y las arquitecturas *many-cores* (*muchos-núcleos* o *muchos-cores*)[67], Hoy en día el avance en entornos multi-core es lento, tedioso y muy especializado. En un futuro inmediato todo CPU será many-core ya no multi-core[68]. En el primer caso, los avances de la arquitectura se centraron en el desarrollo de mejoras con el objetivo de acelerar las aplicaciones, por ejemplo la incorporación de varios núcleos de procesamiento. Ante los límites físicos alcanzados por las computadoras personales, la industria tomó la idea a partir de las supercomputadoras existentes e incorporó procesadores a sus desarrollos. (Piccoli, 2012)

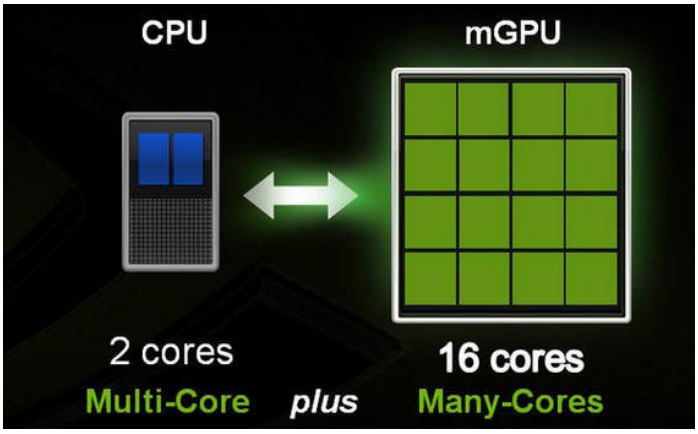

Figura 4.1 Multi-Core versus Many-Cores

Fuente: http://media.bestofmicro.com/GeForce-9300,3-0-163404-3.jpg

Es así que surgieron las computadoras 2, 3, 4, 8 y más procesadores por unidad central (*multicore*). Los *multicores* se iniciaron con sistemas de 2 núcleos y con cada generación se duplica este número, actualmente el procesador Intel *Core* i7 cuenta con versiones de 2 a 6 núcleos y el Intel Xeon E7 con 10 núcleos (hasta 20 threads). En (Hwu, 2008) (Sutter, 2005) se enuncian las características fundamentales de las nuevas arquitecturas, entre las cuales se encuentra el poder de procesamiento. En la actualidad ya no existen computadoras con un único procesador, es más ya está por desaparecer las de sólo 2 *cores*. (Piccoli, 2012)

[67] Many-cores ("muchos núcleos", en traducción literal), http://www.muycomputer.com/2011/09/16/intel-prepara-el-camino-hacia-procesadores-many-core Consultado 17/May/2012
[68] http://www.jdejuegos.com/x/articulos/077_detallesyconclusiones_larrabee.htm Consultado 17/May/2012

ARQUITECTURA MANY-CORES

En el caso de las arquitecturas *many-cores*, los desarrollos se centran en optimizar el desempeño de las aplicaciones. Dentro de este tipo de arquitectura se encuentran las Unidades de Procesamiento Gráfico (*Graphics Processing Unit*, GPU). En su inicio, la primer GPU, GeForce 256, estaba formada por un gran número de pequeños núcleos. Con cada nueva generación, los núcleos se duplicaron, la actual GTX 590 cuenta con 1024 *cores*. Desde sus comienzos, este tipo de arquitecturas *many-core* le llevaron ventaja a los procesadores de propósito general *multicores*, principalmente respecto a las mejoras en el desempeño, a partir del 2009 la diferencia de las velocidades provistas por ambas arquitecturas es de 10 a 1 (GPU-CPU), 1 TB versus 100GB. (Piccoli, 2012)

DISEÑO DE LAS ARQUITECTURAS

Uno podría preguntarse por qué existe una brecha tan grande entre el rendimiento de una CPU *multicore* y una GPU *many-core*. La respuesta la tienen las diferencias en la filosofía de diseño de ambos. Las optimizaciones de las arquitecturas *multicore* son hechas para proveer mejor desempeño a las soluciones secuenciales, es por ello que las optimizaciones se basan en, por ejemplo, proveer lógica de control compleja para la ejecución paralela de código secuencial o incluir memorias caché más rápidas y más grandes para disminuir la latencia de las instrucciones. Si bien la intención es muy buena, logrando tener computadoras rápidas, con 4 o más núcleos, estas no necesariamente mejoran el desempeño de las aplicaciones. (Piccoli, 2012)

Figura 4.2 Diseño de las Arquitecturas

Fuente: (Piccoli, 2012)

La filosofía de diseño de las arquitecturas *many-cores* como la GPU y sus avances están regidos por la industria del video-juego y su constante demanda de mejores prestaciones. La idea subyacente es optimizar el *throughput* de muchos *threads*

ejecutando en paralelo, de manera tal que si alguno de ello está esperando por al finalización de una operación, se le asigne trabajo y no permanezca ocioso. Las memorias caché son pequeñas, su función es ayudar a mantener el ancho de banda definido para todos los *threads* paralelos. Estas características determinan por ello que la mayor parte de la arquitectura está dedicada a cómputo y no a técnicas para disminuir la latencia. (Piccoli, 2012)

Otro punto de discrepancia entre ambos tipos de arquitecturas es el ancho de banda de la memoria, las GPU mantuvieron siempre una brecha en el ancho de banda diez veces superior al de las CPU contemporáneas. Esto obedece a que las arquitecturas de propósito general deben optimizar el ancho de banda para atender a todas las aplicaciones, operaciones de entrada/salida y funciones del sistema operativo coexistentes en el sistema. Por el contrario en la GPU con su modelo de memoria más simple y menor número de limitaciones, es más fácil lograr mayor ancho de banda de memoria. Uno de los chips de Nvidia la GTX 590 soporta alrededor de 327 GB/s y para los Intel Xeon E7 en *multicore*, el ancho de banda de la memoria es de 102GB/s, siendo inferior al provisto por la GPU. (Piccoli, 2012)

Como se desprende de lo expuesto, la filosofía de diseño para las arquitecturas de las CPU y las GPU son distintas, estas últimas son diseñadas como computadoras especializadas en cómputo numérico. Es de esperar que las aplicaciones secuenciales trabajen bien en las CPU, mientras que aquellas con intensivo cómputo numérico lo hagan mejor en la GPU. Poder contar con una arquitectura de trabajo híbrida permite aprovechar las ventajas de la CPU y las GPU. Esto es posible gracias a CUDA (*Compute Unified Device Architecture*) (Kirk, 2010) (NVIDIA., 2011) (Sanders, 2010), el cual fue presentado por Nvidia en 2007 y su principal característica es permitir el desarrollo de aplicaciones en un modelo de arquitectura CPU-GPU. (Piccoli, 2012)

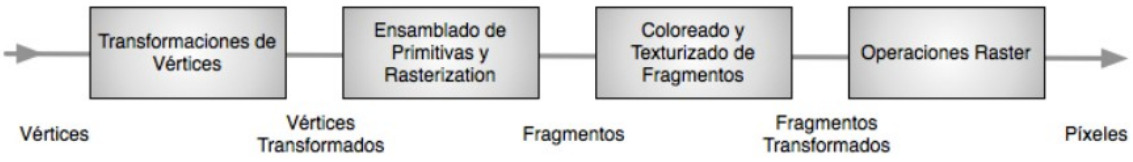

Figura 4.3 Pipeline gráfico de una GPU

Fuente: (Piccoli, 2012)

4.1.3.2 USO DE "COMPONENT OF THE SHELF" O COTS

En realidad, el uso de COTS es consecuencia de la imposibilidad de reutilización del código en un mercado que evoluciona tan rápidamente como el de los video-juegos. El problema de la reusabilidad es que ésta sólo es posible si el software desarrollado se construyó pensando en volver a ser usado (Tracz, 1995a); es decir, no vale con coger el código creado para el título anterior, y utilizarla directamente. Es necesario que ese código que se quiere volver a usar se creara con la reutilización en mente. (Gómez Martín, 2007)

Durante la década de los 90, cuando nosotros situamos la migración de los estudios desde el ensamblador hacia los lenguajes de alto nivel, el área de la reutilización estaba en auge. Se dedicó, y aún se dedica, mucho esfuerzo para generar aplicaciones que automaticen o al menos ayuden en la reutilización del código. Sin embargo, Tracz (1995b) nombra un estudio que revelaba que el coste de desarrollo del código se ve incrementado entre un 30 y un 50 por ciento si se implementa con miras a la reutilización. Además, el diseño de las clases nuevas hay que hacerlo con cautela, para no caer en la sobreingeniería (Kerievsky, 2002), lo que provocaría una pérdida aún más excesiva de recursos. (Gómez Martín, 2007)

Por poner un ejemplo de la razón de este incremento, se puede pensar en un juego que requiera un dado con los números del 1 al 6 en sus caras. La implementación de una tirada, requiere una simple línea en C/C++, utilizando la función rand(), el módulo y el incremento. Sin embargo, implementar un código reutilizable de un dado implica mucho más tiempo y dinero. En primer lugar, es posible que el dado deseado en un futuro no tenga seis sino más caras, por lo que el módulo o clase "CDado" deberá tener un parámetro para indicar el número de caras. Además, con un número de caras variable, lo que aparece en cada una de ellas debería ser configurable. El código de lo que antes era una mera línea, ahora se ha convertid en un fichero de definición y otro de implementación de una clase que habrá que programar y probar, para garantizar su correcto funcionamiento. Cuando este proceso esté completo (o mientras se completa), también habrá que escribir la documentación pertinente para facilitar la reutilización. (Gómez Martín, 2007)

Pero no todo son aspectos negativos; el notable incremento del coste en la escritura del código reutilizable se compensa con el abaratamiento de los desarrollos posteriores, que simplemente tienen que hacer de usuarios de ese software. De esta

forma el esfuerzo dedicado en la construcción de clases o módulos reutilizables se convierte en una inversión del estudio a medio o largo plazo. (Gómez Martín, 2007)

Sin embargo, el problema en la industria del video-juego es, precisamente, la dificultad de pensar a largo plazo, debido a que la ventana de mercado es muy corta. La causa es el rápido avance de las capacidades del hardware y el hecho de que los juegos que se lanzan al mercado deben aprovechar los últimos avances tecnológicos para resultar atrayentes al consumidor. (Gómez Martín, 2007)

Por lo tanto, existen dos barreras que frenan la generación, por parte de los estudios de juegos, de código altamente reutilizable:

- Motivo económico: la creación de código y componentes pensando en su reutilización es más costosa.
- El propio mercado: por dos razones; primero la creación de estos componentes generalmente retrasa la salida del primer título que los utiliza, algo que puede ser de vital importancia, al existir el riesgo de quedarse desfasado tecnológicamente; segundo, la rápida evolución del mercado hace que muchos componentes pensados para su reutilización queden obsoletos antes de poder volverlos a usar.

Únicamente los estudios grandes que disponen de varios proyectos en paralelo pueden plantearse la posibilidad de desarrollar internamente partes del juego que reutilizarán. En ese caso, al salir al mercado varios títulos en u corto espacio de tiempo, los componentes que se reutilizan no han quedado desfasados de un juego a otro. (Gómez Martín, 2007)

El resto de estudios, en general, no pueden hacer frente a la creación de componentes reutilizables. Eso, unido a la necesidad de reducir los tiempos de desarrollo, hace que las prácticas software de dichas empresas están migrando desde los desarrollos en los que todo el código era construido por el propio estudio, a la compra de componentes software implementados por terceros. (Gómez Martín, 2007)

De esta forma, surgieron empresas como RenderWare5, que desarrolla un motor gráfico independiente de la plataforma utilizado por más de 500 juegos entre los que destacan Grand Theft Auto: San Andreas o Call of Duty: Finest Hour, y como Havok que tiene una familia de utilidades, como un motor de animación o de física, y que se ha utilizado en juegos como Half-Life 2 o F.E.A.R.. Conviene también destacar el caso especial de idSoftware y Epic Games, dos estudios de juegos que, si bien la calidad de los mismos en cuanto a guión puede ser discutible, la calidad gráfica que consiguen es indiscutible. Sus juegos pueden entenderse como propaganda de un producto derivado

que también comercializan: la tecnología gráfica. Ambos estudios licencian sus motores gráficos a terceros, que los incorporan a modo de COTS en sus desarrollos. Por nombrar un ejemplo, Epic Games ha licenciado su motor a los estudio de Microsoft y Disney. (Gómez Martín, 2007)

4.1.4 ARQUITECTURAS DIRIGIDAS POR DATOS

Inicialmente, el tamaño de los juegos era muy pequeño, el contenido de la parte gráfica era mínimo, y lo único importante era sacar el máximo rendimiento a unos escasos recursos hardware para conseguir una aplicación que resultara mínimamente divertida. (Gómez Martín, 2007)

Con estos requisitos y máquinas (estamos hablando de la primera mitad de la década de los 80), los juegos se hacían principalmente en ensamblador, y los datos necesarios (como las posiciones de los muros del Comecocos, o la forma de pintar los fantasmas) estaban cableados en el código. Sea como fuere, este fuerte acoplamiento entre el comportamiento del juego (lógica) y los datos necesarios para su ejecución, fue separada mediante el uso de ficheros. Ya durante la década de los ochenta aparecen juegos cuyo núcleo principal (ejecutable), una vez cargado en memoria, leía ficheros de datos de los dispositivos de almacenamiento (disco duro, disquetes o incluso cintas), de los que obtenía la información para dibujar al personaje, el mapa del nivel o las posiciones de los enemigos. Esta separación permitía que el trabajo de desarrollo pudiera dividirse entre la tarea de los programadores, que eran los responsables últimos del ejecutable, y los artistas, que creaban los ficheros con la parte gráfica. (Gómez Martín, 2007)

Hoy por hoy esta división es tan habitual que ya nadie piensa que en otros tiempos no fue así, y se ha pasado a una separación aún mayor: parte del propio ejecutable, o mejor dicho, parte del código que controla el juego se ha sacado fuera del fichero ejecutable propiamente dicho. Esta separación es posible gracias al uso de librerías de carga dinámica proporcionadas por el sistema operativo[69] o mejor aún, el uso de lenguajes de scripts. (Gómez Martín, 2007)

Este modelo ha hecho que los programadores, al contrario de lo que sucedía a principios de la década de los 90, han dejado de ser el centro del desarrollo de juegos (Llopis, 2005a). Hoy en día, el rol del programador es muy distinto: el juego es creado

[69] Biblioteca de enlace dinámico o más comúnmente DLL (sigla en inglés de dynamic-link library) es el término con el que se refiere a los archivos con código ejecutable que se cargan bajo demanda de un programa por parte del sistema operativo.

por los artistas y diseñadores, que son los que construyen el contenido (en inglés, *game content*). De esta forma, su tarea se ha desplazado desde ser el centro del desarrollo a estar al servicio del resto del equipo, proporcionándole los medios necesarios para crear un buen juego. (Gómez Martín, 2007)

Debido a la separación clara entre código y datos, han surgido distintas áreas de programación claramente diferenciadas en los estudios: (Gómez Martín, 2007)

- Motor del juego: código que va en el juego final, pero que no es específico del juego en cuestión. Es la base de código sobre la que se sustenta el juego en sí; se apoya en la tecnología subyacente, y simplifica el desarrollo del resto de la aplicación. Además, en muchas ocasiones, permite la ejecución del juego en distintas plataformas, como PC y consolas. Su funcionalidad típicamente pasa por el motor gráfico (para presenta las escenas 2D y 3D), un motor de física o de colisiones, gestión del sonido, animación, lenguajes de scripts y red. Dada su complejidad, éste es el ámbito de actuación principal de los COTS.
- Código del juego (*game code*): parte del código muy específico de algún juego en particular. Esta parte del código es muy importante, y en muchos casos se realiza, al menos parte de ella, en lenguajes de script; por ejemplo, cómo reaccionan los personajes controlados por el computador ante ciertas situaciones, cómo se comporta la cámara, o cómo se almacenan los puntos del jugador.
- Herramientas: cuanto más orientada a datos es la arquitectura de un juego, más ficheros de datos hay que generar, y por lo tanto, más herramientas para ayudar en esa generación se necesitan. En cuanto a herramientas, pueden ser simples plug-in para otras aplicaciones (como Maya o 3DStudio Max), o programas complejos para edición de mapas (por ejemplo, Worldcraft/Hammer, UnrealEd y Sandbox). El lenguaje utilizado para crear estas herramientas no tiene por qué ser C/C++.

a) Captura del Laberinto del Sultán (Amsoft, 1984).

```
660 MOVE 320,300: DRAW 320,351,1
670 MOVER 9,0: DRAWR 43,-12,0:
    DRAWR 0,-4: DRAWR -43,-4:
    DRAWR 0,20: MOVER 0,-2:
    DRAWR -4,0: MOVER 0,-16:
    DRAWR 4,0
680 MOVER 4,0: DRAWR 0,16,1:
    MOVER 4,-2: DRAWR 0,-14: ...
```

b) Fragmento de código original del juego

Figura 4.4 Ejemplo de juego con datos acoplados en el código

Fuente: (Gómez Martín, 2007)

4.1.5 BUCLE PRINCIPAL

La ejecución de un juego está dirigida por lo que se conoce como el bucle principal (en inglés game loop), que es el responsable de la ejecución de cada una de las tareas requeridas en cada fotograma, y que van encaminadas a crear la ilusión de un entorno vivo y animado. A pesar de que la programación del bucle principal lleva poco tiempo, su importancia es vital, ya que es el eje conductor que va estableciendo qué módulos se ejecutan, cuándo y cada cuánto tiempo. El modelo de bucle principal utilizado define, por ejemplo, si la lógica avanza en intervalos constantes o no, si la entrada del usuario es analizada casi instantáneamente o su procesamiento es retrasado varios fotogramas, etc. (Gómez Martín, 2007)

La lista de tareas que debe realizar el bucle principal son:

- Gestión de la entrada del usuario: dado que un juego es una aplicación inherentemente interactiva, es importante que la lectura de la entrada se realice de forma correcta. Puede involucrar desde simplemente comprobar si se ha pulsado alguna tecla, a la gestión de complicados dispositivos de entrada. En vez de distribuir la lectura de la entrada por las tareas que dependen de ella (por ejemplo, mirar el estado de las teclas cuando se va a actualizar la posición del usuario y comprobar los eventos del ratón cuando se va a actualizar la cámara), se suelen analizar todos los dispositivos de entrada a la vez, almacenando el resultado de esas lecturas. Esos resultados son los que luego consultan las distintas tareas en el momento de su ejecución. De esta forma, se garantiza que la aplicación reaccionará de manera consistente en todo el fotograma. Así, la

lectura de la entrada pasa a ser una tarea por sí misma, que suele situarse al principio del bucle de la aplicación. En general, se ejecuta una vez en cada fotograma (o vuelta del bucle), aunque en juegos en los que la interacción no es tan importante (como los de estrategia) puede hacerse cada varios fotogramas.

- Gestión de la red: en juegos multijugador, la red puede verse como otra forma de entrada, por lo que en algún momento del bucle de juego se deben recopilar todos los mensajes de ésta y procesarlos.
- Simulación o actualización de la lógica del juego: es donde el juego realmente avanza. La simulación se encarga de decidir los comportamiento de los personajes (Inteligencia artificial o la abreviación en inglés IA), ejecutar las acciones de los mismos, actualiza el estado de los objetos, lanzar eventos que provocarán otras futura acciones, etc. También se encarga de que los personajes controlados por el jugador reaccionen ante los eventos recogidos en las fases anteriores.
- Simulación física y gestión de colisiones: se encarga de mover los objetos de acuerdo con la simulación física subyacente (velocidad y aceleración actual), y puede encargarse de actualizar los sistemas de partículas o el avance de las animaciones de los personajes. La simulación física de juegos sencillos puede llevarse a cabo dentro de la etapa anterior de simulación de la lógica del juego, aunque hoy por hoy se tiende a separar. La ejecución de la física puede provocar que los objetos simulados colisionen entre sí. En ese caso, se debe reaccionar ante esas colisiones, tanto siguiendo las propiedades físicas de la naturaleza (por ejemplo, haciendo que una caja rebote en una pared) como las lógicas (haciendo que el jugador pierda energía si una bala colisiona contra él, etc.).
- Dibujado del mundo: todas las etapas anteriores van encaminadas a calcular el nuevo estado del entorno virtual: las nuevas posiciones de los objetos, sus estados de animación, etc. En esta última fase del bucle principal, este estado recién calculado es utilizado para dibujar la nueva escena.

Existen algunas otras tareas de menor importancia que no han sido situadas en ninguna de las fases del bucle, y que, no obstante, no deben olvidarse, como son la actualización del sistema de sonido para que haga todas las mezclas necesarias, el procesado de los eventos enviados por el sistema operativo a la aplicación/ventana (para

reaccionar, por ejemplo, ante cambios de contexto, activación del protector de pantalla, o entrada en modo ahorro de energía), o el envío de paquetes de red construidos durante la simulación. (Gómez Martín, 2007). En general, se puede decir que cada vuelta del bucle al final tiene como resultado la creación de un nuevo fotograma[70]. Generalmente, se utiliza como parámetro para medir el rendimiento el número de vueltas por segundo que se ejecuta el bucle, o lo que es lo mismo, el número de fotogramas por segundo (FPS) que se dibujan en pantalla. Como límite inferior para poder decir que la aplicación es interactiva, suelen aceptarse los 16 fotogramas por segundo (Valente, 2005), aunque el óptimo suele estar entre 50 y 60 (que suele coincidir, o al menos acercarse, a la velocidad de refresco del monitor). (Gómez Martín, 2007)

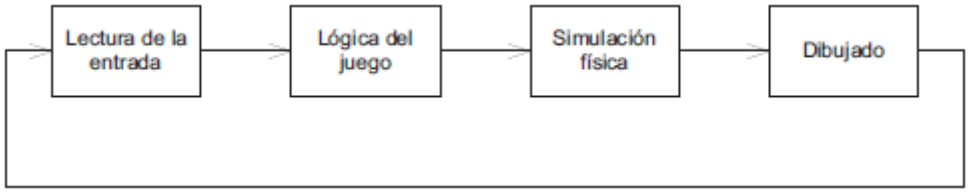

Figura 4.5 Esquema de bucle principal básico

Fuente: (Gómez Martín, 2007)

4.1.5.1 BUCLES MULTIHEBRA

Hoy en día, el hardware ha avanzado lo suficiente como para permitir la ejecución de hebras distintas con paralelismo real. Por ejemplo, en los computadores personales o PCs empieza a ser habitual computadores capaces de ejecutar dos o cuatro hebras; la consola de Microsoft XBox 360 tiene un procesador con tres núcleos, cada uno con hyperthreading, y la Play Station 3 de Sony tiene un procesador central y siete procesadores satélites. (Gómez Martín, 2007)

Esto hace que los arquitectos de software deban plantearse la paralelización[71] de todos los módulos del juego, para aprovechar todas estas capacidades. En el caso del computador personal o PC y de la consola XBox, esta paralelización es relativamente sencilla (son sistemas multiprocesador simétricos, SMP). Sin embargo, en la Play Station 3, las divisiones de los distintos módulos para su paralelización son más complicadas. (Gómez Martín, 2007)

[70] Cada una de las imágenes individuales captadas por cámaras de video y registradas analógica o digitalmente.
[71] Ejecución de más de un cómputo (cálculo) al mismo tiempo usando más de un procesador en una computadora, la paralelización es un factor básico para tener un alto desempeño en los equipos de cómputo y obtener un alto rendimiento o mayor velocidad al ejecutar un programa.

4.1.6 ENTIDADES DEL JUEGO

Un juego, o mejor, un entorno virtual gira en torno a la interacción entre el usuario y el mundo. Debido a esto, podemos decir que existen dos categorías de objetos dentro del entorno: los objetos dinámicos y los estáticos. Los objetos estáticos son inmutables, y el usuario no puede hacer nada con ellos, excepto verlos y chocarse contra ellos. Ejemplos son muros, rocas, etc. (Gómez Martín, 2007)

Los objetos dinámicos son aquellos que hacen del entorno un mundo vivo, que cambian sus propiedades durante la ejecución de la aplicación, ya sea debido a la interacción del usuario o de otros objetos. En el contexto de los juegos, estos tipos de objetos suelen conocerse como entidades, aunque también pueden denominarse objetos de juego (en inglés, game objects) (Plummer, 2004). (Gómez Martín, 2007)

No existe un consenso en la definición de lo que es una entidad, pero aquí nos quedaremos con la dada por Llopis (2005b), que la define como una pieza autocontenida de contenido interactivo. (Gómez Martín, 2007)

Dada la naturaleza de las entidades, es natural aplicar el paradigma de orientación a objetos para codificar su comportamiento. De esta forma, es habitual disponer de una jerarquía de clases para representar los distintos tipos de entidades. Por ejemplo, podemos tener la clase CEntity, de la que hereda una clase CMoveable, que represente a cualquier entidad que puede moverse por el entorno. Por debajo de ésta, tenemos CVehicle, CHuman, CGun, etc. De la clase que representa cualquier avatar, heredarían los distintos enemigos y el propio jugador. (Gómez Martín, 2007)

4.1.7 ARQUITECTURA BASADA EN COMPONENTES

La arquitectura por componentes se caracteriza por:

- Descomposición en componentes funcionales.
- Reutilización del software.
- Conjunto de propiedades.
- Se tiene de manera externa una especificación.

Figura 4.6 Arquitectura basada en componentes

Fuente: http://upload.wikimedia.org/wikipedia/commons/9/9b/Porcomponentes.jpg

En la programación orientada a objetos se aplica el diseño de las entidades basándose en el mecanismo de herencia ya que es muy intuitivo. Sin embargo, existen una serie de problemas que están provocando un cambio en su forma de programación: (Gómez Martín, 2007)

- Descomposición: existen infinidad de formas de descomponer el sistema de objetos de un juego, utilizando distintas clasificaciones. El problema, ya de por sí complicado, de decidir qué clasificación es más adecuada se ve acrecentado con la naturaleza cambiante del juego. Los diseñadores de contenido, según avanza el desarrollo, van tomando decisiones que, de haber sido conocidas al principio, habrían llevado a estructurar la jerarquía de otra forma.
- Código poco flexible: enlazado con los comentarios anteriores, una jerarquía de clases es muy complicada de cambiar. Por lo tanto, ante apariciones de nuevos tipos de entidades no planificados, la solución adoptada suele ser el uso de una sola clase para entidades parecidas pero distintas, la sobrecarga de métodos que un buen diseño nunca habría permitido, etc.
- Clases base grandes: según van creciendo las clases (y entidades) del juego, la jerarquía se hace más grande. En muchas ocasiones, los programadores se

encuentran con que la funcionalidad que necesitan para una clase profunda en la jerarquía, ya ha sido implementada en otra clase hermana.

- Clases difíciles de entender: para conseguir entender el comportamiento de una clase, el programador tiene irremediablemente que comprender primero las clases de toda la línea sucesoria desde la clase base.
- Extensión del comportamiento: en una jerarquía como la de las entidades, es muy habitual que las clases derivadas no se limiten a sobrescribir el comportamiento de una serie de métodos heredados, sino que lo extiendan. Eso, irremediablemente, pasa por invocar en algún momento de la implementación del método al mismo método de la clase padre. Dado que el lenguaje no proporciona ningún mecanismo para obligar al programador de la clase hija llamar recursivamente a la implementación de la padre, se presta mucho a olvidos únicamente detectables a través de la depuración. Una solución posible es utilizar el patrón Template Method (Gamma, 1995), pero en la práctica, dado que la jerarquía suele ser profunda y que el número de métodos por clase que lo requerirían es grande, utilizarlo en todos ellos supondría una sobrecarga de nombres difícil de justificar.
- Aparición de herencia en diamante: la jerarquía de clases al fin y al cabo es una clasificación de las entidades utilizando un cierto criterio. En ocasiones surgen entidades que, por su naturaleza, deben formar parte de dos o más categorías. La implementación obvia es hacer que la nueva clase herede de las dos categorías, implementando la herencia en diamante. Para implementarla, se utiliza la herencia virtual disponible en C++ pero que es en general mal entendida por los programadores, introduce ineficiencias (Lippman, 1996) y acarrea problemas relacionados con la herencia múltiple (Meyers, 1997, Item 43).
- Mayor tiempo de compilación: la herencia es uno de los mecanismos de C++ que provocan mayor acoplamiento entre clases, no sólo desde el punto de vista lógico, sino también físico. Este último tipo de acoplamiento es el que obliga al compilador a tener que procesar todos los ficheros de cabecera de las clases padre, y al enlazador acceder a todos los ficheros objeto para resolver las dependencias. Eso provoca un aumento significativo del tiempo de generación del ejecutable final (Lakos, 1996). (Gómez Martín, 2007)

Una solución es dividir la entidad en un conjunto de funcionalidades independientes que se separan en distintas clases. La entidad u objeto del juego pasa así a ser un mero contenedor de punteros a esas clases, como por ejemplos las que gestionan el objeto gráfico de la entidad, el objeto físico, el emisor de sonido o el controlador de su lógica (o inteligencia artificial). Para poder comunicar cada una de estas clases, existen dos alternativas principales: (Gómez Martín, 2007)

- Que cada una de las clases que implementan la funcionalidad pueda tener un puntero al resto de clases con las que se tiene que comunicar. Por ejemplo, la inteligencia artificial establecerá distintas fuerzas en el objeto físico, por lo tanto tendrá internamente un puntero a él. De acuerdo con la simulación física, ésta actualizará la posición en la que el objeto gráfico tendrá que dibujar el modelo, por lo que el primero tendrá un puntero al segundo.
- Utilizar un sistema de mensajes genérico, de tal forma que uno de los objetos pueda enviar mensajes al resto de ellos informando, por ejemplo, de un cambio en la posición de la entidad.

La creación de una entidad pasa a convertirse en la creación e inicialización de los objetos que implementan las distintas funcionalidades que forman la entidad. Si una entidad no necesita alguna de la funcionalidad (como por ejemplo el emisor de sonido), ese puntero apuntará a NULL. (Gómez Martín, 2007)

La composición viene a evitar un problema asociado con la centralización de todos los objetos del juego en una única clase, hoy día la industria del video-juego está basada en componentes (o librerías) comprados a terceros. Dado que la gran mayoría de estas librerías son utilizadas desde las entidades, sus programadores necesitan tener un gran dominio de las mismas. La descomposición de la entidad en funcionalidades disjuntas permite también la especialización de los programadores, de forma que únicamente los responsables de la programación de una funcionalidad determinada deberán conocer a la perfección la librería correspondiente. (Gómez Martín, 2007)

La generalización del método de composición consiste en hacer que todas las clases que implementan las distintas funcionalidades, hereden de una misma clase común, que representa un componente genérico. La clase que representa la entidad, en vez de tener una serie de punteros como antes, es un contenedor de componentes genéricos (West, 2006). La definición de una entidad consiste en listar de qué componentes estará formada, y cómo configurarlos. La ventaja adicional es que esos componentes se pueden enganchar y desenganchar de la entidad dinámicamente,

dependiendo del momento del juego. El modelo de componentes tiene dos ventajas adicionales: permite almacenar todos los componentes del mismo tipo consecutivos en memoria, lo que agiliza sus recorridos, y la división hace que sea más fácil manejar diferentes hebras, cada una encargándose de una funcionalidad particular. (Gómez Martín, 2007)

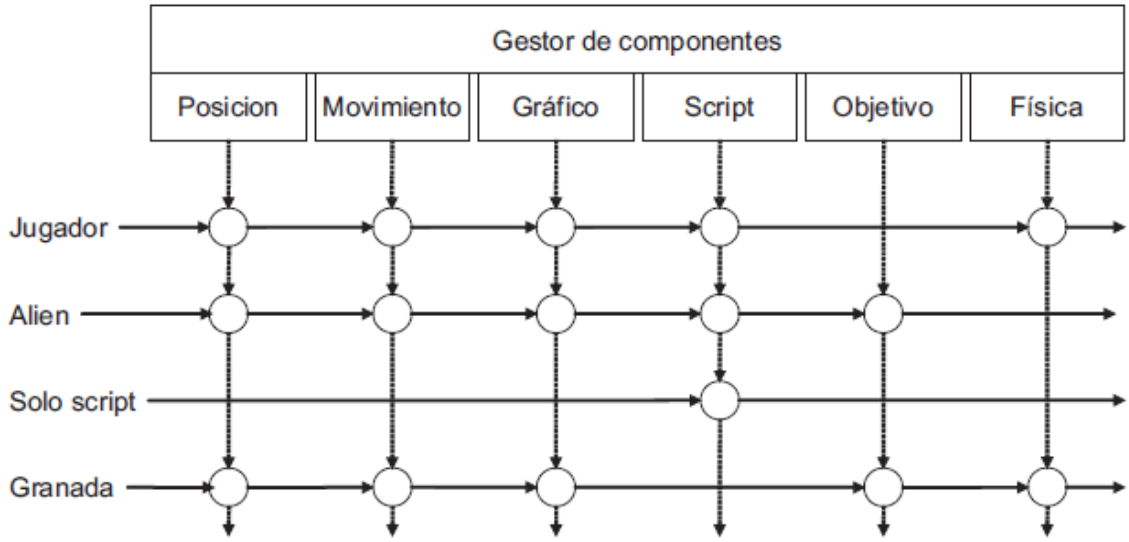

Figura 4.7 Estructura de objetos utilizando componentes (West, 2006).

Fuente: (Gómez Martín, 2007)

4.1.8 LENGUAJES DE SCRIPT

La evolución de los juegos dirigidos por datos que guardan en ficheros todos los recursos gráficos fue sacar fuera del ejecutable parte del comportamiento del juego. Ya hemos visto que una alternativa es dividir el juego en dos partes, el motor de juego y el código específico. Fijando una comunicación entre ambos componentes, se consigue separarlos de tal manera que la parte específica puede ir en una librería de vinculación dinámica (DLL). De esta forma, el equipo de desarrollo puede estructurarse más fácilmente, se reducen las restricciones físicas entre partes del código fuente (Lakos, 1996), lo que reduce los tiempos de compilación y enlazado, y se permite la posterior modificación de parte del juego, ya sea por parte del propio estudio de desarrollo, o por aficionados. (Gómez Martín, 2007)

El uso de lenguajes de script va un paso más allá. En vez de sacar el código específico del juego utilizando librerías dinámicas, se saca el propio código. El motor del juego se convierte así en un intérprete del lenguaje de script particular, y lo invoca cuando lo necesita. Dependiendo del juego, el uso del lenguaje de script será más o

menos anecdótico. Por ejemplo, algunos juegos lo pueden utilizar únicamente para el control de escenas entre niveles o lanzamiento de algún comportamiento predefinido, mientras que otros implementan toda su lógica y comportamiento utilizándolos. (Gómez Martín, 2007)

La decisión de dónde poner la frontera entre el motor del juego y los scripts es importante, y afecta al API resultante entre ambas partes y al rendimiento del juego, una de las desventajas de los lenguajes de scripts es que su ejecución es más lenta que la del código compilado. Por lo tanto, si se delega mucha funcionalidad en el lenguaje de script, se corre el riesgo de ralentizar demasiado el juego. Si, por el contrario, se cede muy poco control al script, se van perdiendo todas las ventajas, ya que muy poco código podrá aprovecharse. (Gómez Martín, 2007)

Desde el punto de vista arquitectónico, el código que maneja un personaje no jugador (en inglés non-player character, NPC) era lo último que quedaba dentro del ejecutable, al haber sacado fuera tanto su modelo y animaciones como las texturas. Por lo tanto, con este último paso se consigue una entidad completamente dirigida por datos, que puede empaquetarse en un único fichero y distribuirlo independientemente del juego. (Gómez Martín, 2007)

4.1.8.1 VENTAJAS Y DESVENTAJAS

Una de las principales ventajas de los lenguajes de script es que se reduce significativamente el tiempo del ciclo de desarrollo editar - compilar – enlazar - ejecutar. Esto es debido a que normalmente un cambio de un fichero de script no requiere la compilación del código del juego y su enlazado con todas las librerías, sino que basta con un simple volcado a disco del cambio. De esta forma, el ciclo que puede durar más de dos minutos en el primer caso se reduce a unos segundos en el último. Algunos motores de juegos están incluso preparados para recargar el script *al vuelo* (o en caliente), de tal forma que no es necesario parar el juego y volverlo a lanzar. (Gómez Martín, 2007)

Esta reducción en el tiempo de ciclo es posible debido a que normalmente el lenguaje de script es interpretado, es decir, durante la ejecución del juego, el fichero se analiza y se ejecutan sus instrucciones interpretándolas una a una. Eso trae consigo una de las desventajas principales de este tipo de lenguajes: su ejecución es mucho más lenta que la un lenguaje compilado. En general, un lenguaje de script interpretado no puede competir en rendimiento con el binario creado a partir del código en C++. Por eso debe

elegirse con cuidado qué partes del juego van a ser programadas utilizando scripts; normalmente serán aquellas que definen el comportamiento de alto nivel, y que por lo tanto no necesitan ejecutarse a menudo. (Gómez Martín, 2007)

Otra causa del menor rendimiento del código de scripts es que los lenguajes poseen características avanzadas que suponen un consumo de CPU para su control. Por ejemplo, la recolección de basura, muy habitual en este tipo de lenguajes, hace que el motor de scripts dedique parte de su tiempo a su gestión. Sin embargo, estas características avanzadas disponibles en los lenguajes de script y que no están en C++ son también una ventaja, ya que facilitan el desarrollo, reduciendo significativamente la duración del proyecto, y por tanto, pueden ser un punto decisivo a la hora de seleccionar uno u otro lenguaje. Si el desarrollo con lenguajes de scripts es lo suficientemente fácil, aparece otra ventaja más: los propios diseñadores o creadores de contenidos del juego pueden modificarlos para cambiar parte del comportamiento. (Gómez Martín, 2007)

4.2 JUEGOS 2D Y 3D

El primer avance en la computación gráfica fue la utilización del tubo de rayos catódicos. Hay dos acercamientos a la gráfica 2d: vector y gráficos raster. La gráfica de vector almacena datos geométricos precisos, topología y estilo como posiciones de coordenada de puntos, las uniones entre puntos (para formar líneas o trayectos) y el color, el grosor y posible relleno de las formas. La mayor parte de los sistemas de vectores gráficos también pueden usar primitivas geométricas de forma estándar como círculos y rectángulos etc. En la mayor parte de casos una imagen de vectores tiene que ser convertida a una imagen de trama o raster para ser vista. [72]

Los gráficos de tramas o raster (llamados comúnmente Mapa de bits) es una rejilla bidimensional uniforme de pixeles. Cada pixel tiene un valor específico como por ejemplo brillo, transparencia en color o una combinación de tales valores. Una imagen de trama tiene una resolución finita de un número específico de filas y columnas. Las demostraciones de computadora estándares muestran una imagen de trama de resoluciones como 1280 (columnas) x 1024 (filas) pixeles. Hoy uno a menudo combina la trama y lo gráficos vectorizados en formatos de archivo compuestos.

Los video-juegos de plataformas en 2D se caracterizan por usar scroll de la pantalla, tanto horizontal como Super Mario Bros o Sonic the Hedgehog como en

[72] Basado en http://es.wikipedia.org/wiki/Computaci%C3%B3n_gr%C3%A1fica Consultado 15/12/2011

vertical como Bubble Bobble y Donkey Kong. Es el estilo de plataformas más clásico y es el más característico de las videoconsolas de 8 bits y 16 bits.

Una de las razones para comenzar a desarrollar primero video-juegos 2D, es que los conceptos involucrados son mucho más simples y fáciles de asimilar, además lo más probable es que varios de estos conceptos ya sean conocidos por quienes lleven un tiempo programando o trabajando con computadores, otra ventaja, es que obtendremos resultados más rápidos, a diferencia del desarrollo de video-juegos 3D, que involucra conocer tópicos más avanzados de matemáticas y de programación. Pero tampoco hay que asustarse, si queremos luego dar el próximo paso a las 3D, ya que hay que recordar siempre que todo lo podemos estudiar y aprender, nada es imposible. A continuación conoceremos la terminología básica, que nos hará comprender de mejor forma los conceptos gráficos involucrados en un video-juego.

4.2.1 TÉCNICAS

Estos términos son aplicados básicamente al desarrollo de aplicaciones 2D y algunos son la base para las 3D.

4.2.1.1 SPRITES

Un sprite es cualquier objeto que aparece en nuestro video-juego. Normalmente tiene asociado algunos atributos, siendo los más básicos una imagen y una posición. (Albornoz Figueroa, 2007)

Figura 4.8 Sprite del personaje Donkey Kong

Fuente: (Albornoz Figueroa, 2007)

Los sprites pueden ser estáticos o dinámicos. Al hablar de sprite estático, nos referimos a un objeto que no cambiará su posición durante el video-juego, por ejemplo la imagen de una llama, piedra, etc. En cambio un sprite dinámico, es aquel que tendrá algún tipo de movimiento, el cual puede ser realizado por el usuario o por el computador, por ejemplo el sprite de un jugador o de los enemigos. (Albornoz Figueroa, 2007)

Representar un sprite en memoria puede ser realizado de muchísimas formas, y puede depender mucho del gusto del programador y su experiencia. Normalmente se almacenan sus atributos en una estructura o clase. (Albornoz Figueroa, 2007)

4.2.1.2 ANIMACIÓN DE SPRITES

Esto es bastante simple, aquí aparece el concepto de *fotograma* o cuadro, que corresponde a una de las imágenes que forman la animación completa de un sprite. Podemos pensar en una secuencia de imágenes que son dibujadas rápidamente, que engañan al ojo humano dando el efecto de un movimiento fluido. Es el mismo método que se utiliza para los dibujos animados. (Albornoz Figueroa, 2007)

El número de fotogramas dibujados en un periodo de tiempo se conoce como tasa de fotogramas, siendo un aspecto muy importante en la animación, ya que de esto depende la calidad de animación que obtendremos. Si cada uno de los fotogramas se muestra muy rápido o muy lento, la ilusión del movimiento se perderá y el usuario verá cada uno de los fotogramas como una imagen separada. (Albornoz Figueroa, 2007)

Una animación muy común puede ser la de un personaje caminando, saltando, o realizando cualquier otra acción. Esto también debe ser representado de alguna forma inteligente en nuestro video-juego, utilizando algún tipo de estructura de datos adecuada, como un vector o lista, normalmente el sprite tendrá como atributo una lista de animaciones (las acciones del personaje, enemigo, etc.), donde cada animación corresponderá a otra lista con cada uno de los fotogramas. (Albornoz Figueroa, 2007)

La totalidad de los fotogramas que forman un personaje, suele almacenarse en un solo archivo de imagen, a esto se le llama sprite sheet, es decir, una imagen donde tenemos secuencias de fotogramas para las diferentes acciones que puede realizar un personaje en un video-juego. (Albornoz Figueroa, 2007)

Figura 4.9 Ejemplo de una animación por sprites

Fuente: (Albornoz Figueroa, 2007)

4.2.1.3 TRANSPARENCIAS

COLOR KEYS

Cuando mostramos una imagen en pantalla, esta siempre se verá como un cuadrado o rectángulo, pero sabemos que dentro tiene la representación de un objeto, personaje, enemigo, ítem, etc. Al dibujar esta imagen queremos ver solo la forma que representa, para esto debemos elegir algún color transparente, pintar con ese color las zonas que no queremos que aparezcan y luego dibujar la imagen ignorando dicho color. (Albornoz Figueroa, 2007)

Normalmente se utiliza el color magenta (255, 0, 255), como color de transparencia, ya que es poco probable que se encuentre en una imagen.

Dependiendo de la API gráfica que utilicemos para desarrollar nuestro video-juego, tendremos una función que realice esta tarea, la de descartar un color determinado al dibujar la imagen, pero no solo usar un color como el magenta es la solución para no ver una imagen en su forma original, también podemos hacer uso de las propiedades del formato gráfico PNG (Portable Network Graphics), ya que este formato guarda un canal alpha con las partes transparentes de la imagen. (Albornoz Figueroa, 2007)

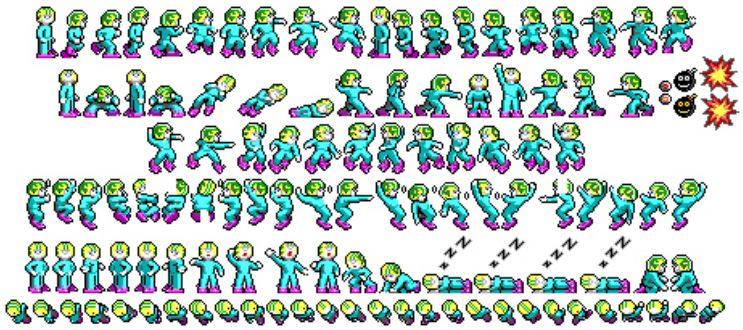

Figura 4.10 Ejemplo de una animación por sprites

Fuente: (Albornoz Figueroa, 2007)

4.2.1.4 MÁSCARA

Existe otro método para ignorar un color en una imagen, el uso de una máscara que se antepone al mapa de bits original, es decir, debemos tener otro mapa de bits que indica cuales son los pixeles transparentes. El color que indica la transparencia es el

negro, y el color blanco la figura que queremos mostrar. Por lo tanto, al dibujar la imagen, recorreremos cada pixel de la máscara, y si este color es blanco, mostraremos el pixel del bitmap original que se encuentra en esa posición. (Albornoz Figueroa, 2007)

Figura 4.11 Ejemplo de Máscara

Fuente: (Albornoz Figueroa, 2007)

4.2.1.5 TRANSFORMACIONES

Cuando dibujamos un sprite en pantalla, tenemos la posibilidad de aplicar algunas transformaciones, las más usadas son las siguientes:

TRANSLATION

Esta es la transformación más simple, corresponde a cambiar la posición del sprite para luego dibujarla en otro lugar de la pantalla, dando el efecto de movimiento, sin duda lo más usado en cualquier video-juego. Y se resume en asignar una nueva posición a las coordenadas (x, y) del sprite.

ROTATION

Otra transformación típica, que consiste en girar el sprite un número determinado de grados. Se usa para mostrar objetos vistos desde otro ángulo, por ejemplo en el video-juego Micro Machines, cuando doblamos cambiamos el ángulo del auto y luego este cambio lo vemos reflejado en la pantalla gracias a la rotación.

También lo podríamos usar para realizar efectos para una presentación, efectos durante el video-juego, etc. Además al rotar una imagen debemos saber con respecto a que punto lo haremos (pivote), por ejemplo en la siguiente imagen realizamos unas rotaciones con respecto al centro de la imagen.

SCALING

Otra transformación muy usada en algunos video-juegos, consiste en escalar una imagen, es decir, cambiar su tamaño (normalmente de forma proporcional), ya sea un aumento o una disminución. Se puede usar para dar un efecto de profundidad, es decir,

si el objeto está más alejado de nosotros lo dibujaremos más pequeño, en cambio si está muy cerca de nosotros, lo dibujaremos de un tamaño más grande.

Suele usarse también en conjunto con la rotación para realizar algunos efectos. Al utilizar esta transformación, hablamos de *scale factor*, es decir, un valor que indica el porcentaje de escalado, donde el valor 1 corresponde al tamaño normal.

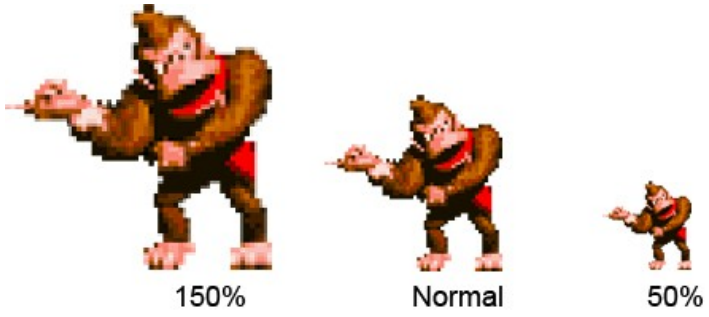

Figura 4.12 Ejemplo de Scaling

Fuente: (Albornoz Figueroa, 2007)

En la imagen anterior nos damos cuenta que al aumentar su tamaño, se produce un efecto no deseado en los bordes de la imagen, conocido como *aliasing*, es decir, las líneas o curvas que forman los bordes de la imagen se ven discontinuas (en forma de escalera). Para esto existe una solución, el antialiasing, una técnica que permite suavizar todos los bordes y así disminuir el efecto de escalera. En toda API gráfica encontraremos una función que realice esta tarea.

FLIPPING

El flipping es un tipo de transformación especial para realizar algunos efectos, básicamente existen dos tipos, Vertical Flip que corresponde al efecto que se produce cuando se refleja un objeto en el agua y Horizontal Flip que corresponde al reflejo de un objeto en un espejo. Con la siguiente imagen quedará mucho más claro.

Figura 4.13 Ejemplo de Flipping

Fuente: (Albornoz Figueroa, 2007)

4.2.1.6 ACTUALIZACION EN PANTALLA

ALPHA BLENDING

Alpha blending o mezclado alpha es una técnica para crear transparencias en una imagen con respecto al fondo. Para lograr esto se agrega un cuarto canal a un color, llamado canal alpha.

Ahora los colores de una imagen se representarán como RGBA. Cada valor alfa de un pixel representa su grado de opacidad, donde un valor cero indica un 100% de transparencia, y a medida que aumentamos su valor se irá haciendo más opaco. El rango va desde 0 a 255.

Para crear un canal alpha en una imagen, tenemos dos opciones, una es almacenar esta información con algún software de edición de imágenes (Photoshop) y en un formato apropiado. (PNG, Portable Network Graphics), o hacerlo posteriormente en el mismo video-juego, utilizando alguna función de la API que nos permita poner un grado de transparencia para la imagen.

El funcionamiento de esta técnica es relativamente simple, ya que se realiza una media ponderada de cada pixel que forma la imagen y los pixeles que conforman el fondo.

Figura 4.14 Ejemplo de Alpha blending

Fuente: (Albornoz Figueroa, 2007)

Esta técnica se puede aplicar a un bitmap completo o solo a un grupo de pixeles de la imagen. Es utilizado para hacer una variedad de efectos en video-juegos, por ejemplo puede usarse para dar un carácter de fantasma a cierto personaje, para hacer un

menú transparente, escribir texto sobre un rectángulo transparente, etc. (Albornoz Figueroa, 2007)

SCREEN BUFFER

El screen buffer es simplemente un área de la memoria de video, donde podemos dibujar algo que se mostrará en la pantalla. Si queremos mostrar un pixel, debemos tener un puntero a la dirección de memoria de video y calcular la posición (offset, desplazamiento) donde queremos dibujar el pixel y luego en esa posición copiar el byte o los bytes que representan el color. (Albornoz Figueroa, 2007)

SURFACE

Este término es la base de la programación 2D. Básicamente es una zona de memoria que puede estar en la RAM o Video RAM y es usada para almacenar un bitmap. Si queremos ver esto en el ámbito de la programación, una Surface es una estructura o clase que tiene como mínimo las misma propiedades de un archivo de imagen, es decir, tendrá un arreglo lineal de datos que contiene cada uno de los pixeles que forma la imagen y además el ancho y alto del bitmap. Dependiendo de la biblioteca gráfica que se use, pueden existir más atributos. (Albornoz Figueroa, 2007)

Una Surface se podría confundir con el concepto de Sprite, pero en general un Sprite tiene como atributo al menos una Surface. (Albornoz Figueroa, 2007)

BLITTING

El blitting es una operación para transferir un bloque de bytes (surface) de un sector de la memoria a otra. Viene del término Blit (Block Image Transfer, Transferencia de Imagen por Bloque) y es una forma de renderizar (dibujar) sprites con o sin transparencias sobre un fondo. Esta técnica puede ser acelerada por hardware, lo cual ayuda a incrementar bastante el rendimiento, de hecho esta operación es una de las más críticas en cualquier video-juego y es la que más usaremos. (Albornoz Figueroa, 2007)

Gracias al blitting podemos armar una escena en un video-juego, por ejemplo en la siguiente imagen vemos que a partir de dos superficies, una con los sprites de los personajes y otra con un fondo, armamos a través de blits una tercera superficie que contiene la escena final. (Albornoz Figueroa, 2007)

En una operación de blitting, podemos especificar en que posición de la superficie destino queremos copiar la superficie origen, incluso que parte de la superficie origen queremos copiar. De esta forma podemos colocar en cualquier posición, superficies dentro de otras. La forma de especificar las superficies origen/destino y sus posiciones, depende de la API gráfica que se esté utilizando. Siempre se nos proveerá de una función de blitting, ya que todas las bibliotecas disponen de al menos una. (Albornoz Figueroa, 2007)

DOUBLE BUFFERING

Para entender esta técnica, antes que todo debemos saber que el monitor ya sea CRT (Tubo de rayos catódicos) o LCD (Pantalla de cristal líquido), no dibuja la imagen en la pantalla instantáneamente, sino que lo hace pixel a pixel, partiendo desde la esquina superior izquierda hasta la esquina inferior derecha. Esto se realiza a una velocidad bastante rápida que el ojo humano no percibe, y a este tiempo, como vimos al principio, se le conoce como frecuencia de refrescado. (Albornoz Figueroa, 2007)

Cuando queremos mostrar una imagen en la pantalla, como ya sabemos colocamos los pixeles de la imagen en la memoria de video y durante el tiempo de refrescado el monitor leerá la información que hay en la Video RAM y la mostrará en la pantalla. (Albornoz Figueroa, 2007)

Ahora imaginemos que queremos mover un objeto por la pantalla, los pasos son simples, primero dibujamos la imagen en una posición, luego la borramos, y la dibujamos en la nueva posición, pero aquí aparecen algunos problemas, si a la mitad del refrescado cambiamos la imagen en la memoria de video, el monitor ahora obtendrá otra información de la imagen y probablemente la parte superior e inferior de la imagen no correspondan, es decir, veremos imágenes superpuestas y además un molesto parpadeo, una posible solución a esto, es esperar a que el monitor termine de refrescar la pantalla, para luego escribir en la Video RAM la nueva imagen. La mayoría de las APIs gráficas disponen de alguna función que espera el refrescado del monitor, evitando así el parpadeo y la superposición de imágenes, pero no del todo. (Albornoz Figueroa, 2007)

Para evitar totalmente el parpadeo, usamos esta técnica llamada Double Buffering, que consiste en tener dos áreas de memoria en la RAM (o Video RAM). Una de estas zonas se conoce como front-buffer, y corresponde a la que se muestra

actualmente en pantalla y también tenemos el back-buffer, que es donde dibujamos los objetos que formarán la escena final. Estas áreas de memoria deben tener las mismas dimensiones del modo de video seleccionado. (Albornoz Figueroa, 2007)

Ahora tenemos dos opciones para esta técnica. Si el back-buffer se encuentre en la RAM, deberemos copiar todo el contenido al front-buffer, es decir, a la memoria de video para que veamos los cambios en el monitor. Para esto realizaremos un proceso de blitting, que dependiendo del modo de video usado, puede ser un poco costoso. (Albornoz Figueroa, 2007)

La otra posibilidad es que el back-buffer también se encuentre en la Video RAM, de ser así, no tendremos que realizar un blitting, sino que haremos algo más simple, intercambiaremos estas dos zonas de memoria, lo cual tiene un costo bastante bajo. A este proceso de intercambio del back-buffer con el front-buffer, se le conoce con el nombre de Page Flipping, y es lo que se suele usar hoy en día, ya que disponemos de más RAM en las tarjetas de video. Además la función de Page Flipping, está implementada en todas las APIs gráficas, así que solo deberemos llamarla en el momento adecuado. (Albornoz Figueroa, 2007)

DIRTY RECTANGLES

Esta técnica es una forma de optimizar la anterior. Con Double Buffering constantemente estamos volcando el contenido completo de una zona de memoria a otra, pero puede haber ocasiones donde no esté sucediendo ningún cambio en pantalla, o tal vez solo haya cambiado una pequeña parte de esta. Es aquí donde aparece la técnica, Dirty Rectangles (rectángulos sucios) y su funcionamiento es bastante simple. Copiaremos a la Video RAM solo las áreas de la pantalla que han cambiado, por lo tanto, cada vez que algún sprite cambie su posición, copiaremos a la memoria de video el área de un rectángulo que rodee al sprite (incluyendo el área donde se encontraba antes) y la colocaremos justo en las mismas coordenadas en la Video RAM. Pero no siempre es recomendable utilizar esta técnica, ya que podemos tener demasiados sprites moviéndose por la pantalla, y ya no sería óptimo estar copiando cada rectángulo a la Video RAM, ya que sería lo mismo que copiar la pantalla completa, en un caso así conviene solo usar la técnica Double Buffering. (Albornoz Figueroa, 2007)

CLIPPING

El clipping es una técnica bien sencilla, consiste en definir una área de recorte para la pantalla, es decir, todo lo que se dibuje fuera de esta área será ignorado y no se mostrará en pantalla. Normalmente esta área de recorte coincide con la resolución elegida para el video-juego, pero puede ser cualquier otra. (Albornoz Figueroa, 2007)

4.2.1.7 SISTEMA DE COORDENADAS

Es importante saber que cuando dibujamos algo en pantalla, siempre tendremos que informar la posición en la cual dibujaremos el objeto. Por defecto la posición de un objeto se encuentra en la esquina superior izquierda, pero algunos prefieren usar como punto de anclaje el centro, y en general puede ser cualquier otro punto. Calcular el centro del punto es tan simple como sumar a la posición x, la mitad del ancho del objeto y a la posición y, la mitad de la altura del objeto.

Nosotros estamos acostumbrados a trabajar en el cuadrante I del plano cartesiano, donde las coordenadas del eje "x" crecen hacia la derecha y las coordenadas del eje "y" crecen hacia arriba, pero al trabajar con gráficos en el computador, esto es relativamente distinto, ya que el eje y se invierte, es decir, ahora el mundo está al revés, las coordenadas en el eje "y" crecerán hacia abajo.

4.2.1.8 SINCRONIZACIÓN EN LOS VIDEO-JUEGOS

Lo ideal en cualquier video-juego, es que todos los objetos se muevan a la misma velocidad, independiente de la velocidad del computador donde se ejecute. Si no nos preocupamos por esto, y ejecutamos nuestro video-juego en un computador antiguo, por ejemplo, en un Pentium de 100 Mhz, probablemente el video-juego se vea muy lento, en cambio sí lo ejecutamos en un computador con un procesador de última generación, un Pentium IV a 2.4 Ghz, se verá tan rápido que será imposible jugar. (Albornoz Figueroa, 2007)

Para solucionar este problema, disponemos de dos métodos.

SINCRONIZACIÓN POR FRAMERATE

El primer método, consiste en sincronizar el framerate, también conocido como FPS o Frames per Second (Frames por segundo). Al hablar de FPS, nos referimos a la frecuencia con que se ejecuta el "*ciclo o bucle principal*" de un video-juego en un

segundo. A mayor cantidad de FPS obtendremos una mayor fluidez en las animaciones. (Albornoz Figueroa, 2007)

En el cine se considera que una velocidad de 24 FPS es suficiente para que el ojo humano perciba una animación fluida, pero en los video-juegos esta cantidad es demasiado baja. Valores adecuados son sobre 60 o 100 FPS. (Albornoz Figueroa, 2007)

El método es bastante sencillo, y lo primero que debemos hacer cuando comienza el ciclo del video-juego, es obtener el tiempo transcurrido hasta ese momento, que normalmente se mide en milisegundos. Luego procesamos lo relacionado al video-juego, ya sea entrada, IA, lógica del juego, detección de colisiones, dibujo de gráficos, etc. y antes de terminar el ciclo, creamos otro loop en el cual vamos obteniendo el tiempo transcurrido hasta ese momento, calculamos la diferencia de tiempo y verificamos si es menor a los FPS que buscamos. De esta forma cada vez que ejecutemos el programa en una máquina diferente, el programa esperará el tiempo adecuado para obtener los FPS. (Albornoz Figueroa, 2007)

Utilizando este método, en computadores más rápidos se verá más fluido, pero si lo ejecutamos en una máquina con un procesador mucho más antiguo del que usamos para desarrollarlo, lo más probable es que se vea bastante lento. (Albornoz Figueroa, 2007)

SINCRONIZACIÓN POR TIEMPO

El segundo método consiste en sincronizar en base al tiempo. En este método no importa el framerate que posea el video-juego, pero aun así los objetos se moverán a la misma velocidad en todas las máquinas. (Albornoz Figueroa, 2007)

4.2.2 GRAFICOS 3D

El término gráficos 3D por computadora se refiere a trabajos de arte gráfico que son creados con ayuda de computadoras y programas especiales. En general, el término puede referirse también al proceso de crear dichos gráficos, o el campo de estudio de técnicas y tecnología relacionadas con los gráficos tridimensionales.

Un gráfico 3D difiere de uno bidimensional principalmente por la forma en que ha sido generado. Este tipo de gráficos se originan mediante un proceso de cálculos matemáticos sobre entidades geométricas tridimensionales producidas en un

computador, y cuyo propósito es conseguir una proyección visual en dos dimensiones para ser mostrada en una pantalla o impresa en papel.

En general, el arte de los gráficos tridimensionales es similar a la escultura, mientras que el arte de los gráficos 2D es análogo a la pintura. En los programas de gráficos por computadora esta distinción es a veces difusa: algunas aplicaciones 2D utilizan técnicas 3D para alcanzar ciertos efectos como iluminación, mientras que algunas aplicaciones 3D primarias hacen uso de técnicas 2D.

Los gráficos 3D se han convertido en algo muy popular, particularmente en video-juegos, al punto que se han creado interfaces de programación de aplicaciones (API) especializadas para facilitar los procesos en todas las etapas de la generación de gráficos por computadora. Estas interfaces han demostrado ser vitales para los desarrolladores de hardware para gráficos por computadora, ya que proveen un camino al programador para acceder al hardware de manera abstracta, aprovechando las ventajas de tal o cual placa de video.

4.3 MOTORES 3D

Un motor de video-juego es un término que hace referencia a una serie de rutinas de programación que permiten el diseño, la creación y la representación de un video-juego. Del mismo modo existen motores de juegos que operan tanto en consolas de video-juegos y sistemas operativos. La funcionalidad básica de un motor es proveer al video-juego de un motor de renderizado para los gráficos 2D y 3D, motor físico o detector de colisiones, sonidos, scripting, animación, inteligencia artificial, redes, streaming[73], administración de memoria y un escenario gráfico. El proceso de desarrollo de un video-juego puede variar notablemente por reusar o adaptar un mismo motor de video-juego para crear diferentes juegos. (Ward, 2008)

Un ejemplo de motor de juego seria el motor gráfico Doom engine, Euphoria o el Rockstar Advanced Game Engine.

[73] Distribución de multimedia a través de una red de computadoras de manera que el usuario consume el producto al mismo tiempo que se descarga.

4.3.1 ESTRUCTURA DE UN MOTOR 3D

4.3.1.1 RENDERER

Renderizado (render en inglés) es un término usado en jerga informática para referirse al proceso de generar una imagen desde un modelo. Este término técnico es utilizado por los animadores o productores audiovisuales y en programas de diseño en 3D.

En términos de visualizaciones en una computadora, más específicamente en 3D, la renderización es un proceso de cálculo complejo desarrollado por un computador destinado a generar una imagen 2D a partir de una escena 3D. La traducción más fidedigna es interpretación, aunque se suele usar el término inglés. Así podría decirse que en el proceso de renderización la computadora interpreta la escena en tres dimensiones y la plasma en una imagen bidimensional.

4.3.1.2 FISICA

La Animación física por computadora o Física de juego es un tipo de programación, en donde supone la introducción de las leyes de Física en un simulador o motor de juego, particularmente en los gráficos 3D por computadora, aunque también existe en gráficos 2D en juegos como Phun. El propósito es hacer que los efectos físicos de los objetos creados o modelados tengan las mismas características que en la vida real, teniendo en cuenta, por ejemplo, gravedad, masa, fricción, restitución, etc.

La simulación de la física en la programación es solo una aproximación cercana a la física real (si bien se acerca mucho a la física que tendría un objeto en realidad, no es igual de realista que en la realidad), y el cómputo es desarrollado usando valores discretos. Hay muchos elementos que forman los componentes de la simulación física.

4.3.1.3 AUDIO/VIDEO

La música de video-juegos, en particular desde comienzos del siglo XXI, puede ser considerada como un género musical por derecho propio, principalmente por tratarse en su mayor parte de música programada, a diferencia de la música grabada en estudio o interpretada en directo.

Hay dos formas de manejar el sonido. Uno es por medio de archivos ".wav"[74] (o similares), lo cual emite un muy buen sonido, pero se requiere de mucha memoria. Por otro lado se pude utilizar archivos MIDI[75], esto reduce la necesidad de memoria, pero los sonidos no son tan buenos, en la actualidad el avance en la compresión del audio ha ayudado a que existan otros formatos alternativos como el MP3.

En el caso del video, esto ha ayudado a las cinemáticas, y debido a los algoritmos de compresión llamado CÓDEC[76], se ha logrado reducir su tamaño, pero no es de extrañarnos que aun los animaciones de las cinemáticas de larga duración o extremas sean hechas por código o en secuencia animada en tiempo real.

4.3.1.4 SCRIPTING

En informática un script o archivo de órdenes o archivo de procesamiento por lotes, es un programa usualmente simple, que por lo regular se almacena en un archivo de texto plano. Los guiones son casi siempre interpretados, pero no todo programa interpretado es considerado un script. El uso habitual de los scripts es realizar diversas tareas como combinar componentes, interactuar con el sistema operativo o con el usuario.

En video-juegos se usan para hacer Pre-scripted Cinematics: usado normalmente en una situación que necesita la explicación en una manera controlada. Para presentar las escenas de la historia, ahora se utilizada el cortar-escenas que presenta la historia en vídeo digital y luego por medio de transiciones se pasa a las gráficas reales del juego.

El scripting le permite al diseñador tomar mando de la escena y manipularla, como colocar objetos o eventos que el jugador no controla. En muy complicado, se necesita de una mente muy metódica y lógica, la mayoría de estos scripts se basan en lenguaje C.

---

[74] También denominado WAVE, apócope de WAVEform audio file format, es un formato de audio digital normalmente sin compresión de datos desarrollado y propiedad de Microsoft y de IBM que se utiliza para almacenar sonidos en el PC, admite archivos mono y estéreo a diversas resoluciones y velocidades de muestreo, su extensión es .wav.

[75] Siglas de la (Interfaz Digital de Instrumentos Musicales). Se trata de un protocolo de comunicación serial estándar que permite a los computadores, sintetizadores, secuenciadores, controladores y otros dispositivos musicales electrónicos comunicarse y compartir información para la generación de sonidos.

[76] Abreviatura de codificador-decodificador. Describe una especificación desarrollada en software, hardware o una combinación de ambos, capaz de transformar un archivo con un flujo de datos (stream) o una señal.

También tenemos el “Visual Scripting Systems”: como lo dice su nombre, permite manejar el script en un ambiente gráfico en lugar de un código escrito, se maneja un carácter real en un ambiente del juego real no basta con que exista un lenguaje de scripting definido, también es necesario contar con las herramientas necesarias para facilitar su uso, aprendizaje, y extensión.

Lo más importante del lenguaje de un motor3D es que aquí los programadores desarrollan toda la lógica del video-juego, junto con la programación de la inteligencia artificial para darle vida e interacción.

4.3.1.5 INTELIGENCIA ARTIFICIAL (IA)

Es la característica más importante que se le atribuye a un motor al lado de la representación de modelos o Render. La IA provee de estímulo al juego. Es crítico en la parte de la forma de juego también llamado “game play”.

La inteligencia artificial de determinado juego puede tornarse muy compleja, primero se debe definir la línea base del comportamiento de los NPC (Non Player Characters - Personajes no Jugables), primero debe definirse qué hace el NPC (patrulla, guarda, etc.), luego se delimita su “visión del mundo”, que es lo es el NPC puede ver del mundo del juego; se debe tomar en cuenta que el personaje no sólo estará en medio del mundo del juego sino que también interactuará con él, después vienen las rutinas de Toma de Decisión

4.3.1.6 REDES

El módulo de red es usado para los video-juegos en línea, aquellos video-juegos jugados vía Internet independientemente de la plataforma, este provee de api para que el desarrollador no programa directamente con la tarjeta de red, esta le provee tanto de la conexión, control de datos, mensajes, sincronización de partidas, hoy en dia el concepto de tener un servidor y varios clientes ha permitido que este módulo tome un grado de importancia alto, ya que algunos juegos una vez terminados se vuelven atractivos al saber que no existe inteligencia artificial en los personajes sino que está siendo controlado por una persona real.

4.3.2 ARQUITECTURA BASICA POR MODULOS

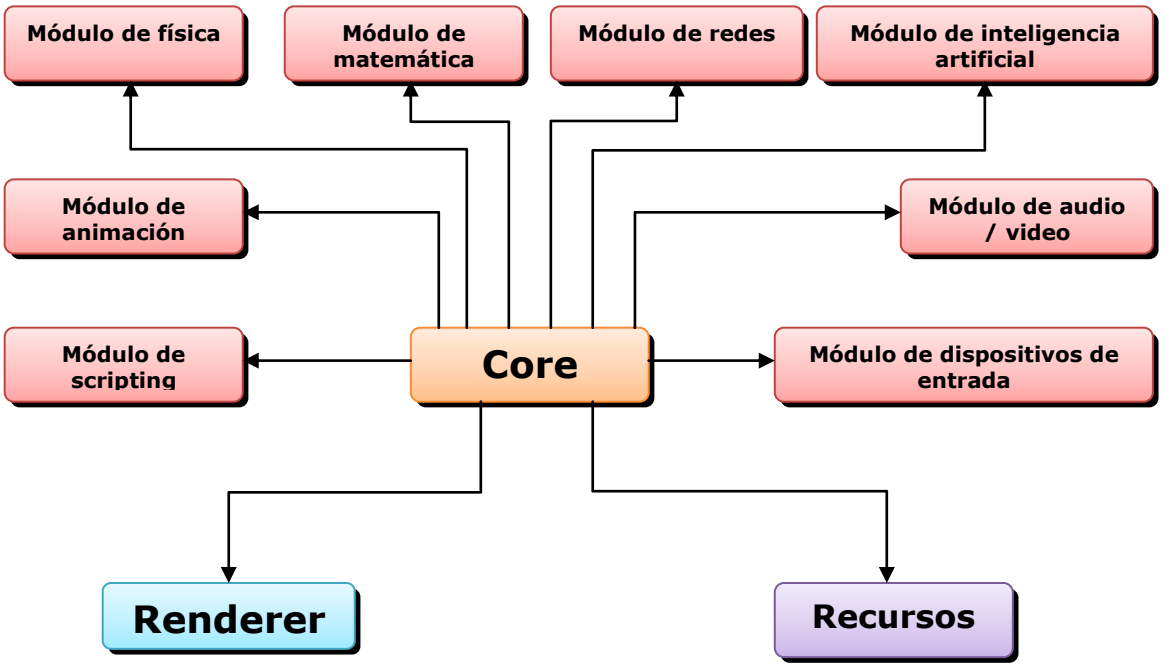

Figura 4.15 Arquitectura básica por módulos

Fuente: Álvarez Pablo Nicolás y Rodriguez Ignacio Gabriel
http://java3dengine.googlecode.com/files/j3dEngine_presentation.pptx

4.3.2.1 MÓDULO DE FÍSICA

Este módulo se encarga de simular la física Newtoniana como rozamiento, velocidad, masa, etc. Cálculo de colisiones, tiempo real vs precisión, cuerpos rígidos vs cuerpos blandos y fluidos, aceleración por hardware sea por PPU. Como Ageia PhysX o la GGPGPU.

4.3.2.2 MÓDULO DE MATEMATICAS

Este módulo se encarga de los cálculos, operaciones complejas como vectores, matrices, quaternions[77], intersección a través de las API del lenguaje.

4.3.2.3 MÓDULO DE REDES

Este módulo se encarga de la red para los Juegos multiplayer o multi-jugador, además funciones de conectividad, abstracción del hardware, optimización para tiempo real, control de perdidas, reducción de latencia y seguridad.

4.3.2.4 MÓDULO DE INTELIGENCIA ARTIFICIAL

Este se encarga de simular comportamiento inteligente en los NPCs[78] (Non-Player-Characters por sus siglas en inglés), desde lógica discreta hasta redes neurales,

[77] También llamados cuaterniones o cuaternios en español, son una extensión de los números reales, similar a la de los números complejos.

máquinas de estado, algoritmos de pathfinding, limitación de capacidad y aceleración por hardware indirecta.

4.3.2.5 MÓDULO DE ANIMACIÓN

Control de las animaciones de los modelos tanto para visualizar y editar, además del realismo de personajes, animación de rostros, algoritmos de "lipsync" y expresiones.

4.3.2.6 MÓDULO DE SCRIPTING

Este se encarga de la utilidad de "Interpretar" el lenguaje, suelen tener un editor de código parta la escritura y compilación; puede ayudar a generar animación por programación, pero sobre todo se escribe la lógica del juego.

4.3.2.7 MÓDULO DE AUDIO/VIDEO

Este se encarga de las funciones multimedia, reproducción, efectos, decodificación, abstracción del hardware o uso de algún hardware especial.

4.3.2.8 MÓDULO DE DISPOSITIVO DE ENTRADA

Este se encarga del acceso a los dispositivos de lectura y entrada de datos, sean botones en el dispositivo o teclados para llevarlas a operaciones de alto nivel, abstracción del hardware o hardware especial como palancas de mando.

4.3.3 ALGUNOS MOTORES USADOS EN LA INDUSTRIA

Crear un "motor 3D" es una tarea muy compleja para un programador o un grupo pequeño de ellos, pero no imposible; el factor preponderante para ponerse a crear uno dependerá en gran medida de cuánto tiempo disponemos para ello; por eso los siguientes apartados de esta tesis pretenden dar a conocer algunos de los motores 3d más populares de la industria para la creación de video-juegos, mencionando sus características, empresa que lo desarrolló, algunas de sus bondades, plataformas entre otras, para que los programadores puedan tomar una decisión acorde a su proyecto y sacar provecho de sus ventajas; además de vislumbrar cuanto ha avanzado el tema tecnológico de software el desarrollo de juegos de video.

[78] NPC, es generalmente parte del programa o juego, y no controlado por un humano.

4.3.3.1 UNREAL ENGINE UDK

DESCRIPCION

Unreal engine es un motor de juego de PC y consolas creados por la compañía Epic Games. Implementado inicialmente en el shooter en primera persona Unreal en 1998, siendo la base de juegos como Unreal Tournament, Deus Ex, Turok, Tom Clancy's Rainbow Six: Vegas, America's Army, Red Steel, Gears of War, BioShock, BioShock 2, Star Wars Republic Commando o Batman: Arkham Asylum. También se ha utilizado en otros géneros como el rol y juegos de perspectiva en tercera persona.

CARACTERISTICAS

Está escrito en C++, siendo compatible con varias plataformas como PC (Microsoft Windows, GNU/Linux), Apple Macintosh (Mac OS, Mac OS X) y la mayoría de consolas (Dreamcast, Gamecube, Wii, Xbox, Xbox 360, PlayStation 2, PlayStation 3). Unreal Engine también ofrece varias herramientas adicionales de gran ayuda para diseñadores y artistas.

La última versión de este motor es el Unreal Engine 3, está diseñado para la tecnología:

- DirectX 9 (para las plataformas Windows XP/Vista/7 de 32/64-bit y Xbox 360)
- DirectX 10 (para plataformas Windows Vista/7 32/64-bit)
- OpenGL (para plataformas Linux, Mac OS X de 32/64-bit y PlayStation 3).

VERSIONES

Unreal Engine 1

Haciendo su primera aparición en 1998, la primera generación Unreal Engine integraba renderizado, detección de colisiones, IA, visibilidad, opciones para redes y manipulación de archivos de sistema en un motor bastante completo. Epic usó este motor para los títulos Unreal y Unreal Tournament.

Unreal Engine 2

La segunda versión del Unreal Engine hace su debut con America's Army en 2002. Esta generación pasó por una reescritura completa del código del núcleo y del

motor de renderizado, además de integrar el nuevo UnrealEd. También incluyó el SDK de Karma physics. Muchos otros elementos del motor fueron actualizados, con mejoras, agregando soporte para la PlayStation 2, GameCube y Xbox.

En la versión Unreal Engine 2.5 fue mejorado el rendimiento y agregadas físicas para vehículos, editor de sistema de partículas para el UnrealEd y soporte 64-bit en Unreal Tournament 2004. Además en la versión especializada de Unreal Engine 2.5 llamada UE2X, se optimizaron características para la Xbox, agregando soporte de efectos de sonido EAX 3.0, siendo usado por ejemplo para el juego Unreal Championship 2.

Unreal Engine 3

El Unreal Engine de tercera generación aparece en 2006, diseñado para PC con soporte DirectX 9/10, Xbox 360 y PlayStation 3. Su motor reescrito soporta técnicas avanzadas como HDRR, normal mapping, y sombras dinámicas. Incluyendo componentes para herramientas complementarias al igual que las anteriores versiones del motor. Se sustituye a Karma por PhysX de Ageia (posteriormente adquirido por NVIDIA), y FaceFX se incluye además para generar animaciones faciales. Epic utilizó esta versión del motor para el video-juego Gears of War y Unreal Tournament 3, posteriormente utilizando una versión mejorada para Gears of War 2.

En el E3 [79] de 2007 Sony anuncia que se asocia a Epic para optimizar el motor para el hardware del PlayStation 3, cambios que se utilizan actualmente por los desarrolladores de juegos de esta plataforma. Asimismo surge el anuncio del desarrollo de una modificación del Unreal Engine para la Wii.

Debido a su política de licencias, Epic obtuvo numerosos contratos con compañías como Atari, Activision, Capcom, Disney, Konami, Koei, 2K Games, Midway, THQ, Ubisoft, Sega, Sony, Electronic Arts, Square Enix, CCP Games y 3D Realms, entre otras.

En la GDC de 2008, Epic revela numerosas mejoras de diseño, entre las que se incluyen: renderizado para mayor número de objetos simultáneos, físicas más realistas para efectos de agua, físicas de texturas corporales, mayor destructibilidad para los

[79] Electronic Entertainment Expo, también conocida como E3, es la convención de video-juegos más importante de la industria.

entornos, IA mejorada y efectos mejorados en luces y sombras con rutinas avanzadas para los shaders. Esta revisión (conocida como Unreal Engine 3.25/5) hizo su debut con el título Gears of War 2.

El Unreal Engine 3 además se aplica en sectores no relacionados con los video-juegos como simulación de construcciones, simuladores de conducción, previsualización de películas y generación de terrenos.

El 5 de noviembre del 2009 Epic Games publicó una versión gratuita del Unreal Development Kit para permitir a grupos de desarrolladores amateur realizar juegos con el Unreal Engine 3.

Unreal Engine 4

En agosto de 2005, el vicepresidente de Epic Games, Mark Rein declara el principio del proyecto para la cuarta generación del motor teniendo en mente al futuro hardware de PC, así como para la séptima generación de consolas de video-juegos. Para ese entonces la única persona encargada del proyecto era el director técnico y fundador de Epic, Tim Sweeney, quien trabajaría en el sistema del núcleo del motor. Sin embargo, ya en 2006 declara que el desarrollo no se iniciaría sino hasta 2008.

A finales de 2008, las declaraciones dadas por Sweeney dan a conocer que el proyecto ya estaba en marcha, con la participación de por lo menos cuatro ingenieros, además, que esta siguiente generación se enfocaría predominantemente a las consolas de video-juegos por encima de la plataforma PC. Por el contrario, posteriormente Mark Rein, menciona que el Unreal Engine 4 para PC estaba próximo y que no se dejaría en desigualdad a esta plataforma.

En recientes declaraciones del presidente de Epic, Michael Capps, dice que debido a que el motor se enfoca en una siguiente generación de gráficos, el motor estaría listo y se utilizaría entre los años 2012 y 2018[80].

El 3 de marzo de 2011, en la conferencia de GCD 2011 en San Francisco son mostradas las primeras imágenes con el motor Unreal Engine 3 repotenciado haciendo

[80] Michael Thompson. "Epic Games: Unreal Engine 4 ready in 2012" (en inglés). http://arstechnica.com/gaming/2008/07/epic-games-unreal-engine-4-ready-in-2012/ Consultado el 30-01-2012.

que aun tenga vigencia para la próxima generación de consolas por varios años más hasta tener listo el Unreal Engine 4.

LICENCIAMIENTO

UDK es **GRATIS** para uso educativo. Las instituciones educativas pueden utilizar el UDK de forma gratuita. No se requiere licencia adicional para los profesores o los estudiantes.

El UDK es gratuito para uso no comercial, pero si se utiliza UDK en un negocio, vender un juego o aplicación, vender servicios o de formación y en relación con ese negocio, se está obligado a firmar un acuerdo Comercial de licencia de uso.

Según el sitio http://www.udk.com/licensing , Si está utilizando UDK internamente dentro de su negocio y la aplicación creada utilizando UDK no se distribuye a un tercero (es decir, alguien que no es su empleado o subcontratista), que están obligados a pagar épica un EPIC un canon anual de 2.500 dólares EE.UU. por instalar "UDK por puesto" de desarrollador al año. Este canon se aplica solamente a "UDK puesto" utilizados para el desarrollo, sin comisión de licencia es necesaria para el hardware en el que sólo las aplicaciones resultantes están instalados.[81]

Si crea un juego o aplicaciones comerciales que utilizan UDK para la venta o distribución a un usuario final o cliente, o si están prestando servicios en el marco de un juego o una aplicación basada en UDK, la opción por puesto, no se aplica. En cambio, los términos de licencia de este acuerdo son $ 99 por adelantado, y una regalía del 0% para usted o su empresa menor a $50.000 dólares en ingresos relacionados a UDK de todos sus juegos UDK o aplicaciones, y una regalía del 25% en los ingresos UDK relacionados de todos sus juegos o aplicaciones basadas en UDK comerciales por encima de 50.000 dólares. Los ingresos relacionados con UDK incluye, el dinero obtenido de: ventas, servicios, capacitación, publicidad, patrocinios, promociones, membrecías, cuotas de suscripción, las transacciones en el juego de alquiler y pagar para jugar. Usted o su empresa no necesitará una licencia comercial para cubrir todas las aplicaciones basadas en juegos UDK o comerciales que se desarrollan.

[81] UDK Licencia http://www.udk.com/licensing consultado 23/May/2012

4.3.3.2 UNITY 3D

DESCRIPCION

Unity3D es un motor de video-juego gratuito, el cual se puede descargar, usar y gratuitamente exportar (ya sea para su comercialización o liberación gratuita) juegos para computador o para la web escribiendo código una vez y obteniendo resultados para varias plataformas. Lo bueno es que podemos adquirir diversas licencias que nos permiten exportar a Android, iPhone, MacOS, Playstation 3 y Xbox 360. [82]

Este es un motor gráfico 3D para PC y Mac que viene empaquetado como una herramienta para crear juegos, aplicaciones interactivas, visualizaciones y animaciones en 3D y tiempo real. Unity puede implementar contenido para múltiples plataformas como *PC, Mac, Nintendo Wii y iPhone.* El motor también puede publicar juegos basados en web usando el plugin Unity web player.[83]

El editor de **Unity** es el centro de la línea de producción, ofreciendo un completo editor visual para crear juegos. El contenido del juego es construido desde el editor y el gameplay se programa usando un lenguaje de scripts.

CARACTERISTICAS

El motor implementa un completo editor de escenarios que nos permite arrastrar y colocar objetos en 3D para animarlos de forma fácil, y para esta tarea podemos utilizar algunos programas de diseño 3D como Blender, Maya, Cinema 4D, 3ds Max, entre otros.

Por otro lado, el editor nos permite trabajar fácilmente con algunos componentes como partículas, colisiones, audio y física y lo más importante es que el editor de escenarios nos proporciona una gran cantidad de opciones para ahorrar tiempo creando los terrenos, por ejemplo, nos permite añadir fácilmente agua o crear elevaciones de terreno en unos segundos.

[82] http://razonartificial.com/2011/04/unity3d/ Consultado 01/Mayo/2012
[83] http://www.mat.ub.edu/futurs_ub/activitats/triptics/motoresgraficos.pdf Consultado 01/Mayo/2012

LICENCIAMIENTO

Unity 3D es gratuito tanto para descargar como para su comercial, En cuanto a la programación, Unity3D nos permite trabajar con 3 lenguajes: Javascript, C# y una implementación de Python. Por otro lado los precios varían mucho dependiendo de la plataforma, por ejemplo podemos comprar la licencia pro que nos permite exportar el juego a varias plataformas a $1.500 dólares, o la versión para exportar a iPhone o Android a $400.

En sí este motor resulta ser una gran alternativa al Unreal Engine sobre todo porque la calidad de los juegos que se pueden realizar con el mismo es realmente alta y dichos resultados se pueden lograr en muy poco tiempo.

Uniy 3D está en la 4ta position del top de motores según la página web http://www.develop-online.net.[84]

VERSIONES

Las siguientes características están tomadas de la página de Unity, http://unity3d.com/unity/whats-new/unity-1.1 en adelante.

- Unity 3D 1.1
    - Versión estable, publicación a la plataforma Windows con un player independiente.
    - Shader de refracción de vidrio y 8 nuevos desplazamientos virtuales bump mapping-shaders.
    - Gizmo en interfaz de dibujo.
    - importación de Maya 4.5 y 5.0.
    - Mejoras en Editor y flujo de trabajo, Física y documentación del lenguaje y API.
- Unity 3D 1.2
    - Efectos gráficos a pantalla completa.
    - Representar las texturas en los materiales.
    - Mejoras en la física.
    - Controlador profesional de primera persona.

[84] http://www.develop-online.net/news/32250/The-top-10-game-engines-revealed Consultado 14-May-2012

    - Extensibilidad de flujo de trabajo.
    - Nuevos script y mejoras en el lenguaje.
- Unity 3D 1.5
    - Web plug-in para navegador en Windows.
    - Animación de Personajes.
    - Binario Universal
    - Alta velocidad de Física para juegos de carros.
    - Filtros para pantalla completa
    - Script para Malla o Mesh de interfaz.
    - Script para texturas.
    - Script para interfaz de partículas.
    - Soporte a lightmap
    - Soporte para importar de Cinema 4D.
- Unity 3D 1.6
    - Unidad a la comunicación del navegador
    - Streaming de archivos ".Unityweb"
    - Soporte para Windows Vista
    - Búsqueda de documentación
    - Efectos de audio
    - información precisa en "Raycasts" en los vértices afectados
    - Completo soporte para .NET DLL.
- Unity 3D 2.0
    - Motor de terreno
    - Sombras dinámicas en tiempo
    - Reproducción de vídeo
    - Soporte de Interfaces gráficas de usuario en los juegos
    - Reproductor Web soporta streaming y compresión
    - Render con DirectX 9.0
    - Servidor de activos o Asset.
    - Soporte juegos en Red y Multijugador.
- Unity 3D 2.1
    - Streaming en Assets Bundles.
    - Streaming en Terrenos

    - Sombras en tiempo real sobre los terrenos
    - Procedimientos para personajes y Animación
    - Editor Aún más extensible.
    - Renderizado con representación de Shader.
    - Soporte a Deshacer o Undo.
    - Mejoras en el rendimiento.
- Unity 3D 2.5
    - Compatibilidad del editor en Windows.
    - Importación de 3ds Max.
    - Mejora de la Usabilidad.
    - Interfaz con pestañas.
    - Precisión en la navegación y herramientas de colocación.
    - Editor completamente personalizable.
    - Soporte e Información a su alcance de forma instantánea.
- Unity 3D 2.6
    - Mejoras en la carga y corre más rápido.
    - Se puede animar todo.
    - Soporte de galería.
    - Visión más nítida de los elementos.
    - Búsqueda en proyectos
    - Imán o "Snappy" en cuadrículas.
- Unity 3D 3.0
    - Editor Unificado
    - Mejoras en Lightmapping
    - Soporte a Audio Magic.
    - Soporte a depuración en el código.
    - Renderizado diferido
    - Efectos de lentes.
    - Administrador de Activos
    - Soporte a Occlusion Culling
- Unity 3D 3.1 / 3.2 / 3.3 / 3.4
    - Tienda de activos de Unity en la WEB.
    - Corrección de errores y fixes.

- Nuevos efectos.
- Mejoras en el editor, gráficos, Audio y físicas.
- Mejor soporter para iOS y Android.
- Mejoras en los algoritmos aplicadas a efectos de substancias.
- Corrección de errores o fix.

- Unity 3D 3.5 / 3.5.1 /3.5.2
  - Sistemas de partículas Shuriken
  - Publicación con Pathfinding.
  - Soporte Google Chrome Native Client.
  - Iluminación lineal del espacio y HDR.
  - Corrección de errores y mejoras en el rendimiento.

4.3.3.3 BLENDER

DESCRIPCION

Blender es un programa informático multiplataforma, dedicado especialmente al modelado, animación y creación de gráficos tridimensionales.

El programa fue inicialmente distribuido de forma gratuita pero sin el código fuente, con un manual disponible para la venta, aunque posteriormente pasó a ser software libre. Actualmente es compatible con todas las versiones de Windows, Mac OS X, Linux, Solaris, FreeBSD e IRIX.

Tiene una muy peculiar interfaz gráfica de usuario, que se critica como poco intuitiva, pues no se basa en el sistema clásico de ventanas, pero tiene a su vez ventajas importantes sobre éstas, como la configuración personalizada de la distribución de los menús y vistas de cámara.

Aun siendo una herramienta relativamente nueva, ha gozado de la aceptación de muchos animadores independientes. En la industria de Generación de gráficos avanza como un proyecto prometedor, si bien las superproducciones no lo han usado para generar secuencias CGI. Existen proyectos actuales que han empezado a usarlo profesionalmente:

- El 18 de febrero de 2010 se estrenó el primer largometraje animado realizado íntegramente con software libre, usando a Blender como herramienta principal; se trata de Plumíferos,1 proyecto que está impulsando el desarrollo de Blender aún más, sobre todo a nivel de animación y manejo de bibliotecas a gran escala.
- Películas tales como Spider-Man 2 que lo ha usado para hacer una previsualización de escenas (Screen-Board Test), han usado de manera incipiente las capacidades del popular programa GNU/GPL.
- Algunas propuestas más llevadas a la producción e integración con gráficos mediante Motion Track tales como "Friday or another day". que es de los primeros esbozos de su uso a 35mm.2
- Otros proyectos hechos en participación de diversos usuarios de Blender incluido Ton Rossendaal el cortometraje Elephants Dream son experimentos de sus capacidades, extendidas gracias a la posibilidad de poder editar su código fuente, aportando de esta experiencia a los demás usuarios con innovaciones fundamentales: Un sistema de control de gestos (Morph system), un sistema de composición de textura y post producción (Composite), entre otros.

CARACTERISTICAS

- Multiplataforma, libre, gratuito y con un tamaño de origen realmente pequeño comparado con otros paquetes de 3D, dependiendo del sistema operativo en el que se ejecuta.
- Capacidad para una gran variedad de primitivas geométricas, incluyendo curvas, mallas poligonales, vacíos, NURBS[85], metaballs.
- Junto a las herramientas de animación se incluyen cinemática inversa, deformaciones por armadura o cuadrícula, vértices de carga y partículas estáticas y dinámicas.
- Edición de audio y sincronización de video.
- Características interactivas para juegos como detección de colisiones, recreaciones dinámicas y lógica.
- Posibilidades de renderizado interno versátil e integración externa con potentes trazadores de rayos o "raytracer" libres como kerkythea, YafRay o Yafrid.

[85] Acrónimo inglés de non-uniform rational B-spline, es un modelo matemático muy utilizado en la computación gráfica para generar y representar curvas y superficies.

- Lenguaje Python para automatizar o controlar varias tareas.
- Blender acepta formatos gráficos como TGA, JPG, Iris, SGI, o TIFF. También puede leer ficheros Inventor.
- Motor de juegos 3D integrado, con un sistema de ladrillos lógicos. Para más control se usa programación en lenguaje Python.
- Simulaciones dinámicas para softbodies[86], partículas y fluidos.
- Modificadores apilables, para la aplicación de transformación no destructiva sobre mallas.
- Sistema de partículas estáticas para simular cabellos y pelajes, al que se han agregado nuevas propiedades entre las opciones de "sombreado" o "shaders" para lograr texturas realistas.

VERSIONES

Las versiones en Blender son conocidas como RAMA y salieron según la siguiente cronología.

- Rama 1.0, En esta etapa se desarrollan las primeras versiones comerciales de Blender, se desarrollan nuevas versiones de Blender para otros sistemas operativos, en junio del 2000 Blender pasa a ser software libre en su totalidad.
- Rama 2.0 en agosto del 2000
- Rama 2.2, En marzo de 2002 Ton Roosendal, el autor original del programa funda en Amsterdam la Blender Fundation, encargada de continuar el desarrollo de Blender y promover su uso, quien es el actual CEO de la Blender Fundation.
- Rama 2.3, se añaden varias herramientas y se realizan muchas mejoras.
- Rama 2.4, en septiembre de 2005 inicia la producción de el cortometraje Elephants Dream, finalizada el 24 de mayo de 2006, se observó que los proyectos de animación obsorvían mucho tiempo al Blender Institute, así en el verano de 2007 Ton Roosendal establece el Instituto Blender, para tener un instituto independiente encargado de los proyectos de animación. El Instituto Blender presenta el 10 de abril de 2008 el cortometraje Big Buck Bunny y en noviembre de 2008 el video-juego Yo Frankie!.

[86] Es un área de "gráficos por computador" que se centra en simulaciones físicas visuales realistas del movimiento y las propiedades de los objetos deformables u órganos blandos.

- Rama 2.5, comienza a desarrollarse el 24 de diciembre de 2009 y llega a la versión estable 2.57 el 13 de abril de 2011. El 18 de febrero de 2010 el estudio argentino Manos Digitales Animation presenta el largometraje Plumíferos. El 27 de septiembre de 2010 el Blender Institute finaliza el cortometraje Sintel (Durian Open Movie).
- Rama 2.6, en esta rama se introdujeron notables novedades, en concreto:
    - 2.60: Codificación FFmpeg. Peper branch:Sonido 3d. 2010 node project.
    - 2.61: Cycles render engine. Motion/camera tracking.
    - 2.62: Booleanas. Herramientas UV.
    - 2.63: BMesh.

LICENCIAMIENTO

Originalmente, el programa fue desarrollado como una aplicación propia por el estudio de animación holandés NeoGeo; el principal autor, Ton Roosendaal, fundó la empresa "Not a Number Technologies" (NaN) en junio de 1998 para desarrollar y distribuir el programa.

La compañía cayó en bancarrota en 2002, entonces los acreedores acordaron ofrecer Blender como un producto de código abierto y gratuito bajo los términos de la GNU GPL a cambio de $100.000. El 18 de julio de 2003, Roosendaal creó sin ánimo de lucro la Fundación Blender para recoger donaciones; el 7 de septiembre se anuncia la recaudación como exitosa (participaron también ex empleados de NaN) y el código fuente se hizo público el 13 de octubre del 2003, Blender está basado en GPL o GNU General Public License.

4.3.3.4 PANDA 3D

DESCRIPCION

Panda3D es un motor de video-juegos que incluye gráficos, audio, E/S, detección de colisiones, así como otras características relevantes para la creación de juegos en 3D.

Panda3D es un proyecto de código abierto y software libre (desde el 28 de mayo del 2008), bajo la licencia BSD Revisada. Versiones anteriores al 28 de mayo de 2008,

no se consideran Software Libre debido a ciertos errores en el diseño de la antigua licencia Panda3D. A pesar de ello, estas versiones antiguas de Panda3D también se pueden utilizar para el desarrollo de video-juegos libres o comerciales sin costo alguno.

El lenguaje de programación de video-juegos para el que fue destinado Panda3D es Python. El motor en sí mismo está escrito en C++, y utiliza un generador-empaquetador automático para exponer la completa funcionalidad del motor en una interfaz de Python. Este enfoque da al programador las ventajas del desarrollo en Python, como el desarrollo rápido o la gestión avanzada de memoria, pero mantiene el rendimiento de un lenguaje compilado en el núcleo del motor. Por ejemplo, el motor es integrado con el recolector de basura de Python, y las estructuras del motor son manejadas automáticamente.

El manual y los programas de ejemplo usan Python, aunque los programadores están trabajando en traducir el manual a C++ y proveer programas de ejemplo en C++.

Un programador que usa Panda3D, típicamente escribe el código en Python, pero también es posible acceder directamente al motor usando código en C++.

Entre los usuarios de Panda3D se incluyen programadores de una enorme cantidad de juegos comerciales, unos pocos de proyectos de código abierto, y una serie de cursos universitarios que aprovechan la corta curva de aprendizaje de Panda3D. La comunidad es pequeña pero activa, las preguntas realizadas en el foro son generalmente respondidas casi al instante.

Panda3D es un motor gráfico de escenas, esto significa que el mundo virtual es inicialmente un espacio cartesiano vacío en el cual el programador de video-juegos inserta los modelos en 3D. Panda 3D no distingue entre modelos 3D "grandes", como el modelo de una presión o una isla, y modelos 3D "pequeños", como el modelado de una mesa o una espada. Ambos modelados grandes y pequeños son creados usando un programa de modelado estándar como Blender, 3ds Max o Maya, cargados en Panda 3D, y después insertados en el espacio cartesiano.

El motor gráfico de escenas de Panda 3D expone la funcionalidad de OpenGL y DirectX de una forma bastante literal. Por ejemplo, OpenGL y DirectX pueden crear niebla. Para habilitar la niebla en Panda3D, se guardan los parámetros de la niebla en un nodo en la escena gráfica. Los parámetros de niebla coinciden exactamente con los

parámetros de funciones equivalentes en las APIs subyacentes. Se diferencia de ellos, en que se guarda en la escena, mientras que OpenGL y DirectX no lo hacen. Por supuesto, también provee de operadores de alto nivel, como la carga de modelos, ejecución de animaciones, detección de colisiones, etc.

Panda3D fue diseñado antes de la existencia de vértices o vertex[87] y pixel shaders[88]. Adquirió soporte para shaders escritos a mano en 2005. No obstante, los usuarios han tardado en aprovechar las modernas técnicas de iluminación por pixel en sus juegos. Los programadores teorizan que esto ocurre porque la programación de shaders puede ser bastante difícil, y por eso muchos programadores de juegos quieren que el motor los pueda manejar automáticamente.

Para remediar esta situación, los programadores de Panda3D, recientemente le han dado al motor la capacidad de sintetizar shaders automáticamente. Esta síntesis ocurre si el modelador de 3D marca un modelado para iluminación por pixel, o si el modelador aplica un "mapa normal", "mapa de brillo", "mapa autoiluminado", o otra capacidad que excede las capacidades de un pipeline[89] de funciones fijas. La intención de la síntesis es renderizar[90] el modelado como el modelador ha previsto, sin ninguna intervención del programador.

CARACTERISTICAS

Panda 3D tiene otras capacidades aparte del renderizado 3D. Los principales son:

- Herramientas de análisis de rendimiento.
- Herramientas de exploración de la escena gráfica.
- Herramientas de depuración
- Un completo pipeline de exportación/importación de arte.
- 3D Audio, utilizando ya sea FMOD, OpenAL o Miles Sound System.
- Detección de colisiones.
- Sistema de físicas, y una integración total para el Open Dynamics Engine.

[87] En teoría de grafos, un vértice o nodo es la unidad fundamental de la que están formados los dibujos o imágenes.
[88] Un sombreador de píxel (del inglés pixel shader, abreviado PS) es un programa de sombreado, normalmente ejecutado en la unidad de procesamiento gráfico.
[89] Conjunto de elementos procesadores de datos conectados en serie, en donde la salida de un elemento es la entrada del siguiente
[90] Proceso de generar una imagen desde un modelo. Este término técnico es utilizado por los animadores o productores audiovisuales y en programas de diseño en 3D.

- Soporte para teclado y ratón.
- Soporte para dispositivos E/S inusuales.
- Máquina de estados finitos.
- GUI.
- Herramientas para redes.

VERSIONES

Pande 3D en sus versiones no ha mejorado drásticamente su entorno, la primera versión fue la 1.0.5 y ha llegado hasta le versión 1.8.0, hoy en día las versiones por debajo de 1.6.0 se las considera obsoletas y se recomienda no usar según el sitio de http://www.panda3d.org/download.php?sdk .

LICENCIA

En el 2002, cuando el motor fue liberado, el objetivo de los desarrolladores fue crear una licencia libre. Sin embargo, la licencia tenía algunos defectos que la hacía no libre: Se requiere que los cambios sean verificados y se prohíbe expresamente la exportación de software a varias naciones con las cuales los Estados Unidos tenía embargos comerciales.

El miércoles 28 de Mayo del 2008, la rama principal del desarrollo de Panda3D cambia a la licencia BSD. Sin embargo, las versiones antiguas siguen usando la antigua licencia.

Panda3D hace uso de multitud de librerías de terceros cuyas licencias tampoco son Software libre, incluyendo FMOD, Nvidia Cg, DirectX, y MFC.

4.3.3.5 ID TECH 4

DESCRIPCION

Id Tech 4, también conocido como el motor de Doom 3, es un motor de video-juego desarrollado por id Software y usado por primera vez en el juego Doom 3. El motor fue diseñado por John Carmack, quien también había creado motores anteriores como los usados en Doom o Quake, los cuales también son reconocidos como avances

significativos en su campo. Básicamente id Tech 4 es una mejora del motor id Tech 3 pero con varias modificaciones para agregar efectos de última generación.

El motor se comenzó a desarrollar alrededor de 1999, originalmente se había planeado que el motor de renderizado sería completamente reescrito pero se mantendrían otros subsistemas como los de accesos a ficheros o el planificador de memoria. Pero después de que el motor de renderizado fuera completado se decidió cambiar el código del motor del lenguaje C a C++ lo que exigió volver a estructurar y escribir completamente el motor por lo que el proyecto termino tardando más de lo esperado.

Id Tech 4 requería como mínimo una tarjeta gráfica compatible con DirectX 8.0 y que poseyera al menos la primera versión de Pixel shader como por ejemplo una Nvidia GeForce 3 o ATI Radeon 8500. Para el E3 del 2002, ya se recomendaban tarjetas superiores para poder correr Doom 3 con buenos gráficos y con una buena velocidad.

Pasaban los meses y aparecían fotos que las revistas de video-juegos criticaban porque tenían demasiado detalle y habría que desactivar los efectos y sombras para poder jugar. Al terminar el juego en 2004, esa fue una de sus mayores críticas ya que necesitaba mucha potencia de hardware para poder ejecutarlo fluidamente, aunque se ha conseguido correr en placas de video más antiguas como algunas voodoo o SIS.

VERSIONES

- Id Tech 1 o Doom Engine, es el motor gráfico que id Software uso para sus video-juegos Doom y Doom II. Este motor gráfico también es usado por Hexen, Heretic, Strife y HacX, y otros juegos producidos por licenciatarios. Fue creado por John Carmack, con las funciones auxiliares escritas por Mike Abrash, John Romero, Dave Taylor y Paul Radek. Originalmente desarrollado en computadoras NeXT, fue portado a DOS para el lanzamiento inicial de Doom, para después ser portado a otras consolas y sistemas operativos.
- Id Tech 2, también conocido como el motor de Quake II, es un motor de video-juego desarrollado por id Software para ser usado en sus video-juegos, principalmente en Quake II. Desde su lanzamiento, id Tech 2 ha sido licenciado para ser usado en otros video-juegos. Una de las características más llamativas

de id Tech 2 es el soporte directo de aceleración mediante tarjeta gráfica, específicamente OpenGL.

- Id Tech 3 (anteriormente conocido como Quake III engine) es un motor de juego desarrollado por id Software para Quake III Arena y se ha utilizado en muchos juegos bajo este motor y el motor del Quake III: Team Arena. En la época, competía con el Unreal engine, ambos motores fueron ampliamente licenciados. id Tech 3 es una mejora sustancial del motor de Quake.

LICENCIA

Jonh Carmack ha dicho que liberaría el código de id Tech 4 al igual que hizo con el resto de desarrollos anteriores de ID. Durante la QuakeCon 2009, Carmack dijo que tenía planeado pedir a Zenimax liberar el código de id Tech 4 después del lanzamiento de Rage.[91]

Finalmente, el 23 de Noviembre de 2011 se liberó bajo licencia GPL versión 3.

[91] Entrevista sobre liberar el código fuente http://www.linuxgames.com/archives/9374 Consultado 30 del enero de 2012.

4.3.3.6 CUADRO COMPARATIVO DE LOS MOTORES

	EMPRESA	PLATAFORMAS	WEB PLAYER	LICENCIA INDIE	INCORPORA
UDK	**Epic**	Windows, Mac OS X, Xbox 360, PS3.	NO	NO	Fonix, SpeedTree, GameSpy, Scaleform GFx, PhysX, Illuminate Labs, Umbra, morpheme, nFringe, HumanIK, Kynapse, Bink, ProFX, AI.implant, Quazal, DigiMask, Game-Link, Wwise, Enlighten
Unity 3D	**Unity Technologies**	PC, Mac, iPhone, Wii, PS3, Xbox.	SI	Gratis/Comercial	PhysX, Mono Editor.
Blender	**Fundación Blender**	Windows, Mac OS X, Linux, Solaris, FreeBSD e IRIX.	NO	GNU-GPL	Propias como primitivas geométricas, cinemática inversa, kerkythea, YafRay o Yafrid.
Panda3D	**Disney**	Windows, Linux, Mac OS X, FreeBSD	NO	BSD	Propias como Open Dynamics Engine, Kinematic, Dynamic and Ray objects, Explosion mechanic.
Id Tech 4	**id Software**	Windows, Mac OS X, Linux, Xbox, Xbox 360, PlayStation 3.	NO	GNU-GPL	Propias, como Unified lighting and shadowing, Shadow volumen, MegaTexture.

Cuadro 4-1 Comparación de los motores 3D

Fuente: Yamil Lambert - 2012

Capítulo 5

Programación del controlador del jugador (Santiago) en el juego Corsarios del Pacifico.

5.1 EL PAPEL DE LA CAMARA EN UN JUEGO EN 3RA PERSONA

En un juego de primera persona la cámara es el punto de vista del jugador, por lo que no hay necesidad de preocuparse de hacer que siga otro objeto alrededor de la escena. El jugador controla la cámara directamente. Las cámaras de primera persona por lo tanto son relativamente fáciles de implementar. (Unity Technologies, 2007)

Sin embargo, un punto de vista de la cámara de tercera persona requiere una cámara que pueda seguir el jugador por todas partes. Esto parece bastante sencillo hasta que te das cuenta de la cámara también tiene que saber cómo evitar objetos que se encuentren entre el personaje del jugador y el punto de vista de la cámara. (Unity Technologies, 2007)

Esto puede conseguirse utilizando raycasting[92] para comprobar si hay objetos no deseados entre la cámara y el avatar del jugador, pero hay algunos casos especiales a considerar. Por ejemplo: (Unity Technologies, 2007)

- ¿Qué sucede si el personaje se apoya contra una pared sólida? Debería la cámara moverse hacia arriba y mirar hacia abajo en el reproductor? ¿Debe trasladarse a un lado?
- ¿Qué pasa si un enemigo se interpone entre la cámara y nuestro avatar del jugador?
- ¿Cómo deben funcionar los controles del jugador? ¿Deberían ser relativos al punto de vista de la cámara? Si es así, esto podría ser muy confuso si la cámara se mueve inesperadamente para evitar un obstáculo.

[92] Raytracing o trazado de rayos es un algoritmo para síntesis de imágenes tridimensionales, se determinan las superficies visibles en la escena que se quiere sintetizar trazando rayos desde el observador (cámara) hasta la escena a través del plano de la imagen. Se calculan las intersecciones del rayo con los diferentes objetos de la escena y aquella intersección que esté más cerca del observador determina cuál es el objeto visible.

Una serie de soluciones para las cámaras en tercera persona se han probado en los últimos años. Puede afirmarse que ninguno ha sido 100% perfecto. Algunas soluciones hacen transparente todo lo que está entre ellos y su enfoque, haciendo paredes o enemigos semitransparentes. Otras opciones incluyen cámaras que siguen el jugador alrededor, pero que, en caso necesario, Pueden moverse a través de paredes y edificios para mantener la vista del jugador consistente, con esto se ha propuesto utilizar el código de tercera persona que viene incluido en unity3d entenderlo y agregarle el código necesario para ajustarse a nuestro personaje en Santiago. (Unity Technologies, 2007)

5.2 CONTROLADOR DEL JUGADOR "SANTIAGO"

El controlador del personaje Santiago del juego "Corsarios del Pacífico" está programado en lenguaje "JavaScript"; usa una cámara en tercera persona y es una muestra de la facilidad de Unity3D de incluir en sus paquetes tanto personajes de pruebas para juegos en primera persona o FPS y juegos en tercera persona o "3rd Person" para que el programador pueda empezar a desarrollar inmediatamente; el concepto de "Tercera Persona" se destaca porque la cámara sigue al personaje desde la parte de atrás, así que en todo momento este es visto completamente por el video-jugador incluyendo todos sus movimiento.

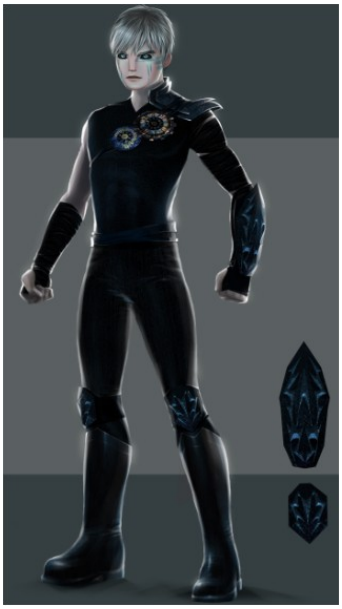

Figura 5.1 Personaje principal del juego Corsarios del Pacífico llamado Santiago

Fuente: Ilustración por Jossie Lara, 2012

El controlador del jugador consta de tres Programas (scripts) escritos en lenguaje Java script.

1. Character Controler
2. Third person Controler
3. Third person Camera

Además se hace necesario disponer de las diferentes animaciones del personaje como “Caminar”, “Patada”, “Puñete” entre otras que conforman todas las posibles acciones que este puede llevar a cabo, estas animaciones pueden ser creadas en cualquier software de animación y modelado 3D, para efectos de nuestra prueba estas fueron hechas en Cinema4D y luego exportadas como un “prefab” o prefabricado[93]; es decir incluidas junto con el modelado del personaje.

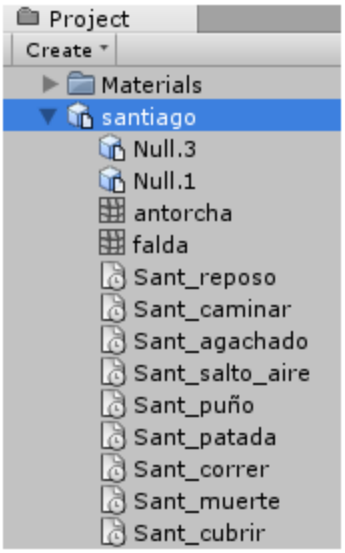

Figura 5-1 Prefabricado del personaje Santiago con sus animaciones.

Fuente: Interfaz de Unity capturada por Yamil Lambert, 2012

Aunque Unity incluye un panel denominado “Animation” donde se pueden ver y crear las animaciones, este es muy básico pero funcional, una vez realizados los movimientos las animaciones se graba como un archivo independiente con extensión “.anim”.

[93] Son un tipo de objeto especial en Unity. Su nombre viene de “prefabricados” un concepto que es usualmente dado para aquellos elementos de construccion que cumplen una funcion específica y se reusan en multiples lugares. http://unitycreativo.com/prefabs Consultado 23/May/2012

Dentro del código y una vez asociado el script al prefabricado que es nuestro personaje, podemos hacer referencia y ejecutar dichas animaciones con las instrucciones:

- _animation.CrossFade(<NombreAnimacion>.name);
- _animation.Play(<NombreAnimación>.name);

La diferencias entre el método **Play()** y **CrossFade()** es que la primera no suaviza la animación si se da inicio a una segunda sin haber terminado la primera; Play() garantiza que la animación se vea completa de inicio a fin pero puede notarse saltos bruscos si otra animación distinta se ejecuta, en nuestro código damos preferencia al segundo método, ver el siguiente ejemplo de referencia.

```
if(_characterState == CharacterState.patada) {

        _animation.CrossFade(this.patadaAnimation.name);
}
```

5.3 LÓGICA DE EJECUCIÓN

Unity incluye funciones predefinidas o comúnmente conocidas como "funciones sobre-escribibles". El nombre viene dado porque estas funciones tienen, con respecto a las "estándar", la peculiaridad de que nos permiten diseñar su contenido, es decir el cuándo y lo que se escribe en código será el qué y el cómo. (Unityscripts, 2011)

A continuación, vamos a explicar el orden en que Unity inicializa los distintos elementos que lo componen cada vez que se carga una escena en el juego: (Unityscripts, 2011)

1. Primero se cargan los objetos (game objects y components).
2. Acto seguido se cargan los scripts que van vinculados a estos objetos, y una serie de funciones (las que veremos a continuación) son llamadas en un orden específico:
    1. **Awake().**
    2. **Start().**
    3. **Update().**
    4. **FixedUpdate().**

5. **LateUpdate().**

UPDATE:

function Update () : void

Esta función es llamada cada fotograma, si el Monobehaviour (script que la invoca) está activo.

Es la función más usada en los scripts, si bien tiene el inconveniente de que cada computador puede tener un framerate distinto, por lo que la misma instrucción de movimiento, por ejemplo, daría pie a diferentes velocidades dependiendo del framerate de cada uno. Para transformar las unidades de actualización de la función de fotogramas a segundos se utiliza Time.deltaTime. (Unityscripts, 2011)

LATEUPDATE:

function LateUpdate () : void

LateUpdate es llamado una vez todas las funciones Update han sido llamadas. Esto nos permite ordenar la ejecución de scripts. Por ejemplo, una cámara que sigue a un objeto debería implementarse en un lateUpdate, pues cabe que el objeto al que sigue se inicialice con un determinado movimiento en update, movimiento que debería tener en cuenta la cámara. (Unityscripts, 2011)

FIXEDUPDATE:

function FixedUpdate () : void

Esta función se ejecuta cada cierto número preestablecido y fijo de fotogramas, lo que hace que no presente los problemas de update. Se debe utilizar en aquellos scripts que impliquen un componente de físicas, y sobre todo, siempre se ha de utilizar cuando haya que añadir una fuerza a un Rigidbody. (Unityscripts, 2011)

AWAKE:

function Awake () : void

Awake es llamada cuando se inicia el script, es decir, cuando se carga la escena, y justo después de que se carguen los objetos (gameobjects y components) a que el

script hace referencia. De tal manera, es útil para inicializar variables o estados del juego antes de que el juego empiece, referenciando si es preciso a los objetos a que hacen mención (y que ya habrán sido como decimos inicializados previamente). (Unityscripts, 2011)

START:

function Start () : void

Es llamada después de Awake y antes de Update. Al igual que awake, sólo es llamada una vez a lo largo de toda la vida del script. Start se diferencia de Awake en que esta sólo es llamada si la instancia del script está habilitada, esto es, tiene su casilla de marcado en el inspector. Así, cuando un gameobject es inicialmente usado en una escena esta función es llamada automáticamente. (Unityscripts, 2011)

5.4 PARAMETROS DE ENTRADA

Para Unity3D la declarativa de una variable pública es considerada un parámetro de entrada que deberá ser dado en el mismo entorno o editor, el archivo "ThirdPersonController.js" define las siguientes variables públicas que han sido traducidas al lenguaje español para una mejor comprensión.

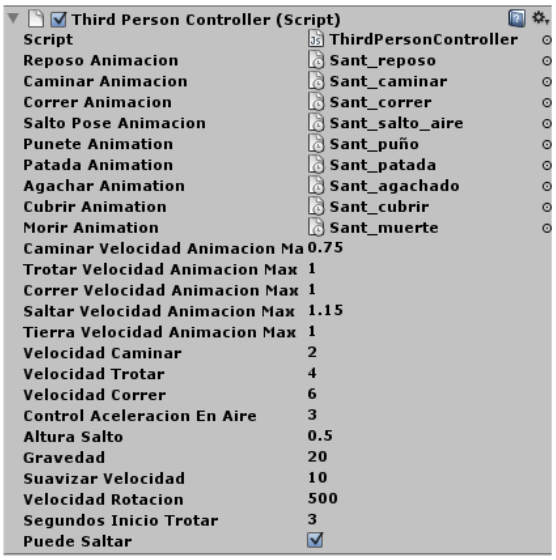

Figura 5.2 Parámetros de estrada controlador de personaje Santiago.

Fuente: Interfaz de Unity capturada por Yamil Lambert, 2012

SCRIPT

Este parámetro recibe la asignación del archivo de código asociado al script que contiene toda la programación, por lo que su parámetro de entrada va en función de su accionar descrito en su nombre de archivo y de selector.

REPOSO ANIMACIÓN

Este parámetro es de tipo "AnimationClip" y recibe la animación del personaje cuando este *no está haciendo nada*, incluso el jugador no ha presionado ningún botón de acción, de tal manera que es denotado por la acción "reposo".

CAMINAR ANIMACIÓN

Este parámetro es de tipo "AnimationClip" y recibe la animación del personaje cuando este *caminando*, es decir que el jugador esta presionando las teclas asociadas al movimiento adelante, atrás, izquierda o derecha; estas están asignadas a las teclas de movimiento del cursor en un teclado.

CORRER ANIMACIÓN

Este parámetro es de tipo "AnimationClip" y recibe la animación del personaje cuando este *corriendo*, es decir que el video-jugador ha presionado primero el botón o tecla de correr (en un teclado es la tecla shift) y al mismo tiempo está presionando las teclas asociadas al movimiento adelante, atrás, izquierda o derecha; estas están asignadas a las teclas de movimiento del cursos en un teclado.

SALTO POSE ANIMACIÓN

Este parámetro es de tipo "AnimationClip" y recibe la animación del personaje cuando este *ha saltado* y el personaje está descendiendo producto del mismo salto, es decir que el video-jugador ha presionado el botón o tecla de salto (en un teclado es la tecla barra espaciadora o spc).

PUÑETE ANIMACIÓN

Este parámetro es de tipo "AnimationClip" y recibe la animación del personaje cuando este *da un golpe o puñete* para defenderse de algún enemigo, es decir que el

video-jugador ha presionado el botón o tecla de golpe (para ejemplo en un teclado se ha asignado la tecla del número uno).

PATADA ANIMACIÓN

Este parámetro es de tipo "AnimationClip" y recibe la animación del personaje cuando este *da una patada* para defenderse de algún enemigo o atacarlo, es decir que el video-jugador ha presionado el botón o tecla de patada (para ejemplo en un teclado se ha asignado la tecla del número dos).

AGACHAR ANIMACIÓN

Este parámetro es de tipo "AnimationClip" y recibe la animación del personaje cuando este *se agacha*, es decir que el video-jugador ha presionado el botón o tecla de agachar (para ejemplo en un teclado se ha asignado la tecla del número tres).

CUBRIR ANIMACIÓN

Este parámetro es de tipo "AnimationClip" y recibe la animación del personaje cuando este *se cubre* para defenderse de algún enemigo que lo ataca, es decir que el video-jugador ha presionado el botón o tecla de cubrir (para ejemplo en un teclado se ha asignado la tecla del número cuatro).

MORIR ANIMACIÓN

Este parámetro es de tipo "AnimationClip" y recibe la animación del personaje cuando este *debe morir* producto de que algún enemigo en el ataque lo ha dejado sin vida.

CAMINAR VELOCIDAD ANIMACION

Este parámetro es de tipo float, y controla la velocidad de la animación de caminar, aunque esta animación tiene una velocidad dada o predeterminada cuando el diseñador de animación la creó, con este parámetro se la puede acelerar o desacelerar a conveniencia.

TROTAR VELOCIDAD ANIMACION

Este parámetro es de tipo float es decir que soporta valores decimales, y controla la velocidad de la animación de trotar, que no es una animación enviada como

parámetro sino que la misma animación de caminar es acelerada, aunque esta animación tiene una velocidad dada o predeterminada cuando el diseñador de animación la creó, con este parámetro se la puede acelerar o desacelerar a conveniencia.

CORRER VELOCIDAD ANIMACION

Este parámetro es de tipo float es decir que soporta valores decimales, y controla la velocidad de la animación de correr, aunque esta animación tienen una velocidad dada o predeterminada cuando el diseñador de animación la creó, con este parámetro se la puede acelerar o desacelerar a conveniencia.

SALTAR VELOCIDAD ANIMACION

Este parámetro es de tipo float es decir que soporta valores decimales, y controla la velocidad de la animación de saltar, aunque esta animación tienen una velocidad dada o predeterminada cuando el diseñador de animación la creó, con este parámetro se la puede acelerar o desacelerar a conveniencia.

TIERRA VELOCIDAD ANIMACION

Este parámetro es de tipo float es decir que soporta valores decimales, y controla la velocidad del desplazamiento del piso o tierra, aunque las animaciones de movimiento tienen una velocidad dada o predeterminada cuando el diseñador de animación la creó, con este parámetro se puede sincronizar ese movimiento con el desplazamiento del personaje según el desplazamiento del piso; por eso se la puede acelerar o desacelerar a conveniencia.

VELOCIDAD CAMINAR

Este parámetro es de tipo float es decir que soporta valores decimales, y controla la velocidad de la animación de caminar con respecto al desplazamiento de la cámara para dar la sensación de avance, con este parámetro se la puede asignar una velocidad de partida cuando el personaje está caminando.

VELOCIDAD TROTAR

Este parámetro es de tipo float es decir que soporta valores decimales, y controla la velocidad de trote al momento de haber estado caminando por un lapso de tiempo dado en “SEGUNDOS INICIO TROTAR” y define el desplazamiento de la cámara para

dar la sensación de avance, se recomienda que este parámetro sea mayor al de "VELOCIDAD CAMINAR".

VELOCIDAD CORRER

Este parámetro es de tipo float es decir que soporta valores decimales, y controla la velocidad de la animación de correr con respecto al desplazamiento de la cámara para dar la sensación del movimiento cuando el personaje está corriendo, se recomienda que este parámetro sea mayor al de "VELOCIDAD TROTAR".

CONTROL ACELERACION EN AIRE

Este parámetro es tipo de float es decir soporta valores decimales, y controla la aceleración del personaje cuando este se mueve y está en el aire, la variable privada que se modifica internamente es inAirVelocity según la siguiente formula, que incluye la Dirección y el tiempo.

```
inAirVelocity += targetDirection.normalized * Time.deltaTime *
ControlAceleracionEnAire;
```

ALTURA SALTO

Este parámetro es de tipo float es decir que soporta valores decimales, y controla la altura del salto del personaje asociado al botón de "Salto", que en el teclado está asociado a la tecla "barra espaciadora".

GRAVEDAD

Este parámetro es de tipo float es decir que soporta valores decimales, y controla el valor de gravedad que se va a tener en el escenario o nivel, la gravedad tienen mucha incidencia en el juego para hacerlo más realista en los saltos y movimientos que involucren la altura como las caídas.

SUAVIZAR VELOCIDAD

Este parámetro es de tipo float es decir que soporta valores decimales, y controla la velocidad cuando se está descendiendo, tomando en cuenta el tiempo, este aplica un suavizado al descender a la velocidad de movimiento dado por las siguientes formulas.

```
Var curSmooth = SuavizarVelocidad * Time.deltaTime;
moveSpeed = Mathf.Lerp(moveSpeed, targetSpeed, curSmooth);
```

VELOCIDAD ROTACION

Este parámetro es de tipo float es decir que soporta valores decimales, y controla la velocidad de rotación del personaje cuando este gira usando botón "adelanta" y al mismo tiempo botón "izquierda" o "derecha"; este valor tiene incidencia con el radio de giro del personaje, a menor valor el radio de giro es menor; a mayor valor el radio de giro también es mayor; este radio denota velocidad de rotación.

SEGUNDOS INICIAR a TROTAR

Este parámetro es de tipo float es decir que soporta decimales, y establece después de cuantos segundos el personaje pasa de caminar a trotar, su valor predeterminado es de 3 segundos.

PUEDE SALTAR

Este parámetro es de tipo boolean, es decir solo permite "verdadero" o "falso"; para la interfaz de unity el parámetro se ve como una casilla de verificación; establece si el personaje puede o no saltar en el escenario o nivel.

5.4.1 CODIGO FUENTE DE THIRDPERSONCONTROLLER

El controlador de tercera persona de Santiago hace uso de otro script en su primera línea llamado "CharacterController", tratado como un componente requerido tal como se muestra en la siguiente línea:

```
@script RequireComponent(CharacterController)
```

Al controlador del personaje se le definen las animaciones como parámetros de entrada de tipo "AnimationClip"; de esta forma es más fácil la asignación de las animaciones para ser controladas dentro del código.

```
public var ReposoAnimacion: AnimationClip;
public var CaminarAnimacion: AnimationClip;
public var CorrerAnimacion: AnimationClip;
public var SaltoPoseAnimacion: AnimationClip;
// ********************************************
// Nuevas animaciones como parámetro
      public var puneteAnimation : AnimationClip;
```

```
      public var patadaAnimation : AnimationClip;
      public var agacharAnimation : AnimationClip;
      public var cubrirAnimation : AnimationClip;
      public var morirAnimation : AnimationClip;
// ************************************************
```

Luego se definen los parámetros para controlar las velocidades de las distintas animaciones del personaje como caminar, trotar, correr etc. Y sus valores predeterminados.

```
public var CaminarVelocidadAnimacionMax : float = 0.75;
public var TrotarVelocidadAnimacionMax : float = 1.0;
public var CorrerVelocidadAnimacionMax : float = 1.0;
public var SaltarVelocidadAnimacionMax : float = 1.15;
public var TierraVelocidadAnimacionMax : float = 1.0;
```

Como el script original de unity solo tenía cuatro estados para su variable CharacterState, se ha agregado cinco estados más para nuestro personaje como es puñete, patada, agachar entre otras dando un total de 10 de estados para Santiago, estos estados son importantes porque según su valor se asignara a la variable "var_animation" de tipo Animation la animación del personaje que corresponda.

```
private var _animation : Animation;
enum CharacterState {
      Idle = 0,
      Walking = 1,
      Trotting = 2,
      Running = 3,
      Jumping = 4,
// ************************************************
// Nuevos estados
      punete = 5,
      patada = 6,
      agachar= 7,
      cubrir = 8,
      morir = 9,
// ************************************************
}
private var _characterState : CharacterState;
```

5.5 ESTADOS DEL PERSONAJE

Los código presentados en este apartado pueden ser consultados de manera detallada en el **Anexo 1** al final de este documento.

5.5.1 CAMINAR

El personaje dispone de una animación denominada "**Sant_cominar**" donde previamente el diseñador de animación ha creado la secuencia de movimiento de manos, pies y dorso para dar la sensación de caminar, la misma que se usará como recurso y es asignado al parámetro de entrada llamado "**CaminarAnimacion**" de tipo AnimationClip.

CONTROLES

Los controles asociados son las teclas del cursor que disponen del movimiento tanto adelante, atrás, izquierda y derecha, pero estos mueven la cámara ubicada detrás del personaje para su dirección que se almacena en la variable targetDirection.

```
var forward = cameraTransform.TransformDirection(Vector3.forward);
forward.y = 0;
forward = forward.normalized;
var right = Vector3(forward.z, 0, -forward.x);
var v = Input.GetAxisRaw("Vertical");
var h = Input.GetAxisRaw("Horizontal");
var targetDirection = h * right + v * forward;
```

ANIMACION

La secuencia de animación del personaje esta creado dentro del mismo prefab denominado "Sant_cominar".

MOVIMIENTO EN EL ESCENARIO

El código que dispone el movimiento en el escenario para la acción de caminar y la animación son los siguientes:

Para el movimiento del personaje con el controlador se almacena en la variable **movement**.

```
// La variable movement toma la dirección, la velocida y un vector que
guarda X, //Y, Z
```

```
var movement = moveDirection * moveSpeed + Vector3 (0, verticalSpeed,
0) + inAirVelocity;

// movement usa el tiempo y no frame
movement *= Time.deltaTime;
// Se obtiene la componente que guarda el controlador
var controller : CharacterController =
GetComponent(CharacterController);

// Bandera para saber si ha ocurrido una colisión
collisionFlags = controller.Move(movement);
```

Para ejecutar la animación según el estado del personaje se usa el siguiente código.

```
// Se evalúa el estado del personaje
if(_characterState == CharacterState.Walking) {
_animation[CaminarAnimacion.name].speed =
Mathf.Clamp(controller.velocity.magnitude, 0.0,
CaminarVelocidadAnimacionMax);
_animation.CrossFade(CaminarAnimacion.name);
}
```

SONIDO

Se dispone del recurso de audio asociado al personaje con los pasos de caminar, dentro del código se lo invoca de la siguiente manera.

```
if(_characterState == CharacterState.Walking) {
      audio.Play();
}
```

5.5.2 TROTAR

El personaje no dispone de una animación para la acción de trote ya que es contralado acelerando la animación de "Sant_caminar" después de 3 segundos.

CONTROLES

Los controles asociados son las teclas del cursor que disponen del movimiento tanto adelante, atrás, izquierda y derecha, pero el personaje trota una vez cumplido los segundos en el parámetro "**SegundosInicioTrotar**", su valor por defecto es 3 segundos.

ANIMACION

El estado trotar no dispone de animación, porque se usa la animación de caminar acelerándola.

MOVIMIENTO EN EL ESCENARIO

El código que controla que el personaje trote transcurrido unos segundos según el parámetro "SegundosInicioTrotar" es:

```
if (Time.time - SegundosInicioTrotar> walkTimeStart){
      targetSpeed *= VelocidadTrotar;
      _characterState = CharacterState.Trotting;
}
```

La animación es controlada evaluando el estado del personaje con el código:

```
if(_characterState == CharacterState.Trotting) {
// Clamp Restringe un valor entre un mínimo y un máximo, sean floats o
ints, es //usado para la animación suave llamada con el método
CrossFade
_animation[CaminarAnimacion.name].speed =
Mathf.Clamp(controller.velocity.magnitude, 0.0,
TrotarVelocidadAnimacionMax);
_animation.CrossFade(CaminarAnimacion.name);
}
```

SONIDO

Se dispone del recurso de audio asociado al personaje con los pasos de trotar, dentro del código se lo invoca de la siguiente manera.

```
if(_characterState == CharacterState. Trotting) {
      audio.Play();
}
```

5.5.3 CORRER

El personaje dispone de una animación denominada "**Sant_correr**" donde previamente el diseñador de animación ha creado la secuencia de movimiento de manos, pies y dorso para dar la sensación de correr, la misma que se usará como recurso y es asignado al parámetro de entrada llamado "CorrerAnimacion" de tipo AnimationClip.

CONTROLES

Los controles asociados son las teclas del cursor que disponen del movimiento tanto adelante, atrás, izquierda y derecha, pero la acción de correr es presionando previamente la tecla "**SHIFT**" o mayúscula, estos mueven la cámara ubicada detrás del personaje para su dirección que se almacena en la variable targetDirection.

```
if (Input.GetKey (KeyCode.LeftShift) | Input.GetKey
(KeyCode.RightShift)){

targetSpeed *= VelocidadCorrer;

_characterState = CharacterState.Running;
}
```

ANIMACION

La secuencia de animación del personaje esta creado dentro del mismo prefab denominado "Sant_correrr".

MOVIMIENTO EN EL ESCENARIO

El código que dispone el movimiento en el escenario para la acción de correr y la animación es el siguiente:

```
if(_characterState == CharacterState.Running) {
_animation[CorrerAnimacion.name].speed =
Mathf.Clamp(controller.velocity.magnitude, 0.0,
CorrerVelocidadAnimacionMax);
_animation.CrossFade(CorrerAnimacion.name);
}
```

SONIDO

Se dispone del recurso de audio asociado al personaje con los pasos de correr, dentro del código se lo invoca de la siguiente manera.

```
if(_characterState == CharacterState. Running) {
      audio.Play();
}
```

5.5.4 SALTAR

El personaje dispone de una animación denominada "**Sant_salto_aire**" donde previamente el diseñador de animación ha creado la secuencia de movimiento de

manos, pies y dorso para dar la sensación de salto, la misma que se usará como recurso y es asignado al parámetro de entrada llamado "**SaltoPoseAnimacion**" de tipo AnimationClip.

CONTROLES

El control asociado para el salto es la tecla "**barra espaciadora**", pero este está condicionado al parámetro de entrada "**PuedeSaltar**" si este está marcado, la función que controla el salto del personaje se llama **ApplyJumping()** junto con **CalculateJumpVerticalSpeed()**.

```
if (PuedeSaltar&& Time.time < lastJumpButtonTime + jumpTimeout) {
  verticalSpeed = CalculateJumpVerticalSpeed (AlturaSalto);
   CalculateJumpVerticalSpeedSendMessage("DidJump",
   SendMessageOptions.DontRequireReceiver);
}
```

Para evaluar su el personaje ha saltado se utiliza las siguientes líneas de código, dentro de la función **ApplyGravity()**.

```
var jumpButton = Input.GetButton("Jump");
if (jumping && !jumpingReachedApex && verticalSpeed <= 0.0){
      jumpingReachedApex = true;
      SendMessage("DidJumpReachApex",SendMessageOptions.DontRequireRec
eiver);}
if (IsGrounded ())
  verticalSpeed = 0.0;
else
  verticalSpeed -= Gravedad * Time.deltaTime;
```

ANIMACION

La secuencia de animación del personaje esta creado dentro del mismo prefab denominado "**Sant_salto_correr**".

MOVIMIENTO EN EL ESCENARIO

El código que dispone el movimiento en el escenario para la acción de saltar está en el método **ApplyJumping()** y la animación es evaluada de la siguiente manera en el código:

```
if(_characterState == CharacterState.Jumping) {
    if(!jumpingReachedApex) {
   _animation[SaltoPoseAnimacion.name].speed =
SaltarVelocidadAnimacionMax
   _animation[SaltoPoseAnimacion.name].wrapMode =
WrapMode.ClampForever;
   _animation.CrossFade(SaltoPoseAnimacion.name);
} else {
  _animation[SaltoPoseAnimacion.name].speed = -
TierraVelocidadAnimacionMax;
  _animation[SaltoPoseAnimacion.name].wrapMode =
WrapMode.ClampForever;
  _animation.CrossFade(SaltoPoseAnimacion.name);
    }
}
```

Se verifica si el salto ha sido alcanzado o no, y determinar la velocidad con respecto a la suelo.

SONIDO

Se dispone del recurso de audio asociado al personaje con los pasos de salto, dentro del código se lo invoca de la siguiente manera.

```
if(_characterState == CharacterState. Jumping) {
      audio.Play();
}
```

5.5.5 PUÑETE, PATADA, AGACHAR, CUBRIR Y MORIR

Las acciones de puñete, patada, agachar, cubrir y morir el personaje dispone de una animación para cada una de ellas que previamente el diseñador de animación ha creado la secuencia de movimiento de manos, pies y dorso según sea el caso, las mismas que se usarán como recurso y serán asignadas a los parámetros de entrada llamado **puneteAnimation, patadaAnimation, agacharAnimation, cubrirAnimation y morirAnimation** todas del tipo **AnimationClip**.

CONTROLES

Para efectos de prueba se ha utilizado las teclas de los números del 1 al 5 para los controles usando KeyCode.Alpha1, 2, 3, 4, 5, pero se pueden asignar a los “Input” Fire1, Fire2, Fire3 etc.

```
…
if (Input.GetKey (KeyCode.Alpha1)){
  _characterState = CharacterState.punete;
} else if (Input.GetKey (KeyCode.Alpha2)){
 _characterState = CharacterState.patada;
} …
```

ANIMACION

La secuencia de animaciones de estas acciones en el personaje se llama.

1. Sant_puño.
2. Sant_patada.
3. Sant_agachar.
4. Sant_cubrir.
5. Sant_muerte.

MOVIMIENTO EN EL ESCENARIO

El código que dispone cada acción en el escenario para la ejecución de las animaciones es el siguiente:

```
…
if(_characterState == CharacterState.punete)
{_animation.CrossFade(puneteAnimation.name);
}   else if(_characterState == CharacterState.patada) {
_animation.CrossFade(this.patadaAnimation.name);
…
```

SONIDO

Se dispone del recurso de audio asociado al personaje para las diferentes acciones.

```
if(_characterState == CharacterState.punete) {
      audio.Play();
}
```

5.6 FUNCIONES PRINCIPALES

5.6.1 FUNCION AWAKE

En la siguiente función "**Awake**" que se ejecuta una solo vez al inicio, se la utiliza para verificar que el personaje tenga una animación y que se hayan asignado las otras animaciones del personaje de lo contrario se asigna nulo y se envía un mensaje correspondiente a la ventana de "LOG" de Unity.

5.6.2 FUNCIÓN UPDATESMOOTHEDMOVIMIENTDIRECTION

La función **UpdateSmoothedMovimientDirection**, se encarga de actualizar y controlar el movimiento del personaje según la dirección del mismo, se valida si este está tocando el suelo con la variable **grounded**; para el control del movimiento se verifica qué botón se está presionando según la acción asignada el estado del jugador como puñete, patada etc; el código considera si el personaje está saltando o moviéndose para el seguimiento de la cámara, para ello se usan las variables **jumping** y **isMoving**.

5.6.3 CODIGO PARA EL SALTO DEL PERSONAJE.

La función **ApplyJumping** se encarga de aplicar el salto al personaje, controla el tiempo que ha transcurrido desde el último salto y si el personaje está tocando el suelo, se valida si el parámetro de entrada **PuedeSaltar** está marcado.

5.6.4 FUNCIÓN APPLYGRAVITY

La función **ApplyGravity** se encarga de aplicar la gravedad al personaje en el juego, se verifica si el mismo se está controlando por parte del jugador con la variable **isControllable**; para luego saber si esta saltando o moviéndose; el movimiento valida si esta en el suelo para asignar la velocidad en sentido vertical.

5.6.5 FUNCIÓN CALCULATEJUMPVERTICALSPEED

La función **CalculateJumpVerticalSpeed**, se encarga de calcular el salto del personaje tomando como base el valor de gravedad y la altura permitida de salto.

5.6.6 FUNCIÓN UPDATE

La función **Update** es la más importante de todas, ya que se ejecuta por cada fotograma (fotograma) que avanza en el juego, es comparable al control **timer** en otros lenguajes de programación, aquí se valida si se controlando al personaje, si esta saltando, se aplica la gravedad, el movimiento con su dirección y suavizado; otro aspecto de esta función es controlar la animación actual según el estado del personaje

denotado en la variable **CharacterState**, aquí se puede notar que el script de unity no ha querido hacer otro para el controlador de las animaciones por estado; pero este cumple su función.

Conclusiones

Desde los inicios de las computadoras de la mano del profesor inglés, Charles Babbaje, pasando las computadoras Mark 1, la ENIAC son la base tanto en hardware como en software para los video-juegos actuales.

Jugar siempre ha sido parte importante en la vida de los seres humanos, los video-juegos son una forma más de dar entretenimiento usando como componente importante un visualizador de gráficos, es decir su impacto entra por los ojos de video-jugador, usa la interacción o jugabilidad entre una o varias personas.

Los video-juegos aportan entre sus características más palpables la interactividad, entretenimiento, jugabilidad, inmersión, simulación entre otras. En la historia de los video-juegos está presente la importancia militar de los países desarrollados, mucha tecnología fue aportada en la guerra fría y la segunda guerra mundial, los video-juegos no estuvieron exentos de crisis y malos momentos tal es así que la historia recoge la crisis sufrida en el año de 1983.

Tomar en cuenta los diferentes géneros desde los Arcades, juegos de estrategia, narrativos, Aventuras graficas o de acción, y su impacto en los niños y jóvenes en cuanto a la violencia que estos pueden cargar, para ello se han creado clasificaciones para que los padres puedan saber con antelación el tipo de juegos a comprar; puesto que los primeros juegos de video carecían de esta violencia, tanto los públicos, el mercado y los mismos video-jugadores han cambiado y se han vuelto más exigentes, estas clasificaciones contribuyen a que ese impacto de los video-juegos en la sociedad sea mínimo, pero no evita el desarrollo de video-juegos violentos o inclusión de temas sexuales o erótico en ellos.

Es importante hacer notar que dentro de la evolución del hardware de las videoconsolas por sus distintas generaciones (7 en total hasta la actualidad) siempre se ha notado un aumento en sus capacidades de procesamiento, capacidades visuales y sonoras hasta el medio de soporte donde están almacenados los juegos, incorporación de accesorios y por supuesto nuevas tecnologías del momento, sin dejar a un lado las consolas portátiles y sus generaciones, las computadores como medio de entretenimiento, los teléfonos básicos hasta llegar a los inteligentes; son hoy por hoy medios donde el usuario puede jugar.

En el software que se usa en el desarrollo de video-juegos uno de los puntos importantes es haber analizado que arquitectura de software se va a usar, porque estos no son sistemas comerciales, así que se hacen presentes tecnologías a APIS graficas como OpengGL y Direct3D; el uso de arquitecturas dirigidas por datos y componentes, y aunque se usan “entidades” o “clases”, concepto que deriva de la programación orientada a objeto, pero el programador cumplen ciertas reglas de ella y otras evita llevarlas al extremo como es el caso de la “herencia”, los lenguajes pasan a hacer “script” manteniendo todas las técnicas 2D y 3D.

Al final es importante recalcar que los primeros video-juegos que se programaron en lenguajes como ensamblador o C/C++ han hecho camino a una nueva forma de crear video-juegos a través de los “Motores 3D”, un tema que se menciona desde 1993 con un mayor énfasis cuando salió al mercado el mítico juego de Doom que uso un motor 3D potenciando el desarrollo de otros y marcando un nuevo modelo de programación de juegos.

Se han visto nacer otros motores de mejor calidad como Unreal Engine, Unity3D, Blender, ID Tech entre otros, mejorando los tiempos de desarrollo, simplificando la lógica del juego y una mejor distribución en equipos de trabajo entre artistas y programadores.

Recomendaciones

Esta tesis ha sido en primera instancia una recopilación de datos bibliográficos para dar a conocer tanto la historia de los video-juegos, su evolución a través de generaciones, los aspectos técnicos en hardware hasta la creación de ellos usando software de computadoras; dejando entrever que las primeras generaciones usaban lenguajes de programación tradicionales que ameritaban un aprendizaje exhaustivo para sacar provecho de ellos.

Los programadores de sistemas tradicionales como los comerciales, bancarios, entre otros; tienen la base para convertirse en programadores de esta nueva industria; conociendo las diferentes técnicas, arquitecturas, lenguajes y simplificando e integrando todo en un entorno de desarrollo llamado “Motor de Juegos” o “GameEngine” por su término en inglés.

Los motores 3D han marcado un antes y un después en el desarrollo de video-juegos y entre sus virtudes podemos destacar el concepto de multiplataforma, es decir la capacidad que tienen estos programas en publicar el mismo juego para varias plataformas como Móviles, Videoconsolas, PC o computadoras Macintosh; el uso de licencias comerciales donde se tiene un abanico desde los más caros como UDK a los más accesibles como Unity3D sin menospreciar a los motores de código abierto o opensource.

Los motores 3D han contribuido a cambiar la forma tradicional de usar los lenguajes de programación para el desarrollo de juegos al usar lenguajes de “scripting”, donde el código es incorporado a los objetos o entidades llamados prefabricados, se usa una sintaxis derivadas de C/C++ hasta nuevas sintaxis no tan comunes como Boo o Python, pero que en todo caso harán mejor o peor el entrenamiento de los nuevos programadores.

Estos entornos usan el concepto de activos o Assets que se importan ayudando a que los tiempos de desarrollo sean más cortos; además de separar la carga de roles entre programadores y artistas; los activos pueden ser modelados 3D, imágenes para texturas, sonido y video.

Es importante mencionar que la documentación de los diferentes motores esta asequible a los programadores a través de internet, con una vasta información, foros donde otros programadores van dejando sus códigos y experiencia, libros desde niveles básicos hasta llegar a los más avanzados, es decir todo al alcance para poder empezar.

Al momento de escoger un motor de juegos se deberán tomar muchos factores a considerar como los costos de licenciamiento, tiempos de desarrollo, tipo de juego a crear, género, plataforma entre otras variables, pero esta tesis quiere encaminar a ese programador que con algo de experiencia puede fácilmente acoplarse e estos nuevos entornos, la mejor opción es Unity 3D, por su facilidad de manejar escenas, costos asequibles tanto para desarrollo video-juegos indie [94] como para una pequeña desarrolladora, el lenguaje de este motor es Javascript vastamente popular, C# que a través de la suite Microsoft Visual Studio ha ganado muchos adeptos, y Boo poco conocido solo decir que es derivado de Pyton.

Unity3D permite publicar el mismo video-juego para consolas como PS3, Xbox y Wii, Móviles con iOS o Android, y las plataformas tradicionales como PC o Macintosh de manera fácil e intuitiva; posee una versión gratuita muy funcional que deja publicar para PC o Macintosh, así que no hay un desembolso previo, si el motor es el correcto puede pagar la licencia de la plataforma a usar.

Además como todo "motor de juego" incorpora otros motores como es el caso de la física y colisiones, sistemas de partículas, edición de terrenos, gestión del audio o otras cualidades como edición de escenas, creación de niveles, prueba en caliente, shaders en las gráficas; que son algunos de los aspectos que un video-juego puede requerir.

La programación de la lógica del juego es mucho más sencilla tanto con los lenguajes de tipo script que estas herramientas incorporan como las APIs que se disponen para ayudar en el desarrollo, pero si el programador o empresa desarrolladora desea crear juegos de gran envergadura esta tesis ha planteado otras alternativas a considerar en el mercado como UDK, Blender, Id Tech entre otros.

[94] Proceso de crear video-juegos sin el apoyo financiero de una distribuidora de video-juegos además de un pequeño equipo de personas.

Bibliografía

1. 20minutos.es. (2008). *Todas las consolas de la historia (Videoconsolas de todas las generaciones).* Recuperado el 14 de Mayo de 2012, de http://listas.20minutos.es/lista/todas-las-consolas-de-la-historia-videoconsolas-de-todas-las-generaciones-301580/

2. Albornoz Figueroa, R. (10 de Enero de 2007). *Conceptos Básicos para el Desarrollo de Video-juegos 2D*. Recuperado el 7 de Mayo de 2012, de http://www.losersjuegos.com.ar/referencia/articulos/conceptos_basicos

3. Barrios, C. (2009). Análisis de diferentes lenguajes y herramientas para el Desarrollo de Video-juegos. Guatemala.

4. Berens, K., & Howard, G. (2008). *The Rough Guide to Videogame.* London: Rough Guides.

5. CURRAN, R. (Dirección). (2007). *La era del Videogame* [Película].

6. Donovan, T. (2010). *Replay The History of Video Games.* Yellow Ant.

7. Entertainment Software Rating Board. (1994). *ESRB*. Recuperado el 22 de 04 de 2012, de http://www.esrb.org/

8. Funversion. (2006). *Video-juegos, la revolución del entretenimiento. Más de 30 años de consola.* Recuperado el 14 de 05 de 2012, de http://funversion.universia.es/video-juegos/reportaje/30anyosconsolas.jsp

9. Gómez Martín, M. A. (2007). *Arquitectura y metodología para el desarrollo de sistemas educativos basados en video-juegos.* Madrid: Tesis Doctoral - Universidad Complutense de Madrid.

10. Indicelatino. (2006). *HISTORIA DE LOS VIDEO-JUEGOS - VIDEOCONSOLAS PORTATILES.* Recuperado el 14 de Mayo de 2012, de http://indicelatino.com/juegos/historia/portatiles/

11. Indicelatino. (2007). *Historia de los video-juegos.* Recuperado el 14 de Mayo de 2012, de http://indicelatino.com/juegos/historia/

12. Informatica. (2009). *Historia de los video-juegos.* Recuperado el 14 de Enero de 2012, de http://archivo.laprensa.com.ni/archivo/2001/mayo/22/informatica/articulos/

13. Juul, J. (2003). *The Game, the Player, the World*. (C. M. J., Editor) Recuperado el 30 de Abril de 2012, de http://www.jesperjuul.net/text/gameplayerworld/

14. Kent, S. L. (2008). *The Ultimate History of Video Games: From Pong to Pokemon.* New York: Three Rivers Press.

15. Kushner, D. (2003). *Masters of Doom: How Two Guys Created an Empire and Transformed Pop Culture.* New York: Random House.

16. Loguidice, B., & Barton, M. (2009). *VINTAGE GAMES - An Insider Look at the History of Grand Theft Auto, Super Mario, and the Most Infl uential Games of All T ime.* London: Focal Press.

17. Martí, J. (2010). *Marketing y video-juegos.* Madrid: ESIC.

18. Pan European Game Information. (2003). *PEGI*. Recuperado el 22 de 04 de 2012, de http://www.pegi.info/es/

19. Pérez Quintanar, J. R. (30 de diciembre de 2004). *Historia de los Video-juegos.* Recuperado el 14 de mayo de 2012, de http://www.rinconsolero.com/Rinconsolero.V2/historia_de_los_video-juegos.htm

20. Perron, B., & Wolf, M. J. (2009). *The Video Game Theory Reader 2.* New York: Routledge.

21. Piccoli, M. F. (Febrero de 2012). Computación de Alto Desempeño en GPU. Argentina, Río Cuarto,.

22. Poole, S. (2000). *Trigger Happy VIDEOGAMES AND THE ENTERTAINMENT REVOLUTION.* New York: Arcade Publishing.

23. Unity Technologies. (2007). *Building a 3D Platform Game in Unity 2.0 traducción por c2estudio.* Recuperado el 14 de Mayo de 2012, de 3D Platform Game: http://unity3d.com/support/resources/tutorials/3d-platform-game

24. Unityscripts. (15 de 10 de 2011). *CLASE MONOBEHAVIOUR (IV).* Recuperado el 05 de 22 de 2012, de http://unityscripts.blogspot.com/2011/10/44-clase-monobehaviour-iv.html

25. Ward, J. (29 de Abril de 2008). *GamecareerGuide*. Recuperado el 01 de Mayo de 2012, de http://www.gamecareerguide.com/features/529/what_is_a_game_.php

26. Wikipedia. (2012). *Videoconsola portátil*. Recuperado el 14 de Mayo de 2012, de http://es.wikipedia.org/wiki/Videoconsola_port%C3%A1til

27. Wikipedia. (2012). *Videoconsolas de cuarta generación*. Recuperado el 14 de Mayo de 2012, de http://es.wikipedia.org/wiki/Videoconsolas_de_cuarta_generaci%C3%B3n

28. Wikipedia. (2012). *Videoconsolas de primera generación*. Recuperado el 14 de Mayo de 2012, de http://es.wikipedia.org/wiki/Videoconsolas_de_primera_generaci%C3%B3n

29. Wikipedia. (2012). *Videoconsolas de quinta generación*. Recuperado el 14 de Mayo de 2012, de http://es.wikipedia.org/wiki/Videoconsolas_de_quinta_generaci%C3%B3n

30. Wikipedia. (2012). *Videoconsolas de segunda generación*. Recuperado el 14 de Mayo de 2012, de http://es.wikipedia.org/wiki/Videoconsolas_de_segunda_generaci%C3%B3n

31. Wikipedia. (2012). *Videoconsolas de séptima generación*. Recuperado el 14 de Mayo de 2012, de http://es.wikipedia.org/wiki/Videoconsolas_de_s%C3%A9ptima_generaci%C3%B3n

32. Wikipedia. (2012). *Videoconsolas de sexta generación*. Recuperado el 14 de Mayo de 2012, de http://es.wikipedia.org/wiki/Videoconsolas_de_sexta_generaci%C3%B3n

33. Wikipedia. (2012). *Videoconsolas de tercera generación*. Recuperado el 14 de Mayo de 2012, de http://es.wikipedia.org/wiki/Videoconsolas_de_tercera_generaci%C3%B3n

Anexo 1

DIFUSIÓN DE LOS GÉNEROS, CARACTERÍSTICAS, PLATAFORMAS, HERRAMIENTAS DE DESARROLLO QUE SE USAN EN LA INDUSTRIA DE LOS VIDEO-JUEGOS

1.- Generalidades

Las computadoras hoy en día son parte de nuestra vida cotidiana, entendamos o no como funcionan internamente; incluso desconocer cómo fueron creados los programas que usamos o nos divierten, este invento no tiene más allá de medio siglo de historia. La idea se la debemos a Charles Babbage[95], profesor inglés que acompañado de su alumna predilecta Ada Byron[96] se involucraron en un proyecto definitivamente visionario para aquella época, por los años de 1830 la máquina analítica para ejecutar programas de tabulación o computación que nunca llegó a realizarse de manera operativa, pero de alguna manera la necesidad militar de hacer cálculos matemáticos para trayectorias balísticas impulsa la creación de una máquina basada en las ideas de Babbage y da el inicio a la creación de una serie de computadores enormes con propósitos muy específicos; hoy en día las computadoras son la base para el desarrollo de video-juegos; así que todo comienza con la historia de las computadoras.

Por definición general un “video-juego” es un software como programa informático, creado para el entretenimiento, que se basa en la interacción o jugabilidad entre una o varias personas denominados jugadores, un controlador o mando; y un aparato electrónico que ejecuta dicho video-juego; este dispositivo electrónico puede ser una computadora, videoconsola de mesa o portátil, máquina recreativa, un teléfono móvil entre otros; los cuales son conocidos como "plataformas". El término "video" en la palabra "video-juego" se refiere a un visualizador de gráficos rasterizados. Además estos poseen ciertas características como la interactividad, el entretenimiento, la jugabilidad, simulación, virtualidad, la Inmersión y su Multiplataforma (computador, videoconsola, televisión, teléfono móvil, etc.).

[95] Se le considera como una de las primeras personas en concebir la idea de lo que hoy llamaríamos una computadora, por lo que se le considera como "El Padre de la Computación"

[96] Considerada como la primera programadora, desde que escribió la manipulación de los símbolos, de acuerdo a las normas para una máquina de Charles Babbage que aún no había sido construida.

2.- Base teórica de los videojuegos

El comienzo de los video-juegos se remonta a finales de los años 40, al borde de la conocida "GUERRA FRIA"[97] y ver como la tecnología surgía en sus inicios desatando el pánico colectivo, esto fue el resultado de las dilatadas tensiones diplomáticas entre el mundo comunista y el liberal.

A medida que los preparativos para la guerra fría se intensificaban un joven físico que trabajaba en un laboratorio nuclear cambiaba para siempre la idea de pulsar un botón. En 1958 William Higinbotham que había trabajado en la primera bomba atómica convirtió dos líneas normales y corrientes; y una pelotita, en la primera experiencia interactiva con un computador llamada "TENIS PARA DOS" en inglés "Tennis for Two" siendo este el primer video-juego de la historia.

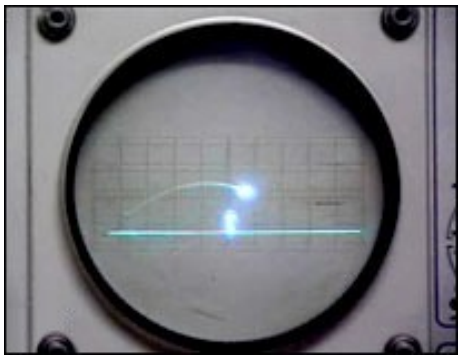

Figura 1 ***Tennis for two*** **desarrollado por Willian Higinbotham en 1958**

Fuente: http://upload.wikimedia.org/wikipedia/commons/a/a9/

La guerra fría se convirtió en una carrera espacial. Steve Ruesell un programador del M.I.T. dio el siguiente paso en los video-juegos con el universo virtual de "SPACEWAR", creado en un computador PDP-1[98]; Si "Tennis for Two" era una inocente distracción para evadirse de la guerra fría el juego de Steve Ruesell era un claro reflejo de la carrera espacial y del miedo mundial a la guerra.

Posteriormente Ralph Baer conocido como el Thomas Edison de los video-juegos, creó la primera videoconsola conocida como la "MAGNAVOX ODYSSEY". Por aquel entonces los comerciales en la televisión que hacían alusión a la empresa

[97] Se denominó FRIA porque nunca se llegó a un conflicto abierto.
[98] El PDP-1 era considerado el primer computador con categoría de computador personal que se podía encender con un interruptor, y se lo consideraba un computador pequeño con el tamaño de una refrigeradora.

MAGNAVOX que había adaptado un simulador de juegos electrónicos que convertirá su televisor en una zona de juegos. Ralph Bear era un joven ingeniero con experiencia en tecnología de televisión y fue contratado para desarrollar tecnología militar durante la guerra fría, pero pronto empezó a trabajar en la "Brawnbox" o caja marrón, el prototipo de la primera consola familiar la "MAGNAVOX ODYSSEY".

El sector de los video-juegos siempre ha estado muy ligado al ejército como cualquier otro sector tecnológico, fueron los militares quienes investigaron con computadores en la segunda guerra mundial para calcular la trayectoria de los misiles, así que desde el principio de la historia de los video-juegos siempre ha estado muy ligado al campo militar. A finales de los años 60 y comienzos de los 70 el movimiento contracultural generó una nueva rama en los video-juegos y Nolan Bushnell fundador de ATARI fue uno de los primeros en verlo.

Cuando la gente discute sobre quien creó los video-juegos hay muchísimos argumentos, pero nadie cuestiona quien creó el negocio de los video-juegos y ese fue Nolan Bushnell, porque él fue el primero en darse cuenta que con ello se podía ganar dinero. Nolan Bushnell recluto a su antiguo compañero Al Arcon para programar el primer juego de salón recreativo, le encargo su primer proyecto donde tenía que construir un sencillo juego de ping-pong con una pelota que se moviese para marcar, cuando ATARI lanzo el juego PONG en el año de 1972 nació la industria del video-juego, así que PONG fue el comienzo de todo, fue el primer juego que era accesible a todos aquellos que no trabajaban en un laboratorio.

Higinbotham, Ruesell, Baer y Bushnell fueron los pioneros que conscientemente o no canalizaron el miedo colectivo hacia sus creaciones virtuales, era obvio que el mundo lo estaba deseando y que los video-juegos eran las armas perfectas de distracción masiva para todo el planeta.

Los primeros video-juegos nacieron como un subproducto de la guerra fría pero en el lejano oriente los primeros juegos japoneses surgieron como consecuencia de un conflicto directo, la segunda guerra mundial, las bombas atómicas de Hiroshima y Nagasaki habían asolado el país. La tecnología se mostró rápidamente como la clave del éxito japonés, el gobierno hacia grandes esfuerzos por encaminar la educación hacia la tecnología, la juventud japonesa estaba altamente alfabetizada en materia tecnológica en ese momento. Japón encontró un aliado en la electrónica y en 1978 Tomohiro

Nishikado coloco a Japón al frente de la industria de los video-juegos gracias una interminable ola de invasores digitales, creando el juego “Space Invaders”, este cambio algo en las mentes de los japoneses porque millones de adolescentes jugaban, la manera tan rápida en la que se extendió solo puede producirse en un lugar como Japón.

Space Invaders fue la expresión digital de un periodo oscuro, pero Japón siempre ha sabido dar un aspecto dulce gracias a sus encantadores personajes cuando llegó el juego de Pac-Man, con muchos colores, este es el ejemplo perfecto que define la rareza japonesa. Toru Iwatani era un joven diseñador que trabajaba en una empresa de juegos japonesa NAMCO, antes de crear la estrella de la historia de los video-juegos, Pac-Man se lanzó en 1980, pero la atmosfera en los salones recreativos eran un tanto salvaje, porque solo había juegos violentos como los de disparar y era un terreno reservado a los chicos. Pac-Man cambio mucho el panorama, mientras que muchos de los primeros juegos se basaban en invasores, acción y disparos.

A principios de los años 80 ya estaba claro que los video-juegos y las formas de vida pixeladas habían sido creadas para quedarse, porque son una forma de arte expresiva y representativa. Surge un nuevo mundo que no solo afecta a los jugadores sino también a sus creadores. También fue la época del dinero fácil y de los grandes negocios; ATARI seguían avanzando su tecnología y empezaron a desarrollar la nueva consola portátil la ATARI VCS "Video Computer System" por sus siglas en inglés.

Posteriormente los juegos perdieron creatividad y se tornaron aburridos, el juego del que hablaba todo el mundo era el de una versión de la película “ET”, que era tan mala, incluso criticada por el mismo Al Arcon, se hicieron demasiados cartuchos del juego de “ET” que fue imposible venderlos, así que las ventas de los video-juegos ATARI sufrieron una caída drástica, 1983 fue el año de la gran quiebra, todo el mundo quería entrar en el negocio había mucha competencia, muchas empresas, los video-juegos estaban por todas partes, pero habían muchos, tan malos que la gente acabo dándole la espalda al medio.

Pero con altos y bajos en la industria de los video-juegos otros actores entraron en escena y en la actualidad para darse cuenta de la importancia de la industria y el mercado de los video-juegos dentro del sector del ocio y el entretenimiento basta con saber que en los últimos años supera con creces el ingreso en taquilla del mercado cinematográfico tanto en los EE.UU como en otros países. El mercado de los video-

juegos generó un volumen de negocio en 2010, solo en los EE.UU., de 25.100 millones de dólares (según fuente NPD Group).

Según José Martí Parreño los importantes crecimientos del sector de los video-juegos (hasta hace apenas unos años cercanos al 25% anual) se debe en parte a que la industria ha sabido captar nuevos segmentos de mercado que hasta ahora le eran totalmente inaccesibles. Queda ya muy lejos la imagen de los consumidores de video-juegos como adolecentes varones fanáticos de la informática. En la actualidad, los video-juegos son consumidos virtualmente por cualquier segmento de edad de ambos sexos. El hecho de que en los EE.UU. las mujeres mayores de 18 años representen el 37% de la población total video-jugadora en 2010 (superando al grupo de varones de 17 años o menores, que representa el 13% de esta población) o que en 2011 el 29% de la población norteamericana mayor de 50 años jugara a los video-juegos (frente a tan solo el 9% en 1999)[99] da buena prueba de esta diversificación y segmentación de sus públicos.

En los últimos años también se han producido importantes cambios de consumo en relación a la edad de los video-jugadores ya que la edad media de los compradores más frecuentes de video-juegos en los EE.UU. es de 37 años (ESA,2011:2). Este dato indica que la media de edad va creciendo puesto que los video-jugadores que empezaron a jugar en su adolescencia o juventud continúan jugando en la edad adulta.

Pero se debe entender que estos son específicos para una plataforma que es el hardware capaz de reproducir un video-juego. Históricamente tanto los computadores como las videoconsolas han sido dos de las plataformas más utilizadas para jugar a los video-juegos aunque no se debe olvidar la edad de oro de las máquinas recreativas (años 80) como un período fundamental en el que muchos jóvenes tuvieron su primer contacto con los video-juegos. En la actualidad la versatilidad que ofrece la telefonía móvil augura un papel determinante de esta plataforma.

Cada plataforma posee unas características e interfaces que determinan en cierta medida tanto el género como el uso de los video-juegos. En los diferentes géneros se han popularizado los Arcade, como paddles (Pong o Breakout), laberintos (Pacman), shoot'em'up (disparar y olvidar como Space Invaders), simuladores, FPS (Juegos en

[99] <http://www.theesa.com>

primera y tercera persona, como DOOM), juegos deportivos, de lucha (Street Fighter), Puzzles (tetris), de estrategias y aventuras como Age of Empireso 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 video-juegos, es un sistema electrónico de entretenimiento para el hogar que ejecuta juegos electrónicos (video-juegos) que están contenidos en cartuchos, discos ópticos, discos magnéticos o tarjetas de memoria. Los primeros sistemas de videoconsolas fueron diseñados únicamente para jugar video-juegos pero a partir de la sexta generación de videoconsolas han sido incorporadas características importantes como multimedia, internet, tiendas virtuales y servicios en línea.

Cabe mencionar que los juegos que incluían violencia, sexo o palabras inadecuados hizo que a partir de los años de 1994 se cree la Entertainment Software Rating Board y en 2003 la Pan European Game Information o PEGI para la clasificación de los video-juegos en grupos idóneos relacionados, acceso por edad y así conocer el materia incluido en ellos.

3.- Hardware usado en la industria de los videojuegos

En la Industria de los video-juegos, las videoconsolas han sido clasificadas en distintas generaciones. Esta clasificación la determina su tiempo de lanzamiento y la tecnología existente en ese momento. Las empresas fabricantes lanzan una nueva consola en determinado tiempo (que puede variar entre 5 o 6 años). Por otro lado, algunas generaciones están señaladas por un número determinado de bits, los cuales determinan el ancho de bus del procesador, (de la segunda generación hasta la sexta generación).

También es muy importante mencionar a las consolas portátiles, un dispositivo electrónico ligero que permite jugar video-juegos y que, a diferencia de una videoconsola clásica, los controles, la pantalla, los altavoces y la alimentación (baterías) están todos integrados en la misma unidad y todo ello con un pequeño tamaño, para poder llevarla y jugar en cualquier lugar o momento. Es la empresa japonesa Nintendo, la que populariza el video-juego portátil. Primero con los juegos Game & Watch entre 1980 y 1991 que permite jugar a un único juego de bolsillo en cualquier lugar. Segundo con la popular GameBoy que permite jugar en cualquier parte con juegos intercambiables. Por último, Nintendo DS y su reciente versión 3D, que consigue

habituar a jugar en cualquier parte a un público no necesariamente interesado en los video-juegos. Tan solo Sega Game Gear y Sony Play Station Portable, han conseguido arrebatar una porción de mercado suficiente para generar beneficios y prolongar su vida comercial durante el ciclo normal de una generación.

4.- Software en el desarrollo de Videojuegos

En cuanto al software de desarrollo debemos mencionar las arquitecturas de los video-juegos, los programadores del software han creado sus propios diseños empezando de cero, o reutilizando diseños de otros. Según Gómez Martín **"Cuando los recursos disponibles crecen, no decrece el esfuerzo necesario para crear los juegos. Al contrario, lo que ocurre es que aumentan sus pretensiones, lo que hace que la complejidad de programación, generación de modelos y recursos en general, también aumente"**. Hoy en día, esa situación ha cambiado, surgiendo una nueva disciplina conocida como arquitecto de software que utiliza un lenguaje común a todos los informáticos del software para detallar diseños de alto y bajo nivel, así que se han planteados soluciones como inclusión de código externo en forma de librerías o módulo completos comprados a terceros o COTS (Component of the shelf); a esto le sumamos que las tarjetas gráficas han evolucionado y ahora son aceleradoras que incluyen procesadores llamados GPU, asi que soluciones como OpenGL (Open Graphics Library) y Direct3D que es parte de DirectX han apaleado en cierta medida estos problemas con sus API para 3D; y que los programadores pueden usarlas sin llegar a programar de cero instrucciones con el hardware, todo esto es con el objetivo de ocultar la complejidad de la interfaz con las diferentes tarjetas gráficas y las capacidades en las diversas plataformas hardware.

Las Arquitecturas dirigidas por datos es otra solución definiendo la separación entre código y datos, plante crear motor del juego, centrando al programador en el código del juego y la polución de herramientas de apoyo, a esto le podemos sumar las arquitecturas basadas en componentes donde prima la descomposición en componentes funcionales, la reutilización del software, conjunto de propiedades y se tiene de manera externa una especificación, por ultimo también están las arquitecturas basadas en módulos donde se asignan módulos para cada cosa como la física, matemáticas, redes, inteligencia artificial, animación, audio, video, etc.

Toda esta complejidad ha hecho que se incorporen otras formas de desarrollo con los llamados "motores de video-juego", un término que hace referencia a una serie de rutinas de programación que permiten el diseño, la creación y la representación de un video-juego, del mismo modo existen motores de juegos que operan tanto en consolas de video-juegos y sistemas operativos. La funcionalidad básica de un motor es proveer al video-juego de un motor de renderizado para los gráficos 2D y 3D, motor físico o detector de colisiones, sonidos, scripting, animación, inteligencia artificial, redes, streaming [100], administración de memoria y un escenario gráfico. El proceso de desarrollo de un video-juego puede variar notablemente por reusar o adaptar un mismo motor de video-juego para crear diferentes juegos, que incluyen renderizado, física, Audio, video, un lenguaje de scripting, la inteligencia artificial, soporte a las redes entre otras cualidades.

Figura 2 Arquitectura basada en componentes

Fuente: http://upload.wikimedia.org/wikipedia/commons/9/9b/Porcomponentes.jpg

5.- Conclusiones

Crear un "motor 3D" es una tarea muy compleja para un programador o un grupo pequeño de ellos, pero no imposible; el factor preponderante para ponerse a crear uno dependerá en gran medida de cuánto tiempo disponemos para ello y el presupuesto. Al momento de escoger un motor de juegos ya creados se deberán tomar muchos factores a considerar como los costos de licenciamiento, tiempos de desarrollo, tipo de juego a crear, género, plataforma entre otras variables, pero esta tesis quiere encaminar

[100] Distribución de multimedia a través de una red de computadoras de manera que el usuario consume el producto al mismo tiempo que se descarga.

a ese programador que con algo de experiencia puede fácilmente acoplarse e estos nuevos entornos, la mejor opción es Unity 3D, por su facilidad de manejar escenas, costos asequibles tanto para desarrollo video-juegos indie[101] como para una pequeña desarrolladora, el lenguaje de este motor es Javascript vastamente popular, C# que a través de la suite Microsoft Visual Studio ha ganado muchos adeptos, y Boo poco conocido solo decir que es un derivado de Pyton.

6.- Recomendaciones

Unity3D permite publicar el mismo video-juego para consolas como PS3, Xbox y Wii, Móviles con iOS o Android, y las plataformas tradicionales como PC o Macintosh de manera fácil e intuitiva; posee una versión gratuita muy funcional que deja publicar para PC o Macintosh, así que no hay un desembolso previo, si el motor es el correcto puede pagar la licencia de la plataforma a usar. Además como todo "motor de juego" incorpora otros motores como es el caso de la física y colisiones, sistemas de partículas, edición de terrenos, gestión del audio o otras cualidades como edición de escenas, creación de niveles, prueba en caliente, shaders en las gráficas; que son algunos de los aspectos que un video-juego puede requerir. La programación de la lógica del juego es mucho más sencilla tanto con los lenguajes de tipo script que estas herramientas incorporan como las APIs que se disponen para ayudar en el desarrollo, pero si el programador o empresa desarrolladora desea crear juegos de gran envergadura existen otras alternativas a considerar en el mercado como motores 3D de la talla de UDK del la empresa EPIC, Blender, Id Tech entre otros, así que manos a la obra para los programadores y explotar nuevos nichos de mercado que cada día nacen con mejores oportunidades que otros ya saturados.

[101] Proceso de crear video-juegos sin el apoyo financiero de una distribuidora de video-juegos además de un pequeño equipo de personas.

Anexo 2

Código Fuente del controlador en tercera persona del personaje

Santiago

```
// Require a character controller to be attached to the same game
object
@script RequireComponent(CharacterController)

public var ReposoAnimacion: AnimationClip;
public var CaminarAnimacion: AnimationClip;
public var CorrerAnimacion: AnimationClip;
public var SaltoPoseAnimacion: AnimationClip;
// ***********************************************
// Nuevas animaciones como parametro
      public var puneteAnimation : AnimationClip;
      public var patadaAnimation : AnimationClip;
      public var agacharAnimation : AnimationClip;
      public var cubrirAnimation : AnimationClip;
      public var morirAnimation : AnimationClip;
// ***********************************************

public var CaminarVelocidadAnimacionMax : float = 0.75;
public var TrotarVelocidadAnimacionMax : float = 1.0;
public var CorrerVelocidadAnimacionMax : float = 1.0;
public var SaltarVelocidadAnimacionMax : float = 1.15;
public var TierraVelocidadAnimacionMax : float = 1.0;
private var _animation : Animation;
enum CharacterState {
      Idle = 0,
      Walking = 1,
      Trotting = 2,
      Running = 3,
      Jumping = 4,
// ***********************************************
// Nuevos estados
      punete = 5,
      patada = 6,
      agachar= 7,
      cubrir = 8,
      morir = 9,
// ***********************************************
}
private var _characterState : CharacterState;
// The speed when walking
var VelocidadCaminar= 2.0;
// after trotAfterSeconds of walking we trot with trotSpeed
var VelocidadTrotar= 4.0;
// when pressing "Fire3" button (cmd) we start running
var VelocidadCorrer= 6.0;
var ControlAceleracionEnAire = 3.0;
// How high do we jump when pressing jump and letting go immediately
var AlturaSalto= 0.5;
// The Gravedad for the character
var Gravedad = 20.0;
// The Gravedad in controlled descent mode
var SuavizarVelocidad = 10.0;
var VelocidadRotacion= 500.0;
```

```
var SegundosInicioTrotar= 3.0;
var PuedeSaltar= true;
private var jumpRepeatTime = 0.05;
private var jumpTimeout = 0.15;
private var groundedTimeout = 0.25;
// The camera doesnt start following the target immediately but waits
for a split second to avoid too much waving around.
private var lockCameraTimer = 0.0;
// The current move direction in x-z
private var moveDirection = Vector3.zero;
// The current vertical speed
private var verticalSpeed = 0.0;
// The current x-z move speed
private var moveSpeed = 0.0;
// The last collision flags returned from controller.Move
private var collisionFlags : CollisionFlags;
// Are we jumping? (Initiated with jump button and not grounded yet)
private var jumping = false;
private var jumpingReachedApex = false;
// Are we moving backwards (This locks the camera to not do a 180
degree spin)
private var movingBack = false;
// Is the user pressing any keys?
private var isMoving = false;
// When did the user start walking (Used for going into trot after a
while)
private var walkTimeStart = 0.0;
// Last time the jump button was clicked down
private var lastJumpButtonTime = -10.0;
// Last time we performed a jump
private var TiempoUltimoSalto= -1.0;
// the height we jumped from (Used to determine for how long to apply
extra jump power after jumping.)
private var lastJumpStartHeight = 0.0;


private var inAirVelocity = Vector3.zero;
private var lastGroundedTime = 0.0;
private var isControllable = true;
function Awake ()
{
      moveDirection = transform.TransformDirection(Vector3.forward);

      _animation = GetComponent(Animation);
      if(!_animation)
            Debug.Log("The character you would like to control doesn't
have animations. Moving her might look weird.");

      /*
public var ReposoAnimacion: AnimationClip;
public var CaminarAnimacion: AnimationClip;
public var CorrerAnimacion: AnimationClip;
public var SaltoPoseAnimacion: AnimationClip;
      */
      if(!ReposoAnimacion) {
            _animation = null;
            Debug.Log("No idle animation found. Turning off
animations.");
      }
      if(!CaminarAnimacion) {
            _animation = null;
```

```
            Debug.Log("No walk animation found. Turning off
animations.");
      }
      if(!CorrerAnimacion) {
            _animation = null;
            Debug.Log("No run animation found. Turning off
animations.");
      }
      if(!SaltoPoseAnimacion && PuedeSaltar) {
            _animation = null;
            Debug.Log("No jump animation found and the character has
PuedeSaltar enabled. Turning off animations.");
      }
}
function UpdateSmoothedMovementDirection ()
{
      var cameraTransform = Camera.main.transform;
      var grounded = IsGrounded();
      // Forward vector relative to the camera along the x-z plane
      var forward =
cameraTransform.TransformDirection(Vector3.forward);
      forward.y = 0;
      forward = forward.normalized;
      // Right vector relative to the camera
      // Always orthogonal to the forward vector
      var right = Vector3(forward.z, 0, -forward.x);
      var v = Input.GetAxisRaw("Vertical");
      var h = Input.GetAxisRaw("Horizontal");
      // Are we moving backwards or looking backwards
      if (v < -0.2)
            movingBack = true;
      else
            movingBack = false;
      var wasMoving = isMoving;
      isMoving = Mathf.Abs (h) > 0.1 || Mathf.Abs (v) > 0.1;
      // Target direction relative to the camera
      var targetDirection = h * right + v * forward;
      // Grounded controls
      if (grounded)
      {
            // Lock camera for short period when transitioning moving &
standing still
            lockCameraTimer += Time.deltaTime;
            if (isMoving != wasMoving)
                  lockCameraTimer = 0.0;

            // We store speed and direction seperately,
            // so that when the character stands still we still have a
valid forward direction
            // moveDirection is always normalized, and we only update
it if there is user input.
            if (targetDirection != Vector3.zero)
            {
                  // If we are really slow, just snap to the target
direction
                  if (moveSpeed < VelocidadCaminar* 0.9 && grounded)
                  {
                        moveDirection = targetDirection.normalized;
                  }
                  // Otherwise smoothly turn towards it
                  else
```

```
                {
                      moveDirection =
Vector3.RotateTowards(moveDirection, targetDirection,
VelocidadRotacion* Mathf.Deg2Rad * Time.deltaTime, 1000);

                      moveDirection = moveDirection.normalized;
                }
            }
            // Smooth the speed based on the current target direction
            var curSmooth = SuavizarVelocidad * Time.deltaTime;

            // Choose target speed
            //* We want to support analog input but make sure you cant
walk faster diagonally than just forward or sideways
            var targetSpeed = Mathf.Min(targetDirection.magnitude,
1.0);
            _characterState = CharacterState.Idle;
            // Pick speed modifier

      //*************************************************************
**
            // Codigo para gregar nuevas teclas y acciones
            if (Input.GetKey (KeyCode.Alpha1)){
                _characterState = CharacterState.punete;
            }
            else if (Input.GetKey (KeyCode.Alpha2)){
                _characterState = CharacterState.patada;
            }
            else if (Input.GetKey (KeyCode.Alpha3)){
                _characterState = CharacterState.agachar;
            }
            else if (Input.GetKey (KeyCode.Alpha4)){
                _characterState = CharacterState.cubrir;
            }
            else if (Input.GetKey (KeyCode.Alpha5)){
                _characterState = CharacterState.morir;
            }
            //
**********************************************************************
            else if (Input.GetKey (KeyCode.LeftShift) | Input.GetKey
(KeyCode.RightShift))
            {
                  targetSpeed *= VelocidadCorrer;
                  _characterState = CharacterState.Running;
            }
            else if (Time.time - SegundosInicioTrotar> walkTimeStart)
            {
                  targetSpeed *= VelocidadTrotar;
                  _characterState = CharacterState.Trotting;
            }
            else
            {
                  targetSpeed *= VelocidadCaminar;
                  _characterState = CharacterState.Walking;
            }
            moveSpeed = Mathf.Lerp(moveSpeed, targetSpeed, curSmooth);

            // Reset walk time start when we slow down
            if (moveSpeed < VelocidadCaminar* 0.3)
                  walkTimeStart = Time.time;
      }
```

```
	// In air controls
	else
	{
		// Lock camera while in air
		if (jumping)
			lockCameraTimer = 0.0;

		if (isMoving)
			inAirVelocity += targetDirection.normalized *
Time.deltaTime * ControlAceleracionEnAire;
	}
}


function ApplyJumping ()
{
	// Prevent jumping too fast after each other
	if (TiempoUltimoSalto+ jumpRepeatTime > Time.time)
		return;

	if (IsGrounded()) {
		// Jump
		// - Only when pressing the button down
		// - With a timeout so you can press the button slightly
before landing
		if (PuedeSaltar&& Time.time < lastJumpButtonTime +
jumpTimeout) {
			verticalSpeed = CalculateJumpVerticalSpeed
(AlturaSalto);
			SendMessage("DidJump",
SendMessageOptions.DontRequireReceiver);
		}
	}
}

function ApplyGravity ()
{
	if (isControllable)	// don't move player at all if not
controllable.
	{
		// Apply Gravedad
		var jumpButton = Input.GetButton("Jump");


		// When we reach the apex of the jump we send out a message
		if (jumping && !jumpingReachedApex && verticalSpeed <= 0.0)
		{
			jumpingReachedApex = true;
			SendMessage("DidJumpReachApex",
SendMessageOptions.DontRequireReceiver);
		}

		if (IsGrounded ())
			verticalSpeed = 0.0;
		else
			verticalSpeed -= Gravedad * Time.deltaTime;
	}
}
function CalculateJumpVerticalSpeed (targetJumpHeight : float)
{
	// From the jump height and Gravedad we deduce the upwards speed
```

```
      // for the character to reach at the apex.
      return Mathf.Sqrt(2 * targetJumpHeight * Gravedad);
}
function DidJump ()
{
      jumping = true;
      jumpingReachedApex = false;
      TiempoUltimoSalto= Time.time;
      lastJumpStartHeight = transform.position.y;
      lastJumpButtonTime = -10;
      _characterState = CharacterState.Jumping;
}
function Update() {

      if (!isControllable)
      {
            // kill all inputs if not controllable.
            Input.ResetInputAxes();
      }

      if (Input.GetButtonDown ("Jump"))
      {
            lastJumpButtonTime = Time.time;
      }

      UpdateSmoothedMovementDirection();

      // Apply Gravedad
      // - extra power jump modifies Gravedad
      // - controlledDescent mode modifies Gravedad
      ApplyGravity ();
      // Apply jumping logic
      ApplyJumping ();
      // Calculate actual motion
      var movement = moveDirection * moveSpeed + Vector3 (0,
verticalSpeed, 0) + inAirVelocity;
      movement *= Time.deltaTime;
      // Move the controller
      var controller : CharacterController =
GetComponent(CharacterController);
      collisionFlags = controller.Move(movement);
      // ANIMATION sector
      if(_animation) {
      Debug.Log(_characterState);
            if(_characterState == CharacterState.Jumping)
            {
                  if(!jumpingReachedApex) {
                        _animation[SaltoPoseAnimacion.name].speed =
SaltarVelocidadAnimacionMax;
                        _animation[SaltoPoseAnimacion.name].wrapMode =
WrapMode.ClampForever;
                        _animation.CrossFade(SaltoPoseAnimacion.name);
                  } else {
                        _animation[SaltoPoseAnimacion.name].speed = -
TierraVelocidadAnimacionMax;
                        _animation[SaltoPoseAnimacion.name].wrapMode =
WrapMode.ClampForever;
                        _animation.CrossFade(SaltoPoseAnimacion.name);

                  }
            }
```

```
            else
            {
               //
**************************************************
               // Codigo para controlar nuevas animaciones
               if(_characterState == CharacterState.punete) {

                        _animation.CrossFade(puneteAnimation.name);

               }
               else if(_characterState == CharacterState.patada) {


      _animation.CrossFade(this.patadaAnimation.name);

               }
               else if(_characterState == CharacterState.agachar) {


      _animation.CrossFade(this.agacharAnimation.name);

               }
               else if(_characterState == CharacterState.cubrir) {

                        _animation.Play(this.cubrirAnimation.name);
                        _animation.wrapMode = WrapMode.Once;
               }
               else if(_characterState == CharacterState.morir) {

                        _animation.CrossFade(this.morirAnimation.name);

               }
                //
**************************************************
                  else if(controller.velocity.sqrMagnitude < 0.1) {
                        _animation.CrossFade(ReposoAnimacion.name);
                  }
                  else
                  {
                        if(_characterState == CharacterState.Running) {
                             audio.Play();
                              _animation[CorrerAnimacion.name].speed =
Mathf.Clamp(controller.velocity.magnitude, 0.0,
CorrerVelocidadAnimacionMax);

      _animation.CrossFade(CorrerAnimacion.name);
                        }
                        else if(_characterState ==
CharacterState.Trotting) {
                              _animation[CaminarAnimacion.name].speed =
Mathf.Clamp(controller.velocity.magnitude, 0.0,
TrotarVelocidadAnimacionMax);

      _animation.CrossFade(CaminarAnimacion.name);
                        }
                        else if(_characterState ==
CharacterState.Walking) {
                              _animation[CaminarAnimacion.name].speed =
Mathf.Clamp(controller.velocity.magnitude, 0.0,
CaminarVelocidadAnimacionMax);
```

```
      _animation.CrossFade(CaminarAnimacion.name);
                        }

                  }
            }
      }
      // ANIMATION sector
      // Set rotation to the move direction
      if (IsGrounded())
      {

            transform.rotation =
Quaternion.LookRotation(moveDirection);
      }
      else
      {
            var xzMove = movement;
            xzMove.y = 0;
            if (xzMove.sqrMagnitude > 0.001)
            {
                  transform.rotation = Quaternion.LookRotation(xzMove);
            }
      }
      // We are in jump mode but just became grounded
      if (IsGrounded())
      {
            lastGroundedTime = Time.time;
            inAirVelocity = Vector3.zero;
            if (jumping)
            {
                  jumping = false;
                  SendMessage("DidLand",
SendMessageOptions.DontRequireReceiver);
            }
      }
}
function OnControllerColliderHit (hit : ControllerColliderHit )
{
//    Debug.DrawRay(hit.point, hit.normal);
      if (hit.moveDirection.y > 0.01)
            return;
}
function GetSpeed () {
      return moveSpeed;
}
function IsJumping () {
      return jumping;
}
function IsGrounded () {
      return (collisionFlags & CollisionFlags.CollidedBelow) != 0;
}
function GetDirection () {
      return moveDirection;
}
function IsMovingBackwards () {
      return movingBack;
}
function GetLockCameraTimer ()
{
      return lockCameraTimer;
```

```
}
function IsMoving ()  : boolean
{
      return Mathf.Abs(Input.GetAxisRaw("Vertical")) +
Mathf.Abs(Input.GetAxisRaw("Horizontal")) > 0.5;
}
function HasJumpReachedApex ()
{
      return jumpingReachedApex;
}
function IsGroundedWithTimeout ()
{
      return lastGroundedTime + groundedTimeout > Time.time;
}
function Reset ()
{
      gameObject.tag = "Player";
}
```