UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

FACULTAD DE CIENCIAS QUÍMICAS E INGENIERÍA

MAESTRÍA Y DOCTORADO EN CIENCIAS E INGENIERÍA

"Arquitectura para el manejo de las emociones de los usuarios en los videojuegos"

QUE PARA OBTENER EL GRADO DE

MAESTRO EN CIENCIAS

Presenta:

José de Jesús Medina Ríos

Bajo la Dirección de:

Dr. J. Reyes Juárez Ramírez

Co-dirigido por:

Dr. Juan Ramón Castro Rodríguez

TIJUANA, BAJA CALIFORNIA, MÉXICO JULIO DEL 2014

Universidad Autónoma de Baja California

FACULTAD DE CIENCIAS QUÍMICAS E INGENIERÍA

COORDINACIÓN DE POSGRADO E INVESTIGACIÓN

FOLIO No. 122

Tijuana, B. C., a 16 de junio de 2014

C. José de Jesús Medina Ríos
Pasante de: Maestro en Ciencias
Presente

El tema de trabajo y/o tesis para su examen profesional, en la Opción TESIS

Es propuesto, por el C. Dr. J. Reyes Juárez Ramírez

quien será el responsable de la calidad del trabajo que usted presente, referido al tema: "Arquitectura para el manejo de las emociones de los usuarios en los videojuegos"

el cual deberá usted desarrollar, de acuerdo con el siguiente orden:

I.- INTRODUCCIÓN
II.- MARCO TEÓRICO: ARQUITECTURA, VIDEOJUEGOS, EMOCIONES
III.- TRABAJOS RELACIONADOS
IV.- MODULO PARA EL TRABAJO DE LAS EMOCIONES
V.- CASO DE ESTUDIO: VIDEOJUEGO "KIN DE MAYAPAN"
VI.- CONCLUSIONES
VII.- ANEXO

UNIVERSIDAD AUTONOMA DE BAJA CALIFORNIA

FACULTAD DE CIENCIAS QUIMICAS E INGENIERIA

Q. Noemí Hernández Hernández
Sub-Director Secretario

Dr. J. Reyes Juárez Ramírez
Director de Tesis

Dr. Luis Enrique Palafox Maestre
Director

Dedicatoria

A mi familia por no preguntar qué es lo que estaba haciendo y no darme computadoras para arreglar, y a mi novia por todos los ánimos y apoyo incondicional dados durante todo este camino...

Agradecimientos

Agradezco a mi madre, que aunque lleva mucho tiempo sin estar conmigo, aún he logrado contar con su apoyo de alguna manera u otra y sin ello no hubiera llegado a donde estoy.

Agradezco a mi novia Wendy, por todas sus palabras de aliento y sus consejos que aunque muchas veces los ignoraba, eran realmente buenos y no los olvidaré fácilmente.

Gracias a todas las personas que estuvieron conmigo en este camino en especial aquellas amistades que les encantaba pasar el tiempo preguntando ¿Y cómo va la tesis?

A mi director de tesis por su guía en esta investigación, por su apoyo que medio en todo momento y sobre todo por aguantarme!

A los miembros del comité de tesis, por las observaciones hechas y los consejos sobre los aspectos a considerar.

Agradezco a los alumnos de licenciatura: Arturo, Alex, Leo, JJ, Chin, Totu y Francisco por su ayuda durante los inicios de la maestría que me apoyaron con el desarrollo de algunas aplicaciones.

También a Asdriel, Paz, Aldo y Barba por su participación en el desarrollo del prototipo de Kin de Mayapan.

Gracias a mis compañeros de posgrado por la ayuda en cuanto a algunos aspectos de la tesis y en la información requerida para ciertos trámites, en especial a Jorge Marín por su ayuda en la programación de Kin de Mayapan

Gracias a CONACYT por el apoyo económico brindado durante mi transcurso en la maestría.

Gracias a los alumnos de licenciatura que participaron en los experimentos y por el tiempo dedicado.

INDICE

Lista de Figuras

Lista de Tablas

RESUMEN

Arquitectura para el manejo de las emociones de los usuarios en los videojuegos

En esta tesis se propone un módulo para el manejo de las emociones para los usuarios de los videojuegos con el fin de obtener información sobre su estado emocional. La propuesta tiene como objetivo diseñar una arquitectura escalable que permita detectar las emociones del jugador durante el juego para hacer cambios en sus los elementos con el fin de manipular las emociones del jugador. Se toma como base el modelo OCC centrado en las entidades Acción-Evento-Consecuencia.

La experimentación está centrada en los videojuegos, ya que son sistemas altamente interactivos que se caracterizan por su alto nivel de atención que requieren de los usuarios y por las emociones que dichas interacciones provocan en ellos.

El módulo propuesto tiene una arquitectura dirigida por eventos, la cual permite un desempeño adecuado en cuanto al acoplamiento. La propuesta consiste en un módulo conformado por 13 clases que se encargan de almacenar los datos del jugador, como su personalidad y tipo de jugador así como de registrar las acciones, eventos y consecuencias ocurridas durante una sesión de juego. También considera la polaridad de cómo estas consecuencias afectan las emociones del jugador, censar de alguna manera el estado de la persona cuando ocurre esto y hacer un análisis estadístico de cómo el usuario reaccionó a dichos sucesos, además de almacenar los resultados de estos. Estos cálculos estadísticos nos permiten ya sea realizar cambios en tiempo de ejecución al juego (en caso de ser necesario) o hacer un análisis con la información obtenida para tener un marco de referencia de cuáles ajustes se podrían realizar y el por qué.

El caso de estudio consistió en la elaboración de un videojuego utilizando una arquitectura tradicional y sin manejo de emociones; posteriormente se le implementó el módulo de manejo de emociones y se observaron los cambios realizados sobre su arquitectura y rendimiento comparado entre las versiones, además de realizar pruebas con varios jugadores para observar qué tipos de reacciones tenían al jugar.

En este caso, el censado de emociones fue realizado con el uso de una Webcam ya integrada en la computadora. La implementación del videojuego fue realizada en lenguaje de programación Java, haciendo uso del framework Libgdx, para funciones adicionales.

Los resultados obtenidos indican que es posible manejar el censado de emociones de esta manera y realizar cambios al juego en base a estos, además de generar estadísticas que pueden ser utilizadas posteriormente, para mejorar la experiencia del usuario.

Palabras claves: Arquitectura, emociones, módulo emocional, videojuegos, modelado usuario.

ABSTRACT

Architecture for managing the emotions of users in videogames

This Thesis presents a module for the management of user emotions of videogame users, in order to obtain information about the emotional state from the user. The proposal has as an objective to design a scalable architecture that permits the detection of player's emotions during gameplay to make changes to its elements with the end of manipulation his emotions. It is based on the OCC model, centered on the entities Action-event-consequence.

The experimentation is centered on videogames because they are highly interactive systems that are characterized by his high attention requirements by the users and the emotions that provoke on them.

The proposed module is based on an event-driven architecture which gives the adequate performance in terms of coupling. The proposal consist on a module formed by 13 classes that are in charge of storing the player data, like his name, personality and player type, then it will register the actions, events and consequences that happen during a game session, it considers the polarity of how the consequences affect emotionally the player-self and it senses in some manner the emotional state of the person when this happens and it makes and statistical analysis of how it reacted to that event and finally it stores the results from this analysis. This result permits us to realize changes to the game during run-time (in case that is needed) or to make an analysis of the information given to use it as a reference to which adjustments could be made and why.

The case study consisted on the elaboration of a videogame making use of a traditional game architecture without the management of emotions, later the emotional management module was added and the changes were noted on the architecture and performance by comparing both versions, also there where test by various players to see what kind of reaction they show while playing.

On this case the emotional censing was made via Webcam already integrated on the computer. The implementation was made on the videogame using Java programming language and making use of the LibGdx Framework for additional functions.

The obtained results shows that is possible to manage the census of the emotions by this way and that you can make changes on the game based on this, also by generating statistics that could be utilized for the betterment of the user experience

Keywords: Architecture, Emotions, Emotional Module, Videogames, User Modelling.

Capítulo I

Introducción

I. Introducción

I.1 Problemática

En la actualidad el uso de los videojuegos como una forma de entretenimiento (Poels, van den Hoogen, Ijsselsteijn, & de Kort, 2012), es muy popular entre los jóvenes y adultos, estadísticas de la ESA (*Entertainment Software Association*), indican que dentro de los Estados Unidos de América, en promedio se cuenta con al menos una consola, computadora o teléfono inteligente con la capacidad de ejecutar un videojuego, además el promedio de edad de un jugador es de 31 años (Association, 2014)

Inicialmente los videojuegos no eran tan complejos, por lo que podían ser elaborados por una sola persona, en la actualidad debido al aumento en la complejidad, ha conllevado a formar grandes equipos de trabajo y al surgimiento de requerir a personal especializado en cierta área determinada (Blow, 2004; Morelli & Nakagawa, 2011; Murphy-Hill, Zimmermann, & Nagappan, 2014).

Actualmente, el diseño de videojuegos está enfocado a un grupo del público (Gilleade & Dix, 2004) que comparten gustos similares y tratan de proveer una misma experiencia para ellos, además se consideran otros aspectos (Novak, 2011; Sylvester, 2013) tales como la demografía y la cultura a donde se pretende presentar el juego, también se consideran que emociones se pueden llegar a suscitar cuando la persona juegue, por ejemplo, poner un escenario terrorífico para asustar a la persona, o poner una escena intensa de acción, para provocarle euforia, sin embargo esto solo se ve en fase de planeación y no se sabe si realmente surte el efecto cuando el usuario juega al videojuego.

Dentro de estos video juegos en algunos casos se utiliza la información introducida por el jugador, ya sea para nombrar un personaje o tomar decisiones sobre una trama argumental, o simplemente configuraciones para el juego, además algunos juegos hacen una evaluación sobre el desempeño del jugador, como los puntos obtenidos u enemigos eliminados u observar el comportamiento de un usuario sobre ciertos escenarios pero son utilizadas con el propósito de comparar desempeños entre otras personas (B Medler, 2009; Ben Medler, John, & Lane, 2011; Zoeller, 2010).

La información adquirida por algunas partes del perfil de usuario nos permitirá realizar modificaciones a algunos aspectos del video juego para que de acuerdo a esto, sea de mejor agrado para el jugador (Gilleade & Dix, 2004) dentro de algunos juegos ya existentes, al detectar un bajo desempeño por parte del jugador (ejemplo: morir frecuentemente), se le ofrece cambiar a una dificultad

más fácil (*Devil May Cry*), mientras que en otros se realiza un ajuste automático en base al desempeño del jugador (*Max Payne*, *Left 4 Dead*), pero en este caso está enfocado a un análisis estadístico por parte de las acciones del jugador y no por las expresiones o emociones que este experimenta, en este caso nos enfocamos en el aspecto afectivo, de cómo el video juego puede "percibir" el estado de ánimo o emocional conforme a lo que ocurre dentro del juego.

El propósito de este trabajo es añadir algunos aspectos del perfil de usuario a considerar dentro de un video juego, por ejemplo el perfil de usuario considera aspectos: afectivos, cognitivos, demográficos, físicos y de experiencia.

Un video juego al ser un software, es posible, o deberían aplicarse el uso de metodologías y herramientas de la ingeniería de software, tales como la elaboración de un diseño de arquitectura. En el contexto de esta investigación, la arquitectura de software se puede definir como los componentes o elementos que forman parte de él y las propiedades y relaciones que hay entre ellos (Bass, Clements, & Kazman, 2003; Shaw & Garlan, 1996)

La arquitectura de software, en el área de los video juegos, como tal no es un tema que se acostumbra a documentar en la industria, debido a que en sus inicios al ser sencillos y por una sola persona o un grupo pequeño, esta era la que conocía el código y la encargada de darle mantenimiento (Blow, 2004), a medida que el tamaño de los video juegos fue en aumento, se utilizaron diversos módulos en su elaboración, pero existen algunos en común tales como : Administrador de Archivos, Gráficos, Sonido, Lógica de Juego y Simulación (Doherty, 2003; Rollings & Morris, 2003; Sollenberger & Singh, 2012); además estos módulos ya no son necesariamente programados completamente desde cero, si no se compra o utilizan algunos módulos generados, también conocidos como "middleware" que le permite a los programados y diseñadores enfocarse más en las funcionalidades nuevas y tratar de innovar en la jugabilidad que podría haber sido retrasada invirtiendo tiempo de desarrollo en algo ya hecho (Adams, 2013; Deloura, 2009b).

El tipo de arquitectura comúnmente utilizado a los inicios era conocido como una arquitectura monolítica, debido a que todos los componentes fueron elaborados para un juego en específico y en ocasiones no era posible la reutilización de código. En cambio con la nueva tendencia del uso de "*middleware*", se caracterizan por el uso de una arquitectura que incluye e utiliza componentes o COTS (del inglés *Component Of The Shelf*) en el cual se utilizan módulos ya existentes para integrar con el nuevo código generado para reducir el tiempo de desarrollo y hacer uso de métodos ya probados, en

sus inicios los más utilizados eran los *frameworks* gráficos (Adams, 2009; Deloura, 2009ª; Sollenberger & Singh, 2012).

Siguiendo el modelo integral de perfil de usuario (Mejía Figueroa, 2013) (ver Figura I.1), para lograr un sistema usable es necesario conocer qué tipo de usuario hará uso del sistema, por lo tanto se requiere conocer a cierto grado las características de él, para poder elaborar un perfil, se tienen que considerar aspectos cognitivos, demográficos, físicos, psicológicos y aspectos emocionales, para ello en trabajos recientes se propone un modelo para el manejo de emociones dentro de los videojuegos, considerando las acciones, eventos y consecuencias de los actos del usuario.

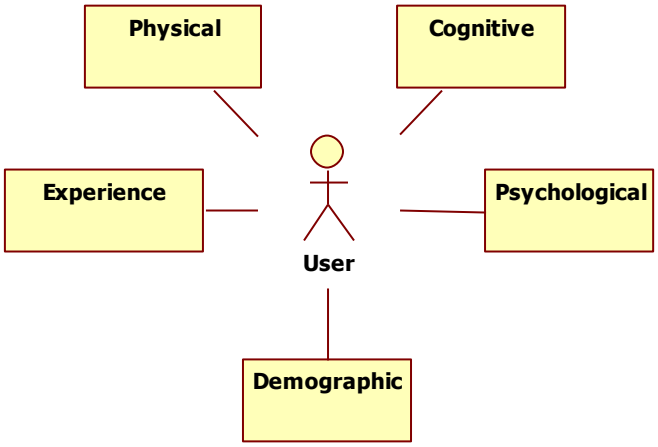

Figura I.1 Modelo integral del usuario

Se han hecho investigaciones sobre como agregar las respuestas biológicas del jugador dentro de un juego (Ambinder, 2011; Dekker & Champion, 2007; Nacke, Kalyn, Lough, & Mandryk, 2011), pero el propósito es utilizar la información ya existente dentro del video juego para evaluar en qué estado se podría encontrar el jugador en determinado momento, la intención es generar un módulo que pueda aplicarse a varios tipos de juegos, tratando de estar lo menos acoplado posible, es decir que no sea dependiente de varias clases dentro de la arquitectura del juego ya existente.

La propuesta consiste en integrar dentro de la arquitectura del video juego un módulo afectivo (encargado de procesar las entradas en las acciones, eventos y consecuencias) y utilizar una aplicación externa que interpreta la información generada para procesar dichas entradas. El propósito es afectar en menor medida posible a la arquitectura del video juego ya creado o como considerar su integración desde las fases de diseño; cuando nos basamos en un juego ya existente es preferible interactuar con las clases que contengan la mayor cantidad de información requerida por el nuevo módulo, esta

información podría ser obtenida de manera directa, por medio de un archivo o a través de una interface (API).

I.2 Objetivos

O1: Diseñar un módulo de manejo de emociones dentro de la arquitectura de software para un videojuego.

O2: Determinar dónde es mejor llevar a cabo la recolección y procesamiento de la información, si dentro de la arquitectura ya existente o en una aplicación externa.

I.3 Metas

M1 (O1, O2): Desarrollar un prototipo de videojuego de género plataforma que considere las acciones, eventos y consecuencias realizadas por el jugador.

M2 (O2): Hacer un análisis del comportamiento detectado de los jugadores, conforme a su tipo de personalidad.

I.4 Hipótesis

H1: Un módulo de manejo de emociones puede ser incluido en la arquitectura de un videojuego con un bajo costo de acoplamiento

H2: En un videojuego una emoción puede ser apreciada con los elementos mínimos tales como acción, evento, consecuencia y aspectos emotivos del jugador.

I.5 Metodología de trabajo

Para el desarrollo de la investigación se procedió con la siguiente metodología:

1. Investigación de las arquitecturas de videojuegos en trabajos científicos, revistas y libros de la industria de videojuegos.
2. Investigación sobre modelos de emociones con representaciones computacionales y observar cómo implementarlos.
3. Formulación de hipótesis sobre la creación de un módulo de emociones para videojuegos.
4. Selección de un modelo de emociones para ser implementado.
5. Diseño de un videojuego tomando en consideración las arquitecturas previamente investigadas.

6. Creación de un módulo para el manejo de emociones dentro de un videojuego, que considere si el usuario sonríe o no, en base a las acciones, eventos y consecuencias ocurridas dentro del juego.
7. Integración del módulo con el videojuego creado.
8. Evaluación de la arquitectura resultante
9. Experimentación con el videojuego para generar reportes de las reacciones de los jugadores.
10. Evaluación de los resultados obtenidos.

I.6 Organización del documento de tesis

El presente documento consta de 6 capítulos y un anexo que describen el trabajo de investigación realizado.

- **Capítulo 1** : En el presente capitulo se presenta una introducción al trabajo de tesis, así como la definición de los objetivos, metas e hipótesis a cumplir con esta investigación
- **Capítulo 2**: En este capítulo se describe el marco teórico, se define que es un video juego, su historia, su clasificación, la evolución en su desarrollo y las fases desarrollo que este conlleva, además de detallar los tipos de arquitectura utilizados en su creación así como una introducción a las emociones y al cómputo afectivo
- **Capítulo 3**: En este capítulo se muestran los trabajos relacionados con la investigación presentada, con respecto al modelado de jugador, videojuegos adaptativos, tipos de jugadores y activadores emocionales.
- **Capítulo 4**: En este capítulo se describe la propuesta del módulo para manejo de las emociones dentro de un videojuego.
- **Capítulo 5**: En este capítulo se detallan los aspectos del caso de estudio manejado, así como el procedimiento para la experimentación realizada y se analizan los resultados obtenidos.
- **Capítulo 6**: En este capítulo se concluye el trabajo de tesis y se analizan los posibles trabajos a futuro que pueden desarrollarse en base a lo observado.

Capítulo II

Marco Teórico: Arquitectura, Videojuegos, Emociones

II. Marco teórico: Arquitectura, Videojuegos, Emociones

II.1 Juego

Los juegos surgen por parte del deseo humano por realizar actividades recreativas y sociales, el objetivo principal es pretender que se está dentro de alguna situación, sobre la cual existe una meta a lograr que solo puede ser alcanzada cuando se actúa conforme a ciertas reglas estipuladas, dichas reglas son el conjunto de instrucciones que indican como se de jugar; la meta es el objetivo particular que se intenta alcanzar (Adams, 2013).

El hecho de jugar involucra que se esté participando dentro de un juego, sobre el cual las decisiones que se toman pueden afectar los sucesos que pueden ocurrir, siempre y cuando estas acciones sean validadas sobre las reglas (Adams, 2013). Los juegos son interactivos debido a que se requiere de la participación de los jugadores para que se pueda proseguir (Adams, 2013).

Las reglas (Salen & Zimmerman, 2004) son definiciones e instrucciones que los jugadores aceptan cumplir durante la duración del juego, estas reglas tienen diversas funciones las cuales pueden ser:

- **El sentido del juego**: Representa el objetivo del juego
- **Gameplay (Forma de Juego)**: Retos y acciones que el juego permite y ofrece al jugador
- **Secuencia de juego:** el progreso de actividades que se hacen dentro del juego.
- Las metas del juego

Condiciones Terminales Frasca (Frasca, 2003) proponía otros tipos de reglas :

- **Reglas de Manipulación**: Define que puede hacer el jugador en el juego
- **Metas**: Definen la meta del juego
- **Meta reglas:** Definen como el juego puede ser alterado.

II.2 Videojuego

Un videojuego es la representación de un juego dentro de la computadora (Adams, 2013), funciona bajo el mismo concepto, pero las reglas suelen estar ocultas, además ahora la interacción es entre el humano (jugador) y la computadora, esta se efectúa a través de dispositivos de entrada y salida (Rollings & Morris, 2003); la computadora presenta una situación al humano a la cual este responde por medio de un dispositivo de entrada, la computadora al recibir esta respuesta, realiza una simulación en base a la entrada y produce una nueva salida que es presentada al humano, este proceso se sigue efectuando

hasta que el juego es dado por terminado por el jugador (Rollings & Morris, 2003; Zimmerman, 2006) véase Figura II.1.

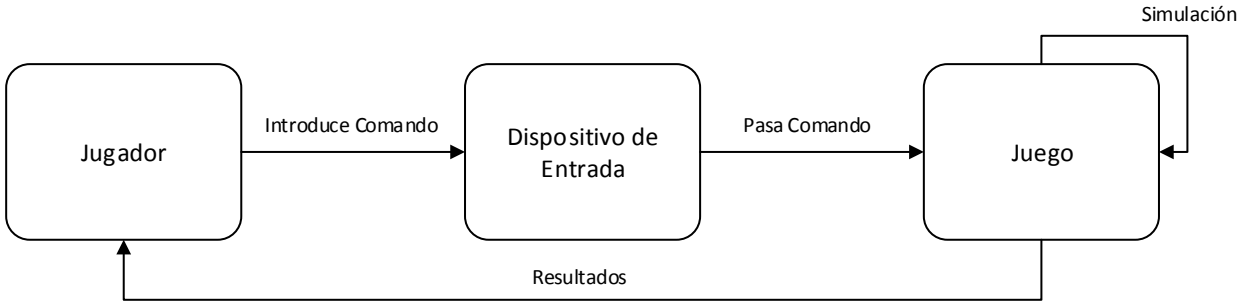

Figura II.1: Interacción de humano con videojuego

II.3 Historia de los Videojuegos

La historia de los video juegos comenzó con la creación de "Tenis for two" desarrollado por William Higinbotham en 1958 (Herman, 2001), simulando una partida de ping pong dentro de un osciloscopio, con los avances tecnológicos en 1972 surge la primera consola *Oddysey* diseñado por Ralph Baer (Herman, 2001), con la capacidad de conectarse a la televisión, después le siguió el *Atari 2600* en 1977.

Años después surge el boom de las arcades, en esta época surgen algunos video juegos que hasta la fecha son reconocidos, como *Space Invaders* (1979), *Pac-Man* (1980) y *Asteroids* (1981), en esta época era común solo guardar los puntajes más altos y almacenar las iniciales de la persona (B Medler, 2009).

En la década de los 80's surge una nueva generación de consolas, provenientes de Japón el Nintendo Entertainment System (1983) y el Sega Master System (1985); en estos tiempos no se contaba con la suficiente memoria para almacenar información, por lo cual se utilizaban contraseñas para acceder a los niveles avanzados, aunque al final existieron cartuchos que contaban con una batería para almacenar el estado del juego (B Medler, 2009).

Dentro de la siguiente generación se encuentran el Super Nintendo Enterteinment System (SNES) y Sega Mega Drive, se enfocaron en mejorar los apartados gráficos (Aún en 2D), además de que se encontraban más juegos con batería para almacenar avances. Después llegaron las primeras consolas con capacidad de gráficos en tres dimensiones, PlayStation (1994), Sega Saturn (1994) y Nintendo 64 (1996), algunos eran basados en discos compactos y hacían uso de memorias externas para almacenar los avances, mientras que el Nintendo 64 seguía utilizando cartuchos con batería interna.

Nuevamente un salto en la tecnología vino a dar un cambio al mundo de los video juegos con otra generación de consolas, esta vez conformado por el Nintendo GameCube (2001), Sony PlayStation 2 (2000), Sega Dreamcast (1998) y Microsoft con el Xbox (2001), en esta generación empezó a surgir en algunas consolas el uso del Internet para jugar partidas multijugador, incluye la creación de tablas de records mundiales por las cuales se competía por alcanzar un puesto.

Dentro de la 7ma generación de las consolas, se caracterizó por los fabricantes concentrándose más en el poder gráfico de la consola, Microsoft con Xbox 360 (2005) y Sony con el PlayStation 3 (2006), mientras tanto Nintendo lanzó el Wii (2006), intercambiando el poder gráfico por un control innovador basado en gestos físicos y movimientos con un mando a distancia. Con el tiempo esta innovación como modo de entrada fue adaptada por Microsoft y Sony con sus propios dispositivos, Kinect y Move respectivamente. En esta generación se caracterizó por darle importancia a la creación de perfiles dentro de la consola, que permitía a los jugadores capturar su información personal y almacenar sus logros (objetivos cumplidos dentro del contexto de un video juego) y poder comprarlo con sus amigos, además de aumentar el uso de tablas de puntaje a nivel mundial, regional y entre amigos (Ben Medler et al., 2011).

Ahora, recientemente se lanzaron al mercado la nueva generación de consolas el Nintendo Wii U (2012), aumentando el potencial gráfico comparado con su versión anterior pero optando nuevamente por innovar en cuanto al uso de los controles, proponiendo ahora un control/tablet. Por su parte Microsoft y Sony presentaron el Xbox One (2013) y el PlayStation 4 (2013) respectivamente, aumentando drásticamente el poder gráfico y de procesamiento comparado con sus versiones anteriores, enfocadas más al uso de las redes sociales y nuevas versiones mejoradas de sus dispositivos de captura de movimientos.

Los video juegos a través de su historia han ido evolucionando en muchos aspectos, siendo el más notorio los gráficos, pero también hay un cambio en cuanto a contenido, ya que antes eran reglas sencillas con un objetivo claro, el cual era divertir; con el paso del tiempo se han introducido nuevos aspectos, como una trama para desarrollar aun personaje, una historia intrigante, con la complejidad de un libro, pero sin perder la interacción que caracteriza un video juego.

Debido a estos cambios de complejidad sobre como mostrar los gráficos o las acciones posibles a ejecutar, la programación de videojuego se ve afectada a que esta incrementa sus módulos necesarios para ejecutar diversas tareas (Blow, 2004), por ejemplo, antes no se utilizaban inteligencias artificiales tan complejas, por lo que un módulo especializado no era tan necesario, en la actualidad ya hay

bastantes entidades dentro del juego que requieren un comportamiento ante lo que ocurre a su alrededor, desde una luz hasta un personaje, por lo que el tamaño de los programas ha ido en aumento, antes todo solía contenerse en menos de 1MB, ahora todo un juego puede llegar a necesitar hasta 25GB, aunque una buena porción de este espacio es ocupado por los elementos gráficos y audio.

II.4 Géneros de Videojuegos

Los videojuegos al igual que otros medios de entretenimiento, como lo son la televisión, libros y películas, existen los géneros que son utilizados para describir en cierto grado de que tratara, un género puede indicar tema, ubicación o algunos elementos que la conforman, estas clasificaciones son asignadas generalmente bajo el contexto de la jugabilidad y retos presentados, sin importar el tema u historia de este.

Algunos autores han propuesto algunas clasificaciones de videojuegos (Adams, 2013; Novak, 2011; Wolf, 2002). y en artículos del área científica se hace uso de géneros utilizados por los reporteros de la industria (Pinelle, Wong, & Stach, 2008).

La siguiente es una lista de géneros más comunes en la actualidad:

1 Acción

Dentro de estos juegos suelen presentarse retos de habilidades físicas y coordinación para el jugador, también pueden presentar a cierto grado acertijos a resolver, pero de una baja complejidad.

Este género se caracteriza por requerir una buena coordinación ojo mano y una buena velocidad de reacción.

Debido a que algunas características de este género pueden variar en gran medida ente varios videos juegos, existen alguno sub-géneros como lo son:

a) Disparos (Shooter)

Los jugadores toman acción a distancia con diversos tipos de armas, el apuntar es una habilidad muy importante para el jugador, usualmente hay que enfocarse en las áreas alrededor del personaje y los objetivos por cumplir. Existen dos tipos de juegos de disparos, de dos dimensiones y tres.

i) 2D

Usualmente cuentan con una vista aérea o lateral, se utilizan personajes ante un sinnúmero de enemigos que atacan simultáneamente mientras debes esquivar los ataques, los niveles pueden representar escenarios fantásticos y las físicas pueden variar de la realidad

ii) 3D

Son juegos más realistas (Comparados con los de 2D), presentan niveles más familiares, además de que las físicas suelen ser apegadas más a la realidad, este sub-género a su vez, tiene otras clasificaciones:

(1) **Disparos en riel:** Consiste en un camino guiado a través de espacios específicos.

(2) **Tácticos:** Suelen ser simuladores militares, dentro del cual los movimientos se realizan con planeación previa y se evitan acciones impulsivas.

(3) **Horror-Supervivencia**: Muestran un ambiente tenebroso y usualmente se cuenta con escasos recursos y se requiere de mucha exploración.

(4) **Arena:** Comunes en el ambiente multijugador, son combates frenéticos con diversos modos de juego, es uno de los subgéneros que demanda más velocidad de reacción por parte del jugador.

b) Plataforma

Presenta mundos caricaturescos dentro del cual el jugador maneja al personaje a través de diversos escenarios, sobre los cuales tiene que esquivar obstáculos además de enfrentar los enemigos que se encuentren en el camino hasta llegar al final del nivel, pueden ser en 2D o 3D;

c) Peleas

Simulan peleas mano a mano, aunque a veces pueden involucrar el uso de movimientos exagerados o alguna simulación de una arte marcial real, pueden ser en 2D o 3D.

d) Puzzle Rápidos

Requieren que el jugador resuelva problemas lo más rápido posible, la complejidad de estos juegos suele ser simple para poder mantener un ritmo rápido, el reto principal es tener la acabildad lógica para resolver los problemas.

e) Acción Aventura

Es una categoría híbrida que combina ciertas características de los juegos de acción y aventura, requiere del uso de habilidades físicas, además se puede contar con una historia narrativa que involucre múltiples personajes.

f) Ritmo Danza/Música

Juegos basados en música que requieren de esfuerzo y coordinación para completar una secuencia de acciones relacionadas con la danza, tocar un instrumento realizando las acciones mostradas en pantalla.

2 Estrategia

En este género la mayoría del reto presentado es un conflicto estratégico, donde se muestra una gran variedad de acciones posibles, la victoria se adquiere con una buena planeación y toma de decisiones. Los retos presentados en este género son logísticos, tácticos, económicos y de exploración, se pueden dividir en dos categorías

a) **Por Turnos:** En estos se maneja un ejército con cierta cantidad de movimientos y acciones delimitadas por un turno, al finalizar el turno, será turno del contrincante de realizar sus movimientos.

b) **Tiempo Real:** El conflicto está en constante movimiento y las tácticas se tienen que ir adaptando conforme a la situación que se está desenvolviendo, la toma de decisiones es crucial para ganar.

3 Rol

El jugador controla uno o más personajes en una serie de aventuras, la victoria consiste en terminar todas estas aventuras. Los personajes aumentan de nivel y adquieren nuevas habilidades conforme se avanza en el juego, los retos incluyen combate táctico, logística, economía, exploración y solución de acertijos.

a) **Guerra:** Contiene combate donde se utiliza un pequeño grupo de personajes, el juego mantiene seguimiento a las estadísticas de los personajes utilizados.

b) **Acción:** Generalmente se utiliza un solo personaje y se requiere un poco más de habilidad física que un juego de rol normal, puede contener una narrativa donde el jugador tiene la opción de interactuar eligiendo algún dialogo.

c) **Aventura:** Mundos detallados, sobre el cual el jugador toma control de un personaje ya existente y los retos son más de solución de acertijos que de combate.

4 Deportes

Simulan algunos aspectos de los deportes atléticos ya sean reales o imaginarios, puede ser que se controle a los personajes del equipo, se administre al equipo o ambos.

5 Simulación Vehicular

Crea una sensación de estar al mando de un vehículo real o imaginario, el objetivo de los reales es dar una experiencia muy cercana a la realidad.

6 Simulación de Construcción y Administración

En este tipo de juegos la mayoría del reto son económicos e involucran crecimiento, ya sea teniendo la administración de una ciudad, país o en la construcción de puentes o caminos; o simplemente administrando un negocio.

7 Aventura

Es una historia interactiva donde el jugador controla un personaje, la narrativa y exploración son elementos esenciales de este género.

8 Vida Artificial

Rama de la ciencia artificial, involucra el simulado de un proceso biológico. Los juegos están orientados a entretener por medio de la observación de la evolución de la simulación.

9 Puzzle

Aunque varios de los géneros se mencionan que cuentan con acertijos o puzzles a resolver, en este género se caracteriza por tener un enfoque exclusivo en resolver los acertijos a través de varios niveles, que van en aumento de dificultad, son atractivos visualmente, están enfocados en problemas de lógica.

La variedad de géneros indica una diferencia en cuanto a las acciones que pueden ser ejecutadas por el jugador, aunque en la arquitectura general la estructura puede ser común, dentro de ciertos módulos se deben de considerar diversos aspectos y dar mayor consideración a algunos módulos que otros, por ejemplo en los videojuegos del genero simulación vehicular se tiene el objetivo de mostrar escenarios reales y además que funcione apegado a las físicas reales (Adams, 2013), por lo tanto el modulo gráfico y el de simulación son los que reciben el mayor enfoque.

II.5 Desarrollo de Videojuegos

Un video juego, como todo proyecto de software conlleva una planeación y desarrollo, dentro de la industria algunos pasos pueden variar, estos son algunas de las fases que se pueden encontrar (Novak, 2011; Rollings & Morris, 2003).

1. **Concepto**: Dedicada a la idea inicial del juego y el inicio de la documentación que lo describe, durante esta etapa el equipo es relativamente pequeño.
2. **Pre-Producción**: Se desarrollan guías de diseño, planeación de la fase de producción y la descripción del documento del diseño del juego (trama, personajes, ambiente, temática, género, etc.) y diseño técnico.
3. **Prototipo:** La meta es obtener el primer software funciona, para demostrar las característica del juego, permite empezar a localizar los elementos que forman parte de la experiencia general del usuario.

4. **Producción:** Esta puede llevar desde semanas hasta años de programación, dentro de esta fase se hace uso de las metodologías de desarrollo iterativos o tipo cascada, para asegurarse del lograr cumplir con el desarrollo del juego en tiempo y forma.
5. **Localización** : En esta parte los aspectos de la jugabilidad e idioma son ajustados para ser aceptados por algunas partes del mercado internacional, esto debido a que en algunas regiones o países, es necesario apegarse a ciertas restricciones impuestas por los gobiernos en cuanto al contenido permitido (sangre, violencia, simbología, cultura).
6. **Alpha:** En esta fase ya se puede jugar de principio a fin, lo que permite hacer una evaluación de la diversión y experiencia del usuario.
7. **Beta:** El objetivo principal es corregir errores, se mejora la usabilidad y el juego se somete a certificación.
8. **Gold** : El juego está terminado y se envía a producción
9. **Post-Producción:** En esta etapa se realizan actualizaciones y parches a errores no detectados previamente, con el fin de mejorar la experiencia del usuario, además incluye la producción de contenido extra.

En sus inicios el desarrollo de videojuegos era solo lógica de circuitos que se encargaba de mostrar los gráficos y respetar las reglas, como fue el Breakout (1976) publicado por Atari, con el paso del tiempo, el auge de las consolas y computadoras, llego el Apple II sobre el cual se desarrollaron una gran cantidad de videojuegos en lenguaje ensamblador, juegos como *Karateka* o *Prince of Persia* de Jordan Merchner, con el avance tecnológico los juegos empezaron a ser programados en lenguaje C, aunque inicialmente era visto como un retraso ya que requería más recursos y no estaba tan optimizado como las rutinas hechas en ensamblador, pero se demostró que se podían hacer mejores cosas en menor tiempo, uno de los primeros juegos en demostrar las ventajas fue Doom (Cass, 2003; Rollings & Morris, 2003).

Los videojuegos eran sistemas pequeños (Blow, 2004; Folmer, 2007) que solo requerían tareas básicas como manejar entradas y salidas, además de poder mostrar gráficos 2D en manera rápido, reproducir audio y las simulaciones requeridas y ser ejecutado a una velocidad razonable.

Con el aumento de los recursos de cómputo, la próxima evolución del desarrollo fue a utilizar otros lenguajes de programación como C++ y el inicio del modelado 3D, los limitantes del pasado se fueron quedando atrás y se nota una mejora en el aspecto gráfico e interfaces de usuario.

Se inició el desarrollo de bibliotecas generalizadas para encargarse de manejar ciertos aspectos del juego, como el cálculo de las físicas, procesamiento de audio, búsqueda de caminos y detección de colisiones.

Así es como surgen los game engines, un conjunto de módulos para las funciones necesarias para crear un videojuego, que incluyen desde manejo de sistemas entrada y salida, audio, funciones básicas de inteligencia artificial, físicas; todo esto con la intención de aumentar la velocidad de desarrollo y hacer que dentro del proceso de desarrollo se enfoquen más a cosas nuevas y no a métodos ya probados; con esto se pretende tener juegos de mejor calidad e innovadores en busca de nuevas formas de entretenimiento (Deloura, 2009a).

Durante el desarrollo del videojuego la arquitectura no es necesariamente un aspecto que se considere de mucho peso (Murphy-Hill et al., 2014), solo se desea el producto final, aunque al final cuando se requiere regresar a mantener algún modulo o reutilizarlo, los desarrolladores llegan a arrepentirse de no dedicarle el tiempo.

En cuanto al diseño del juego, lo que principalmente se definen es la jugabilidad y los elementos del juego que van surgiendo así como las definiciones iniciales de las reglas del juego y los objetivos que tienen que cumplirse, descritas en lenguaje natural y no con descripción formal (Reyno & Cubel, 2009), ya si logran evocar emoción al jugador, durante las fases de prototipo, el diseñador asume que se va por buen camino, el diseño se enfoca a un sector del público y trata de darle gusto aprovechando las características de ellos, a esto se le conoce como diseño enfocado en el usuario (Charles & Black, 2004).

II.5.1 Elementos de un videojuego

Los videojuegos se caracterizan por contar con un game loop, esto es un ciclo que siempre se repite en todo juego y que se puede salir del cuándo el jugador desea dejar de jugar o cuando pierde en un escenario (véase Figura II.2), dentro de él simplemente se recibe la entrada del jugador, que puede ser desde un comando o nada, y esto es procesado para seguir avanzando el juego, donde todas las entidades u objetos que no son directamente controlados por el usuario, se mueven o cambian su comportamiento con la nueva información recibida.

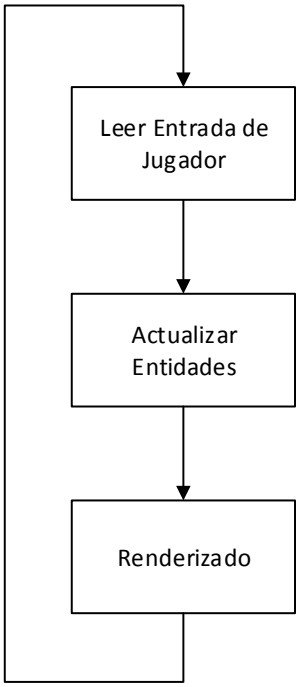

Figura II.2: Game Loop

Existe un metamodelo que define todos los elementos necesarios para un sistema de juego (las definiciones que conforman a los elementos del juego) (véase Figura II.3) (Reyno & Cubel, 2009).

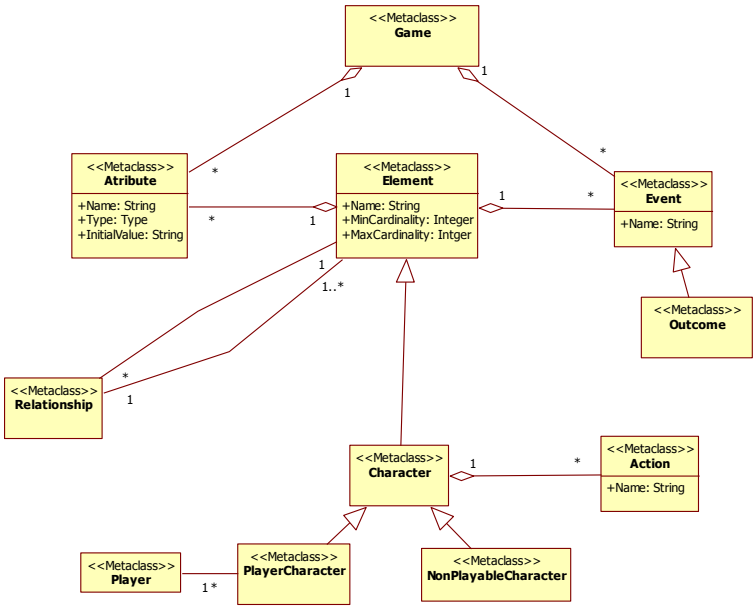

Figura II.3: Metamodelo de Sistema de Juego

Además de considerar a los personajes como un elemento, existen otros, como lo es la narrativa, interfaz gráfica, los niveles dentro del juego, el mundo que lo compone (Adams, 2013).

II.5.2 Motivación del Jugador

Los videojuegos, al ser un medio de entretenimiento, su principal objetivo es el de mantener el interés del jugador, para poder lograr esto se consideran las motivaciones del jugador, ¿Por qué juega? (Frome, 2007; Gilleade & Dix, 2004; Novak, 2011; Oxland, 2004).

Usualmente los jugadores buscan un reto, cuando ellos no encuentran diversión o el reto suficiente estos se aburren y dejan de jugar (Oxland, 2004).

Durante la fase de la creación del diseño de un videojuego, se considera primero a que público se va dirigido y en base a ello se decide qué características tendrá el juego, tratando de incluir a la mayor cantidad de gente posible, excluyendo a los que de plano no les agradaría el juego (Adams, 2013), durante esta fase se planea lo que es como ir evolucionando la historia del juego y los retos que estarán presentes, en ocasiones se preparan situaciones frustrantes (Gilleade & Dix, 2004) para el jugador, para elevar el desafío, pero no se tiene que exagerar o el jugador terminara enojado y dejaría de jugar.

Existe la teoría psicológica de Flow (Csikszentmihalyi, 2008), que cuenta con los siguientes componentes:

- Una actividad desafiante que requiere habilidades
- Mezcla de acción y conciencia
- Metas claras
- Retroalimentación directa e inmediata
- Concentración en la tarea delante.
- Sentido de control
- Perdida de la auto conciencia
- Alteración en el sentido del tiempo

Dentro de los videojuegos existe una experiencia similar a lo que el jugador experimentan en una sesión de juego, se pierde la noción del tiempo, presiones e intereses (Holt & Mitterer, 2000).

La intensión de la teoría de Flow en los videojuegos (Chen, 2007) es como encontrar mantener ese punto intermedio, para mantener al jugador interesado pero tampoco desesperado (Gilleade & Dix, 2004), también conocido como la zona de flujo, para poder lograr estar en este estado es necesario estar haciendo ajustes para que el jugador no se dirija hacia los sentidos extremos y quede en el medio.

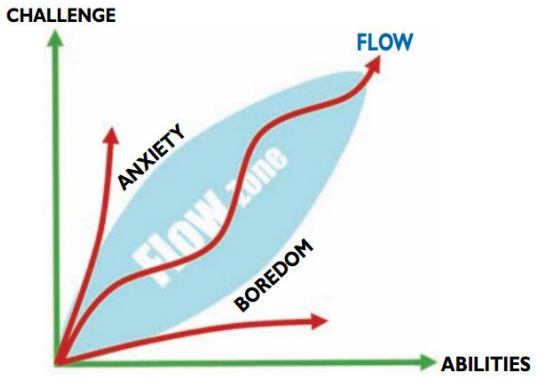

Figura II.4: Flow (Chen, 2007)

Usualmente un juego bueno es el que está ajustado de tal menare que el "flow" se mantiene a un ritmo entretenido para el jugador, usualmente los diseñadores tratan de llegar a esto haciendo ajustes al juego conforme a los resultados obtenidos durante las pruebas (Chen, 2007).

II.6 Arquitectura de Software

El ISO 1471-2000 (ISO/IEC, 2007) define una arquitectura como:

La arquitectura es la organización fundamental de un sistema encarnada en sus componentes, sus relaciones entre sí y con el medio ambiente y los principios que guían su diseño y evolución.

Otras definiciones que se encuentran son:

Una arquitectura es un grupo de decisiones significantes sobre la organización de un sistema de software, la selección de elementos estructurales y sus interfaces que lo componen, junto con su comportamiento especificado en la colaboración entre los elementos, la composición de los elementos en subsistemas progresivamente más grandes y en el estilo arquitectónico que guía esta organización (Kruchten, 2004).

La arquitectura de software de un programa o sistema computacional es la estructura o estructuras del sistema, que comprometen elementos de software, las propiedades visibles de estos elementos y la relación entre estos (Bass et al., 2003) .

II.6.1 Estilos de Arquitectura

Al igual que con los patrones de diseño de software, con el tiempo se fueron creado soluciones para necesidades de arquitectura para algunos problemas encontrados durante el desarrollo de software.

Un estilo de arquitectura define una familia de sistemas en términos de patrones con organización estructural. Específicamente un estilo de arquitectura define un vocabulario de componentes, tipos de conectores y restricciones sobre cómo pueden combinarse (Shaw & Garlan, 1996).

II.7 Métricas de Calidad de la Arquitectura de Software

Los métricas de calidad de software, surgen como una forma de poder evaluar una arquitectura, de tal manera que se encuentre una forma d reducir los costos de mantenimiento y modificación del software (Stevens, Myers, & Constantine, 1974), existen varias métricas de calidad, pero para este caso nos enfocaremos en las de cohesión y acoplamiento.

II.7.1 Cohesión

Es una medida en que tan bien se relacionan las responsabilidades de los atributos dentro de un componente. Una alta cohesión (un componente con responsabilidades en común) es considerado bueno, mientras que una baja cohesión (componente con pocos o ninguna responsabilidad en común) son considerados malos.

Una alta cohesión es buena porque hace que la arquitectura es fácil de entender y el mantenimiento también se haría más sencillo ya que los arreglos estarían contenidos dentro del componente en vez de en varios. Un componente es más reusable cuando tiene una alta cohesión debido a que este tiene un grupo de responsabilidades que es más probable a ser usado en conjunto que unas que estén al azar (Eeles & Cripps, 2009).

II.7.2 Acoplamiento

Esta métrica está relacionada con la cohesión, esta indica la fuerza de la asociación entre componentes, los componentes que están fuertemente acoplados están enlazados estrechamente así que los componentes son dependientes el uno del otro, mientras que los componentes débilmente acoplados tienen una asociación ligera o inexistente En este caso un acoplamiento débil tiende a ser más reusable, debido a que por definición pueden ser extraídos de un sistema y agregados a otro con una mayor facilidad (Eeles & Cripps, 2009) .

La estabilidad en la ingeniera de software, está relacionada con la cantidad de esfuerzo necesario para realizar un cambio o mover un paquete, en este caso se refiere a la dependencia que tiene el paquete hacia otros y los que otros que dependen de él (Martin, 2003), para ello se utilizan las siguientes formulas :

Acoplamiento Aferente (C_a) Numero de clases fuera del paquete que depende de clases dentro de este paquete.

Acoplamiento Eferente (C_e): Numero de clases dentro de este paquete que dependen de clases fuera de este paquete.

Inestabilidad:

$$I = \frac{C_e}{C_a + C_e}$$

II.8 Arquitectura de Software en Videojuegos

Como todo tipo de software, los videojuegos cuentan con algunos tipos de arquitectura utilizados en su desarrollo, pero debido a que la industria de los videojuegos a veces no cree en las buenas prácticas de la ingeniería de software, como lo es el diseño arquitectónico o quieren mantenerlo como un secreto de industria, no se cuenta con la documentación requerida, se hace lo que sea necesario para tener algo funcional (Murphy-Hill et al., 2014), por lo que estas pueden variar entre varias compañías de la industria, a continuación se muestran algunos de los tipos de arquitectura encontrados en la literatura, este tema no es muy investigado en lo que se concierne a la creación de referencias de arquitectura (Morelli & Nakagawa, 2011; Plummer, 2004).

II.8.1 Modelos de Arquitectura

II.8.1.1 Capas

Folmer (Folmer, 2007) propone una arquitectura de referencia por capas (véase Figura II.5), a continuación se describen dichas capas:

- **Capa de interfaz de juego**: Comprende a los objetos y componentes que contienen la lógica de juego, en esta paca todos especifico a los objetos dentro del juego son encontrados, así como la interfaz de usuario y la lógica utilizada.
- **Capa de dominio específico**: Esta capa encapsula módulos que son específicos para el dominio de los juegos tales como: gráficos, físicas, red, sonidos, entre otros.
- **Capa de infraestructura**: Estas son componentes que podrían ser reutilizados en diferentes dominios y con propósito general, servicios como la entrada y salida, administración de base de datos y abstracción de hardware, entre otros.

- **Software de la plataforma**: Software requerido para hacer funcionar el juego sobre determinada plataforma en la que se desea utilizar el videojuego.

Figura II.5: Arquitectura de referencia

II.8.1.2 Monolíticas

En este tipo de arquitectura el desarrollador se encarga en la totalidad del proyecto (véase Figura II.6) y se requiere tener experiencia en todos los campos que son requeridos para su elaboración, la desventaja de esto es que a veces se reutiliza código de proyectos anteriores y suelen introducir duplicidad de código o nuevos errores (Sollenberger & Singh, 2012), este tipo de arquitectura era el más utilizado en los inicios del desarrollo de videojuegos puesto que contaban también con una complejidad menor comparado con los producidos hoy en día, debido a esto era posible que una persona controlara todos los aspectos. Un ejemplo de arquitectura monolítica se puede apreciar en la Figura II.7.

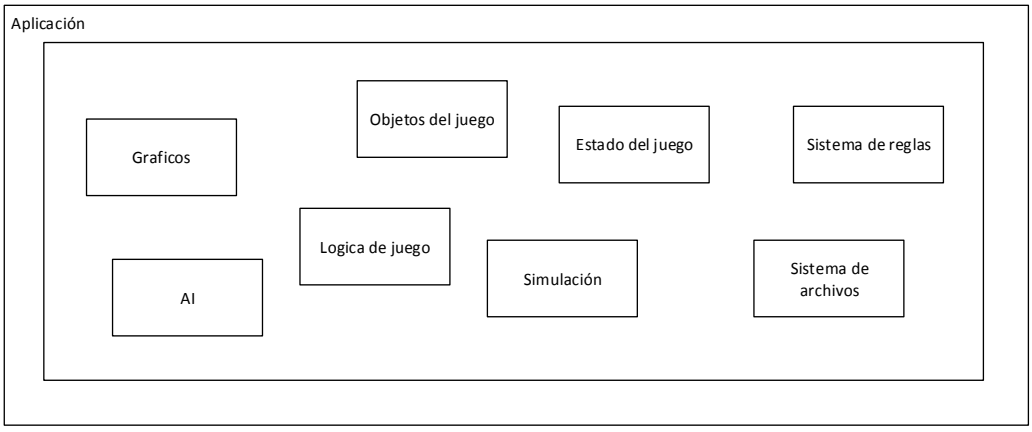

Figura II.6 : Arquitectura Monolítica

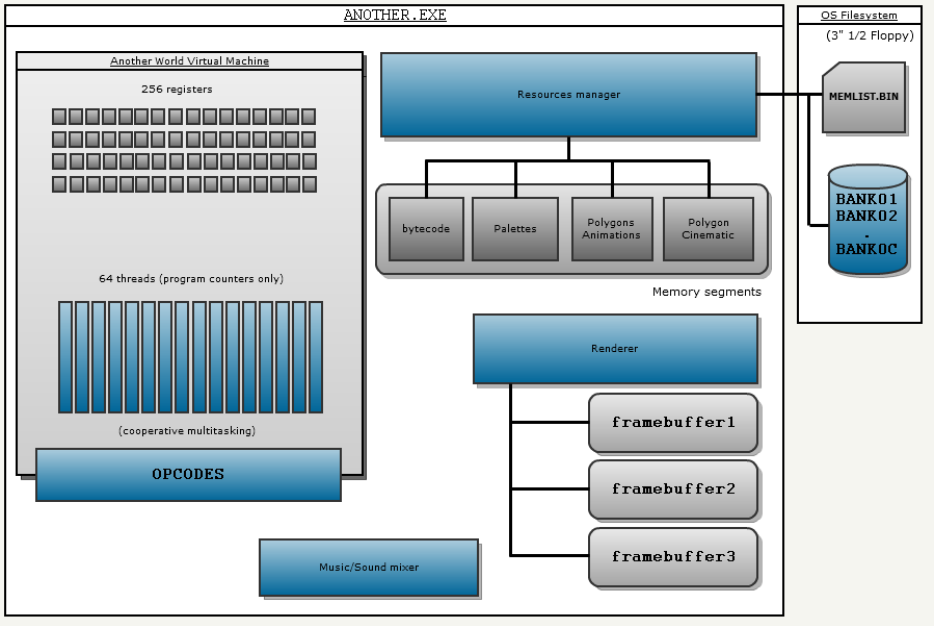

Figura II.7 : Arquitectura de Another World (Sanglard, 2011)

II.8.1.3 Modelo Vista Controlador

McShaffry (McShaffry & David Graham, 2013) propone el uso del modelo vista controlador como arquitectura para crear videojuegos y muestra una arquitectura de alto nivel, que separa en algunos módulos importantes al juego (véase Figura II.8).

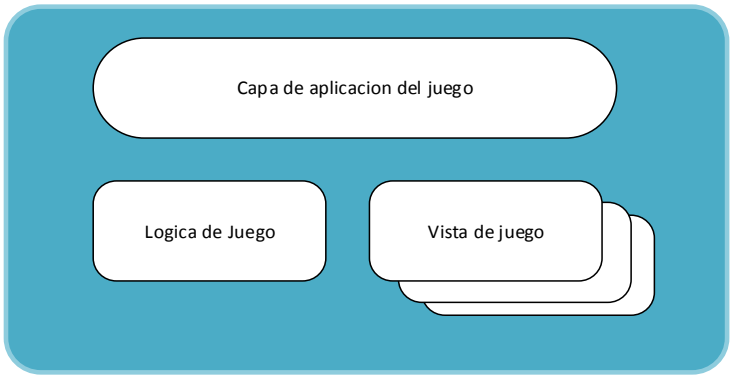

Figura II.8: Arquitectura videojuego alto nivel

Dentro de esta arquitectura se encuentran las siguientes capas:

- **Capa de aplicación de juego**: se encarga del manejo de los subsistemas de hardware o del sistema operativo, principalmente esta capa concierne sobre que dispositivo se ejecuta el juego, sea una computadora o consola de videojuegos, también se encarga de recibir los

comandos de los dispositivos externos, así como el manejo de archivos y memoria, inicializar el juego.

- **Capa de lógica de juego**: Es la parte más importante del videojuego, ya que define todo el mundo, que contiene y cómo interactúan entre si y cómo reaccionan, podría ser debido al jugador introduciendo comandos o a la Inteligencia Artificial moviendo a los personajes, un diagrama sobre cómo se compone esta capa puede verse en la Figura II.9, los subsistemas podrían ser descritos como los siguientes :

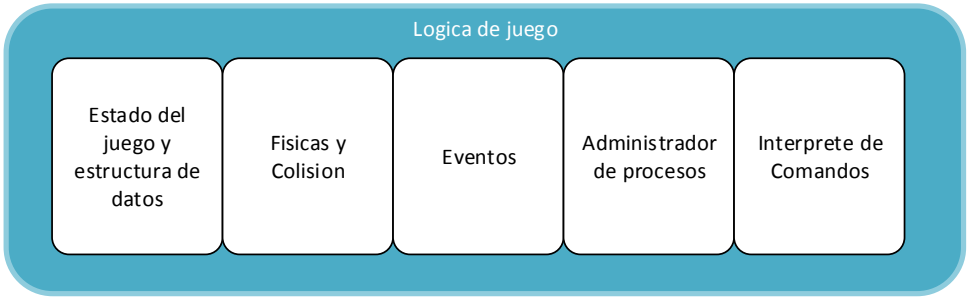

Figura II.9: Capa lógica de juego

  - **Estado de Juego y Estructura de Datos:** Guarda las propiedades de los objetos del juego, además se encarga de almacenarlos de manera eficiente.
  - **Físicas y Colisiones:** Parte muy importante del juego donde se llevan a cabo la simulación de como se ve afectado un objeto debido a la gravedad o al entrar en contacto con otro objeto.
  - **Eventos:** Los eventos sirve para notificar otras partes del sistema cuando se suscita algún cambio.
  - **Administrador de Procesos**: Administra el poder de procesamiento que se requieren ejecutar por cada objeto, ya sea para cálculos que requieren poder de procesamiento y los elimina cuando su tarea ha sido finalizada.
  - **Interprete de comandos**: Como su nombre lo indica, se encarga de darle un significado al comando recibido a uno que puede ser interpretado por el juego.
- **Vista de juego**: responsable de presentar el estado del juego y de recibir y enviar los comandos de entrada a la lógica de juego, esta puede ser vistas para él jugador o para la inteligencia artificial

- **Vista para los Humanos**: Responde a los eventos del juego y conocer como dibujar la escena y enviar información a los otros dispositivos de salida, como las bocinas, ver Figura II.10.

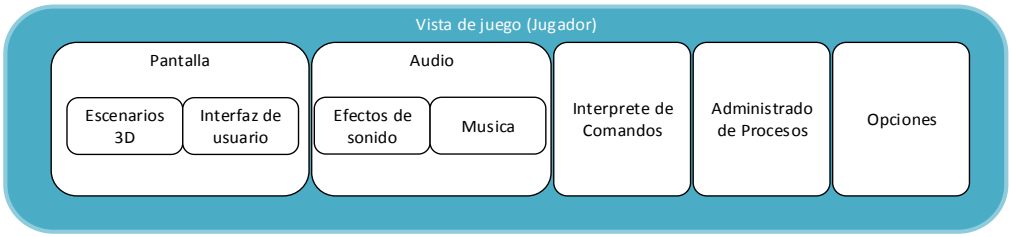

Figura II.10: Capa vista de jugador

  - **Gráficos** : Muestra en pantalla los objetos que hacen una escena del juego y la interfaz gráfica, hace uso del GPU para completar esta tarea, este es uno de los aspectos que más utiliza procesamiento dentro del videojuego y se tiene que estar haciendo a una velocidad rápida, para generar una experiencia deseable.
  - **Audio**: Reproduce los efectos de sonido y música requeridos por el videojuego en determinado momento.
- **Vista para la IA :** Dentro de este, se analiza el estímulo que recibe la Inteligencia artificial, como se movieron los objetos, si colisiono con alguno y debido a esto la Inteligencia debe tomar una decisión respecto a esto.

II.8.1.4 Component off the Shelf (COTS)

Este tipo de arquitectura es en la que se adquieren diversos componentes ya diseñados por terceros (Deloura, 2009b; Folmer, 2007; Morelli & Nakagawa, 2011; Plummer, 2004) también conocidos como *middleware*, estos están especializados en realizar acciones que pueden resultar repetitivas programar, tal como el renderizado del mundo virtual, para esto ya se cuenta con una seria de rutinas y modelos de vistas, que pueden ser reutilizados (programándolos para que así lo sean) estos son conocidos como Rendering engines (Gregory, 2009), también existen motores de físicas, audio, inteligencia artificial, gestores de archivos, el hacer uso de uno o varios de estos permiten a los programadores ahorrar tiempo reinventando la rueda y así enfocarse en programar los módulos necesarios que hacen diferente a su juego. Un ejemplo, aunque con módulos creados por una misma compañía se pueden observar en la Figura II.11.

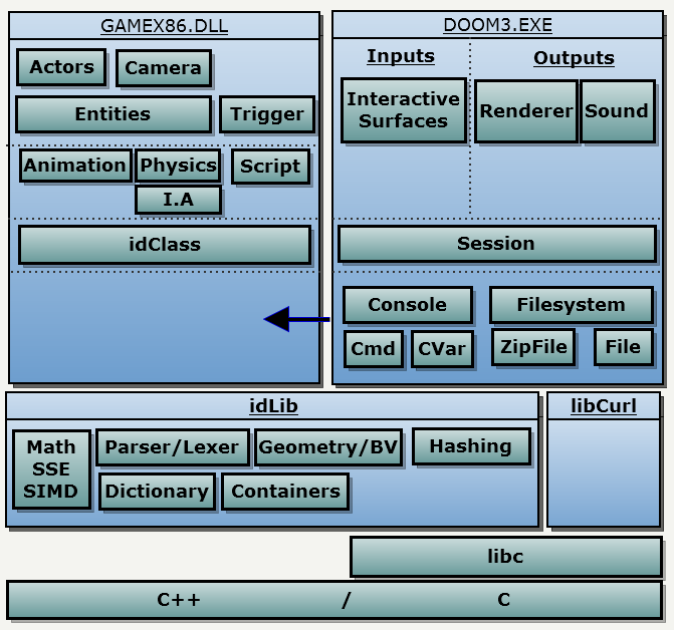

Figura II.11: Arquitectura videojuego Doom 3 (Sanglard, 2012)

II.8.1.4.1 Doherty

La descripción general de una arquitectura para videojuegos casi no es mencionada en la literatura o en artículos científicos, una aproximación fue propuesta por Doherty (Doherty, 2003) en la que maneja una arquitectura de alto nivel conformada por cuatro componentes (véase Figura II.12).

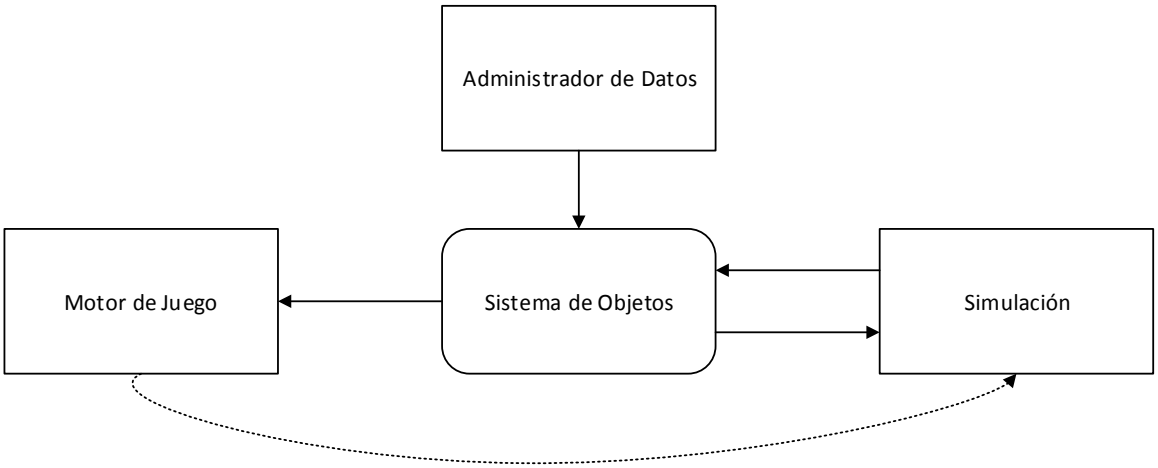

Figura II.12 Arquitectura Doherty

- **Administrador de Datos**

Se encarga de manejar el sistema de archivos, de cargar los datos persistentes, los recursos del juego (sprites, audio, modelos) y almacenar información.

- **Motor de Juego**

Es el componente encargado de interactuar con el usuario, ya sea recibiendo las entradas para enviarlas a la simulación o para mostrar al usuario el estado actual del mundo virtual, por medio de gráficos y elementos de audio, mostrando al representación grafico del estado actual del sistema de objetos a como fue afectado por la simulación, además se encarga de manejar la comunicación con el hardware por medio de la interface con el sistema operativo (véase Figura II.13)

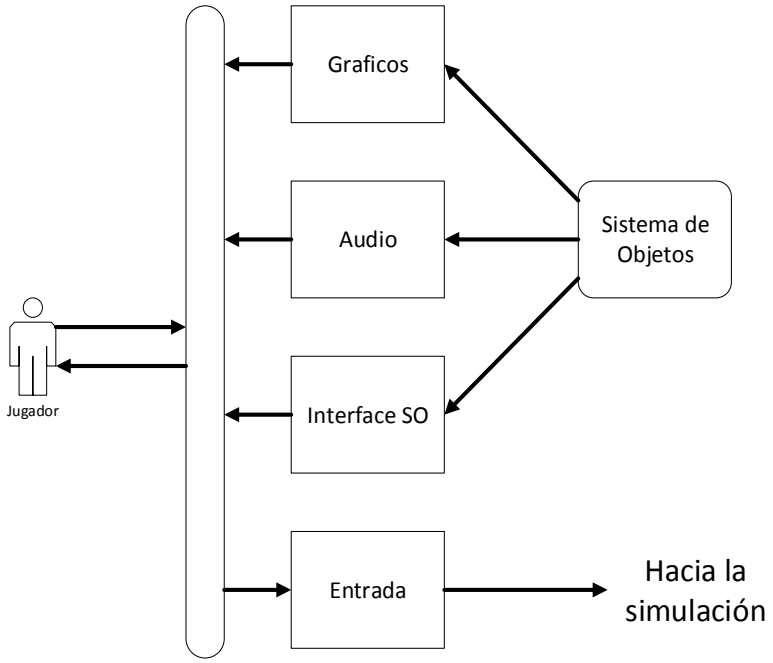

Figura II.13: Motor de Juego (Doherty)

- **Sistema de Objetos**

Es el encargado de mantener el estado de la información que describen los objetos del mundo del juego, estos residen en memoria.

- **Simulación**

Dentro de este módulo se encuentra la parte principal del juego, que es la encargada de mantener las reglas del juego y al recibir una entrada por medio del motor del juego, la simulación lo utiliza para hacer reaccionar a los objetos del juego, mover personajes, actualizar información, elegir el comportamiento de los enemigos, tomar decisiones que se requieren para representar el estado del juego (véase Figura II.14).

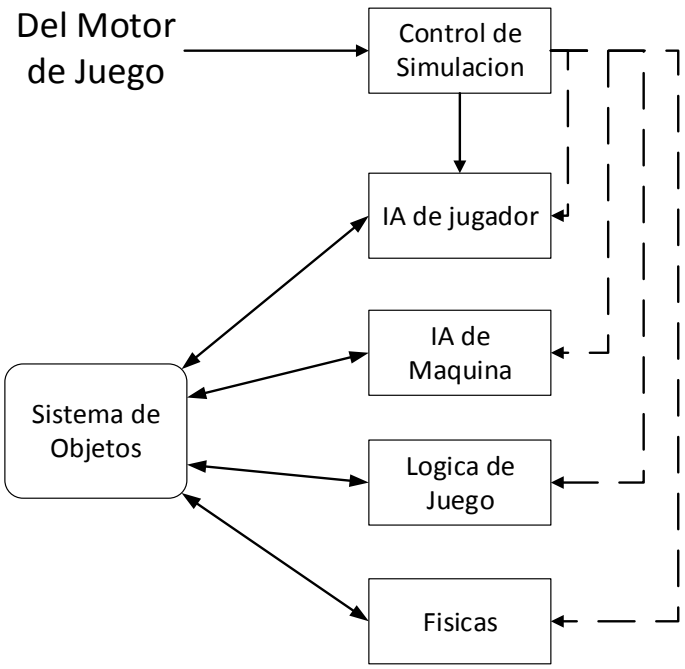

Figura II.14: Simulación (Doherty)

II.8.1.4.2 Rollings & Morris

Esta arquitectura descrita por estos autores no esta tan detallada (Rollings & Morris, 2003), pero muestra los subsistemas que lo conforman (véase Figura II.15) y se enlistan a continuación:

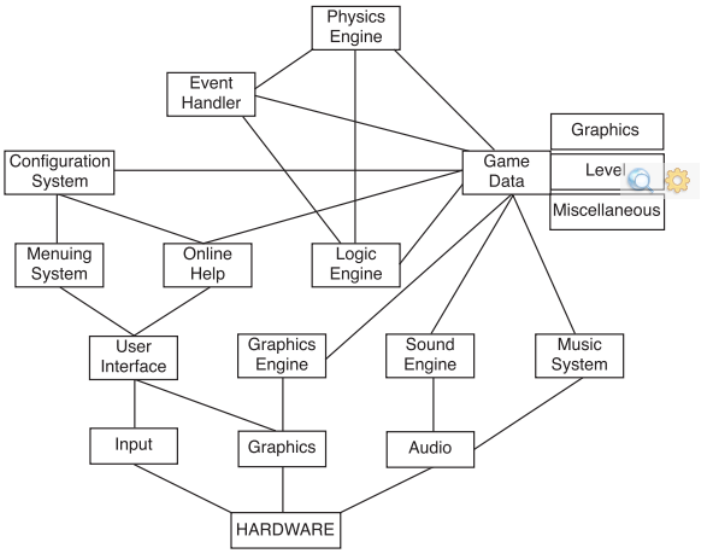

Figura II.15 : Arquitectura Rollings & Morris

Subsistemas Primarios:

- Interfaz de Usuario
- Manejador de eventos bidireccional
- Motor de Datos (gráficos, niveles, audio, etc.)
- Sistema de Dinámica (Colisiones y cálculo de físicas en general)
- Motor Lógico (El corazón del juego)
- Motor Gráfico
- Motor de Sonido
- Abstracción del Hardware (interfaz con las tarjeta gráfica y de sonido, y comunicación con controles)

Subsistemas Secundarios:

- Configuración del juego
- Sistema de menús
- Instrucciones y sistemas de ayuda
- Sistemas de Música

Estos subsistemas pueden ser reutilizados de otras partes.

II.8.1.4.3 Game Engines

Este tipo de arquitectura es más bien un COTS todo en uno, ya que se encarga de manejar lo que se utilizara con diversos componentes internos, de lo único que se debe de preocupar el desarrollador que utiliza el game engine, es la configuración de este, el contenido a utilizar y programar la lógica del juego y los objetos que lo conforman, (véase Figura II.16) el uso de los game engines en la actualidad es el método de desarrollo más popular debido a que hay una gran variedad y permiten a pequeños grupos de desarrollo no desperdiciar tiempo programando todo desde cero y se enfocan más a las funcionalidades deseadas (Deloura, 2009a; Novak, 2011), entre los engines más conocidos en la actualidad se encuentran: *Unreal Engine*, *Unity*, *CryEngine.*

Usualmente los game engine se caracterizan por tener dos tipos de módulos, los módulos núcleo, los cuales pueden ser reutilizados a lo largo de distintos juegos y son los que requieren mejor optimización en cuanto a rendimiento y están los módulos de *gameplay* los que son creados y utilizados específicamente para el juego que se está realizando (Nordmark, 2012).

La principal desventaja de los game engines es que están enfocados hacia un tipo o género de juego en general y hacia algún hardware en específico (Computadora o Consola) (Anderson, Engel, Comninos, & McLoughlin, 2008; Gregory, 2009; Plummer, 2004), una de las razones por la cual no se cuenta con un engine generalizado es por el hecho de que se podría desperdiciar el poder permitido por alguna consola o no se podría aprovechar al máximo algunas funciones útiles para un género de juego, aunque pueden ser adaptados a otros géneros, haciendo ajustes al código fuente que puede variar de un cambio simple a cambiar la totalidad de cómo funciona un módulo. Usualmente los game engines eran realizados para juegos de disparos en primera persona "*FPS*" por lo que se solían encontrar trabas al tratar de hacer un juego de otro género; los creadores de estas herramientas han tomado nota y las versiones más recientes cuentan con mucha más flexibilidad.

Los game engines, están formados por una gran cantidad componentes que trabajan en varias capas (Gregory, 2009), que van desde el manejo a bajo nivel de comunicación con dispositivos de hardware, administración de memoria, archivos y otros recursos, además cuenta con una librería matemática que puede ser utilizada por los otros subsistemas contenidos, como puede ser el de simuladores físicos e inteligencia artificial y una parte muy importante de los engines es el renderizado de los gráficos, en ocasiones permiten conectar otros middlewares para especializarse en algunas secciones del juego.

Otros módulos que puede contener es el del manejo de redes y un lenguaje de scripts (Moore & Sward, 2006), estos lenguajes de script son utilizados como una alternativa para hacer uso de los elementos contenidos dentro del game engine, usualmente cuentan con su propio lenguaje, aunque en algunas ocasiones permiten codificar en lenguajes de alto nivel, tales como C# o JavaScript, en otras ocasiones las compañías dan acceso total al código fuente del game engine lo que permite realizar una manipulación más a fondo de los módulos, esto usualmente en el lenguaje de programación de C++.

Los game engines al ser elaborados de una manera modular, permiten a los creadores realizar actualizaciones y mejoras a estos, sin necesidad de alterar demasiado a los otros módulos que forman al game engine.

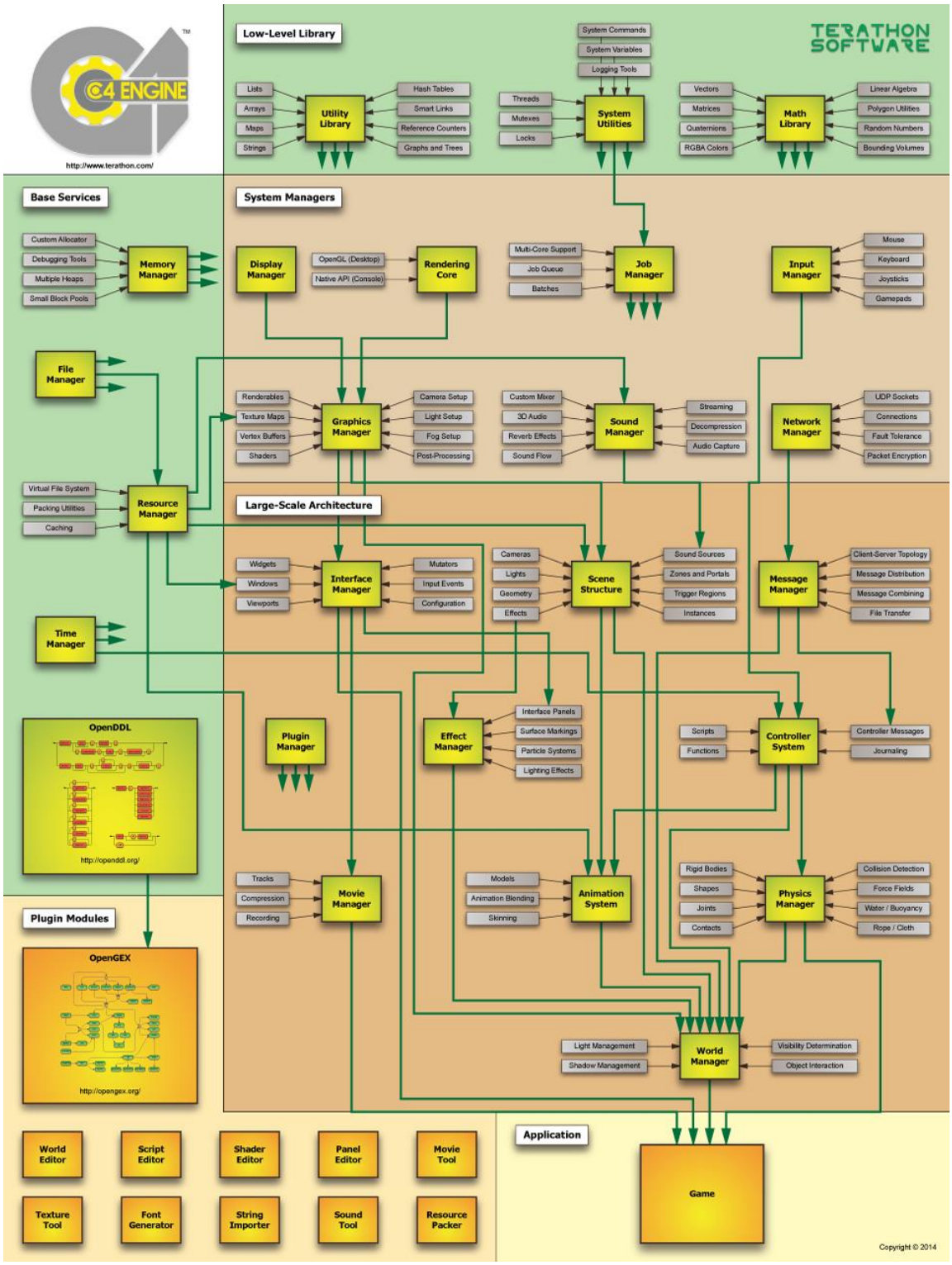

Figura II.16: Arquitectura C4 game engine

Ejemplo de Arquitectura de Componentes

Para el videojuego de código abierto Battle for Wesnoth (Shimooka & White, 2012), un juego de Estrategia por Turnos, se optó por una arquitectura de componentes (véase Figura II.17), usando la biblioteca SDL (*Simple DirectMedia Layer*), para tener acceso a desplegado de gráficos, uso de audio/video, manejo de entradas y sus eventos, esto les permite portabilidad en diversas plataformas (Windows, Linux y Mac).

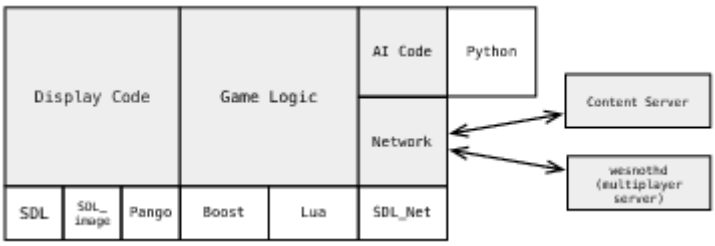

Figura II.17: Arquitectura de Battle for Wesnoth

Para poder tener una aumento en la flexibilidad del juego (la generación de nuevo contenido y configuraciones) se hace uso de un lenguaje de datos el Wesnoth Markup Language (WML), que es utilizado por personas sin mucho conocimiento de programación, para crear nuevos tipos de unidades, por lo que al estar estructurado de esta manera, permite expandir sobre el contenido del juego utilizando el mismo código base, siempre y cuando lo que se desea agregar no intente añadir nuevos comportamientos que requerirán modificar el núcleo del juego.

II.8.2 Estilos de Arquitectura

II.8.2.1 Orientada a Objetos

La manera tradicional de ver a los objetos del juego y cada uno de estos tiene sus respectivas subclases, aquí se pueden subdividir por las características que tienen en común (Gestwicki, 2012), como puede verse en la Figura II.18

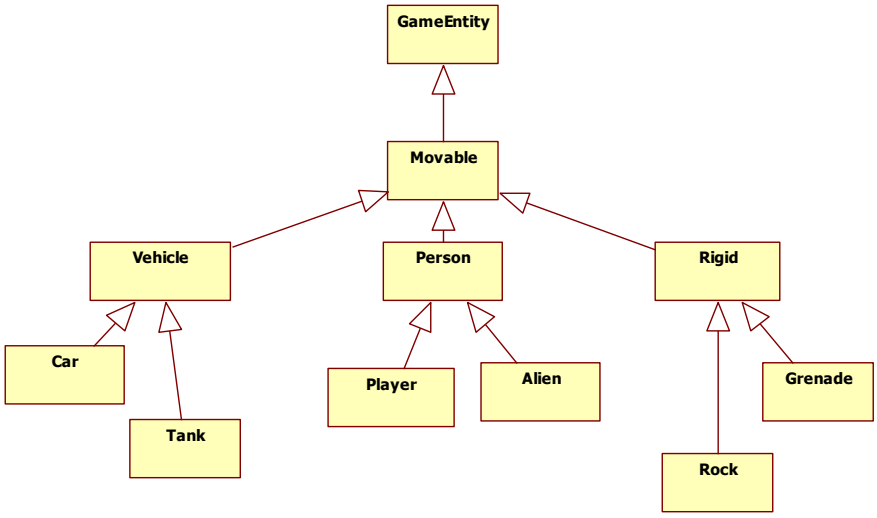

Figura II.18: Arquitectura Orientada a Objetos

II.8.2.2 Sistema de Entidades

Dentro de este tipo de arquitectura cada objeto del juego es llamado una entidad, el comportamiento de cada entidad está definido por cada componente que se le agrega, aquí tiene más valor la composición que la herencia. Los componentes solo contiene datos que son manipulados por medio de sistemas, cada sistema se encarga de manipular todas las entidades que contengan un componente que le pertenezca a dicho sistema (Gestwicki, 2012), este funcionamiento se puede observar en la Figura II.19.

Se cuenta con un ***Entity Manager*** responsable de mantener el grupo de agregaciones de una ***Entity*** con un grupo de ***Components,*** provee los métodos para su creación, destrucción, agregación, acceso y remover componentes, cada Juego contiene un ***Entity Manager*** y los sistemas requeridos para operar los componentes con los que cuentan las entidades, las entidades y componentes no contienen algún tipo de lógica solo datos, la lógica del juego está contenida dentro de cada sistema.

El sistema de entidades es recomendable, cuando el juego crece en bastante complejidad en cuanto a las entidades requeridas, debido a que si se utiliza la herencia puede que las clases base se saturen de demasiados métodos.

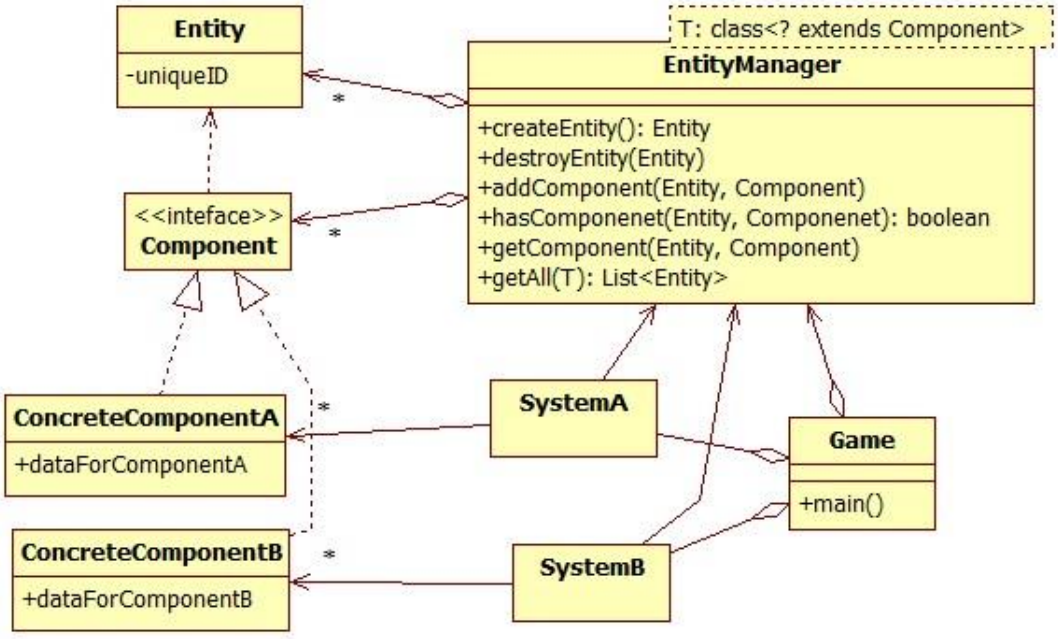

Figura II.19: Sistema de entidades

II.9 Emociones

Las emociones, son una experiencia que tiene todos los humanos, la cuál puede ser percibida de diferentes maneras, dependiendo de la persona por lo que tener una definición concreta es algo complejo, Scherer (Scherer, 2005) la define como:

> *Emoción es un episodio de cambios sincronizados e interrelacionados en los estados de todos o la mayoría de los subsistemas en respuesta a la evaluación de un estímulo externo o interno que es relevante a aspectos importantes del organismo.*

II.10 Computo Afectivo

El cómputo afectivo es el estudio y desarrollo de sistemas y dispositivos que pueden reconocer, procesar y simular las emociones humanas (Picard, 2000).

Las áreas principalmente en la que se enfoca la computación afectiva son: la detección y reconocimiento emocional y el perimir que las computadoras den una respuesta emocional, la computación afectiva puede clasificarse en cuatro clases (véase Tabla II.1), una forma de mejorar la detección del estado afectivo de una persona además de tener un modelo computacional, es haciendo el uso de sensores, que nos den datos en tiempo real, para alimentar al sistema (Picard, 2000).

Tabla II.1: Categorías del cómputo afectivo

Computadora	No puede expresar afecto	Puede expresar afecto
No puede percibir Afecto	I	II
Puede percibir afecto	III	IV

La descripción de las categorías es la siguiente:

I. La mayoría de los sistemas se encuentra en esta categoría.

II. Se enfoca en crear voces generadas por computadora con entonación natural y capacidad de mostrar caras con expresiones naturales.

III. Percibe a la computadora, percibe el estado afectivo del usuario, permitiéndole ajustar sus respuestas de tal manera que permita una mejor asistencia.

IV. Maximiza la comunicación entre el humano y la computadora, provee potencialmente una manera de trabajo más personal y/o amigable para el usuario.

II.10.1 Juegos Afectivos

El futuro de los juegos afectivos, está en que los videojuegos pueden ser lo suficientemente inteligentes para extraer los comandos del jugador no solo por el control, si no por el habla, gestos, comportamiento y su estado emocional (véase Tabla II.2), lo que permitiría adaptar al juego acorde a esto (Kotsia, Zafeiriou, & Fotopoulos, 2013).

Tabla II.2: Fuentes de Información Afectiva

Tipo	Subtipo
Visión	Expresión Facial
	Acciones del cuerpo
Interfaz Cerebro Computadora	Electroencefalograma (EGG)
	Espectroscopia del infrarrojo cercano
Mediciones Fisiológicas	Sensor de Cerebro
	Sensor de Ritmo Cardiaco
	Sensor de Respiración
	Sensor de Conductividad en la piel
	Captura de Movimientos

Los beneficios que puede traer esto el involucrar estas mediciones, de acuerdo a Sykes y Brown (Sykes & Brown, 2003) son:

- La habilidad de generar contenido dinámicamente respecto al estado afectivo del jugador
- La habilidad de comunicar el estado afectivo de un jugador a otros.
- Adoptar nuevas mecánicas de juego respecto al estado afectivo del jugador.

En el ámbito de la industria, Valve (Ambinder, 2011), ha demostrado el uso del algunas formas mediciones para alterar sus juegos ya existentes tales como EGG, revisión de imágenes y sensores de ritmo cardiaco, se han realizado algunos experimentos pero aún no se ha realizado algún anuncio de alguna publicación comercial. Mientras tanto Microsoft por su parte haciendo uso del nuevo sensor Kinectv2 (Microsoft, 2013), que han demostrado medir el calor corporal, ritmo cardiaco y hasta cierto punto emociones y mejoras a sus anteriores funciones como la detección del esqueleto, el cual aún no ha sido demostrado en los videojuegos de su actual consola y para investigadores saldrá la versión de Windows este verano.

Capítulo III

Trabajos Relacionados

III. Trabajos Relacionados

III.1 Modelado de jugador

Recientemente se han estado realizando investigación sobre el modelado del jugador (Bakkes, Spronck, & van Lankveld, 2012; Charles & Black, 2004; Houlette, 2003; Karouzaki & Savidis, 2012; Pedersen, Togelius, & Yannakakis, 2009; Sharma, Mehta, Ontanón, & Ram, 2007; Smith, Lewis, Hullett, Smith, & Sullivan, 2011; Yannakakis, Spronck, Loiacono, & André, 2013), con la finalidad de lograr desarrollar un videojuego adaptativo, es decir que se ajuste a las necesidades y/o habilidades del jugador, esto surge debido a que no todos los jugadores interactúan de igual manera (Charles & Black, 2004) y la adaptación es realizada con la intención de mantenerlos entretenidos por más tiempo.

El modelado de jugador es el estudio de modelos computacionales de los jugadores de juegos, esto incluye la detección, modelado, predicción y expresión de las características del humano que se manifiestan por medio de patrones cognitivos, afectivos y de comportamiento (Yannakakis et al., 2013).

Existen diversos autores que proponen un modelado de jugador (Smith et al., 2011), no todos convergen sobre el mismo tipo de dominio, mientras que algunos se enfocan al nivel de las reacciones humanas, otros simplemente utilizan el propio sistema de reglas de un videojuego; algunos se enfocan en la persona, otros en una subclase de jugadores y aplicables a todo tipo.

La intención de esta investigación es proponer parte de un modelo de jugador que consiste en el estado emocional del jugador y que el videojuego actué conforme a este, para esto es necesario establecer un modelo de las emociones que son experimentadas en el transcurso de una partida y tomar una acción conforme a los cambios detectados.

Existe investigación conforme a crear un perfil de jugador de videojuego Bakkes (Bakkes et al., 2012) menciona que se cuenta con varias formas de modelar el comportamiento de un jugador:

- Modelado de acciones
- Modelado de técnicas
- Modelado de estrategias
- Perfil de Usuario

III.1.1 Modelado de acciones

Consiste en listar todos los estados del juego y las probabilidades de que cada acción ocurra en dicho estado, este modelado es comúnmente utilizado en juegos de mesa, tal como el ajedrez, donde

las acciones suelen ser pocas y estas se pueden llegar a evaluar los próximos movimientos, este tipo de modelado es útil para implementar inteligencias artificiales, siendo útil para tareas sencillas, pero difícil de generalizar.

III.1.2 Modelado de técnicas

Consiste en la observación de una o varias unidades organizadas con un objetivo en común o una seria de acciones que llevan a una meta, este modelado usualmente se genera basado en casos que se determinan mediante observación, este modelo es útil sólo para los juegos con un bajo nivel táctico.

III.1.3 Modelado de estrategias

Consiste en la observación de tendencias o planes a largo plazo de los jugadores, este puede ser utilizado en crear inteligencias artificiales adaptativas para contrarrestar dichas estrategias, este modelo toma tiempo ya que se van capturando nuevas estrategias conforme se van detectando, esta técnica es útil para la generalización, pero requieren una gran cantidad de observaciones.

III.1.4 Perfil de Usuario

El enfoque que se le ha dado es establecer perfiles sociológicos y psicológicos verificables, que proveen motivación o explicaciones a comportamientos observados. Un perfil destaca las características internas del jugador, mientras que un modelo captura el comportamiento externo; una desventaja del perfil de usuario es que es único por jugador (Bakkes et al., 2012) y se requiere de algunas herramientas que elaboren el perfil de una vez y pueda ser utilizado por diversas aplicaciones, pero la gran ventaja es que da una amplia cantidad de información sobre la experiencia y satisfacción del jugador.

III.1.5 Modelado de Personalidad

Zhou (Zhou et al., 2007), habla sobre la creación de un modelo de usuario, el cual considera algunos rangos de la personalidad, por medio de atributos de preferencia, actitud, tendencia del personaje, conocimiento y habilidades. Menciona que la personalidad del usuario puede ser descrito en tres grupos de atributos:

- Información demográfica: edad, tiempo trabajando, grado académico, especialidad, etc.
- Características del usuario: actitudes, cualidades del personaje, preferencias, habilidades, puntos de conocimiento, etc.
- Comportamiento de usuario: Como se comporta el usuario, como actúa, si toma decisiones rápido, en qué sentido moral, que tan rápido presiona los comandos, como mueve los ojos, tiempos de respuesta, expresiones faciales, etc.

Proponen que para obtener el perfil psicológico del jugador es necesario hacer ciertas pruebas, pero que debería integrarse en pequeñas dosis, cada vez que entra el usuario, con el fin de formar un perfil aproximado y no estarle tomando demasiado tiempo al jugador para que este no se aburra.

III.2 Tipos de Jugadores

Los tipos de jugadores, son clasificaciones las actividades que les gusta realizar a ciertos jugadores, las preferencias de juegos que tienen, a continuación se muestran algunas clasificaciones que se han realizado.

Según Richard Bartle (Bartle, 1996) evaluado a los comportamientos de los jugadores de MUD (Aventura multijugador), detecto los siguientes tipos de jugadores:

- **Socializadores**: disfrutan de aprender y comunicarse con lo demás jugadores.
- **Asesinos**: Disfrutan manipular a los demás jugadores.
- **Triunfador**: Disfruta de interactuar con el mundo del juego (pasear, hacer todo lo que se puede hacer, tener el nivel máximo).
- **Exploradores**: Disfrutan manipular el juego.

Después Bartle (Bartle, 2003), reanalizo los tipos creados ya que en teoría todas las personas en algún momento pasan por todos los tipos diferentes, por lo que en 2003 propuso ocho tipos de jugador:

- **Oportunistas**: Si ven la oportunidad de obtener algo la toman, buscan que hacer y no saben que es hasta que lo encuentran, si se llegan a encontrar con algún problema, dejan lo que se proponían y buscan otra cosa.
- **Planeadores**: Tienen una meta fijada y sus acciones son con el fin de llegar a dicha meta, si se encuentran con un obstáculo, buscan la manera de pasarlo.
- **Científicos**: Hacen sus teorías y experimentan sobre ellas además buscan la explicación de los fenómenos y adquieren conocimiento.
- ***Hackers***: Experimentan para revelar significado, entienden de manera intuitiva el mundo virtual, pero no requieren probar lo que saben.
- ***Networkers***: Buscan personas con las cuales interactuar, se esfuerzan por conocer a los demás y aprenden sobre ellos.
- **Amigos**: Interactúan principalmente con las personas que conocen bien, disfrutan su compañía y los aceptan como son.

- ***Griefers***: Les gusta atacar a los demás, sin razón alguna, les gusta ser conocidos con una mala reputación.
- **Políticos**: actúan precavidos, manipulan sutilmente a las personas, explican sus acciones con las contribuciones hacia la comunidad, apunta a ser grandes y con buena reputación

Existe otra clasificación de tipos de jugador, la cual está basada en los ejes de personalidad obtenidos de las pruebas Myers-Briggs (Briggs & Myers, 2010), cada eje es opuesto y cada persona tiene diferentes valores entre estos ejes, los cuales son:

- Introversión (I) contra Extroversión (E)
- Sentir (S) contra Intuición (N)
- Pensar (T) contra Sentimiento (F)
- Juzgar (J) contra Percibir (P)

Bateman y Boon (Bateman & Boon, 2005) proponen una clasificación de cuatro personalidades de jugadores, en base a esta prueba, definen la forma de juego que prefieren y las subdividen en dos subtipos, casual y hardcore, el primero es el que juega esporádicamente y el ultimo son los que juegan demasiado tiempo.

Conquistadores

Tipos de Personalidad: ISTJ, INTJ, ESTJ y ENTJ.

Características Comunes

- Interesados en ganar o terminar el juego
- Completar el juego
- Ganar es lo más importante
- Derrotar a los demás (Multijugador)
- Avanzar rápido por el juego (Ignoran objetivos opcionales)
- Consideran parte de la historia del juego como irrelevante, solo las partes principales
- Disfrutan competir en línea y demostrar que son los mejores
- Necesidad de ganar argumento

Disfrutan discutir de juegos Hardcore:

- Entender el juego (Saber todos los movimientos y como usarlos)
- Completar el juego (Si no lo completan, sienten que no lo terminaron)
- Genero Popular :

Juegos de Acción Casual:

- Buscan diversión mientras compite por ganar

- Se Involucran en competición directa
- Genero Popular :
    - Juegos de disparos en primera persona (FPS)
    - Carreras

Administradores

Tipos de Personalidad: ISTP, INTP, ENTP y ESTP

Características Comunes

- Buscan reto estratégico
- Buscan reto táctico
- Interesados en ser “maestros” del juego
- Disfrutan juegos abiertos (sin un fin determinado)
- Géneros especializados
    - Estrategia
    - Administración
    - Construcción
- Juegan con calma y se rinden si el reto es demasiado
- Disfrutan la historia de manera intelectual

Hardcore

- Deseo de gameplay estratégico
- Poseer una seria de herramientas para superar los retos presentes
- Superar los retos debido a sus habilidades

Casual

- Desean realismo

Viajero/Vagabundo

Tipos de Personalidad: INFP, ENFP, ISFP y ESFP

Características Comunes

- Buscan una experiencia divertida
- Dejan de jugar cuando ya no les parece divertido
- Les interesan los personajes y sus características emocionales
- Hablan de los juegos que les gustan

Hardcore:

- Interesados en terminar el juego de manera adecuado
- Juegan demasiados juegos

Consideran que algunos juegos son difíciles Casual:

- Necesitan un juego fácil
- Quieren sentir que están logrando algo dentro del mundo del juego
- No es necesario que se les indique que están siendo retados

Participantes

Tipos de Personalidad: ESFJ, ISFJ, ENFJ, e INFJ.

Características Comunes:

- Juegan por una experiencia social
- Orientados por la historia
- Participa con otros jugadores en un contexto emocional
- Están interesados en la narrativa, ya sea de un grupo de personas o personajes
- Se interesan en los personajes del juego
- Disfrutan juegos multijugador (preferentemente no online)

Hardcore:

- Enfocados en los RPG (dan sentimiento participativo)

Lo primordial es la colaboración Casual:

- Disfrutan jugar como parte de un grupo
- Disfrutan juegos cooperativos

III.3 Ajuste de dificultad en Videojuegos

Los ajustes de dificultad, dentro de los videojuegos surgen debido a que se requiere permitir a que diferentes jugadores con distintas habilidades logren experimentar el juego y mantener su interés (Gilleade & Dix, 2004) y entretenerlo la mayor parte del tiempo, ya que si al jugador se le presenta un reto imposible, este dejara de jugar debido a la incapacidad de poder superarlo o al contrario no darle retos y sentirá una victoria vacía y perdería el interés (Charles & Black, 2004).

Los tipos de dificultades pre-definidos en un videojuego tradicional vienen desde la fase de diseño (Charles & Black, 2004), en la cual se prueban con distintas dificultades y se analizan en la etapa de pruebas y se toman las que dan un mejor reto o balance en cuanto a las posibilidades de superar los retos, al final se le presenta al jugador una elección limitada de posibles dificultades, con niveles que no suelen cambiar o contienen elementos estáticos, el jugador al ser una entidad cambiante con el paso del tiempo, las dificultades disponibles pueden dejar de ser de su agrado y así reducir la experiencia (Gilleade & Dix, 2004), otra característica de los videojuegos actuales es que no se suele hacer un monitoreo del desempeño del usuario mientras está jugando, para hacer ajustes al reto o a la experiencia (Charles & Black, 2004).

Una de las maneras de saber cuándo y cómo hacer el ajuste de dificultad viene de la recolección de datos sobre el jugador (Charles & Black, 2004) y análisis de los atributos numéricos dentro del juego, que sirven para describir el estilo de juego de un individuo, cada uno de los atributos se pueden asociar con alguna habilidad (Houlette, 2003), pero se debe de considerar que los cambios realizados no cambien drásticamente al juego de manera significante, en la Figura III.1 se puede observar cómo funciona de esta manera:

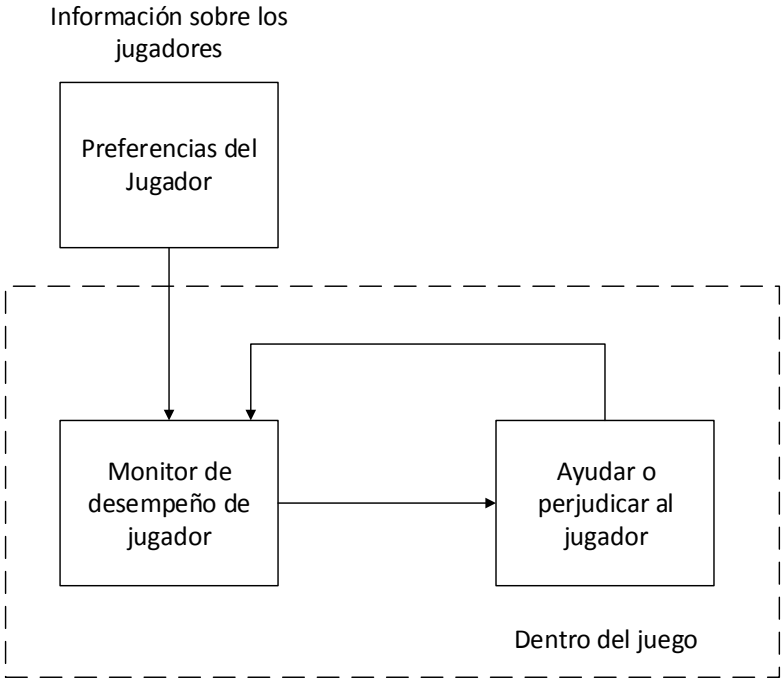

Figura III.1: Juego que cambia en respuesta al desempeño del jugador (Charles & Black, 2004)

Si se tiene un modelado de los tipos de jugadores, podría utilizarse en conjunto con las preferencias del jugador para evaluar su desempeño y así adaptar el juego a determinado individuo (véase Figura III.2) (Charles & Black, 2004).

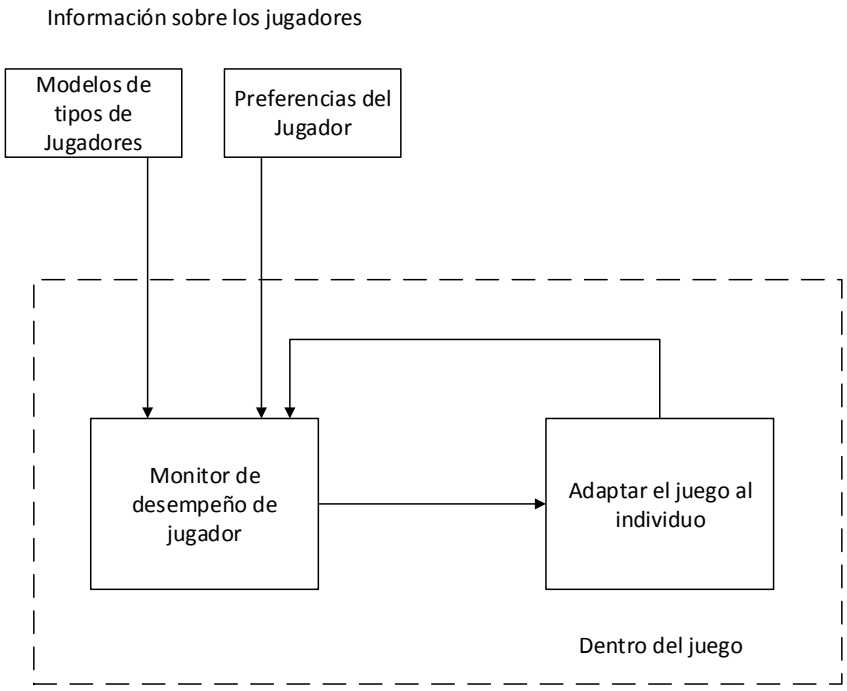

Figura III.2 : Juego que cambie conforme a un individuo

Lo ideal es que el videojuego sea capaz de adaptarse dinámicamente al jugador, para que la experiencia del usuario sea la adecuada para él (Bakkes et al., 2012), esto sería posible a través de analizar las emociones del jugador (Gilleade & Dix, 2004).

III.4 Emociones en videojuegos

Los videojuegos contienen un flujo dinámico de eventos y acciones, por lo que los juegos tienen el potencial de generar una variedad de emociones a través del tiempo (Zhou et al., 2007).

Frome (Frome, 2007), menciona varios tipos de aspectos que pueden influir en las emociones del jugador, y menciona las maneras en las que estas pueden ser provocadas:

- Emociones de Juego: emociones generadas por eventos del juego, como ganar, perder, realizar un logro y frustración.
- Emoción narrativa: emociones basadas en los personajes, historia, ambiente y eventos (de la historia).
- Emoción a artefactos: emociones provocadas por el aspecto estético del juego.
- Emoción ecológica: reacciones psicológicas basadas en experiencia en el mundo real.

III.4.1 David Acevedo

David Acevedo (Acevedo, 2009) hizo una propuesta (véase Figura III.3) basado en el modelo emocional OCC (Ortony, Clore, & Collins, 1990), que se encargaba de cambiar el comportamiento de una IA, que simulaba tener emociones, influenciadas por las acciones del personaje controlado por el jugador, el modelo, maneja los siguientes módulos :

- Entorno: Mundo donde se alojan los objetos y agentes, los agentes con sus acciones alteran el entorno.
- Evento: Señales que utilizan los objetos dentro del sistema para comunicarse unos con otros.
- Acontecimiento: conforme a la teoría OCC, es una creencia acerca de lo que sucede en el entorno, además incluye los resultados de las acciones realizadas por los agentes.
- Generador de Acontecimientos (GA): Generar eventos que encapsulan una abstracción con la información relevante para los agentes de un acontecimiento ocurrido.
- Habilidades: Representa las capacidades del personaje con efecto sobre el entorno.
- Comportamientos: Conjunto de habilidades definidas que se ejecutan en secuencia.
- Modulo Afectivo Individual (MAI): Representa las emociones y la intensidad con las que las manifiesta un individuo, además de representar las mentas, normas y actitudes para valorar el impacto de los acontecimientos sobre las metas y las acciones en base a reglas.
- Sistema Cognitivo (SC): Se encarga de tomar la información relevante para producir un comportamiento en el agente.

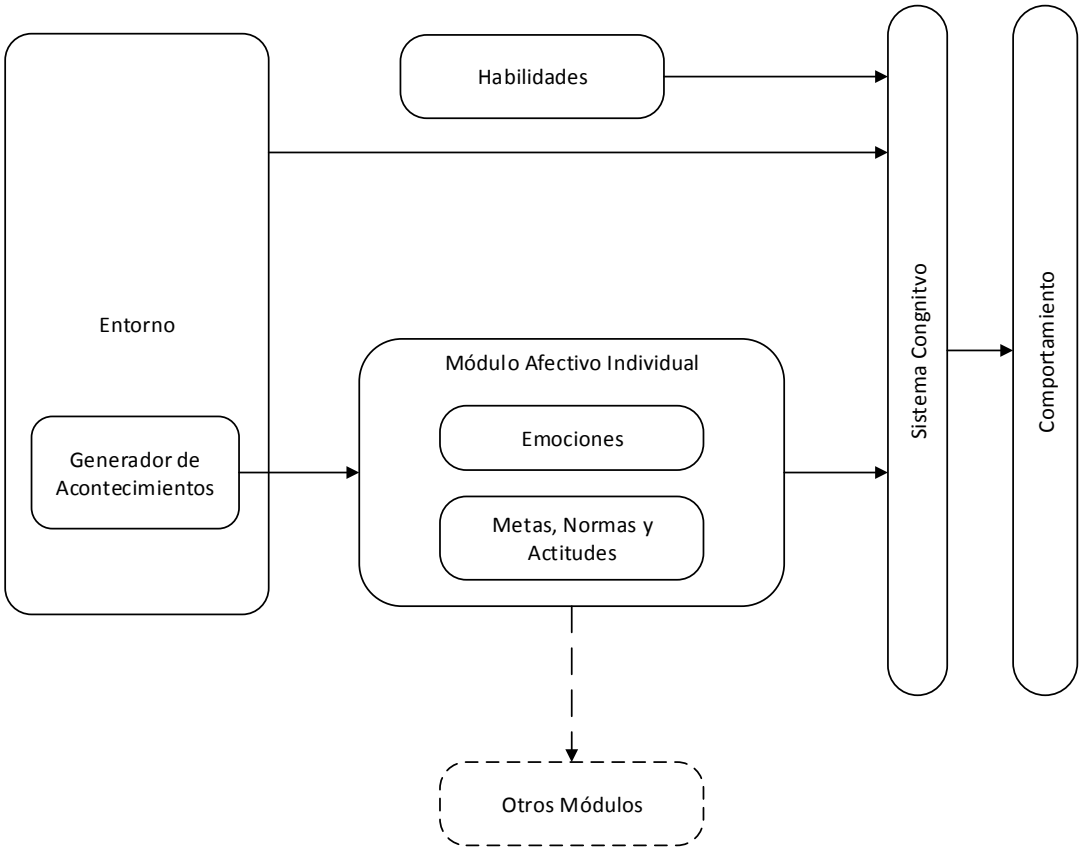

Figura III.3: Arquitectura Emotiva (Acevedo, 2009)

III.4.2 Activadores Emocionales

Los activadores emocionales son las cosas u observaciones que causan una emoción, algunos activadores básicos de emoción son (Sylvester, 2013):

- A través de aprendizaje
- A través de hechos sobre los personajes
- A través del reto
- A través de la interacción social
- A través de la adquisición
- A través de la música
- A través del espectáculo
- Por medio de la belleza
- A través del ambiente
- A través de tecnología novedosa
- A través de amenazas

Estos activadores, pueden ser analizados y conocer las circunstancias en la que estos ocurren y sería posible conocer de qué manera afectan al jugador o si se manifiestan dentro del juego desarrollar. Dentro de su libro Sylvester, menciona que estos pueden ser considerados como factores que se desean provocar emociones al jugador, sin embargo esto es en un medio de a prueba y error y no algún tipo de modelado.

Capítulo IV

Módulo para el Manejo de las Emociones

IV. Módulo para el manejo de las emociones

Tomando como base al modelo OCC (Ortony et al., 1990), se formuló una propuesta para indicar cómo el usuario se ve afectado emocionalmente, en base a las acciones que éste realiza en el juego y que tiene consecuencias palpables para él mismo. Se consideran los eventos y consecuencias que tiene toda la secuencia para el usuario.

Esta propuesta va enfocada hacia los videojuegos, debido a su alto grado de interacción y cambios emocionales que pueden provocar, en lugar de otros tipos de aplicaciones que rara vez logran esto.

IV.1 Propuesta del manejo de emociones (Acción – Evento – Consecuencia)

Esta propuesta busca cómo detectar un cambio emocional del jugador en base a las acciones, eventos y consecuencias ocurridos dentro del juego, tomando en cuenta las siguientes consideraciones:

- **Acción** : Realizar un movimiento dentro del juego (comando de entrada)
- **Evento**: Suceso provocado por una acción que afecta al juego.
- **Consecuencia**: Las repercusiones que deja el evento a los intereses o intensiones del jugador.
- **Activador**: Es el conjunto de una acción, evento y consecuencia el cual tiene un valor emocional para el jugador, ya sea positivo o negativo.

Sobre estas definiciones consideramos a dos actores que participan dentro de un videojuego, el humano (jugador) y la máquina (Inteligencia Artificial). Los efectos de esta interacción se proponen como las siguientes consideraciones:

Consideración 1: Cuando el jugador realiza una acción positiva para él, y el evento y respectiva consecuencia también le son positivos, su emoción tiende a una intensidad positiva.

Consideración 2: Cuando el jugador realiza una acción positiva para él, pero su evento y consecuencia le perjudican, su emoción aumenta en intensidad negativa.

Consideración 3: Cuando la máquina realiza una acción positiva para ella, y su evento y consecuencia también lo son para ella positivos, la emoción del jugador aumenta en intensidad negativa.

Consideración 4: Cuando la máquina realiza una acción negativa para ella, y su evento y consecuencia también le perjudican a ella, la emoción del jugador aumenta en intensidad positiva.

Al tener un análisis del estado emocional del jugador, podría ser posible hacer cambios al videojuego para tratar de manipular dicho estado, para llevarlo a uno deseado, ya sea por diseño o algún medio automatizado, al basarnos en los activadores emocionales (Sylvester, 2013) y al ver parte de los elementos que conforman a un videojuego se puede llegar a proponer algunos cambios sobre algunos de estos elementos que se listan a continuación:

- Personajes
- Narrativa
- Aspecto visual del juego
- Cambios a los escenarios
- Cambios a la dificultad
- Cambiar la información mostrada al usuario

Dentro de este módulo de emociones, en base al metamodelo EMOCC (Juárez-Ramírez, Marquez-Olivas, & Medina-Ríos, 2014), se representan las características del juego y cómo influyen sobre el humano, este puede verse en la Figura IV.1, Figura IV.2, Figura IV.3 y Figura IV.4. Los atributos que se encuentran dentro de cuadros turquesa son los considerados por el módulo de manejo de las emociones.

De todos los elementos que conforman a un videojuego, en este trabajo de tesis se pone énfasis solamente en el estado del juego, sobre este se consideran las acciones, eventos y consecuencias, además de considerar al jugador en los aspectos de nombre y personalidad. También se considera un estado emocional, en este caso este está especificado por una polaridad (positiva, negativa) y el “stage” del juego, que representa el nivel donde camina el personaje.

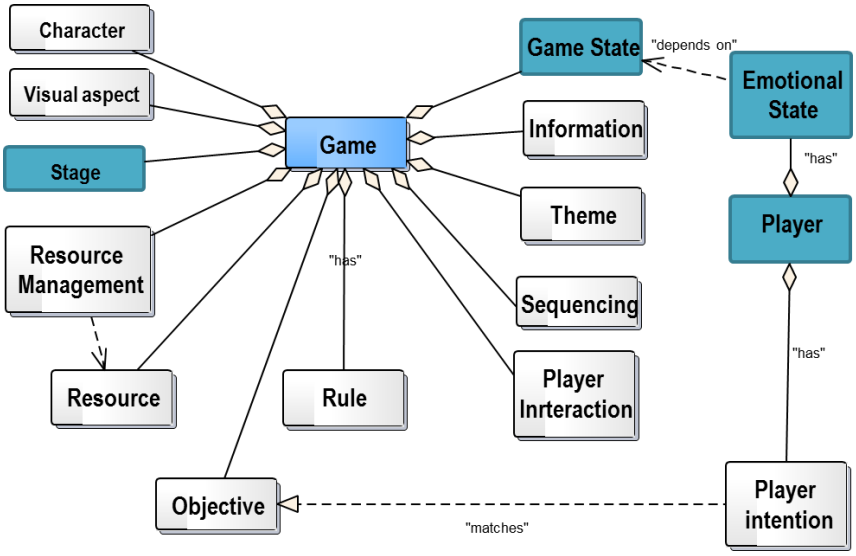

Figura IV.1: Metamodelo de elementos de un videojuego

El metamodelo de las emociones en videojuegos (véase Figura IV.2) describe como una persona se ve afecta sobre las acciones que este realiza sobre un videojuego, ya que este produce una consecuencia visible para el que lo afecta emocionalmente con respecto a la intención que éste tiene.

Dentro del módulo tomamos en cuenta al jugador y la consecuencia que se percibe (en este caso la expresión que refleja a la cámara).

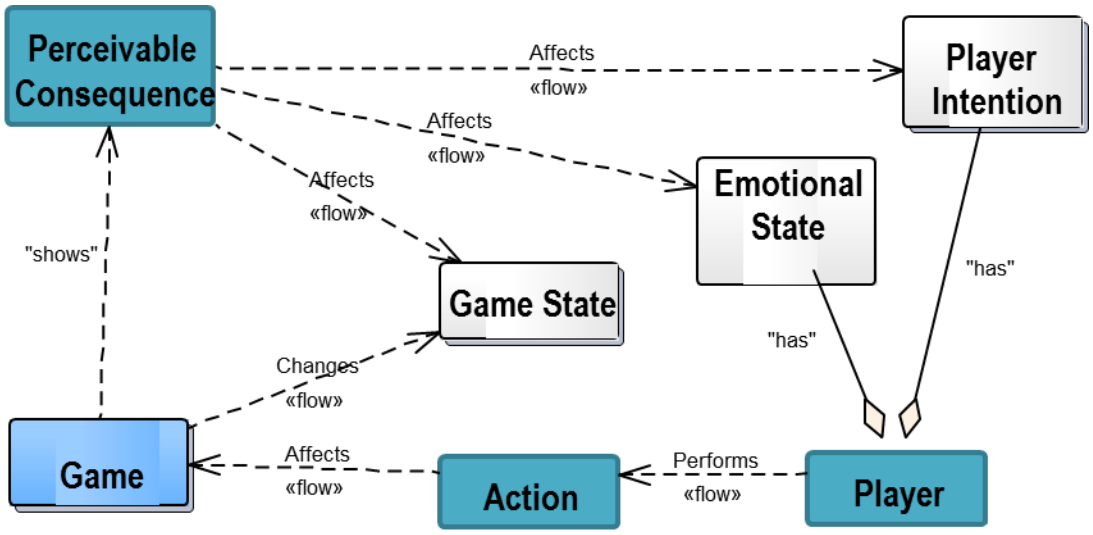

Figura IV.2: Metamodelo de emociones en videojuegos

En la ontología de las emociones (véase Figura IV.3) se describe cómo es que se llega a producir una emoción dentro del ser humano, en este caso se produce por medio de un estímulo que es considerado una acción que conduce a un evento, otro tipo de estímulo, que finalmente desencadena una

consecuencia; dicha consecuencia será evaluada por el humano dependiendo de los factores que este les dé, por ejemplo, importancia, acercarlo a un objetivo o a un deseo personal. En base a esto el humano hace su evaluación y demuestra una respuesta a través de diversos aspectos como pueden ser: expresiones faciales, gestos físicas, tono de voz y señales biológicas. Con esto se determina una valencia (positividad o negatividad) y el arousal (intensidad) con la que se mostrara la emoción, otro de los factores que influye sobre esto es la personalidad del humano y que tanto espera el resultado que ocurrió.

Dentro del módulo para el manejo de las emociones se consideran los conceptos de acción, evento y consecuencia como los sucesos que generan a una emoción, así como considerar si esta tiende a negativa o positiva. Además hacemos consideración de la experiencia que tiene el humano por medio de la expresión facial. Esto con la finalidad de obtener al menos algún tipo de medición sobre el estado del ser humano.

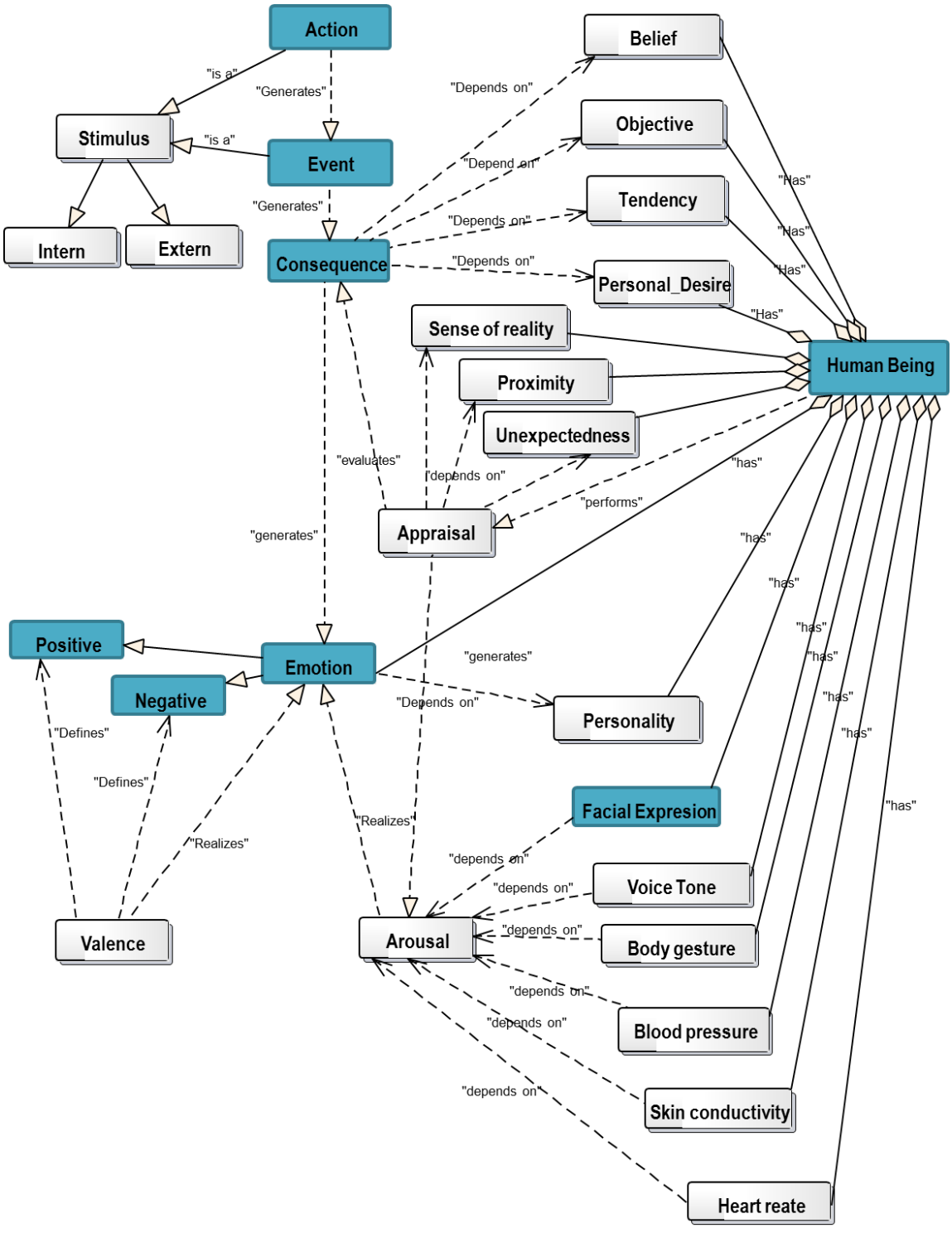

Figura IV.3: Ontología de la emoción

Sobre el estado emocional (véase Figura IV.4), la emoción y el humor son parte de un estado psicológico, que cuenta con una duración, expresividad, intensidad y polaridad. El humor es generado

por un evento no específico, mientras que un evento específico genera una emoción, que puede o no estar influenciada por el humor actual.

Dentro del módulo, consideramos como un evento especifico afecta al humano, en este caso por el sentido de polaridad de una consecuencia, siendo solo positiva o negativa.

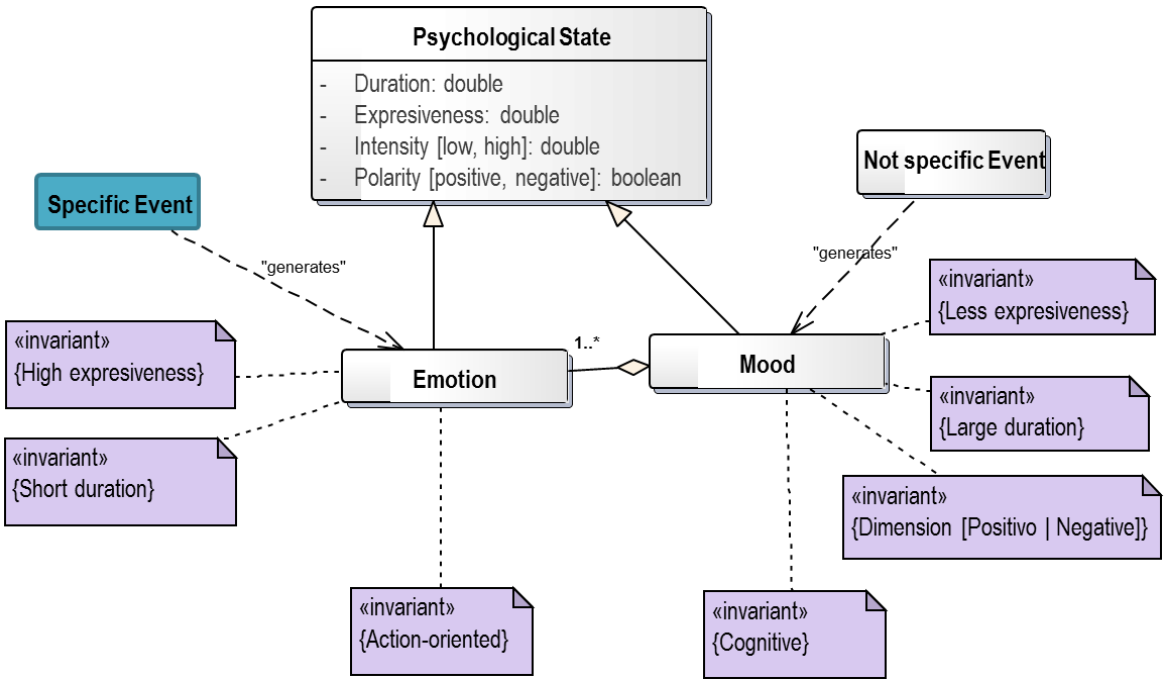

Figura IV.4: Emoción y humor como un esta psicológico (Estado Emocional)

IV.1.1 Diseño de la arquitectura

Dentro de la realización del módulo de emociones se toma en consideración que se tiene pensada una integración a un videojuego con un tipo de arquitectura que no sea la del *game engine*, esto debido a que dentro de los *game engines* se manejan sus propios lenguajes especiales en cuanto a reportar eventos (Sollenberger & Singh, 2012) que pueden o no complicar un en cierto grado, por lo tanto esta arquitectura está pensada para integrarse en videojuegos que sean realizados con arquitectura monolítica o de componentes.

El diseño de este módulo está basado en estilo de arquitectura dirigida por eventos (Bass et al., 2003; Michelson, 2006), la razón de la elección de este estilo es debido a la naturaleza de los videojuegos y a las propuestas que se tienen realizadas, ya que se tiene que hacer un registro de las acciones, eventos y consecuencias (en este caso, todos son manejados como eventos) y al recibir cierta condición (una consecuencia) se procede a hacer un análisis del estado emocional del jugador cuando se suscitó el último evento.

Para llevar a cabo esto, se propone un nuevo módulo que interactúe con el videojuego, encargado de llevar un registro de las acciones, eventos y consecuencias, saber cuándo ocurren y cuáles cambios en los estados emocionales podría llegar a provocar, para esto se propone la arquitectura conformada por 13 clases que se describen a continuación:

- **EmotionModule**

  Clase principal del módulo que aplica el patrón de diseño *singleton*, es la encargada de recibir la información de cuando ocurren los cambios dentro del videojuego. Esta información la envía a la clase GameLogger para su registro; cuenta con una lista de todas las posibles acciones, eventos, consecuencias y cambios emocionales posibles dentro del juego, así como tener una base de datos de los jugadores, en caso que se tenga más información detallada de este.

  - **Atributos (véase Figura IV.5) :**
    - **instance:** Instancia de la clase, para asegurarnos de que solo existan una instancia (patrón singleton).
    - **sessionNumber:** Numero de la sesión de juego actual.
    - **subjectName:** Nombre del jugador actual (se recibe dato del juego).
    - **actions:** Lista de las acciones posibles dentro del juego.
    - **events:** Lista de los eventos posibles dentro del juego.
    - **consequences:** Lista de las consecuencias posibles dentro del juego.
    - **emotionChanges:** Lista de los cambios emocionales que pueden ocurrir debido al juego.
    - **personsDatabase:** Lista que contiene a todos los jugadores de los que se cuente registro.
    - **currentPlayer:** objeto de la clase tipo Jugador que describe al jugador actual
    - **log:** objeto del tipo Log que se utiliza principalmente para registrar las acciones, events y consecuencias recibidas por a través del juego.
    - **frec_pos_pos :** atributo estadístico de cuentas veces se encontró una reacción positiva a una consecuencia positiva para el jugador.
    - **frec_pos_x :** atributo estadístico de cuantas veces se encontró una reacción positiva a una consecuencia negativa para el jugador.
    - **frec_neg_pos:** atributo estadístico de cuantas veces se encontró una reacción negativa a una consecuencia positiva para el jugador.

- **frec_neg_x:** atributo estadístico de cuentas veces se no se encontró una reacción a una consecuencia negativa para el jugador.

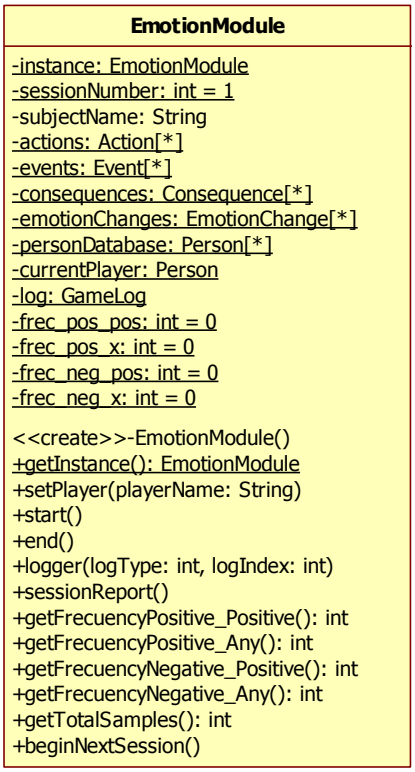

Figura IV.5: Clase EmotionModule

- **LogIndex**

  Tiene la descripción de todas las acciones, eventos y consecuencias dentro del juego, esta clase es utilizada por el módulo y por el juego, para facilitar el registro de los eventos ocurridos con el fin de que estos se registren correctamente.

  - **Atributos (véase Figura IV.6):**
    - Los atributos de esta clase son dependientes del juego que se están implementando, los únicos constantes son los que definen el tipo **:**
      - TYPE_ACTION: indica que se está registrando una acción.
      - TYPE_EVENT: indica que se está registrando un evento.
      - TYPE_CONSEQUENCE: indica que se está registrando una consecuencia.

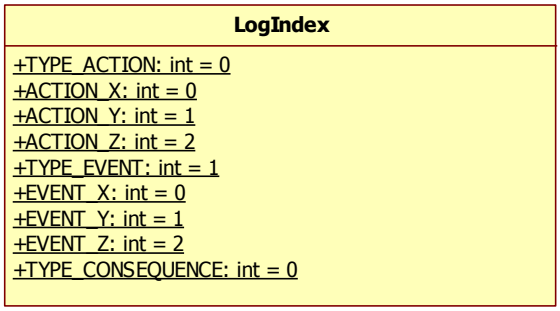

Figura IV.6: Clase LogIndex

- **Action**

  Descripción de una acción que puede ocurrir en el juego por medio de un identificador y un nombre.

  - **Atributos (véase Figura IV.7):**
    - **id:** identificador de la acción.
    - **name:** nombre de la acción.

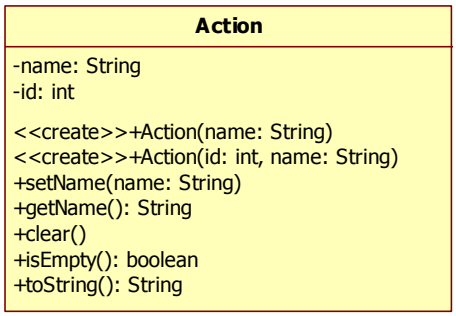

Figura IV.7: Clase Action

- **Event**

  Descripción de un evento que puede ocurrir en el juego, por medio de un identificador y un nombre.

  - **Atributos (véase Figura IV.8):**
    - **id:** identificador del evento.
    - **name:** nombre del evento.

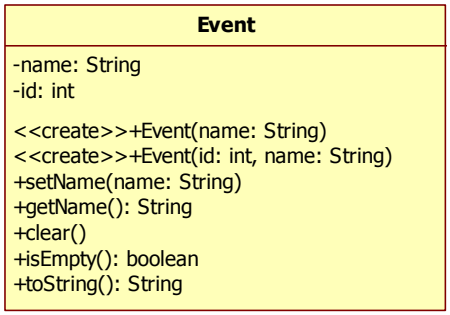

Figura IV.8: Clase Event

- **Consequence**

  Descripción de una consecuencia que puede ocurrir en el juego, por medio de un identificador y un nombre

  - **Atributos (véase Figura IV.9):**
    - **id:** identificador de la consecuencia.
    - **name:** nombre de la consecuencia.

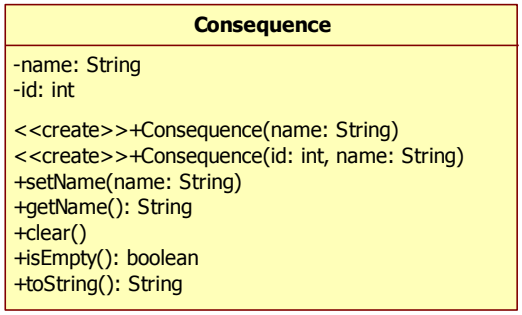

Figura IV.9: Clase Consequence

- **EmotionChange**

  Descripción del efecto que tiene una consecuencia sobre el jugador, para este caso solo se maneja un string que indica si el efecto tendrá características positivas o negativas para él.

  - **Atributos (véase Figura IV.10):**
    - **consequence:** consecuencia asociada al cambio emocional.
    - **polarity:** polaridad a la cual se dirige el cambio de emoción.

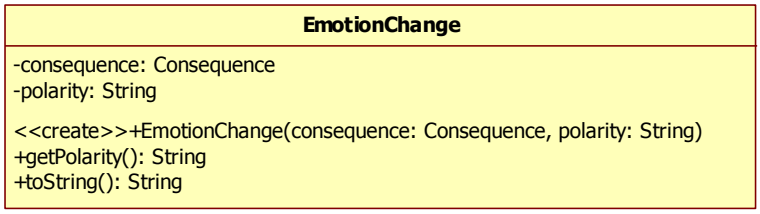

Figura IV.10: Clase EmotionChange

- **Person**

  Describe a una persona en base a un identificador, nombre y personalidad.

  - **Atributos (véase Figura IV.11):**
    - **id:** identificador de la persona.
    - **name:** nombre de la persona.
    - **personality:** características de la personalidad de la persona.

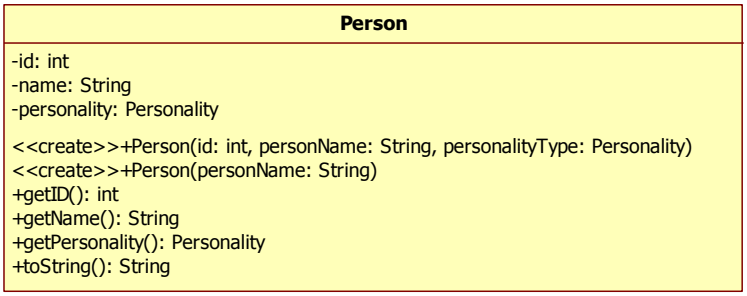

Figura IV.11: Clase Person

- **Personality**

  Descripción de la personalidad de una persona conforme a los cinco atributos del Big Five y la personalidad del jugador.

  - **Atributos (véase Figura IV.12):**
    - **extraversion:** identificador de la consecuencia.
    - **openness:** nombre de la consecuencia.
    - **agreeableness :** valor del rasgo agreeableness respecto al Big 5.
    - **conscientiousness:** valor del rasgo conscientiousness respecto al Big5.
    - **neurotism:** valor del rasgo neurotism respecto al Big 5.
    - **strongerTrait:** nombre del rasgo con el valor más alto.
    - **playerType:** tipo de jugador.

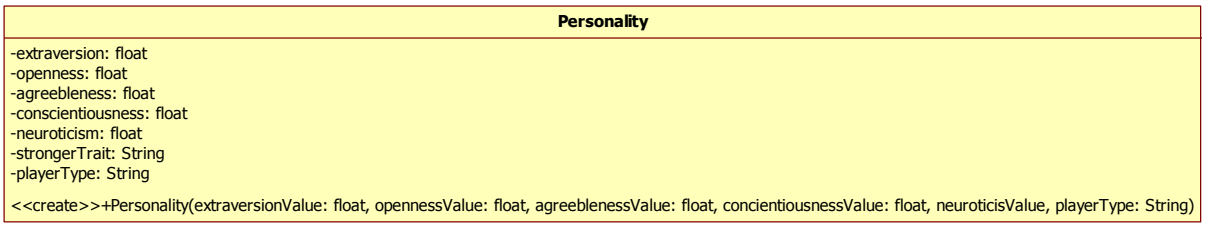

Figura IV.12: Clase Personality

- **XMLReader**

  Se encarga de leer los archivos XML que contienen la descripción de las acciones, eventos, consecuencias, cambios en los estados emocionales y jugadores que son utilizados por este módulo.

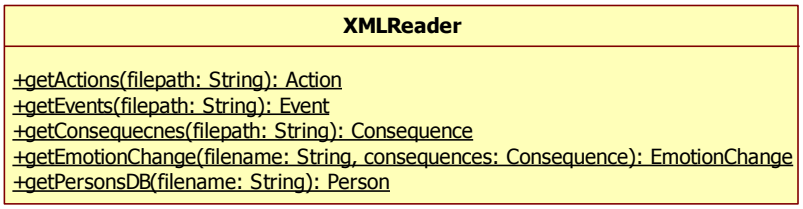

Figura IV.13: Clase XMLReader

- **GameLog**

  Se encarga de registrar las acciones, eventos y consecuencias dentro de un archivo de texto y además activa e inicializa *threads* encargados de realizar el censado del jugador, en este caso: tomar las fotos del jugador.

  - **Atributos (véase Figura IV.14):**
    - **currentAction:** acción actual (última acción registrada).
    - **currentEvent:** evento actual (último evento registrado).
    - **currentConsequence:** consecuencia actual (última consecuencia registrada).
    - **currentPolarity:** polaridad de la consecuencia actual (última consecuencia registrada).
    - **actionTime:** hora cuando ocurrió la última acción.
    - **eventTime:** hora cuando ocurrió el último evento.
    - **consequenceTime:** hora cuando ocurrió la última consecuencia.
    - **idPict:** identificador la imagen actual.

- **sessionNumber:** número de sesión.
- **subjectName:** nombre del jugador actual.

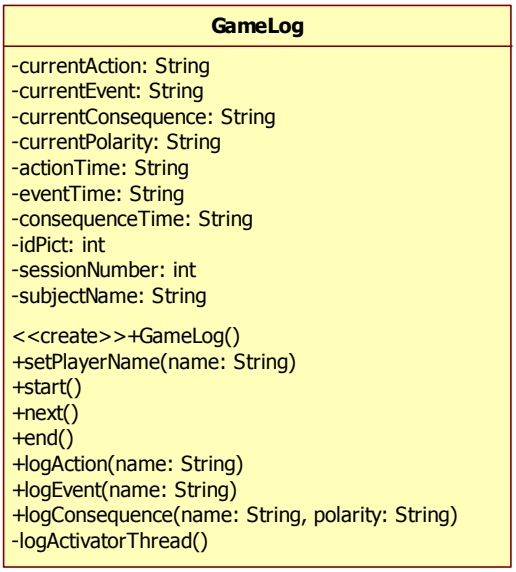

Figura IV.14: Clase GameLog

- **Camera**

  Se encarga de la inicialización y uso de la webcam para posteriormente analizar las fotos tomadas, esta clase hace uso de la biblioteca de visión por computadora *OpenCV*, para hacer uso de la cámara y leer las imágenes captadas.

  - **Atributos (véase Figura IV.15):**
    - **cam:** objeto que representa la cámara de captura (Webcam).
    - **frame:** almacena una foto obtenida del dispositivo de captura.

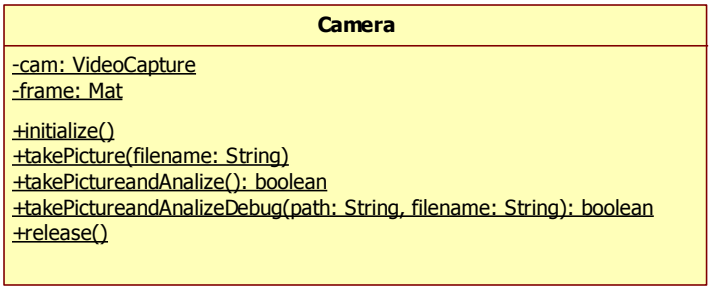

Figura IV.15: Clase Camera

- **PictureAnalyzerT**

  Clase que hereda de Thread, esta es creada cada vez que se detecta una secuencia, su naturaleza es de esta manera para no interrumpir el ritmo de juego del jugador y no esperar que se termine el análisis de imagen para continuar. Se encarga de enviar a tomar una foto y registrar los resultados obtenidos en archivo.

  - **Atributos (véase Figura IV.16):**
    - **actionName:** indica el nombre de la acción a registrar.
    - **eventName:** indica el nombre del evento a registrar.
    - **consequenceName:** indica el nombre de la consecuencia a registrar.
    - **polarity:** indica la polaridad de la consecuencia hacia el jugador.
    - **path :** ruta donde se almacenara la imagen.
    - **actionTimeStamp:** tiempo en el que se registró la acción.
    - **eventTimeStamp:** tiempo en el que se registró el evento.
    - **consecuenceTimeStamp:** tiempo en el que se registró la consecuencia.
    - **idPicture :** identificador de la foto.

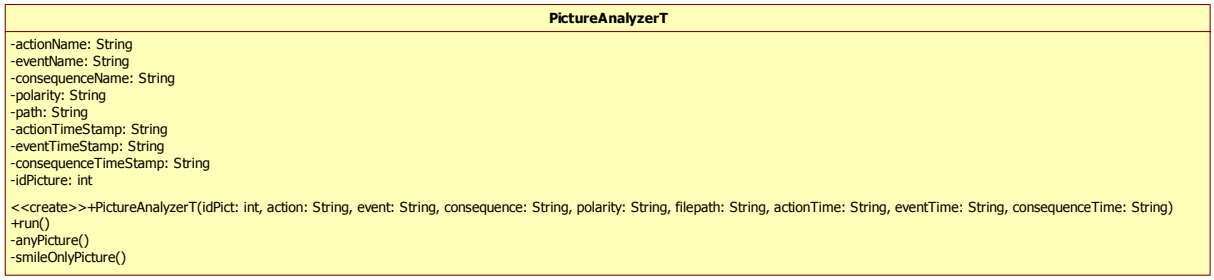

Figura IV.16: Clase PictureAnalyzerT

- **PictureAnalizer**

  Dentro de esta clase, se hace uso de la biblioteca OpenCV, haciendo uso de Cascade Classifiers para en análisis de las imágenes, en este caso se utilizan dos, un detector de rostro (archivo contenido en la descarga de la biblioteca) y uno para detectar sonrisas. En este caso se utilizó un clasificador de código abierto creado por Hromada (Hromada, Tijus, Poitrenaud, & Nadel, 2010) esta clase almacena el resultado del análisis y además genera archivos de las imágenes tomadas y marca los objetos encontrados (en caso de existir).

- **Atributos (Figura IV.17):**
  - **face_detector:** *cascade clasifier* que se utiliza para la detección de cara.
  - **currentEvent:** *cascade clasifier* que se utiliza para la detección de sonrisa.

PictureAnalyzer
-face_detector: CascadeClassifier -smile_detector: CascadeClassifier +smileDetector(frame: Mat): boolean +smileDetectorFileSaverDebug(frame: Mat, path: String, filename: String): boolean

Figura IV.17Clase PictureAnalyzer

La Figura IV.18 muestra la arquitectura general, indicando las relaciones entre estas clases descritas que conforman el módulo del manejo de emociones.

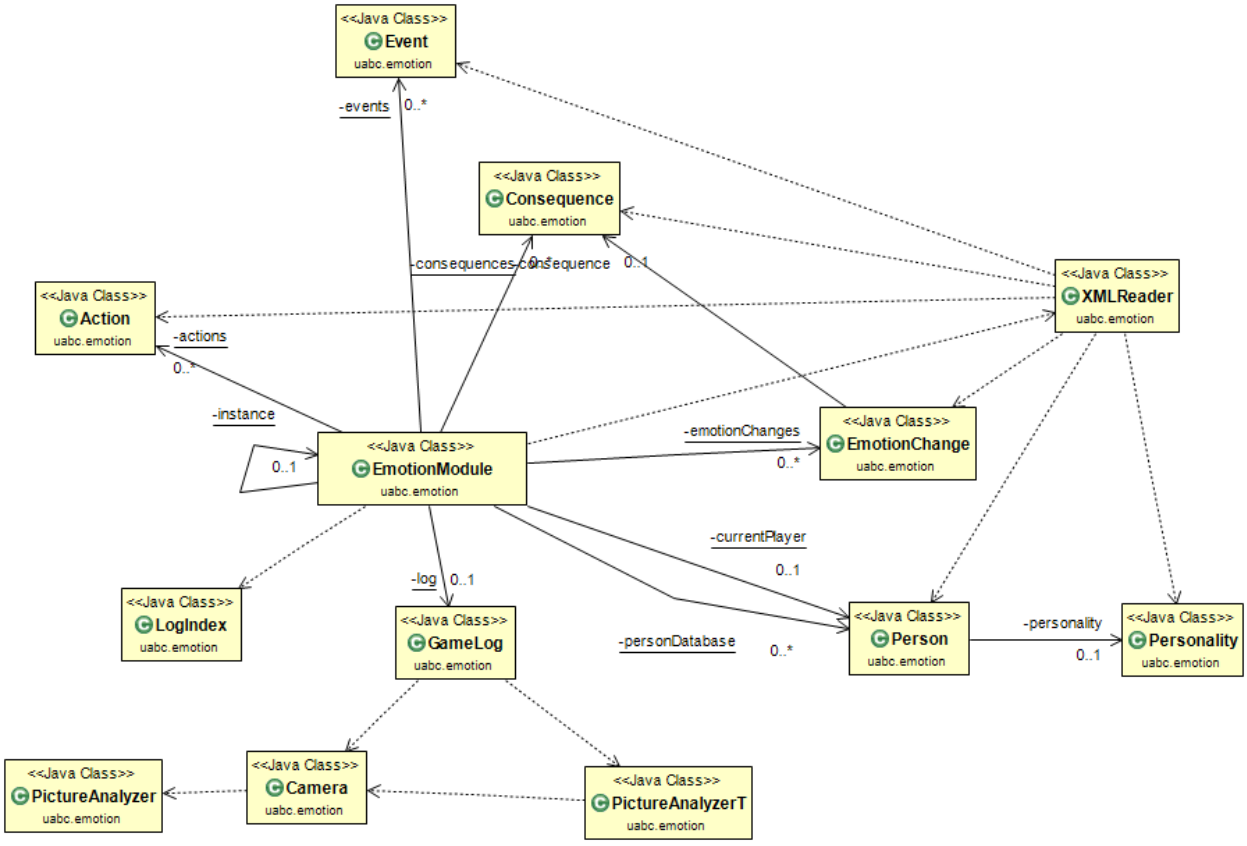

Figura IV.18: Diagrama de relación entre clases módulo manejo de emociones

Se requiere de tres archivos XML, los cuales describen lo siguiente:

- Acciones, eventos y consecuencias posibles dentro del juego

- Relación entre consecuencia y cambio emocional que se espera (positivo, negativo)
- Descripción de los jugadores: nombre, personalidad Big Five y tipo de personalidad de jugador.

IV.1.2 Funcionamiento

Este módulo recibirá por parte del juego, cuándo y cuál de cada acción evento y consecuencia se deba registrar. Al registrarse una consecuencia, se activan los siguientes pasos:

1. Tomar foto del jugador.
2. Analizar foto e indicar si sonríe o no.
3. Almacenar el resultado (acción, evento, consecuencia, cambio, sonrisa) en una cadena de texto.

Al finalizar una sesión, se crean archivos con la descripción de todas las consecuencias suscitadas, en un formato especifico (véase Tabla IV.1).

Tabla IV.1: Formato de registro de consecuencia

# Foto	Acción	Tiempo-Acción	Evento	Tiempo-Evento	Consecuencia	Tiempo-Consecuencia	Efecto Deseado	Efecto-Sonrisa	Sonrisa

Al final del archivo se añade a manera de reporte la siguiente información:

- Total de Sonrisas.
- Total de Consecuencias.
- Total de Sonrisas sobre un evento que se espera un resultado positivo.
- Total de Sonrisas sobre un evento que se espera un resultado negativo.
- Total de eventos que se esperan resultados negativos, que no obtienen una reacción.
- Total de eventos que se espera resultado negativo y no se obtuvo una reacción.
- Proporción de sonrisas detectadas entre total de consecuencias.

Al analizar la información obtenida, nos es posible sacar estadísticas de cuando el jugador estaba sonriendo y que combinación de sucesos fue lo que lo llevo a esta expresión, además es posible hacer una relación entre la personalidad y la reacción. Esta estadística puede ser utiliza por los desarrolladores de videojuegos para realizar ajustes para que el juego tenga el efecto emocional deseado o para visualizar posibles ajustes dinámicos, según sea su intención.

El módulo hace uso de una base de datos (actualmente en un archivo XML) que contiene el nombre, valores de atributos de Big 5 (OCEAN) y personalidad del jugador, esto con el fin de que el algún futuro esto sea considerado para alguna modificación generalizada o incluso llegar a realizar una a nivel personal.

Para este caso la cámara es utilizada como un medio para realizar el censado de una manera no invasiva, con el fin de darle comodidad al jugador. El único requisito es que su cabeza completa este dentro del rango de visión de la cámara.

El modulo podría ser expandido de diferentes maneras, haciendo uso de otros sensores para aumentar la efectividad del censado emocional, incluso el modulo actual podría mejorarse si se obtienen clasificadores para ciertas expresiones faciales, que indiquen ciertas emociones.

Una de las intenciones de este módulo es el que no se requiera tanto acoplamiento con el juego, para esto se tiene que llevar a cabo un análisis del código ya existente y con esto localizar en donde se suscitan o detectan las acciones, eventos y consecuencias; el caso ideal es que se encuentren dentro de una sola clase, pero esto depende ya de como el juego fue diseñado y programado.

IV.1.3 Integración dentro de arquitectura del juego

Para la integración del módulo a un videojuego, es necesario realizar una seria de pasos:

- Identificar las acciones, eventos y consecuencias que pueden ocurrir dentro del juego.
- Ubicar en donde se activan cada uno de estos.
- Crear archivos XML, donde se describan las acciones, eventos y consecuencias y añadirlos al LogIndex.
- Llamar a los métodos requeridos, en el momento que ocurre una acción, evento o consecuencia y cuando finaliza una sesión.

Dependiendo de esto y los resultados arrojados por el modulo, se determina que acciones tomar con respecto del jugador, estos aspectos pueden variar y pueden ser:

- Realizar cambios en la dificultad.
- Cambiar el aspecto de los escenarios.
- Cambiar la información que se despliega.
- Cambiar al personaje.

El caso ideal es que las clases del juego solo se comuniquen con la clase ***EmotionModule,*** además del ***LogIndex*** dentro del módulo de emociones, el primero para hacer uso para registrar los acontecimientos y el último para tener acceso al índice de estos.

El diagrama que se muestra en la Figura IV.19 contiene el flujo de actividades para el manejo de emociones. Funcionando de la siguiente manera, explicando el flujo básico y sus flujos alternos:

Flujo Básico:

1. Jugador realiza una acción dentro del juego por medio de un comando.
    a. El juego registra la acción dentro del módulo.
2. El juego consigue el evento que ocurre debido a la acción del jugador
    a. El juego registra el evento dentro del módulo.
3. El juego revisa la consecuencia ocurrida y actúa conforme a esta.
    a. El juego registra la consecuencia dentro del módulo
    b. El modulo toma una foto del jugador en ese momento.
    c. El módulo se encarga de analizar la foto
    d. El módulo almacena los resultados dentro de un archivo de texto correspondiente a la sesión actual.
4. El juego revisa si el personaje sigue vivo, el personaje sigue vivo.
5. El juego vuelve al paso 1.

Flujo Alterno 1:

1. Jugador realiza una acción dentro del juego por medio de un comando.
    a. El juego registra la acción dentro del módulo.
2. El juego consigue el evento que ocurre debido a la acción del jugador
    a. El juego registra el evento dentro del módulo.
3. El juego revisa la consecuencia ocurrida y actúa conforme a esta.
    a. El juego registra la consecuencia dentro del módulo
    b. El modulo toma una foto del jugador en ese momento.
    c. El módulo se encarga de analizar la foto
    d. El módulo almacena los resultados dentro de un archivo de texto correspondiente a la sesión actual.
4. El juego revisa si el personaje sigue vivo, si no lo está, procede a solicitar un análisis de resultados.

    a. El módulo revisa el archivo de texto generado para conseguir el conteo de lo detectado y lo envía al juego.
5. El juego evalúa el reporte y dependiendo de los resultados puede: aumentar la dificultad, disminuir dificultad o dejarla como esta.
6. El juego vuelve al paso 1.

Flujo Alterno 2:

1. Jugador realiza una acción dentro del juego por medio de un comando.
    a. El juego registra la acción dentro del módulo.
2. El juego consigue el evento que ocurre debido a la acción del jugador
    a. El juego registra el evento dentro del módulo.
3. El juego revisa la consecuencia ocurrida y actúa conforme a esta.
    a. El juego registra la consecuencia dentro del módulo
    b. El modulo toma una foto del jugador en ese momento.
    c. El módulo se encarga de analizar la foto
    d. El módulo almacena los resultados dentro de un archivo de texto correspondiente a la sesión actual.
4. El juego revisa si el personaje sigue vivo, este aun lo está.
5. EL juego revisa si el personaje termino el nivel.
    a. El módulo revisa el archivo de texto generado para conseguir el conteo de lo detectado y lo envía al juego.
6. El juego evalúa el reporte y dependiendo de los resultados puede: aumentar la dificultad, disminuir dificultad o dejarla como esta.
7. El juego vuelve al paso 1.

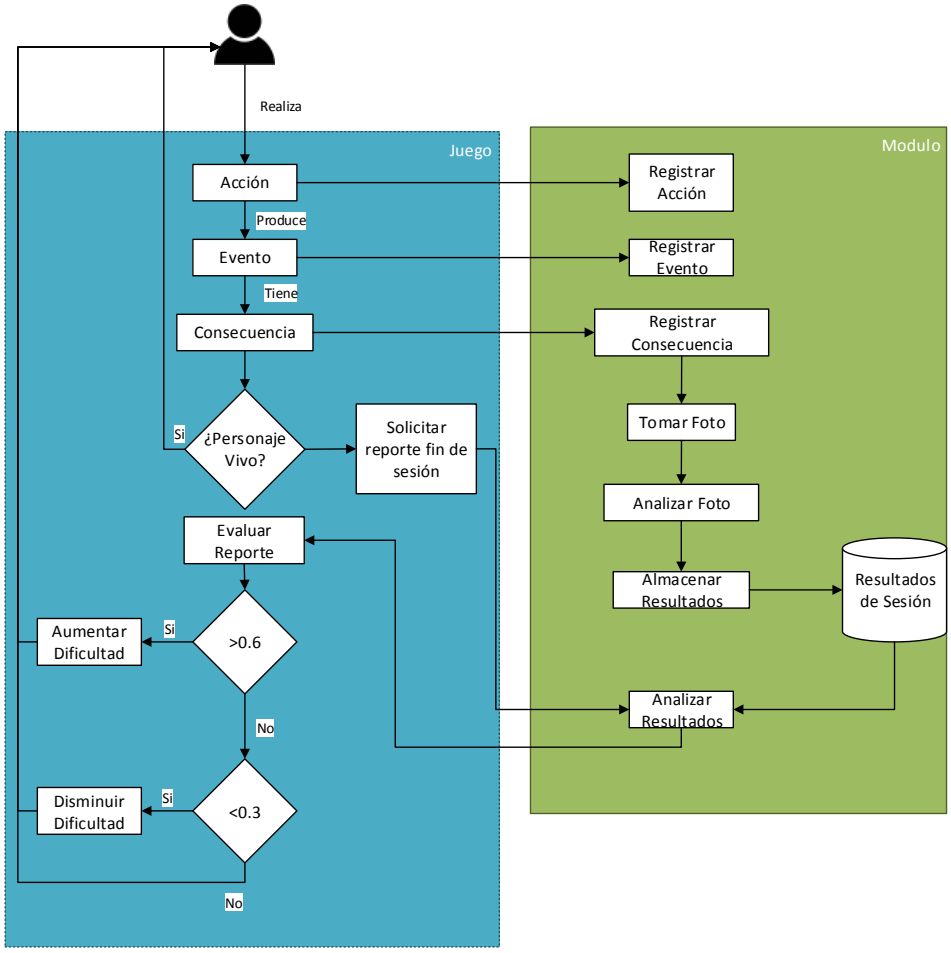

Figura IV.19: Funcionamiento del módulo para el manejo de las emociones

Capítulo V

Caso de Estudio: Videojuego “Kin de Mayapan”

V. Caso de Estudio: Videojuego “Kin de Mayapan”

V.1 Caso de estudio con videojuego tipo plataforma

V.1.1 Kin de Mayapan

El principal caso de estudio consistió en la elaboración de un videojuego de género plataforma con temática de la cultura maya, con gráficos en dos dimensiones, esto último con el fin de simplificar la parte del diseño.

Una de las razones por la cual se eligió desarrollar un videojuego de genero plataforma es porque no son tan complejos a la hora de jugar y son bastante conocidos, por lo que nos ahorraría tiempo en que la persona pase por una fase de aprendizaje larga y así ya pueda estarse expresando más naturalmente y no estando confundido por la manejara de jugar.

El juego se caracteriza por requerir que el juego cuente con reacciones rápidas para evitar los obstáculos o para eliminar a los enemigos que se encuentran en el camino, mientras se recolectan objetos, para ayudar en el camino.

En la elaboración de este videojuego se contó con la participación de estudiantes de licenciatura de la carrera de ingeniera en computación dentro de UABC Tijuana (programación, música y efectos) y alumnos de la carrera de diseño gráfico de UABC Valle de las Palmas (gráficos).

El videojuego fue desarrollado haciendo uso del framework Libgdx (“Libgdx,” 2014), lo que nos dio acceso a realizar un desarrollo rápido al no preocuparnos por interactuar directamente con el hardware. Este framework nos permite el manejo de: gráficos, audio, dispositivos de entrada y cargar mapas de Tiled. Algunas imágenes del juego en ejecución pueden ser observadas en la Figura V.2 y Figura V.3

El objetivo principal del juego es llegar a la meta de los dos niveles existentes, eliminando o pasando por alto a los enemigos encontrados, mientras se esquivan trampas mortales, con la posibilidad de recolector mazorcas de maíz.

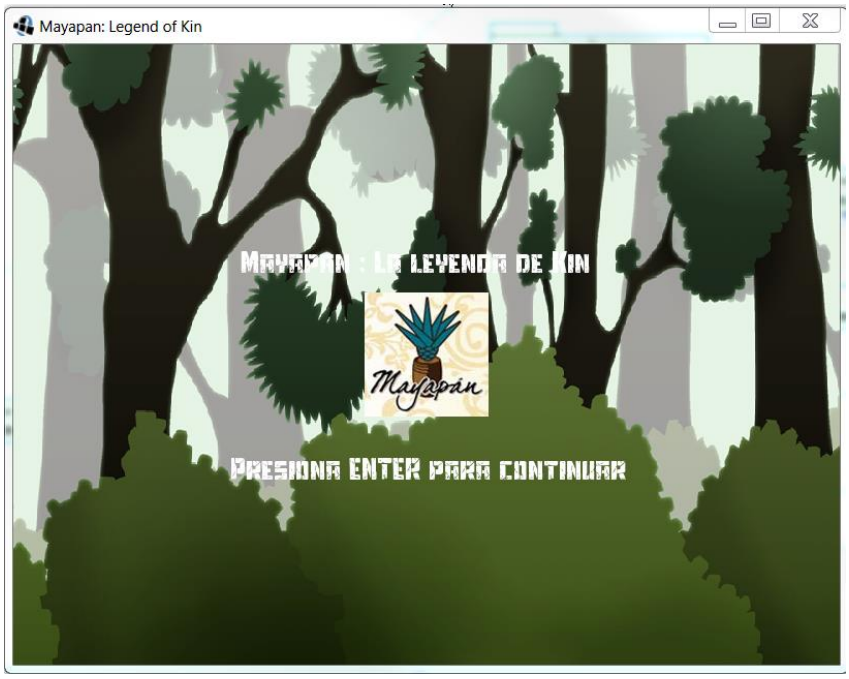

Figura V.1: Pantalla principal kin

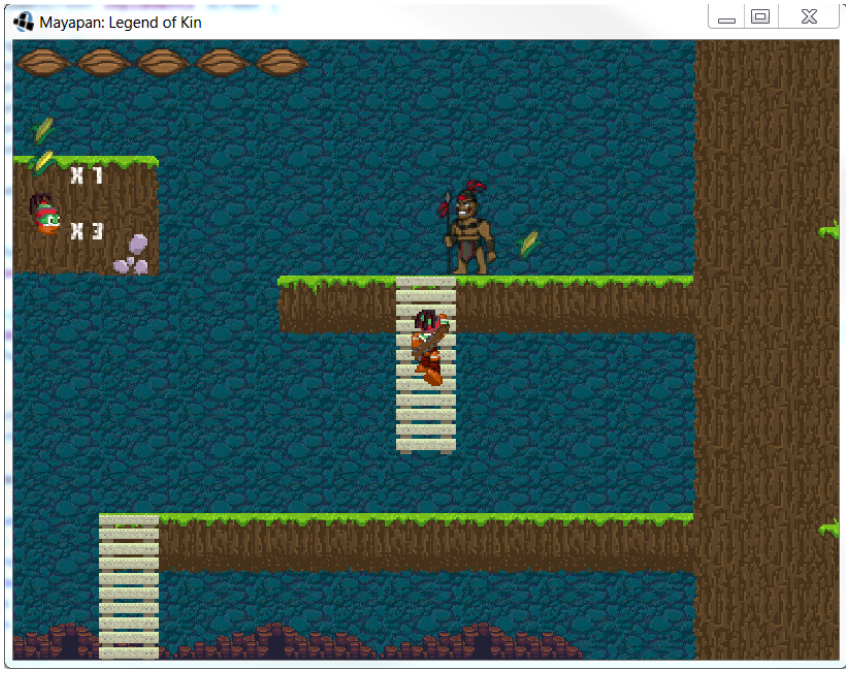

Figura V.2: Kin dentro del juego

V.1.1.1 Arquitectura

Sobre el diseño de la arquitectura se tomó la decisión de realizarlo basado en la propuesta hecha por Doherty (Doherty, 2003), ya que es una de las más detalladas que se encontraron, además de no ser tan complejo como hacer un *game engine*. Debido a esto, la arquitectura está conformada por cuatro

módulos principales, que se encuentran subdivididos en tareas específicas, la arquitectura puede verse en la Figura V.3 y su descripción es la siguiente:

- **Data Manager**: Encargado de cargar los archivos de audio e imágenes requeridas por los elementos del juego (entidades y pantallas).
- **Game Engine:** Encargado de mostrar el estado del juego al jugador y pasar los comandos introducidos por este a las demás partes del juego.
    - **Graphics :** Manejo de las pantallas que son utilizadas a lo largo del juego y utilizadas para mostrar el estado del juego
    - **Input:** Manejo de las entradas realizadas por el usuario.
    - **Tween:** submodulo auxiliar que hace uso de una biblioteca para animaciones gráficas, para ser utilizada para agregar algunos efectos de transición sobre las pantallas al iniciar el juego.
- **Object System**: Mantienen todas las entidades que son utilizadas por el juego además se encarga de tener el estado más actualizado sobre el juego.
    - **Entities**: La representación de todos los elementos que componen el juego, en este caso las entidades son los personajes, enemigos y objetos que interactúan entre si dentro del juego.
    - **World**: Se encarga de mantener el estado del juego de un nivel, llevando el registro del estado de todas las entidades que se encuentran dentro del mundo del juego.
- **Simulation:** Se encarga de llevar a cabo todas las manipulaciones sobre los objetos del juego conforme al paso del tiempo o a la entrada realizada por el jugador.
    - **AI**: Módulo básico de comportamiento de los enemigos conforme a cierta situación, simula introducir comandos para que los objetos no controlados por el jugador se muevan.
    - **GameLogic**: Encarga de revisar colisiones entre los objetos, para activar los comportamientos de estos, se encarga de manipular las entidades y llevar el control de los cambios que se afecten.
    - **StateMachine:** Modulo auxiliar encargado de crear máquinas de estado, en este caso utilizado para el comportamiento de las entidades e indicar en qué estado (del juego) se encuentran.

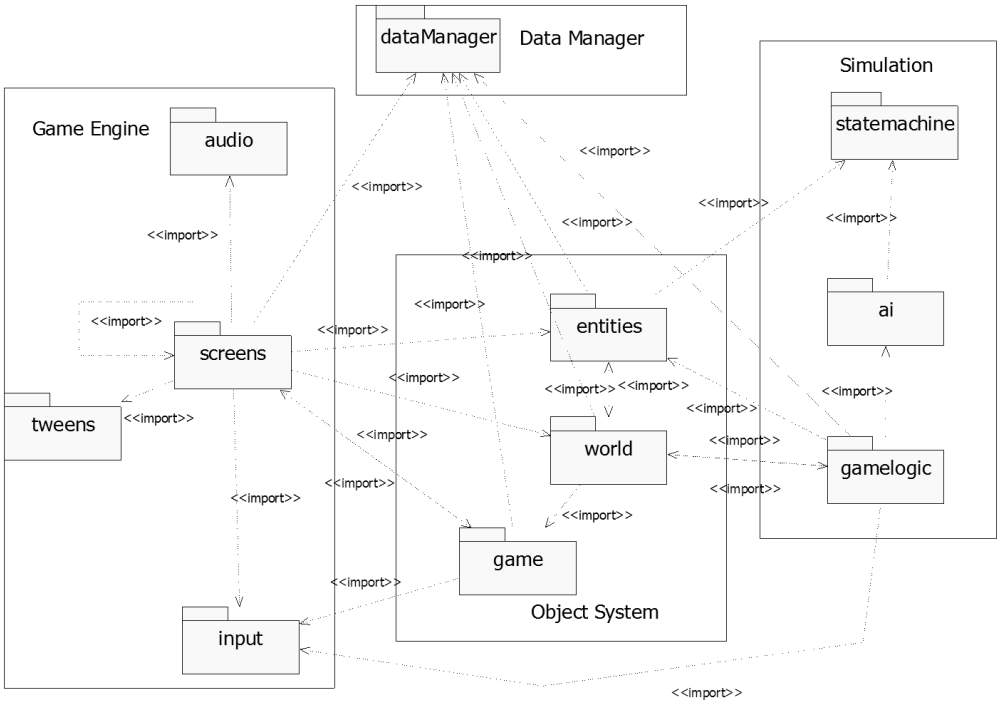

Figura V.3: Arquitectura Kin de Mayapan

V.1.1.2 Implementación de Módulo de Emociones en un videojuego

Como fue explicado en el capítulo anterior, para la implementación del módulo de emociones dentro de un videojuego, es necesario primero llevar a cabo un análisis de las acciones, eventos y consecuencias que ocurren dentro de este, siempre y cuando el juego no cuente con un sistema parecido, como un manejador de eventos, en este caso se tomarían en cuenta los eventos estipulados y detectados por los desarrolladores.

En este caso, el juego fue elaborado sin un motor de eventos, pero estos se localizan fácilmente en un par de clases; además se llevó a cabo un análisis de las posibles acciones, eventos y consecuencias que pueden suceder dentro del juego, estos resultados son visibles en la Tabla V.1

Tabla V.1: Acciones, eventos y consecuencias en Kin de mayapan

Acciones	Eventos	Consecuencias
Moverse	Recibir Golpe Corto Alcance	Morir
Saltar	Recibir Golpe Largo Alance	Recibir daño a vida (corto alcance)
Saltar y Moverse	Golpear Enemigo Corto Alcance	Recibir daño a vida (largo alcance)
Doble Salto	Golpear Enemigo Largo Alcance	Recibir daño a escudo (corto alcance)
Doble Salto y Moverse	Tocar Enemigo	Recibir daño a escudo (largo alcance)
Caer	Tocar Item	Enemigo dañado (corto alance)
Caer y Moverse	Tocar Maíz	Enemigo dañado (largo alcance)
Subir Escalera	Alcanzar Checkpoint	Enemigo Matado
Subir Escalera y Moverse	Caer Fuera del Mapa	Obtener Poder
Ataque Corto Alcance	Llegar al final de Nivel	Incrementar Maíz
Ataque Corto Alcance en el aire		Subir Vida
Ataque Largo Alance		Subir Escudo
Ataque Largo Alcance en el aire		Activar Checkpoint
		Terminar Nivel
		Fin del Juego (Perder)
		Terminar Juego (Ganar)

En este caso las acciones se encuentran dentro de la clase del personaje, ya que este reacciona a las entradas, mientras que los eventos y consecuencias ocurren dentro de la simulación, debido que estas son resultado de la interacción entre las entidades o el mapa.

Sobre la clase principal del juego, se inicializará el módulo y solo dentro de las demás clases que manejen la detección de acciones, eventos y consecuencias se hará uso del módulo de emociones.

Para el caso de alguna ajuste al juego debido a los resultados obtenidos por el modulo emocional, también se requerirán añadir esas clases, un caso deseado seria que alguna de las clases ya utilizadas tuviera esta capacidad, en cuanto a los ajustes para este caso de estudio serán cambios al nivel de dificultad, al realizar este cambio, se alteran aspectos del nivel para permitir ya sea más fácil acceso a algunas plataformas o tratar de dejar un camino complicado que requiere de movimientos precisos por parte del jugador y aumentar la cantidad de objetos curativos y disminuir enemigos, la Figura V.4 y la Figura V.5 muestran los ejemplos del diseño del primer nivel conforme a múltiples dificultades.

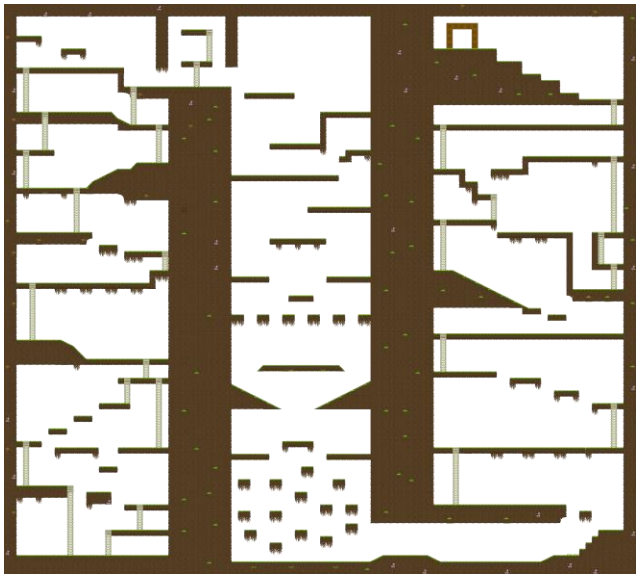

Figura V.4: Primer nivel dificultad fácil

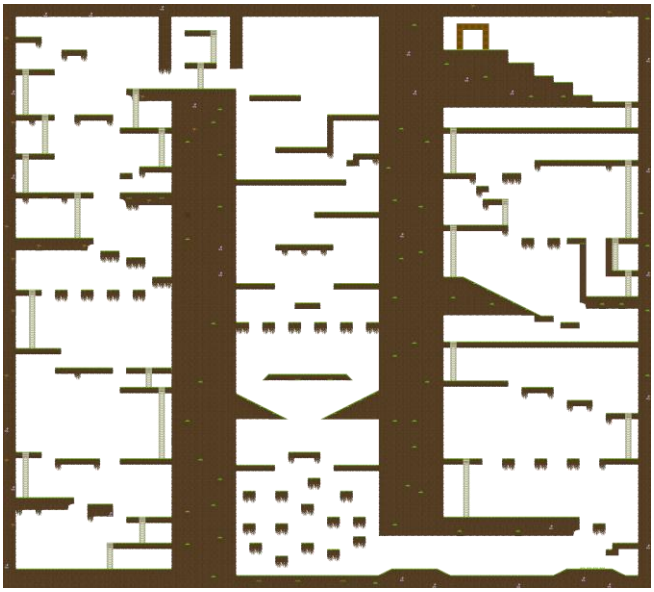

Figura V.5: Segundo nivel dificultad normal

El resultado de la integración puede verse en la Figura V.6, las clases inferiores son las que forman al módulo.

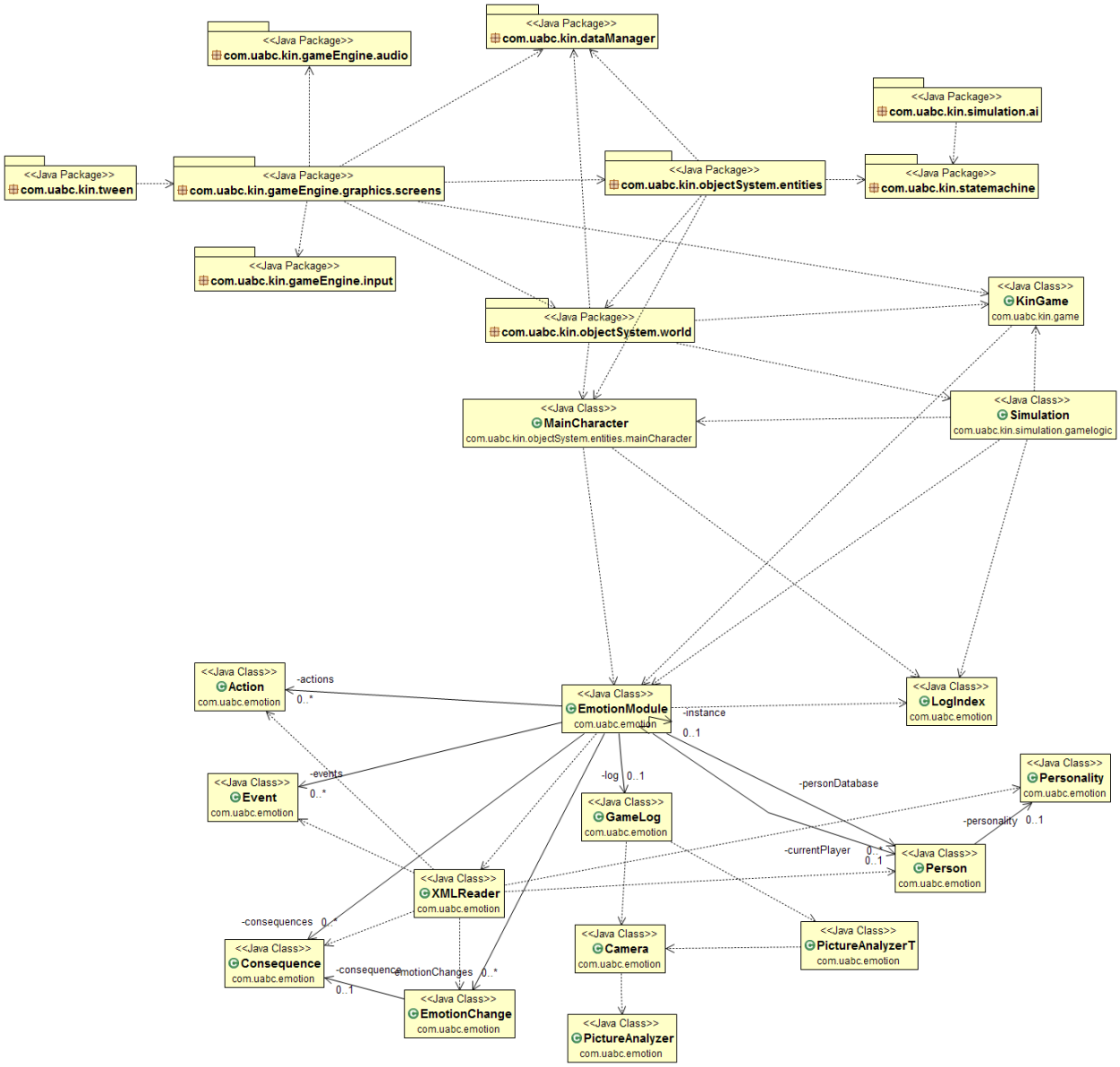

Figura V.6: Arquitectura de Kin de Mayapan con Modulo Emocional

El impacto sobre el desempeño del juego sin un módulo de emociones es notable, debido a que se están ejecutando varios threads en paralelo, encargados de realizar la toma y análisis de foto, debido a esto y para darle tiempo al módulo de terminar de procesar todos estas entradas cada vez que se registra una consecuencia de fatalidad para el personaje o un cambio de nivel. Como puede verse en la Figura V.7, se hace una breve espera de 2 segundos, para asegurarnos que todo termine, este tiempo puede ser reducido dependiendo de las capacidades del equipo de cómputo que se esté utilizando, debido que para asegurarnos de la precisión de las fotos, estas son almacenadas por el modulo, esto

también es una causa de impacto en el rendimiento, para esto el módulo cuenta con un modo de solo analizar, pero no almacenar la imagen resultante que reduce el coste de procesamiento.

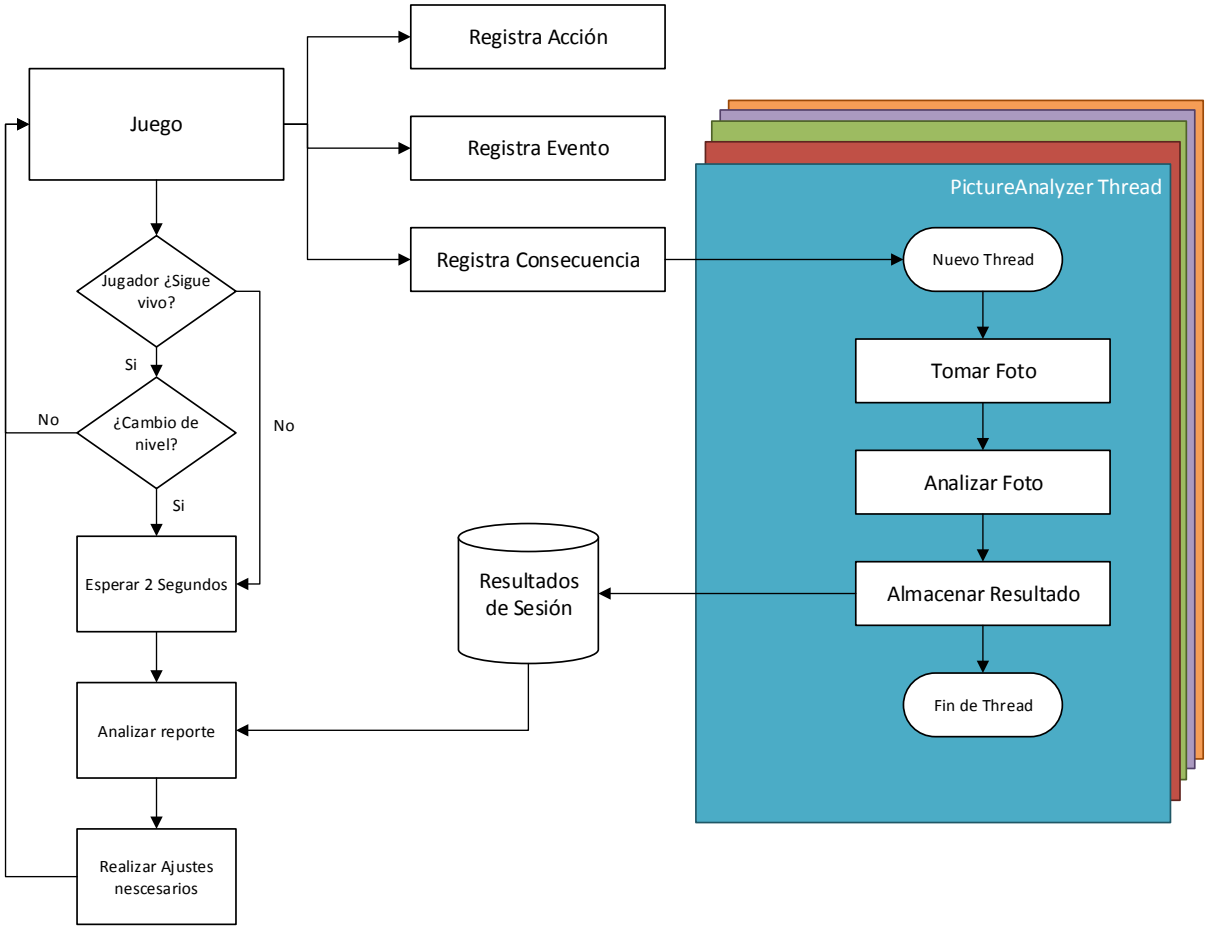

Figura V.7: Diagrama de flujo, toma de foto

V.1.1.3 Análisis de Acoplamiento

Del módulo de emociones solo dos clases son utilizadas por el videojuego al que se desea implementar : **EmotionModule** y **LogIndex**, en el caso de Kin de Mayapan, **EmotionModule** es utilizado 3 veces, una por la aplicación principal, otra por el personaje principal (registro de acciones) y el último por la simulación (registro de eventos y consecuencias), mientras que **LogIndex** solo es utilizado por las clases que hacen algún tipo de registro al Log, entonces tenemos solo dos clases, el personaje principal y la simulación, el acoplamiento y la cohesión previamente existente dentro de la arquitectura del juego no se vio afectado.

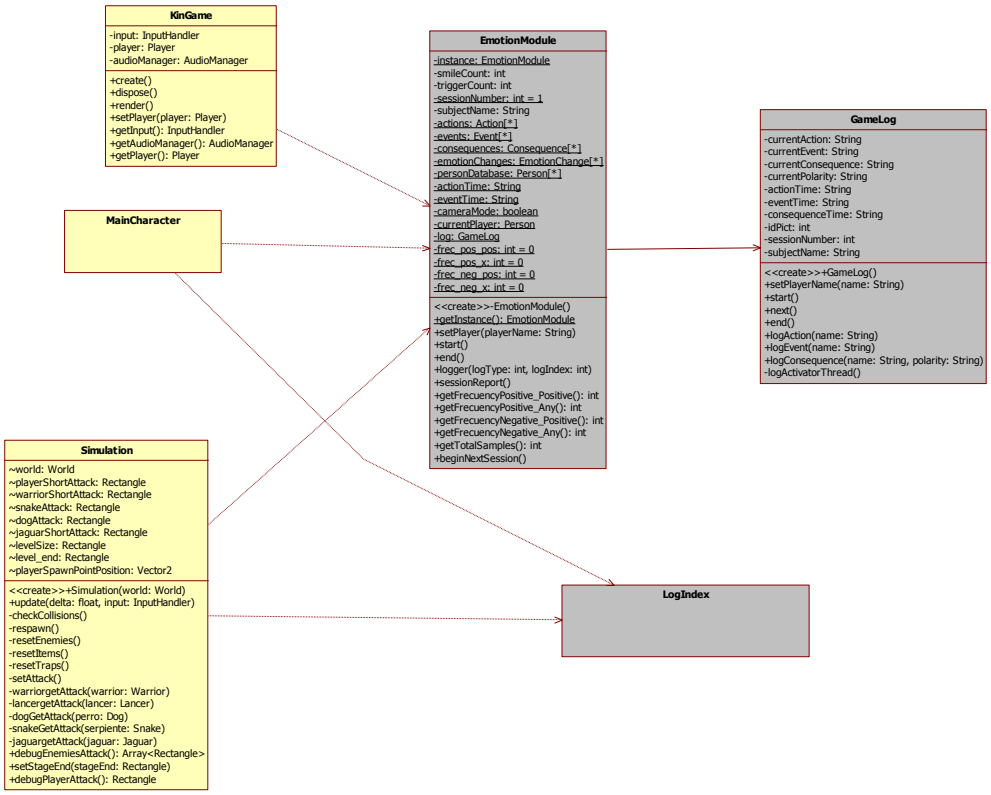

Figura V.8: Relación entre juego y módulo

Aplicando las formulas e acoplamiento (véase Capitulo II, Sección 5.2) sobre el antes y después de la arquitectura se obtienen los siguientes resultados, esto basándonos en las clases que ahora interactúan con el módulo:

Arquitectura Normal

- GameLogic (Simulation).
  - C_a = 1
  - C_e= 18
  - Inestabilidad = 0.94
- MainCharacter (ObjectSystem).
  - C_a = 2
  - C_e= 9
  - Inestabilidad = 0.81

- KinGame (Game).
    - C_a = 10
    - C_e= 5
    - Inestabilidad = 0.333

Arquitectura con Emociones.

- GameLogic (Simulation).
    - C_a = 1
    - C_e= 20 (2)
    - Inestabilidad = 0.95
- MainCharacter (ObjectSystem).
    - C_a = 2
    - C_e= 11(2)
    - Inestabilidad = 0.84
- KinGame (Game).
    - C_a = 10
    - C_e = 6(1)
    - Inestabilidad = 0.375

Estos resultados arrojan un ligero incremente en cuanto a acoplamiento eferente se refiere (la cantidad de bibliotecas importadas y usadas), esto era de esperarse debido a que es requerido que exista una conexión entre el juego y el modulo, al mantener las conexiones mínimas nos aseguramos de que solo algunas clases requieran cambios en ser necesarios en lugar de tenerlo distribuido sobre una infinidad de clases.

Un bajo valor en inestabilidad indica que una clase estable, lo que significaría que es una clase que no se cambiaría, cuando una clase tiene un valor alto en inestabilidad, quiere decir que la clase es inestable lo que asume que esta clase puede requerir cambios para ajustarse a las necesidades y que no es estática, en este caso como estamos hablando de un videojuego, que su estado es cambiante es razonable que sea una clase inestable, debido a la dependencia que se tiene de varios elementos que forman parte del juego, por lo que un aumento en ello era considerado pero no fue tan drástico.

V.1.1.4 Análisis de Desempeño

Para tomar en cuenta el desempeño que tiene la aplicación, se hizo uso de la herramienta VisualVM lo que nos permite ver el estado de la memoria y CPU de determinadas aplicaciones

Para este experimento se hizo uso de dos versiones del videojuego, una sin el módulo de manejo de emociones y otra con él, se ejecutaron por un lapso de tiempo similar y posteriormente se procedió a revisar los reportes generados por la herramienta. Los reportes se pueden ver en la Figura V.9 y Figura V.10 la cantidad de tiempo que requiere cada thread utilizado, debido a que el módulo emocional crea un thread por cada foto, estos se ven demasiados, pero simplemente duran alrededor de 100ms cada uno, por lo que no es un retraso considerable, más si se maneja en threads ajenos al principal.

Call Tree - Method	Time [%]	Time	Time (CPU)	Invocations
LWJGL Application		150484 ms (100%)	150484 ms	1
LWJGL Timer		150484 ms (100%)	0.000 ms	1

Figura V.9: Uso de CPU arquitectura normal

Call Tree - Method	Time [%]	Time	Time (CPU)	Invocations
Finalizer		94.8 ms (100%)	94.8 ms	1
LWJGL Application		166093 ms (100%)	148116 ms	1
LWJGL Timer		166093 ms (100%)	0.000 ms	1
Thread-10		97.2 ms (100%)	97.2 ms	1
Thread-100		98.9 ms (100%)	98.9 ms	1
Thread-101		99.7 ms (100%)	99.7 ms	1
Thread-102		92.3 ms (100%)	92.3 ms	1
Thread-12		100 ms (100%)	100 ms	1
Thread-14		99.7 ms (100%)	99.7 ms	1
Thread-15		99.7 ms (100%)	99.7 ms	1
Thread-16		98.3 ms (100%)	98.3 ms	1
Thread-17		91.8 ms (100%)	91.8 ms	1
Thread-18		91.8 ms (100%)	91.8 ms	1
Thread-19		99.5 ms (100%)	99.5 ms	1
Thread-20		81.9 ms (100%)	81.9 ms	1
Thread-21		100 ms (100%)	100 ms	1
Thread-22		100 ms (100%)	100 ms	1
Thread-23		99.7 ms (100%)	99.7 ms	1
Thread-24		100 ms (100%)	100 ms	1
Thread-25		97.5 ms (100%)	97.5 ms	1
Thread-26		99.8 ms (100%)	99.8 ms	1
Thread-27		95.0 ms (100%)	95.0 ms	1
Thread-29		100 ms (100%)	100 ms	1
Thread-30		96.5 ms (100%)	96.5 ms	1
Thread-31		98.6 ms (100%)	98.6 ms	1
Thread-32		94.9 ms (100%)	94.9 ms	1

Figura V.10: Uso de CPU en arquitectura con módulo integrado

V.1.1.5 Procedimiento de la Experimentación

Para este experimento se contó con la participación de 28 usuarios, estudiantes de la carrera ingeniería de computación, en el caso de algunos de ellos se les solicitó realizar encuestas para conocer su tipo de personalidad y tipo de jugador. De los 28 sujetos de prueba, solo se tomaran en consideración 23, debido a que durante las pruebas cinco quedaron fuera de foto por lo que no funcionó correctamente la detección facial.

Las encuestas realizadas sobre personalidad fueron las siguientes:

- Encuesta de Big-Five que consistía en 50 preguntas
    - http://personality-testing.info/tests/BIG5.php
- Prueba de Tipología de Jung, consistiendo en 72 preguntas
    - http://www.humanmetrics.com/cgi-win/jtypes2.asp

La primera nos sirvió para obtener los valores de los rasgos de personalidad y el último para determinar qué tipo de jugador se es en base a DGD (Hobo, 2004). Estas pruebas fueron realizadas a algunos durante clase, otros fueron contestados en sus casas enviando los resultados obtenidos a un correo electrónico para su recopilación.

Esta información será utilizada al final para determinar si hay alguna relación entre el tipo de personalidad y la cantidad de expresiones mostradas.

A los sujetos se les dio una breve explicación del juego y se le explicaron los controles, posteriormente se les dejo jugar le juego a su gusto por un tiempo de 10 minutos, como único requisito era el estar con la Webcam activada y estar mirando hacia el juego y reaccionar libremente sin sentirse observados, además para aumentar la efectividad del análisis de rostros, se les solicito a los participantes no usar lentes, en caso de poder ver sin ellos.

Los experimentos fueron realizados en una laptop con las siguientes características

- Sistema Operativo: Windows 7 Home Premium 64-Bits
- CPU: Intel Core i7 3610QM @ 2.30Ghz
- RAM: 16 GB
- Cámara Web
- Versión de Java JDK: 1.7

A los sujetos se les dejo jugando el videojuego por una duración de 10 min cada uno, si llegaban a perder el juego, se les solicitaba que continuaran.

Los niveles presentados son los mismos para cada sujeto, esto con el fin de realizar una comparación en circunstancias similares, debido a que cada persona es diferente, no todos tendrán la misma experiencia, por lo que también se consideran los tiempos que tardan en llegar a un evento y la frecuencia de estos.

De los 28 sujetos de prueba, solo se tomarán en consideración 23, debido a que durante las pruebas cinco quedaron fuera de foto por lo que no funcionó correctamente la detección facial.

Dentro de este juego, una sesión es considerada desde que el personaje aparece hasta que pierde una vida o pierde el juego.

V.1.1.6 Resultados

En la Tabla V.2 se muestra un resumen de los sujetos que participaron en el experimento y que se contaba con sus respectivas personalidades, el rasgo de mayor valor está indicado en color amarillo.

Tabla V.2: Relación Sujetos/Personalidad

Sujeto	Big-Five (OCEAN)					Myers-Briggs	Tipo de Jugador	Personaje
	Extraversion	Openness	Agreeableness	Conscientiousness	Neuroticism			
S1	47	75	72	52	65	INFJ	Participant	Warrior
S2	52	50	48	57	42	ENTJ	Conqueror	Warrior
S3	89	87	77	82	37	ENFJ	Participant	Warrior
S4	20	70	65	65	27	INFJ	Participant	Warrior
S5	57.5	82.5	62.5	60	60	ENFJ	Participant	Warrior
S6	45	70	42.5	35	52.5	INTP	Manager	Warrior
S7	62.5	65	55	52.5	55	ENFJ	Participant	Priest
S8	67.5	75	60	32.5	65	ENFP	Wanderer	Priest
S9	12	45	32.5	42	52	ISTP	Manager	Warrior
S10	57.5	67.5	80	52.5	32.5	ENFJ	Participant	Priest
S11	72.5	72.5	75	57.5	20	ENFJ	Participant	Priest
S13	67.5	52.5	65	80	25	ENFJ	Participant	Warrior
S16	77.5	85	72.5	62.5	42.5	ENFJ	Participant	Warrior
S17	47.5	60	75	65	37.5	INFJ	Participant	Priest
S19	60	60	55	42.5	77.5	ENFP	Wanderer	Civilian
S25	45	40	65	27.5	67.5	ISFP	Wanderer	Priest
S26	90	92.5	37.5	50	65	ENTJ	Conqueror	Warrior
S28	65	85	75	60	62	ENFJ	Participant	Civilian

Dentro del tiempo que jugaron los sujetos, algunos tenían más experiencia con videojuegos que otros. Cada sesión es considerada una vida del juego, por lo tanto entre más sesiones se tengan registradas, la persona murió en varias ocasiones.

Dentro de este juego, se lleva a cabo un ajuste en la dificultad cada vez que el personaje muere, y se toma en consideración la proporción de las consecuencias contra las sonrisas detectadas, debido a esto si un jugador sonreía demasiado, al morir se encontraría con un reto más elevado, en caso contrario si casi no sonreía la dificultad bajaba tratando de darle una oportunidad de llegar más fácil a la meta, tratando de ofrecerle gusto con esto, las condiciones utilizadas para este caso fueron las siguientes :

- Si la relación esta entre un rango de 0.3 y 0.6, la dificultad se mantendrá.
- Si la relación es mayor a 0.6, la dificultad aumentara para tratar de poner serio al jugador.
- Si la relación es inferior a 0.3, la dificultad disminuye, para tratar de animar al jugador.

Algunos jugadores comentaron que si se dieron cuenta de los cambios al nivel, pero no le tomaban tanta importancia en cuanto a enojarse por el cambio, la mayoría contaba con comentarios positivos y no les parecía aburrido el juego, con excepción de un sujeto de prueba que cada instante preguntaba cuanto faltaba para terminar de jugar.

Otros comentarios circulaban a que el juego estaba muy difícil (aunque se encontraran en la dificultad más sencilla), también dieron aviso de algunos errores encontrados, que serán considerados para versiones posteriores.

La Tabla V.3 muestra la sumatoria de todas las sesiones y cuantas consecuencias llego a experimentar cada jugador.

Tabla V.3: Resumen de pruebas

Sujeto	# Sesión	Consecuencias	Sonrisas	Sonrisa / Positiva	Sonrisa / Negativa	Positiva / Sin reacción	Negativa / Sin reacción	Proporción	Sonrisas x Sesión
1	7	369	0	0	0	318	51	0.0%	0
2	16	432	0	0	0	322	114	0.0%	0
3	3	292	3	3	0	264	25	1.0%	1
4	10	337	0	0	0	269	68	0.0%	0
5	7	220	33	27	6	148	39	15.0%	4.71
6	4	94	80	60	20	20	4	85.1%	20
7	10	248	77	53	24	126	45	31.0%	7.70
8	7	410	215	186	29	170	25	52.4%	30.71
9	8	331	59	64	9	202	46	17.8%	7.38
10	13	235	53	31	22	126	56	22.6%	4.08

11	6	286	11	9	2	238	37	3.8%	1.83
12	8	331	17	14	3	257	57	5.1%	2.13
13	10	283	103	81	22	138	42	36.4%	10.30
14	10	270	0	0	0	203	67	36.4%	0
16	7	266	4	3	1	214	48	1.5%	0.57
17	9	205	80	58	22	98	27	39.0%	8.89
19	7	313	45	41	4	237	31	14.4%	6.43
22	4	229	3	2	1	196	30	1.3%	0.75
23	10	269	70	54	16	157	42	26.0%	7
25	7	246	169	140	29	62	15	68.7%	24.14
26	5	346	1	1	0	311	34	0.3%	0.20
27	5	286	24	21	3	227	29	8.4%	4.80
28	10	207	24	18	6	130	53	11.6%	2.40

El la Figura V.11 se puede observar un ejemplo de una sonrisa detectada, mientras que en la Figura V.12 se ve cuando una cara es detectada pero no se encuentra una sonrisa.

Figura V.11: Sonrisa detectada

Figura V.12: Solo cara detectada

Con los resultados obtenidos, se encontró el caso de que solo algunas personas tenían algún tipo de sonrisa detectada, mientras que algunos sujetos de plano no sonreían en ningún momento en el que se estaba capturando alguna imagen, esto quizá a que no es una persona muy expresiva o se sentía observado debido a la experimentación.

Haciendo un análisis empírico (debido a que se conoce a algunos de los sujetos), se llega a asumir o concluir que las personas que tienden a ser muy expresivas en cuanto a su manera de pensar, también lo son a la hora de reaccionar al jugar un videojuego, además que algunos de ellos son fanáticos de los videojuegos.

Sobre la cantidad de sonrisas detectadas, se procedió a hacer una análisis sobre cuántas de estas cumplen con la propuesta de que una consecuencia positiva para el usuario lo lleva a mostrar una reacción u emoción positiva, además de considerar la personalidad del jugador, solo fueron consideradas 18 personas debido a que son las únicas que habían respondido los cuestionarios, bajo este análisis se obtiene un promedio del 78% de las consecuencias con alguna reacción detectada obtuvieron una respuesta positiva bajo un evento positivo, como puede verse en la Tabla V.4, sobre la consideración de las consecuencias a evaluar, solo se consideraron en las que se notaba claramente una sonrisa.

Tabla V.4: Relación consecuencia con reacciones positivas

Sujeto	Correctas	Evento Positivo	%	Personalidad / Atributo Mayor	Tipo Jugador
S1	0	0	0%	Openness	Participant
S2	0	0	0%	Conscientiousness	Conqueror
S3	3	3	100%	Extraversion	Participant
S4	0	0	0%	Openness	Participant
S5	17	14	82%	Openness	Participant
S6	30	23	77%	Openness	Manager
S7	46	29	63%	Openness	Participant
S8	73	61	84%	Openness	Wanderer
S9	8	6	75%	Neuroticism	Manager
S10	0	0	0%	Agreeableness	Participant
S11	9	7	78%	Agreeableness	Participant
S13	62	48	77%	Conscientiousness	Participant
S16	4	3	75%	Openness	Participant
S17	40	29	73%	Agreeableness	Participant
S19	38	34	89%	Neuroticism	Wanderer
S25	61	47	77%	Neuroticism	Wanderer
S26	1	1	100%	Openness	Conqueror
S28	20	14	70%	Openness	Participant

En promedio, según la Tabla V.5, se puede observar que las personas con un rasgo mayor de *Neurotism*, suelen ser las más expresivas cuando se encuentran bajo una consecuencia posible para él, se podría decir que también los de *Extraversion,* pero como de estos solo se tenía una muestra, no se puede llegar a algo concluyente.

Tabla V.5: Promedio de sonrisas por personalidad Big Five

Personalidad	Cantidad	Promedio
Openness	9	61%
Conscientiousness	2	39%
Extraversion	1	100%
Agreeableness	3	50%
Neuroticism	3	81%

En cuanto al tipo de personalidad en la Tabla V.6, se observa que las personas con una personalidad tipo *Wanderer* son las que fueron más expresivas en cuanto a emociones positivas en consecuencias positivas para sí mismo, mientras que los menos expresivos eran los *Conquerors*.

Tabla V.6: Promedio de sonrisas por tipo de jugador

Tipo Jugador	Cantidad	Promedio
Participant	11	56%
Conqueror	2	50%
Manager	2	76%
Wanderer	3	83%

V.1.1.7 Discusión de Resultados

Dentro de los resultados obtenidos podemos darnos cuenta que el desempeño del juego (véase Capítulo V, Sección 1.1.3 y 1.1.4) a como es normal, tiende a ser un poco inferior a cuando no se tiene el modulo integrado, esto era de esperarse debido al procesamiento necesario para la obtención de información de las fotos, en este caso el detección de sonrisas, pero el impacto no es notable a la hora de jugar.

Respecto a las métricas de acoplamiento, los valores aumentan debido a la integración del módulo con algunas clases del juego, esto tiene un impacto mínimo sobre ellas, debido a que solo se agrega a las clases indispensables que lo requieran, por lo que en caso de requerir mantenimiento o agregar alguna nueva acción, evento o consecuencia, solo hay que enfocarse en las clases que actualmente lo implementan.

La experiencia de usuario no fue igual para todos debido a que los jugadores no contaban con tanta experiencia o contacto con el juego utilizado ya que era la primera vez que lo miraban, pero se adaptaron sencillamente a los controles, por la familiaridad del modo de juego con otros del mismo género, los resultados obtenidos por medio de las expresiones faciales detectadas (véase Tabla V.4), nos permite plantear o analizar una preferencia a expresarse si se cuenta con cierto tipo de personalidad (véase Tabla V.5) o se es un tipo de jugador (véase Tabla V.6). Durante el juego de manera automática se sacaba un promedio de cuantas sonrisas se detectaron, comparado con las consecuencias ocurridas, en base a este análisis se procedía a cambiar el nivel de dificultad. Algunas personas si se dieron cuenta de los cambios, en cambio otros simplemente estaban concentradas en superar el reto que no se percataban de ello.

Con los resultados nos damos cuenta que los jugadores batallaron algo en poder pasar los niveles, pero el reto dado no los frustró demasiado como para querer dejar de jugar, durante los 10 minutos de juego.

Las personalidades influyen en cierto grado a como se suelen expresar las personas, durante la experimentación se encontraron personas poco expresivas que no sonreían ante los estímulos proporcionados por el juego, usualmente estos contaban con un valor bajo en *Extraversion* en lo que conforma al Big Five.

Sobre el comportamiento de la arquitectura, una vez que fue integrado, todo siguió funcionando de la manera correcta, siempre y cuando las llamadas a los métodos del módulo estuvieran en secuencia correcta. La arquitectura siguió respondiendo con el mismo desempeño al permitir la misma jugabildiad presentada antes, esto gracias a que el módulo de manejo de emociones funciona a través de threads, quizá el único cambio era un pequeño tiempo de "carga" agregado al final de cada muerte del personaje, para dejar finalizar los threads para tener un reporte más completo.

Sobre el módulo, es posible integrarlo a cualquier juego pero se requiere de algunas adaptaciones, debido a que se tiene que agregar las acciones, eventos y consecuencias al LogIndex, esto con el fin de que dentro de la arquitectura del juego estos se puedan registrar con mayor facilidad y evitar errores de ingresar valores inválidos, además si se desea agregar ajustes al juego en tiempo de ejecución, será necesario por parte del desarrollador definir reglas para dichos cambios dentro de la arquitectura del juego, tomando como base los resultados de los reportes arrojados por el módulo.

Un aspecto importante es que al hacer uso del módulo estamos considerando nuevos aspectos que usualmente no son considerados en el mundo de los videojuegos, la emoción del jugador. Si al fin en cuentas lo que se busca es entretener a la persona, ¿cómo se sabe que se está logrando el objetivo? Esto agrega un nuevo medio de interacción que el usuario puede experimentar; sentir una clase de retroalimentación por parte del juego en base en sus expresiones podría resultar en una experiencia más placentera debido a que el reto es constante en base a sus habilidades (Chen, 2007) y no a como este estipulado directamente, lo cual aumentaría el potencial de que un juego sea más longevo y el usuario no pierda el interés tan rápidamente.

Capítulo VI

Conclusiones

VI. Conclusiones

El objetivo principal de la tesis fue la creación de un módulo para el manejo de las emociones e integrarlo a la arquitectura de un videojuego ya existente, en este caso el módulo fue desarrollado siguiendo el modelo OCC (Ortony et al., 1990) y basado en el metamodelo EMOCC (Juárez-Ramírez et al., 2014) con la finalidad de poder tener una caracterización de la emoción sobre el juego, y que éste pueda tomar alguna decisión en base a la información proveída.

El uso de las emociones para mejorar el potencial de los juego es un campo, que está aumentando considerablemente (Ambinder, 2011; Dekker & Champion, 2007; Nacke et al., 2011; Sollenberger & Singh, 2012) aunque con fines distintos, algunos buscan entretener más a la persona mientras que otros tienen objetivos con enfoques educativos con el fin de mejorar el aprendizaje. Estos principios mencionados durante este trabajo son una base para la detección de emociones que puede ser usado en cierta cantidad de aplicaciones.

Fue posible integrar un módulo para el manejo de las emociones sobre un videojuego creado desde cero que originalmente no fue diseñado para soportarlo, la integración resultó ser exitosa a manera de acoplamiento esperado era el mínimo posible, pues se considera que pocas clases deben de estar involucradas para controlar mejor el problema.

Dentro de la experimentación se obtuvieron algunos resultados sobre cómo se comportan las personas al estar jugando, desde algunas que sonríen mucho a otros que de plano no sonríen, sería interesante hacer uso de más sensores para obtener la información al respecto. Quizá fue en parte al exceso de dificultad demostrado, que aunque no los aburrió o frustró demasiado, no les permitía avanzar más en el juego para llegarse a tranquilizar o expresarse de otras maneras.

Con el trabajo realizado se demuestra que es posible adaptar un módulo de manejo de emociones a un videojuego, para recolectar información que podría permitir realizar un análisis estadístico de qué clase de consecuencias ocurren más y cuáles son las que son molestas para el jugador y poder buscar la manera de reducirlas o aumentarlas, según sea el caso.

La inclusión de las emociones en los videojuegos nos permitirá explorar nuevas áreas en cuanto a la interacción con las personas y cómo se puede ir adaptando ya sea las interfaces o contenido para mantener la atención del usuario. En algunos casos se ha trabajado con personajes no controlados que expresen emoción en base al desempeño o comportamiento del jugador (Acevedo, 2009; Karouzaki &

Savidis, 2012) pero el hacer uso de lo que siente el usuario en determinados momentos (Ambinder, 2011; Gilleade & Dix, 2004) es uno de los caminos más interesantes ya que en tiempo real se conoce el estado del jugador en ciertos aspectos, aunque en esos casos solo se basan en la lectura de sensores. En este trabajo de tesis se basa que para que haya una reacción de emoción tiene que haber un estímulo por esa razón le damos consideración a los acciones eventos y consecuencias para determinar si hay una relación entre esto y las emociones reflejadas. También queda pendiente un análisis con más tipos de personalidades para ver si es posible encontrar alguna tendencia a las expresiones reflejadas, con estos aspectos se intenta mejorar la experiencia del usuario tratando de que él tenga una experiencia más agradable para él, y dependiendo del contenido del juego podría incluso dejársele un mensaje positivo.

Este trabajo solo es el comienzo de la integración de emociones sentidas por el usuario al juego, pero de aquí parten otras maneras de poder interactuar con los juegos, y aún queda investigación por delante sobre cómo mejorar estos aspectos en cuanto a la detección de las emociones.

Reflexiones sobre el aprendizaje en investigación

La investigación científica no es algo sencillo o fácil de entender en las primeras etapas de lo que es la maestría; en sus inicios parecía muy fácil o complicado. A lo largo de cada semestre se fueron agregando requerimientos sobre cómo y para qué utilizar el módulo de emociones, y en ocasiones nos desviamos por caminos equivocados debido a nuestro enfoque y pensamiento práctico con que egresamos de la licenciatura, llevando un poco a la frustración, pero con la guía de tutor al final el trabajo se logró sacar adelante, dejando muchas experiencias con respecto a lo que es el método científico y la investigación en general.

VI.1 Trabajo Futuro

Existe aún la necesidad de seguir probando el modelo, quizá con algún juego más complejo para ver el comportamiento y cómo deberían de ser las decisiones que debería del juego utilizar la información proveída para realizar modificaciones sobre el juego (Chen, 2007; Yannakakis et al., 2013) para asegurar una mejor experiencia del usuario.

Sobre el tema de emociones, aún queda mucho por investigar, como por ejemplo, cuáles otros sensores podrían integrarse al módulo y cómo combinando sus lecturas se podría tener una opinión más certera para identificar la emoción que el usuario experimenta.

Otro tema interesante sería hacer este módulo más dinámico, puesto que en su forma actual de su implementación se basa en sesiones para realizar cambios, el hacerlo en tiempo real se supone un

requerimiento elevado de procesamiento para poder seguir manteniendo la ejecución del juego a una velocidad estable mientras se analizan los resultados y se hacen los cambios pertinentes. Un modo interesante sería la generación de niveles procedurales (Zook & Lee-Urban, 2012), sobre el cual se podría diseñar un juego de dos dimensiones con un personaje caminando por un nivel que se va generando mientras este avanza, tomando en consideración como reacciona ante ciertos peligros, por ejemplo, si se detecta que no le gusta cierto tipo de enemigo, procurar mostrarlo menos para no frustrarlo tanto o al contrario, según sea el objetivo del juego.

VII

Referencias

VII. Referencias

Acevedo, D. (2009). *Diseño de una arquitectura para incorporar emociones en videojuegos*. Universidad Nacional Autónoma de México.

Adams, E. (2013). *Fundamentals of game design* (3rd ed., p. 576). New Riders.

Ambinder, M. (2011). Biofeedback in gameplay: how Valve measures physiology to enhance gaming experience. In *Game Developers Conference*.

Anderson, E. F., Engel, S., Comninos, P., & McLoughlin, L. (2008). The case for research in game engine architecture. In *Proceedings of the 2008 Conference on Future Play Research, Play, Share - Future Play '08* (pp. 228–231). doi:10.1145/1496984.1497031

Association, E. S. (2014). Essential facts about the computer and video game industry : 2014 Sales, Demographic, and Usage Data. *Retrieved April*.

Bakkes, S. C. J., Spronck, P. H. M., & van Lankveld, G. (2012). Player behavioural modelling for video games. *Entertainment Computing*, *3*(3), 71–79. doi:10.1016/j.entcom.2011.12.001

Bartle, R. (1996). Hearts, clubs, diamonds, spades: Players who suit MUDs. *Journal of MUD Research*, *1*, 19. doi:10.1007/s00256-004-0875-6

Bartle, R. (2003). *Designing Virtual Worlds*. *New Riders Publishing In* (Vol. p, p. 741). doi:10.1093/carcin/bgs054

Bass, L., Clements, P., & Kazman, R. (2003). *Software Architecture in Practice*. *Architecture* (Vol. 2nd, p. 528). Addison-Wesley Professional.

Bateman, C., & Boon, R. (2005). Myer-Briggs Typology and Gamers. In *21st Century Game Design* (p. 352). Charles River Media.

Blow, J. (2004, February). Game development: Harder than you think. *Queue*, (February), 28–37.

Briggs, I., & Myers, P. (2010). *Gifts differing: Understanding personality type*. Nicholas Brealey Publishing.

Cass, S. (2003). Masters of doom: How two guys created an empire and transformed pop culture [Book Review]. *IEEE Spectrum*, *40*. doi:10.1109/MSPEC.2003.1197486

Charles, D., & Black, M. (2004). Dynamic Player Modelling: A Framework for Player-centred Digital Games. *Proceedings of 5th International Conference on Computer Games: Artificial Intelligence, Design and Education (CGAIDE'04)*, *Microsoft* , 29–35.

Chen, J. (2007, April 1). Flow in games (and everything else). *Communications of the ACM*, *50*(4), 31–34. doi:10.1145/1232743.1232769

Csikszentmihalyi, M. (2008). *Flow: The Psychology of Optimal Experience*. (1rs, Ed.) (p. 336). Harper Perennial Modern Classics.

Dekker, A., & Champion, E. (2007). Please biofeed the zombies: enhancing the gameplay and display of a horror game using biofeedback. *Proc. of DiGRA*, 550–558.

Deloura, M. (2009a, May). Game Engine Showdown. *Game Developer*, 7–12.

Deloura, M. (2009b, August). Middleware showdown. *Game Developer*, 8–14.

Doherty, M. (2003). *A software architecture for games*. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.89.2884&rep=rep1&type=pdf

Eeles, P., & Cripps, P. (2009). *The process of software architecting* (1st ed., p. 432). Pearson Education.

Folmer, E. (2007). Component Based Game Development – A Solution to Escalating Costs and Expanding Deadlines ? In *Component-Based Software Engineering 10th International Symposium, CBSE* (pp. 66–73).

Frasca, G. (2003). Simulation versus narrative. In *The video game theory reader* (pp. 221–235).

Frome, J. (2007). Eight Ways Videogames Generate Emotion. In *In Procs. Situated Play, Digital Games Research Association 2007 Conference (DiGRA 2007)* (pp. 831–835). Tokyo.

Gestwicki, P. (2012). The entity system architecture and its application in an undergraduate game development studio. In *Proceedings of the International Conference on the Foundations of Digital Games - FDG '12* (p. 73). doi:10.1145/2282338.2282356

Gilleade, K. M., & Dix, A. (2004). Using frustration in the design of adaptive videogames. In *Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology - ACE '04* (pp. 228–232). New York, New York, USA: ACM Press. doi:10.1145/1067343.1067372

Gregory, J. (2009). *Game engine architecture* (p. 864). CRC Press.

Herman, L. (2001). *Phoenix: The Fall and Rise of Videogames* (3rd editio., p. 388). Roleta Press.

Hobo, I. (2004). *Demographic Game Design: How to make game design as valuable as marketing* (p. 8).

Holt, R., & Mitterer, J. (2000). Examining video game immersion as a flow state. *108th Annual Psychological Association*.

Houlette, R. (2003). Player Modeling for Adaptive Games. In *AI Games Programming Wisdom 2* (pp. 557–566).

Hromada, D., Tijus, C., Poitrenaud, S., & Nadel, J. (2010). Zygomatic Smile Detection: The Semi-Supervised Haar Training of a Fast and Frugal System: A Gift to OpenCV Community. In *2010 IEEE RIVF International Conference on Computing & Communication Technologies, Research, Innovation, and Vision for the Future (RIVF)* (pp. 1–5). IEEE. doi:10.1109/RIVF.2010.5633176

ISO/IEC. (2007). ISO/IEC Standard for Systems and Software Engineering - Recommended Practice for Architectural Description of Software-Intensive Systems. *ISO/IEC 42010 IEEE Std 1471-2000*.

Juárez-Ramírez, R., Marquez-Olivas, J. A., & Medina-Ríos, J. J. (2014). *EMOCC Ontology –a Computational Representation of Emotions: Towards assessing User Experience in Human-Computer Interaction*. Tijuana.

Karouzaki, E., & Savidis, A. (2012). A Framework for Adaptive Game Presenters with Emotions and Social Comments. *International Journal of Computer Games Technology*, *2012*, 1–18. doi:10.1155/2012/929814

Kotsia, I., Zafeiriou, S., & Fotopoulos, S. (2013). Affective Gaming: A Comprehensive Survey. In *2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops* (pp. 663–670). IEEE. doi:10.1109/CVPRW.2013.100

Kruchten, P. (2004). *The Rational Unified Process: An Introduction* (p. 310). Addison-Wesley Professional.

Libgdx. (2014). Retrieved May 27, 2014, from http://libgdx.badlogicgames.com/

Martin, R. C. (2003). Agile Software Development: Principles, Patterns, and Practices. Retrieved from http://dl.acm.org/citation.cfm?id=515230

McShaffry, M., & David Graham. (2013). *Game coding complete* (4th ed., p. 960). Cenegage Learning PTR.

Medler, B. (2009). Generations of game analytics, achievements and high scores. *Eludamos. Journal for Computer Game Culture*, *3*(2), 177–194.

Medler, B., John, M., & Lane, J. (2011). Data Cracker: developing a visual game analytic tool for analyzing online gameplay. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'11)* (pp. 2365–2374). New York: ACM. doi:10.1145/1978942.1979288

Mejía Figueroa, A. (2013). *Modelo para la conservación de los atributos de calidad de mantenibilidad y flexibilidad de la arquietcura del software con la integración del modelo de usuario*.

Michelson, B. M. (2006). Event-Driven Architecture Overview. *Architecture*.

Microsoft. (2013). Kinectv2. Retrieved May 23, 2014, from http://www.microsoft.com/en-us/kinectforwindows/discover/features.aspx

Moore, M. E., & Sward, J. (2006). *Game Design and Development : Introduction to The Game Industry* (1st ed., p. 500). Prentice Hall.

Morelli, L., & Nakagawa, E. (2011). A Panorama of Software Architectures in Game Development. *23rd Inter. Conf. on Software Engineering & Knowledge Engineering*, 752–757.

Murphy-Hill, E., Zimmermann, T., & Nagappan, N. (2014). Cowboys, Ankle Sprains, and Keepers of Quality: How Is Video Game Development Different from Software Development? In *International Conference Software Engineering (ICSE)*.

Nacke, L., Kalyn, M., Lough, C., & Mandryk, R. (2011). Biofeedback game design: using direct and indirect physiological control to enhance game interaction. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems CHI 2011*, 103–112.

Nordmark, N. (2012). *Software Architecture and the Creative Process in Game Development*. Norwegian University of Science and Technology.

Novak, J. (2011). *Game Development Essentials: An Introduction* (3rd ed., p. 512). Cengage Learning.

Ortony, A., Clore, G., & Collins, A. (1990). *The cognitive structure of emotions*. England: Cambridge University Press.

Oxland, K. (2004). *Gameplay and design*. Pearson Addison Wesley.

Pedersen, C., Togelius, J., & Yannakakis, G. N. (2009). Modeling player experience in Super Mario Bros. *2009 IEEE Symposium on Computational Intelligence and Games*. doi:10.1109/CIG.2009.5286482

Picard, R. (2000). *Affective computing*. Cambridge, Ma.: MIT Press.

Pinelle, D., Wong, N., & Stach, T. (2008). Heuristic evaluation for games: usability principles for video game design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)* (pp. 1453–1462). New York, NY, USA: ACM.

Plummer, J. (2004). A flexible and expandable architecture for computer games, (December).

Poels, K., van den Hoogen, W., Ijsselsteijn, W., & de Kort, Y. (2012). Pleasure to play, arousal to stay: the effect of player emotions on digital game preferences and playing time. *Cyberpsychology, Behavior and Social Networking*, *15*(1), 1–6. doi:10.1089/cyber.2010.0040

Reyno, E. M., & Cubel, J. Á. C. (2009). A Platform-Independent Model for Videogame Gameplay Specification. In *Breaking New Ground: Innovation in Games, Play, Practice and Theory: Proceedings of the 2009 Digital Games Research Association Conference.*

Rollings, A., & Morris, D. (2003). *Game architecture and design: a new edition* (p. 960). New Riders.

Salen, K., & Zimmerman, E. (2004). *Rules of Play: Game Design Fundamentals*. *Leonardo* (Vol. 37, p. 670). doi:10.1093/intimm/dxs150

Sanglard, F. (2011). "Another World" source code review. Retrieved May 01, 2014, from http://fabiensanglard.net/anotherWorld_code_review/index.php

Sanglard, F. (2012). Doom3 Source Code Review: Introduction. Retrieved May 01, 2014, from http://fabiensanglard.net/doom3/index.php

Scherer, K. R. (2005). What are emotions? And how can they be measured? *Social Science Information*, *44*, 695–729. doi:10.1177/0539018405058216

Sharma, M., Mehta, M., Ontanón, S., & Ram, A. (2007). Player modeling evaluation for interactive fiction. In *In Third Artificial Intelligence for Interactive Digital Entertainment Conference (AIIDE), Workshop on Optimizing Player Satisfaction*.

Shaw, M., & Garlan, D. (1996). *Software Architecture: Perspectives on an Emerging Discipline* (p. 242). Prentice Hall.

Shimooka, R., & White, D. (2012). Battle for Wesnoth. In A. Brown & G. Wilson (Eds.), *The Architecture of Open Source Applications Elegance, Evolution, and a Few Fearless Hacks* (p. 432).

Smith, A. M., Lewis, C., Hullett, K., Smith, G., & Sullivan, A. (2011). An Inclusive View of Player Modeling. In *FDG '11 Proceedings of the 6th International Conference on Foundations of Digital Games* (pp. 301–303). doi:10.1145/2159365.2159419

Sollenberger, D. J., & Singh, M. P. (2012). Koko: an architecture for affect-aware games. *Autonomous Agents and Multi-Agent Systems*, *24*(2), 255–286. doi:10.1007/s10458-010-9160-3

Stevens, W., Myers, G., & Constantine, L. (1974). Structured design. *IBM Systems Journal*.

Sykes, J., & Brown, S. (2003). Affective Gaming Measuring emotion through the gamepad. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems* (pp. 732–733). doi:10.1145/765891.765957

Sylvester, T. (2013). *Designing Games: A Guide to Engineering Experiences*. O'Reilly Media.

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, *1*. doi:10.1109/CVPR.2001.990517

Wolf, M. J. (2002). Genre and the video game. In *The medium of the video game* (pp. 113–134).

Yannakakis, G. N., Spronck, P., Loiacono, D., & André, E. (2013). Player Modeling. *Artificial and Computational Intelligence in Games*, *6*, 45–59. doi:http://dx.doi.org/10.4230/DFU.Vol6.12191.45

Zhou, C., Yu, X., Dong, Y., Tian, J., Cui, Q., & Hu, L. (2007). Affective computation driven personalization modeling in game-based learning. In *Proceedings - 2007 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Workshops, WI-IAT Workshops 2007* (pp. 87–90). doi:10.1109/WIIATW.2007.4427546

Zimmerman, E. (2006). Narrative, interactivity, play, and games: Four naughty concepts in need of discipline. In *First Person: New Media as Story, Performance, and Game* (pp. 154–164). MIT Press.

Zoeller, G. (2010). Game Development Telemetry. In *Proceedings of the Game Developers Conference*.

Zook, A., & Lee-Urban, S. (2012). Skill-based mission generation: A data-driven temporal player modeling approach. In *Proceedings of the 3rd Workshop on Procedural Content Generation in Games*.

Capítulo VIII

Anexo

VIII. Anexo

VIII.1 OpenCV

OpenCV (Open Computer Vision) es una librería de programación con un enfoque a la visión de computadora en tiempo real, originalmente fue desarrollada por Intel y actualmente es mantenida por itseez.

La biblioteca tiene una gran cantidad de algoritmos optimizados de computación visual y de aprendizaje máquina, estos algoritmos pueden ser utilizados para detectar y recocer caras, identificar objetos, clasificar acciones humanas en video, rastrear movimientos de cámaras, grabar objetos en movimiento, búsqueda de imágenes, remover ojos rojos, seguir movimientos de ojos, etc.

Esta librería es multi-plataforma, teniendo interfaces con C++, C, Python, Java y Matlab, soportando uso desde Windows, Linux, Android y Mac, además se están desarrollando interfaces para que funcione sobre CUDA y OpenCL.

VIII.1.1 Uso durante la tesis

Esta biblioteca, fue utilizada para la detección de sonrisas de los jugadores, para esta caso se utilizó el medio de detección, se hizo uso de los *Cascade Classifiers*, que cosiste en una serie de clasificadores los cuales si uno tiene una detección correcta, se procede hacia los demás, el primer *cascade clasifier* fue realizado por Viola y Jones (Viola & Jones, 2001). Su funcionamiento consiste en posicionar varias regiones de interés sobre la imagen e ir analizándolas con clasificadores, regresando valores verdaderos cuando es posible que se encuentre un objeto que se está buscando, sobre el área determinada, así se analiza toda la imagen y se ignoran las secciones donde no se encontró nada y se prosigue a descartar las áreas donde al seguir avanzando no se encuentra el objeto deseado.

VIII.1.2 Procedimiento utilizado

Para la identificación de sonrisas, se hace por medio de dos pasos, en primer lugar utilizamos un clasificador de detección de rostro y posteriormente se hace una búsqueda de sonrisa, a continuación se describen los pasos:

1. Se obtiene la imagen de la cámara (véase Figura VIII.1).

Figura VIII.1: Imagen de cámara

2. En caso de ser necesario se reducen sus dimensiones (con el fin de acelerar el procesamiento).
3. Se transforma la imagen de color a una de blanco y negro.
4. Se le hace una ecualización de histograma (mejorar el contraste) a la imagen resultante (véase Figura VIII.2).

Figura VIII.2: Imagen con histograma ecualizado

5. Se analiza la imagen con un *Cascade Classifier* para rostro.
6. Se obtiene una región con el rostro detectado (véase Figura VIII.3).

Figura VIII.3: Rostro detectado

7. Sobre esta región, consideramos solo la mitad de altura del recuadro obtenido, con el fin de tratar de dejar la parte inferior de la cabeza (véase Figura VIII.4).

Figura VIII.4: Nueva región de interés

8. Se analiza el punto obtenido con un *Cascade Classifier* para sonrisas.
9. Se obtiene una región con la sonrisa detectada (véase Figura VIII.5).

Figura VIII.5: Sonrisa detectada

10. Se almacena una imagen con los resultados obtenidos, sobre la imagen original (véase Figura VIII.6).

Figura VIII.6: Imagen con resultados señalados

Con esto realizado, la función regresaba un valor “true” al detectar una sonrisa y “false” en caso contrario o incluso al no encontrar un rostro.