UNIVERSIDAD DE LAS FUERZAS ARMADAS - ESPE

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA DE SISTEMAS

APLICACIÓN DE LA METODOLOGÍA OOHDM Y TÉCNICAS DE INTELIGENCIA ARTIFICIAL EN LA SOLUCIÓN DEL DESARROLLO DE UN VIDEOJUEGO, ENFOCADO A NIÑOS DE 6 A 10 AÑOS, UTILIZANDO LA TECNOLOGÍA GDI+ BASADO EN C# Y WIIMOTE, PARA SU APLICACIÓN EN LA EMPRESA VIRTUAL LEARNING SOLUTIONS

PREVIO LA OBTENCIÓN DEL TÍTULO DE:

INGENIERÍA EN SISTEMAS E INFORMÁTICA

POR:

TORRES VINUEZA MARCELO DANIEL

SANGOLQUÍ, OCTUBRE DEL 2013

CERTIFICACIÓN

Certifico que el presente trabajo fue realizado en su totalidad por el señor Marcelo Daniel Torres Vinueza, como requerimiento parcial a la obtención del título de INGENIERO EN SISTEMAS E INFORMÁTICA.

1 de septiembre de 2013.

_____

Ing. Margarita Zambrano
Profesora Directora

CERTIFICACIÓN

Certifico que el presente trabajo fue realizado en su totalidad por el señor Marcelo Daniel Torres Vinueza, como requerimiento parcial a la obtención del título de INGENIERO EN SISTEMAS E INFORMÁTICA.

1 de septiembre de 2013.

_____

Ing. Carlos Prócel
Profesor Codirector

DECLARACIÓN DE AUTORÍA DE RESPONSABILIDAD

Yo, Marcelo Daniel Torres Vinueza, certifico que contribuí directamente a la creación de este manuscrito, a la génesis y análisis de sus datos, por lo cual estoy en condiciones de hacerme públicamente responsable de él, y acepto que mi nombre figure como su autor.

1 de septiembre de 2013.

_____

Marcelo Daniel Torres Vinueza
Autor.

AUTORIZACIÓN

Yo, Marcelo Daniel Torres Vinueza

Autorizo a la ESCUELA POLITÉCNICA DEL EJÉRCITO la publicación, en la Biblioteca Virtual de la Institución, del trabajo "Aplicación De La Metodología OOHDM Y Técnicas De Inteligencia Artificial En La Solución Del Desarrollo De Un Videojuego, Enfocado A Niños De 6 A 10 Años, Utilizando La Tecnología GDI+ Basado En C# Y Wiimote, Para Su Aplicación En La Empresa Virtual Learning Solutions", cuyo contenido, ideas y criterios es de mi exclusiva responsabilidad y autoría.

1 de septiembre de 2013.

_____

Marcelo Daniel Torres Vinueza
Autor.

DEDICATORIA

Dedico este trabajo a toda mi familia, la cual ha comprendido y ha puesto su parte para culminar con esta ardua tarea que se me ha puesto al frente; y a mis amigos, sin los cuales no habría encontrado la suficiente entereza y ganas para trabajar al máximo.

"No se aprende nada importante en la vida. Simplemente se recuerda."

Carlos Ruiz Zafón

MARCELO TORRES VINUEZA

AGRADECIMIENTOS

Un especial agradecimiento a Dios, el que me ha puesto las personas (las que se han quedado a mi lado y las que no) y los sucesos (no los cambiaría por nada) que han hecho que me convierta en la persona que soy.

Mis padres y hermanos; los cuales, a su manera, han influenciado en mí y me han dado su apoyo incondicional a su momento.

A mis amigos, esos hermanos de espíritu, cuya presencia da complemento a mi vida.

A mis profesores, en especial a los que ha apoya la culminación de este trabajo, gracias por el conocimiento que han impartido a mi mente.

Finalmente, a las Escuela Politécnica del Ejército, ha sido mi *Alma Mater* en muchos aspectos en mi vida.

MARCELO TORRES VINUEZA

ÍNDICE

ÍNDICE DE FIGURAS Y TABLAS

RESUMEN

El presente trabajo tiene como objetivo la construcción de un **Videojuego** Educativo llamado *Puzzlemote*, un juego de rompecabezas tipo **N-Puzzle**, creado específicamente para niños en edad escolar (de 6 a 10 años), como parte de una solución de productos educativos de la empresa Virtual Learning Solutions. Se utilizó como metodología la de **Diseño Hipermedia Orientada a Objetos** para su diseño y construcción, con un motor gráfico basado en la **Tecnología GDI+** de Windows, además de usar un motor de Razonamiento Hacia Adelante y Hacia Atrás (**Inteligencia Artificial**) para desarmar y armar de manera didáctica el rompecabezas. El software final posee un sistema de registro de producto para que la empresa pueda venderla con licencias. Así mismo, se creó una biblioteca de conexión con el Dispositivo de Interfaz Humana (un dispositivo de entrada) conocido como **Wiimote**, de la empresa japonesa Nintendo, para que el jugador interactúe con el juego usando dicho control a distancia. Finalmente, los rompecabezas están hechos con animales ecuatorianos autóctonos; como son el cóndor, el perezoso, la tortuga galápago, etc., con el propósito que los niños generen un poco de conciencia hacia lo que es de su país.

Palabras Clave: Videojuego, N-Puzzle, Diseño Hipermedia Orientada a Objetos, Tecnología GDI+, Inteligencia Artifical, Wiimote

CAPÍTULO 1: DEFINICIÓN DEL PROYECTO

1.1. INTRODUCCIÓN

Gracias al crecimiento exponencial de la tecnología, explicado varias veces por Ray Kurzweil en una conferencia realizada en Madrid en junio de 2010, el aprendizaje del uso de las Tecnologías de Información se hace cada vez más intuitivo, tanto así que los niños de ahora desarrollan con facilidad el conocimiento y la habilidad necesaria para manejar todos los artículos de tecnología que existen en el mercado. Pueden manejar con facilidad equipos informáticos como: computadoras, celulares, hasta video juegos dirigidos hacia un nicho mucho más maduro.

Así mismo, el desarrollo de nuevas técnicas de educación (que faciliten la adquisición de conocimientos por parte de la niñez) se hace mucho más sencillo, a la vez que se complican cada vez más para los desarrolladores de tecnología. La Inteligencia Artificial es una herramienta poderosa a la hora de generar un sistema, que permita a un niño de manera didáctica, el desarrollo de su memoria a una temprana edad.

La creación de estas nuevas formas de manejo de información, permiten con un solo movimiento de nuestras manos, realizar una acción sobre un sistema informático cualquiera. Un ejemplo claro de ello es, sin duda, el Wiimote, el cual permite conectarse a un equipo por medio de tecnología Bluetooth. Esta herramienta ha demostrado que es fácil de adaptar a los requerimientos de desarrollo, además que es fácil de obtener y es asequible a los bolsillos de los usuarios.

1.2. FORMULACIÓN DEL PROBLEMA

Virtual Learning Solutions es una empresa que se encarga del desarrollo de soluciones informáticas, las cuales satisfacen la demanda de productos para la educación virtual, tanto por sistemas de escritorio, como por medio e-learning. Esta empresa ha descubierto un nicho de mercado que aún no ha sido explotado correctamente en nuestro país: la generación de programas para la educación de niños de temprana edad por medio de sistemas informáticos.

El problema de generar estas herramientas, radica en su costo elevado al momento de investigación, planificación y desarrollo, de tal manera que permita a la empresa obtener una diferencia en relación a sus competidores inmediatos. Tomando en cuenta lo anterior, se presenta la necesidad de desarrollar un sistema con herramientas ya existentes en el mercado, lo que permitirá abaratar costos de producción para la compañía.

A partir de esto, se ve la oportunidad de generar un software que permita a un niño, que se encuentre entre los 6 a 10 años, desarrolle su memoria de una

manera didáctica, basándose en un sistema que gestione, tanto Inteligencia Artificial, como un control Wiimote, de una forma más fácil e interactiva, aprender a desarrollar su mente.

1.3. IMPORTANCIA Y JUSTIFICACIÓN

Los juegos son unas herramientas que han demostrado que su uso facilita la enseñanza y/o aprendizaje, tanto de los educadores como los educandos. Así, se puede provocar que un niño desarrolle su memoria de manera más temprana, y permitiendo cimentar una absorción de información de mejor manera.

Así mismo, la adquisición de equipos que se usan en este tipo de tecnologías, suele tener un costo elevado. La necesidad de abaratar la compra de equipos lúdicos se ve implícita tanto en el desarrollo de nuevas herramientas, así como el uso de equipos que ya existen dentro del mercado.

La aplicación que se desarrollará permitirá tener las bases teóricas, prácticas, habilidades y destrezas en la formación integral de los niños generando en ellos un interés y un aprendizaje fácil mediante el uso juegos educativos interactivos en 2D, utilizando imágenes de animales del ecuador, para también generar una identidad nacional

1.4. OBJETIVOS

1.4.1. General

Aplicar la metodología OOHDM y técnicas de Inteligencia Artificial en la solución del Desarrollo de un Videojuego, enfocado a niños de 6 a 10 años, utilizando la tecnología GDI+ basado en C# y Wiimote, para su aplicación en la empresa Virtual Learning Solutions.

1.4.2. Específicos

- Revisar y documentar los conceptos teóricos acerca de videojuegos didácticos y motores de juegos, así como la aplicación de la Inteligencia Artificial.
- Revisar y documentar las distintas fases de la metodología OOHDM y su aplicabilidad para el desarrollo de videojuegos.
- Realizar el análisis y el diseño de la aplicación 2D utilizando la metodología OOHDM con UML.
- Elaborar la documentación de la técnica de razonamiento de Inteligencia Artificial llamada “Encadenamiento Hacia atrás y Hacia adelante”.
- Realizar y documentar la librería de control para el manejo del hardware Wiimote.
- Elaborar un video juego, usando 2D, la librería Wiimote, el razonamiento “Encadenamiento Hacia atrás y Hacia adelante”, que contenga imágenes de animales ecuatorianos.

1.5. ALCANCE

El sistema contemplará el videojuego didáctico “Puzzlemote”, específico para las edades de 6 a 10 años, con la opción de seleccionar entre 3 tipos de rompecabezas (8, 15 y 24 piezas), los cuales mostrarán entre 6 imágenes de animales del ecuador distintos, y guardar el tiempo de la partida jugada.

A esto se agregará la librería de conexión al Wiimote, el cual permitirá su uso dentro del “Puzzlemote”, está librería se encargará de manejar los datos de entrada y salida de la aplicación.

Se usarán archivos planos para el manejo de información (registro de usuarios y configuración). Esto debido a la imposibilidad de usar base de datos entre diferentes máquinas.

Así mismo, se pondrá un módulo de seguridad, el cual impedirá por parte de terceros la distribución sin permiso del producto final.

Finalmente se usará la técnica de Inteligencia Artificial de razonamiento “Encadenamiento Hacia atrás y Hacia adelante”, con el fin de que si el usuario no pudo resolver el rompecabezas, el sistema lo resolverá paso a paso por sí mismo.

1.6. METODOLOGÍA DE DESARROLLO

El desarrollo del presente trabajo se hará por medio de la Metodología de Diseño de Hipermedia Orientada a Objetos (OOHDM por sus siglas en inglés), la cual permite una planificación más adecuada para el desarrollo de un sistema

informático de tipo Diseño, de una manera ágil y de fácil control por parte del desarrollador.

1.7. HERRAMIENTAS DE DESARROLLO

SOFTWARE	UTILIZACIÓN
GDI+	Manejo de librerías gráficas
Fireworks	Diseño de fondos
C#.NET	Programación de la aplicación
GIMP	Edición de imágenes
XML	Manejo de archivos

CAPÍTULO 2: MARCO TEÓRICO

2.1. VIDEOJUEGOS Y APRENDIZAJE

2.1.1. El Juego Como Actividad De Aprendizaje

Una de las actividades que los animales más evolucionados (mamíferos o aves, por citar ejemplos claros) poseen, es sin duda el juego. Este se realiza para que los más jóvenes aprendan los conceptos básicos del comportamiento que deben tener (la caza para alimentarse o la defensa contra los depredadores).

De la misma manera, desde los albores de la humanidad, los niños y jóvenes adultos han ido adquiriendo el conocimiento y comportamiento necesario para su supervivencia por medio del juego. Como la misma naturaleza del hombre invita a tener curiosidad de cuál es la razón e importancia del juego en la crianza, también desde la antigüedad se encuentran ejemplos de investigación concretos sobre la preocupación y estudio del juego[1] (Gorris, 1997).

[1] J. M. Gorris hace referencia a las obras de Platón y Aristóteles, donde se reconoce el valor práctico del juego en la educación del niño (Gorris, 1997: 9)

Hasta ahora, investigadores de los campos de educación han mostrado interés en el juego como actividad de aprendizaje; algunos han desarrollado sus propias teorías al respecto, y la potencialidad del mismo. Existen muchos trabajos y variados estudios sintetizados de muchas teorías[2], pero todas pueden demostrar que es lo que hace especial al juego como una herramienta en la educación de nuestra sociedad.

Borja en su trabajo dice que “el juego, al igual que el lenguaje, es una constante antropológica, la que se encuentra en todas las civilizaciones a lo largo de la historia y en cada etapa de las mismas” (Borja, 1980).

Así mismo, se puede apreciar gracias a la constancia histórica y por la misma vida diaria, que el juego se manifiesta durante todas las etapas de la vida humana: “a todas las personas, tengan la edad que tengan, les gusta jugar” (Martín et al., 1995); por lo tanto, se podría decir que esta actividad cumple diferentes funciones y adquiere diferentes significados para cada individuo, y por cada período de su vida.

Definir la palabra juego es algo que es casi subjetivo en relación personal o social. Se podría decir que el juego es “una acción u ocupación libre, la cual se desarrolla dentro de los límites temporales y espaciales determinados, según unas reglas absolutamente obligatorias, aunque libremente aceptadas, acción que tiene en su fin en sí misma y que va acompañada de un sentimiento de tensión y/o alegría, y de una conciencia de ‘ser de otro modo’ que la vida corriente. Este concepto trata

[2] Las obras de Gorris (1997), Elkonin (1998), López Rodríguez (1998), Cañeque (1999) o López Mantalla (1993) tratan sobre muchas de las teorías con mayor detalle

de abarcar el porqué de los juegos en los animales, niños y adultos: juegos de cálculo y azar, exhibiciones y representaciones"[3].

Otros dicen que el juego es "una ocupación separada, cuidadosamente aislada del resto de la existencia y realizada por lo general dentro de los límites precisos de tiempo y lugar"[4]. Aquí se puede apreciar que la identificación y descripción de las características de un juego como tal.

Basado en todo lo anteriormente dicho, se puede concluir ciertas cualidades que posee el juego. Lo caracterizan y diferencian del resto de las actividades cotidianas que posee un animal o humano. Aunque podrían notarse obvias estas cualidades, se debe inducir que existen otras características no visibles, estas se pueden encontrar en los resultados de practicar el juego. En conclusión, un juego posee características intrínsecas y extrínsecas de la actividad como tal.

2.1.1.1 Características Intrínsecas del Juego

Se pueden decir que son:

- **El juego es una actividad libre.**
- **Es autotélico.**
- **Es una actividad placentera.**
- **Es una actividad ficticia.**
- **Esta limitada en el tiempo y el espacio.**

[3] La notoriedad de Huizinga se debe a dos de sus obras: El otoño de la Edad Media (Herfsttij der Middeleeuwen- 1919) y Homo Ludens (1938).
[4] Caillois es un escritor de libros de filosofía, sociología, psicología y pedagogía; autor de más de veinte obras literarias, entre las que destacan: Los juegos y los hombres: la máscara y el vértigo (1958), El Mito y los Hombres (1938) y El hombre y lo sagrado (1939), entre otras.

- **Se regula por reglas y/o normas.**
- **Es una actividad global.**

Estas características están relacionan entre sí; como por ejemplo: no se puede imaginar que el juego no tenga cierto tipo de placer al desarrollarse en el tiempo y espacio que se desee.

2.1.1.2 Características Extrínsecas del Juego

Los autores Haywood et al. (1993) y Franch y Martinell (1994) en cada una de sus obras, resumen las características potencialmente subyacentes que se encuentra en el juego. Estas pueden afectar las cuatro dimensiones fundamentales del individuo: las dimensiones motora, intelectual, afectiva y social.

- **El juego y su influencia en el desarrollo motor.** El juego provoca movimientos físicos, estimula los gestos y su precisión al momento de usarlos, la fuerza y velocidad, etc., mejorando la capacidad de percepción de y de los sentidos.
- **En el desarrollo Intelectual.** Al jugar, se debe comprender como hacerlo, como encontrar la solución para poder terminarlo, como funciona en si las herramientas o elementos que posea el juego, elaborar rutas o estrategias para terminarlos, etc. Esta es la característica es la que más interesa para el desarrollo del proyecto actual.
- **En el desarrollo Afectivo.** El juego permite una evasión temporal de la realidad, dejar los problemas a un lado, y al ser una actividad libre y placentera, un individuo puede comportarse de tal manera que puede

mejorar su estado de ánimo. Un juego puede cambiar el estado afectivo de una persona, permitiendo que exista una mayor estimulación para la maduración individual. Por ende, el juego permite a una persona sentirse mejor consigo misma y en algún grado hacia los demás.

- **En el desarrollo Social.** Jugar entre varios individuos permite que estos entablen relaciones entre ellos. Los juegos pueden ser improductivos en lo relacionado a la parte económica, pero tiene una elevada productividad relacional. El juego promueve el aprendizaje de la vida social del grupo en la que se utiliza. Genera la simbología para transmitir valores y actitudes entre personas, lo que quiere decir que “los juegos son situaciones inventadas que permiten la participación de mucha gente en algún patrón significativo de su propia vida corporativa” (McLuhan, 1996).

2.1.2. El Fenómeno Social de los Videojuegos

Los videojuegos constituyen un fenómeno popular que se inserta en el proceso de desarrollo tecnológico que experimenta nuestra sociedad. Se introdujeron por primera vez en los Estados Unidos a principios de los años setenta, con un éxito sin precedentes en los salones recreativos hasta entonces ocupados por máquinas antiguas (ejemplo como los tragamonedas).

Los videojuegos constituyen una de las actividades de entretenimiento más populares de la actualidad. Además, desde la segunda mitad de la década de los

ochenta, se ha ampliado el campo de acción y ha sobrepasado la frontera del entretenimiento abriendo posibilidades de uso en el ámbito educativo.

Limitar el concepto de videojuego a una actividad exclusivamente lúdica supone obviar las potencialidades instructivas o educativas del mismo, estudiadas a partir de numerosas investigaciones (Estallo, 1995). Sin embargo, no se debe olvidar que estos videojuegos didácticos pueden llegar a perder el sentido propio del juego desde el momento en que quien los utilice lo haga con el objetivo de aprender y no por el simple hecho de jugar.

Los videojuegos (Provenzo 1991) son algo más que un producto informático. También son un negocio, para quienes los manufacturan y los venden, y una empresa comercial sujeta como todas, a las fluctuaciones del mercado.

2.1.3. Tipos de Videojuegos y Propuestas de Clasificación

Los videojuegos se puede clasificar por varios criterios: según el *hardware* que utilizan, los contenidos del juego, grupo objeto, etc.

Algunos autores[5] dividen a los juegos como:

a) **Juegos de laberinto. –** El jugador guía al personaje del juego a través de laberintos, en los que la dificultad de juego aumenta progresivamente.

[5]Bloom, S. (1982). *Video invaders*. Nueva York: Arco Publishing. Citado por Brusa, 1987.

b) **Juegos de escalada. –** El personaje del juego escala hacia la parte superior de la pantalla, intentando evitar o destruir todo aquello que se interpone en su camino.

c) **Juegos de invasores del espacio. –** El enemigo desciende de la parte superior de la pantalla mientras el protagonista del juego se mueve de lado a lado, en la parte inferior de la misma, e intenta destruir a los invasores antes de que lo alcancen.

Otros autores, a partir de las características generales del desarrollo del juego, clasifican los videojuegos en siete tipos distintos (Martínet al., 1995):

- **Arcade.** Son los juegos de ordenador más tradicionales. En ellos el jugador a través de un personaje debe superar una serie de obstáculos de creciente dificultad, matar a los enemigos que le atacarán y coger una serie de objetos que le serán útiles en el transcurso del juego. También se organiza dentro de esta clasificación los simuladores deportivos, y los juegos de lucha o los juegos de construcción, en los que se deben ir encajando distintas piezas para ir formando figuras determinadas a gran velocidad y con una dificultad que aumenta progresivamente.
- **Aventura.** Parten de la idea de conseguir un objetivo determinado en un ambiente de aventura y peligro en el que el jugador deberá superar dificultades, resolver problemas o enigmas, o derrotar a sus enemigos.
- **Estrategia.** En este tipo de videojuegos se suele reproducir una situación compleja en la que el jugador debe controlar una serie de variables para lograr una meta concreta.

- **Juegos de rol.** Son una simulación de los juegos de mesa que llevan el mismo nombre, donde el ordenador juega el papel de director del juego y contiene las reglas del mismo.
- **Simuladores.** Reproducciones muy sofisticadas de aparatos o actividades complejas como, por ejemplo, los simuladores de vuelo, de conducción de vehículos o de realización de deportes concretos.
- **Educativos.** Juegos en los que prima una finalidad más educativa que de puro entretenimiento.
- **Juegos de mesa.** Reproducciones de gran parte de los juegos de mesa tradicionales.

Otra clasificación menos popular se podía también decir de las siguientes[6]:

- **La perspectiva del jugador:** Hace referencia a la forma en que el jugador se incorpora a la actividad de juego.
    - Manipulación de objetos – El jugador controla los movimientos y actividades de un objeto, vehículo, arma, nave, etc.
    - Manipulación del personaje – El jugador controla los movimientos y actividades de un personaje.
    - En primera persona – Expresa una ilusión de que está enfrentándose directamente con la actividad que se desarrolla en la pantalla.
- **El campo de percepción:** Toma en consideración el cómo presentar un universo particular a partir de un programa de juego determinado.

[6]Garner realizó una investigación centrada en las máquinas de videojuegos que funcionan con monedas *(coin-operated video games)* por lo que en su clasificación no incluye los videojuegos domésticos.

- De límites fijos – Se presenta al jugador los límites donde tendrá lugar el juego.
- De ventana fija – No se define los límites conceptuales del juego, solamente los personajes u objetos pueden moverse más allá de los ejes visibles, dando la sensación de que el juego se lleva a cabo en un universo y que cada pantalla o ventana representa un área del mismo.
- De ventana móvil - La apariencia de un foco de visión móvil en lugar de fijo.

- **Las capacidades interactivas:**
  - Competición individual.
  - Competición interactiva indirecta: los jugadores van alternando sus turnos.
  - Competición interactiva directa: los jugadores utilizan los objetos o personajes del juego para competir entre ellos.
  - Cooperación/competición interactiva directa: los jugadores comparten el esfuerzo para vencer a los enemigos generados por el juego, sin que se excluya la competencia.
- **El escenario de juego:** Naturaleza del universo del juego en el que se desenvuelve el jugador:
  - Realidad auto-generada – Específicamente diseñada para el juego.
  - Realidad transferida – Simulación de nuestra vida diaria.
  - Realidad negociada – Combina las dos realidades anteriores en un contexto determinado.

Sin embargo, una de las clasificaciones más completa (Estallo, 1995) es la que combina dos criterios distintos: las habilidades y recursos psicológicos necesarios para el juego y, el desarrollo y temática del juego en sí:

- **Juegos de Arcade:** Requieren un ritmo rápido de juego, exigiendo tiempos de reacción mínimos y una atención focalizada; apenas cuentan con un componente estratégico.
- **Juegos de Plataforma:** El jugador se halla en un escenario bidimensional desplazándose de izquierda a derecha y de arriba a abajo.
- **Laberintos:** El escenario que reproduce un laberinto de considerable extensión.
- **Competiciones deportivas:** Reproducen deportes distintos.
- **Juegos de acción:** Los escenarios varían al eliminar el número de enemigos suficiente.
- **Juegos de simulación:** Simulan actividades o experiencias raramente accesibles en la vida real. Permiten al jugador asumir el mando de situaciones o tecnologías específicas. Entre sus características principales destacan:
    - La baja influencia del tiempo de reacción en comparación con los juegos de Arcade.
    - La utilización de estrategias complejas y la necesidad de enfrentarse a situaciones nuevas que exigen conocimientos específicos sobre la simulación.

- **Simuladores instrumentales:** Primeros en comercializarse. Tienen su origen en los simuladores de vuelo utilizados en el entrenamiento de pilotos aéreos.
- **Simuladores situacionales:** El jugador asume un papel específico determinado por el tipo de simulación (juegos de ajedrez).
    - **Juegos estratégicos:** el jugador adopta un papel específico y sólo conoce el objetivo final. Con frecuencia los personajes son de ficción y provienen del mundo de la literatura y del cine.
- **Reproducciones juegos de mesa:** con el mismo desarrollo que sus originales.

A pesar de todo, existen videojuegos que difícilmente pueden excluirse o incluirse en alguna de las categorías descritas.

2.1.4. La Investigación Sobre Videojuegos

El interés por demostrar los efectos de los videojuegos, ya sean positivos o negativos, es la motivación de la mayoría de las investigaciones desarrolladas. Al revisar los estudios llevados a cabo en este campo se descubrió que los videojuegos han sido estudiados desde diversas perspectivas. La Tabla 2.1 muestra una síntesis de las principales líneas de investigación sobre los videojuegos.

Tabla 2.1. Líneas de investigación sobre videojuegos.

El perfil de los jugadores
▪ Diferencias de género ▪ Edad, lugar de juego, preferencias
Los efectos positivos y negativos de los videojuegos
▪ Videojuegos y agresividad ▪ Videojuegos y adicción ▪ Videojuegos y habilidades sociales ▪ Videojuegos y rendimiento escolar ▪ Potencial instructivo de los videojuegos
Otras aplicaciones sociales para los videojuegos
▪ Como medio didáctico ▪ Como test psicológico ▪ Como herramienta para el tratamiento

No todos los videojuegos son iguales y, lógicamente, no todos presentan el contenido violento de la misma forma - por ejemplo, no es igual un videojuego de violencia fantástica que uno de violencia humana. Así mismo, se puede concluir que, como toda actividad, los videojuegos generan polémica porque si su contenido no es el apropiado, puede crear daño en el perfil psicológico de un individuo (sin importar la edad, sexo o estructura social).

2.1.5. Juego de Rompecabezas

Un rompecabezas es un problema o enigma que se debe solventar por medio de algún tipo de genialidad[7]. En un rompecabezas básico, se debe colocar con lógica las piezas o fichas en orden, de tal manera que se encuentre una solución deseada para el problema.

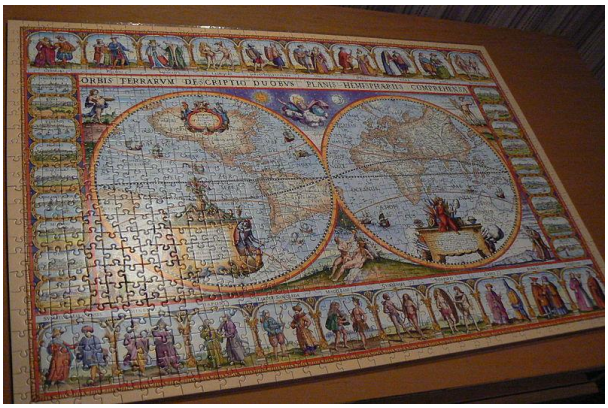


Figura 2. Los rompecabezas normalmente forman figuras cuando están completas

Normalmente los rompecabezas son usados como una forma de entretenimiento, aunque también concentran problemas serios en las matemáticas o problemas lógicos; que, si se puede resolverlos, pueden ayudar significativamente en la investigación matemática o del campo en donde se encontraron dichos problemas tipo rompecabezas.

Algunos rompecabezas se los puede resolver con ciertos patrones ya definidos, manejándolos en un orden particular. Las personas que tienen una alta

[7] Kendall G., Parkes A. y Spoerer K. (2008) "A Survey of NP-Complete Puzzles"

capacidad inductiva pueden encontrar una solución de una manera más fácil que otras personas.

2.1.5.1. Historia de los rompecabezas

El primer rompecabezas como juego de enseñanza, fue creado cerca del año 1760, por el entonces creador de mapas y grabador británico John Spilsbury[8], coloco un mapa en una hoja de madera, la cual aserró en los bordes de cada línea divisoria entre los países marcados en él. Con el resultado del corte, Spilsbury enseñaba geografía a sus alumnos. Cuando este método se hizo famoso, el uso de este rompecabezas se mantuvo como la mejor manera de enseñanza en la materia[9].

Al inicio del siglo 20, revistas y periódicos descubrieron que si agregaban competencias de armado de rompecabezas a sus subscritores, las ventas de diarias aumentaban en gran porcentaje. Estos rompecabezas incluían letras, palabras, números, formas y acertijos.

2.1.6. N-Puzzle

Los n-Puzzle (del inglés *puzzle*, rompecabezas), fueron creados por Sam Lloyd cerca del año 1874[10]. Esta versión mostraba 16 bloques numerados, los cuales se colocaban juntos en filas de 4, cada una sumando 34. El juego comenzó

[8] Hannas, Linda. *The English jigsaw puzzle, 1760-1890.* Wayland, 1972, p. 20, se lo acredita como el creador de los rompecabezas como se los conoce ahora, el "jigsaw puzzle".
[9] Esta historia se encuentra en el archivo "Jigsaw Puzzle History" que se encuentra en la página web http://www.jigsaw-puzzle.org/jigsaw-puzzle-history.html
[10] *The 15 Puzzle*, de Jerry Slocum & Dic Sonneveld, 2006.

a ser facturado por la “Escuela Americana para Sordos”, en Nueva York, en diciembre de 1879.

Noyes Chapman intento aplicar una patente al producto en febrero de 1880. Aun así, las patentes fueron rechazadas, ya que no eran tan diferentes de las patentes de otro tipo de n-Puzzle, el “Puzzle Block” (bloques de rompecabezas) del señor Ernest U. Kinsey[11].

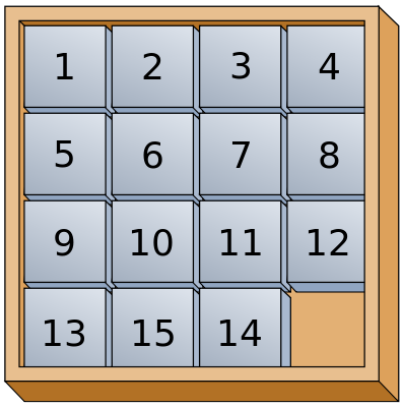


Figura 2.2 El 15-Puzzle sin solución de Sam Lloyd

Capacidad de Solución al problema 15-Puzzle

La figura 2.2 muestra un caso en el 15-Puzzle es imposible de resolver. Las fichas 14 y 15 están intercambiadas entre sí. Este rompecabezas probablemente no tiene solución debido a que cuando se intenta moverlo hacia un estado de solución, requeriría un cambio de la variable constante o *invariante*[12].

[11] *Ídem*
[12] En matemáticas, una invariante es la propiedad de una clase de objetos matemáticos que se mantiene sin cambios cuando transformaciones de cierto tipo son aplicados a esa clase.

Johnson & Story (1879[13]) usaron un argumento de paridad que mostraba que la mitad de las posiciones en el n-puzzle eran imposibles de resolver, sin importar el número de movimientos que se puedan realizar. Esto sucede si se considera que la configuración de la invariante como una función que no varía dentro del cualquier movimiento válido, y usarlo para partir el espacio de todos los posibles estados dentro de dos clases equivalentes de estados alcanzables e inalcanzables entre sí[14].

Estos autores también demostraron que todas los tableros que posean fichas *m x n* (siendo m y n al menos el número 2) en sus permutaciones son todas solucionables.

Wilson (1879[15]) estudio la analogía del 15-puzzle con grafos arbitrarios conectados y no separables (a un grafo se lo llama separable si al retirar un vértice aumenta el número de componentes). En este estudio, se demostró que, exceptuando por polígonos y en un excepcional grafo de 7 vértices, es posible obtener todas las permutaciones; al menos que el grafo sea bipartito, en cuyo caso, las permutaciones exactas se obtendrán para ese grafo.

[13] Johnson, Wm. Woolsey; Story, William E. (1879), "Notes on the "15" Puzzle", American Journal of Mathematics (The Johns Hopkins University Press) 2 (4): 397–404
[14] Johnson, Wm. Woolsey; Story, William E. (1879), "Notes on the "15" Puzzle", American Journal of Mathematics (The Johns Hopkins University Press) 2 (4): 397–404
[15] Wilson, Richard M. (1974), "Graph puzzles, homotopy, and the alternating group", Journal of Combinatorial Theory. Series B 16: 86–96

Para versiones más grandes de n-puzzle, encontrar una solución es más sencillo, pero encontrar la mejor solución es extremadamente difícil[16][17]. Para el 15-puzzle, el número de pasos para su solución varía entre 0 a 80 movimientos de fichas, en un promedio de 43 movimientos[18]. Un 8-puzzle puede ser resuelto en no menos de 31 a 24 movimientos de fichas.

2.2. INTRODUCCIÓN A LA INTELIGENCIA ARTIFICIAL

Las técnicas de Inteligencia Artificial permiten la representación del conocimiento junto a un mecanismo de inferencia, mediante el cual se obtienen conclusiones después de un proceso de razonamiento o deducción[19].

El mecanismo de IA es aprovechado en la creación de Sistemas Inteligentes, los mismos que pueden utilizarse para la generación de procesos de Enseñanza – Aprendizaje.

De esto se puede deducir que si se desea crear un proceso de enseñanza más óptimo, es indispensable que se introduzca la IA en la mayoría, por no decir todas, las capas de la educación de niños y jóvenes.

---

[16] Daniel Ratner, Manfred K. Warmuth. Finding a Shortest Solution for the N × N Extension of the 15-PUZZLE Is Intractable. Conferencia Nacional de Inteligencia Artificial, 1986.
[17] Ratner, Daniel; Warmuth, Manfred (1990). "The (n2−1)-puzzle and related relocation problems". Journal of Symbolic Computation 10 (2): 111–137
[18] A. Brüngger, A. Marzetta, K. Fukuda and J. Nievergelt, The parallel search bench ZRAM and its applications, Annals of Operations Research 90 (1999), pp. 45–63.
[19] Maikel León Espinosa, Zenaida García Valdivia, "La Inteligencia Artificial en la Informática Educativa", Revista de Informática Educativa y Medios Audiovisuales Vol. 5(10), págs. 11-18. 2008

2.2.1. La Computadora en el Ámbito Educativo

En épocas anteriores, se trataba de justificar que las computadoras podían utilizarse como objeto o medio de enseñanza dentro del plan de estudios de una disciplina, si no en la misma de la computación. Hoy en día, se puede afirmar que si no existe dentro del proceso educativo el uso masivo de computadoras, ese proceso tiene bastantes carencias al impartir conocimiento. Por lo tanto, es razonable pensar que se debe planear de manera consiente en que facetas de la etapa educacional el uso de la computadora.

Existen cuatro formas de utilizar esta máquina en el proceso docente educativo, las cuales son:

- Lograr el dominio del aprendizaje por el reforzamiento y ejercitación.
- Realizar proceso de aprendizaje por descubrimiento.
- Generar procesos de búsqueda en contexto de interacción.
- Favorecer procesos de construcción de conocimiento.

Lo anterior no es necesariamente rígido, pero cada una de estas formas tiene su variante y suelen estar casi siempre tomadas en cuenta al momento de planificar procesos de enseñanza educativa[20].

Al analizar la computadora en la dimensión del medio enseñanza – aprendizaje, se hace necesario conocer al conjunto de recursos informáticos

[20] Maikel León Espinosa, Zenaida García Valdivia, "La Inteligencia Artificial en la Informática Educativa", Revista de Informática Educativa y Medios Audiovisuales Vol. 5(10), págs. 11-18. 2008

diseñados con la intención de usarse en el contexto conocido como "*Software Educativo*"[21].

Entonces, salta la pregunta ¿Cómo utilizar la computadora ante cada tipo de situación educativa? Para lograr que estas máquinas cumplan con su cometido, es necesario dotarlas del software educativo necesario y de calidad, lo que debe medirse en términos del conocimiento, que sean capaces de representar y transmitir correctamente la información que contienen.

Para la creación de un software educativo correcto, se debe considerar lo siguiente:

- Determinar la existencia de un problema educativo a resolver.
- Asegurar que la computadora efectivamente tiene ventajas cualitativas sobre medios educativos para resolver el problema[22].

Ventajas de los Software Educativos

León Espinosa y García Valdivia (2008) demuestran que el uso de la computadora en sus diversas modalidades ofrece, sobre otros métodos de enseñanza, ventajas tales como:

- Participación activa por parte del alumno en la construcción de su propio aprendizaje.

[21] Ídem
[22] Ídem

- Interacción entre el alumno y la máquina.
- La posibilidad de dar una atención individualizada al estudiante.
- Permite el desarrollo cognitivo del estudiante.
- Control del tiempo y secuencia del aprendizaje por el alumno.

2.2.2. Definición de Inteligencia Artificial

La Inteligencia Artificial (IA) es un campo de las ciencias computacionales que busca "que las computadoras hagan cosas que al momento las personas hacen mejor"[23]. La IA permite la representación del conocimiento junto a un mecanismo de inferencia, mediante el cual se obtienen conclusiones después de un proceso de razonamiento o deducción, permitiendo a la computadora la capacidad de aparentar un raciocinio al momento de resolver problemas.

La IA ofrece técnicas para enfrentar dos clases de problemas:

- Los que no poseen un algoritmo conocido para su resolución.
- Los que por su tamaño o dimensión, son inaplicables métodos o algoritmos conocidos para su solución.

2.2.3. División de la Inteligencia Artificial

Se podría decir que la IA se divide en 2 escuelas de pensamiento:

- Inteligencia Artificial Convencional

[23] BELLO, R. Aplicaciones de la Inteligencia Artificial, Universidad de Guadalajara. 2002

- Inteligencia Artificial Computacional

Inteligencia Artificial Convencional

La IA *Convencional* está basada sobre todo en el análisis formal y estadístico del comportamiento humano cuando este se encuentra en diferentes escenarios[24].

De esto, se puede dividir la IA Convencional en muchas formas o aspectos, siendo algunas:

- **Razonamiento Basado en Casos**: Este permite de una manera más fácil tomar decisiones resolviendo problemas concretos que van surgiendo en una actividad.
- **Sistemas Expertos**: Estos deducen soluciones basados en un conocimiento previo del ambiente o contexto en el que el sistema se aplica, además de conocer las reglas y/o relaciones que existen en dicho contexto.
- **Redes Bayesianas**: Son soluciones que surgen mediante la deducción de estadísticas que existen en un problema presente.
- IA Basada en Comportamientos: Son sistemas complejos que tiene autonomía y son auto-regulables y pueden evolucionar de acuerdo a lo necesario del caso.

Inteligencia Artificial Computacional

Es también conocida como IA Sub-simbólica. Es la que implica un desarrollo o aprendizaje interactivo; esto quiere decir, que el sistema es capaz de aprender si

[24] Elaine Rich y Knight Kevin. Inteligencia Artificial. Segunda Edición. McGraw Hill: México, 1994.

existen modificaciones de alguno de los parámetros que posee dicho sistema[25]. Este aprendizaje se realiza gracias a los datos empíricos que tiene de base (Rich y Kevin, 1994). Alguno de los métodos que posee la IA Computacional son:

- **Máquinas de Vectores Soporte.** Sistemas que permiten reconocimiento de patrones genéricos de gran potencia.
- **Redes Neuronales.** Sistemas con grandes capacidades de patrones.
- **Modelos ocultos de Markov.** Aprendizaje basado en dependencia temporal de eventos probabilísticos.
- **Sistemas Difusos.** Técnicas para lograr el razonamiento bajo incertidumbre.
- **Computación Evolutiva.** Aplica conceptos inspirados en la biología, como por ejemplo, la población, mutación y supervivencia del más apto para generar soluciones sucesivamente mejores para un problema. Estos métodos a si vez se dividen en *Algoritmos Evolutivos* (un ejemplo de esto es el algoritmos genéticos) e *Inteligencia Colectiva* (un ejemplo d este es el algoritmo de la hormiga).

2.2.4. Técnicas de Inteligencia Artificial

Se podría clasificar a las técnicas de IA en dos tipos de categorías: *las técnicas básicas* y las *tecnológicas*[26]*.*

[25] Elaine Rich y Knight Kevin. Inteligencia Artificial. Segunda Edición. McGraw Hill: México, 1994.
[26] Stuart Rusell y Norving Meter. Inteligencia Artificial: Un Enfoque Moderno. Prentice Hall: México, 1996

Técnicas Básicas

Se las conoce así porque son la base en diversas aplicaciones de IA. Estas pueden ser (Rich y Kevin, 1994):

- Búsqueda Heurística de Soluciones
- Representación del Conocimiento
- Deducción Automática
- Programación Simbólica
- Redes Neuronales

Técnicas Tecnológicas

Se puede decir que estas técnicas son la combinación de varias técnicas básicas, y suelen estar orientadas a resolver familia de problemas (Rich y Kevin, 1994). Suelen estar mucho más cerca de las aplicaciones finales. Las siguientes son solo algunas de estas técnicas:

- Robótica
- Visión
- Lenguaje Natural
- Sistemas Expertos

2.2.5. Inteligencia Artificial en el Proceso Enseñanza – Aprendizaje

La educación es una de entre las múltiples áreas de aplicación de la IA. Si se plantea su uso en el desarrollo de software educativo, la IA permite al desarrollador:

- La mejor adaptación de las características de los estudiantes, teniendo en cuenta el historial de actuaciones de los alumnos y no a una respuesta aislada[27].
- La generación de problemas, soluciones y diagnósticos del cómo y cuándo se necesite una sesión de aprendizaje[28].
- Resolver problemas complicados de manera que su forma de operar sirva de guía para el alumno[29].
- Organizar el saber disponible sobre alguna materia, posibilitando su aplicación directa a la solución del problema.
- Preservar el conocimiento para su utilización futura.
- Captar y presentar en diferentes formas las respuestas que recibe o proporciona.
- Reconocer una extensa gama de errores de razonamiento.
- Proveer conjuntos de problemas distintos y graduar su dificultad relativa.

El software educativo requiere de grupos multidisciplinarios donde intervengan al menos educadores y especialistas en computación[30]. Estos programas abarcan finalidades muy diversas que pueden ir desde la adquisición de conceptos al desarrollo de destrezas básicas, o la resolución de problemas. Existen

[27] Maikel León Espinosa, Zenaida García Valdivia, "La Inteligencia Artificial en la Informática Educativa", Revista de Informática Educativa y Medios Audiovisuales Vol. 5(10), págs. 11-18. 2008
[28] Ídem
[29] LENAT, D. Building Large KnowledgeBased Systems. Addison-Wesley 1990
[30] SHNEIDERMAN, B. Diseño de interfaces de usuario. Estrategias para una interacción persona-computadora efectiva. México: Addison Wesley. 2006

diferentes tipos de software educativos que pueden utilizarse en el proceso de enseñanza-aprendizaje entre los que se encuentran:

- Programas de ejercitación.
- Tutoriales.
- Sistemas expertos[31].
- Programas de demostración.
- Simuladores.
- Repasadores.
- Juegos.
- Sistemas de aplicación.

El relativo y creciente uso de la computación en la Educación está más relacionado con el impacto que la informática ha tenido en el mundo moderno y continuará teniendo. Hay que ver a la computadora como un medio complementario a otros a que puede utilizar el profesor, pero este medio debe superar las limitaciones de los medios educativos convencionales, enfrentando el reto que le imponen los últimos avances tecnológicos.

[31] DURKIN, J. Expert Systems. Design and Development. Prentice Hall Inter-national. 1994

2.2.6. Sistemas Expertos

Kastner define a un sistema experto como “un programa de computadora que resuelve problemas que requieren experiencia humana, mediante el uso de representación del conocimiento y procedimiento de decisión”[32].

Son sistemas diseñados para actuar como un experto humano en un dominio o área de conocimiento particular[33]. Son parte de los sistemas basados en el conocimiento (SBC), en los cuales aparecen representados los conocimientos de un dominio determinado, de tal forma que dicha representación sea procesable por un programador informático (Césari, 2012). La finalidad de un SE es la resolución de problemas del dominio para el que ha sido creado, aplicando técnicas de razonamiento sobre el conocimiento que alberga su base de conocimiento.

Estos sistemas son capaces de crear máquinas que actúan como seres humanos, capaces de hacer algo parecido a *razonar* y *pensar*. “Actúan como un especialista humano en un dominio particular o área de conocimiento” (Césari, 2012). Para hacer esto, un experto humano transmite la información que posee al sistema, y el usuario lo utiliza para encontrar soluciones a problemas del área con la misma eficacia del especialista, tal vez incluso más eficiente. Así mismo, el sistema puede transmitir el conocimiento al usuario, como si fuera un maestro.

La principal característica de un sistema experto es que es capaz de diferenciar o “separar los conocimientos almacenados (base de conocimiento) del

[32] Kastner, J. y Hong S. “*A review of expert systems*”. European journal of operational research, p 285 – 292. 1984
[33] Césari, Matilde. *Sistemas Experto*. Facultad Regional de Mendoza, Laboratorio Dharma, Argentina, p 1 – 19. 2012

programa que los controla (motor de inferencia)"[34]. Los datos propios de un determinado problema se almacenan en una base distinta (base de hechos).

El éxito de un sistema experto se encuentra en el conocimiento sobre el tema que trata y la capacidad que tiene para aprender.

Estructura básica de un Sistema Experto

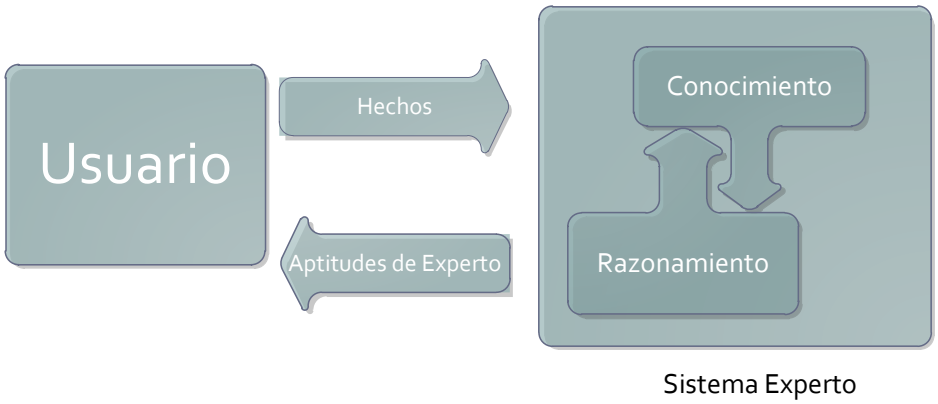


Figura 2.3 Esquema básico de un SE

En la figura 2.3 se puede apreciar que los SE pueden dividirse en 2 módulos: el de conocimiento y el de razonamiento[35]. El módulo de conocimiento está conformado en base a reglas. Estas son dadas por la persona experta relacionada con el sistema. Mientras tanto, el módulo de razonamiento es constituido por los mecanismos de inferencia que permite generar conclusiones a partir de un conjunto de reglas del módulo de conocimiento. Este sistema debe refinarse de una manera iterativa, hasta conseguir un resultado deseable.

[34] Césari, Matilde. *Sistemas Experto*. Facultad Regional de Mendoza, Laboratorio Dharma, p 1 – 19. 2012
[35] Césari, Matilde. *Sistemas Experto*. Facultad Regional de Mendoza, Laboratorio Dharma, Argentina, p 1 – 19. 2012

Aplicaciones de Sistemas Expertos

Los SE tienen una gran variedad de aplicaciones en la actualidad, siendo solamente algunos:

- Transacciones bancarias
- Control de tráfico
- Problemas de planificación
- Diagnósticos médicos

Ventajas de los Sistemas Expertos

Al usar este tipo de sistemas, se debe tomar en consideración un análisis de factibilidad y de coste-beneficio para la empresa o negocio en el que se desea aplicar un SE. El desarrollo y la adquisición de uno es normalmente caro, pero el mantenimiento y su costo a largo plazo es relativamente bajo. De todas maneras, si se aplica un SE bien planificado, existirá una alta ganancia en términos de tiempo, dinero y precisión.

Clasificación de Sistemas Expertos

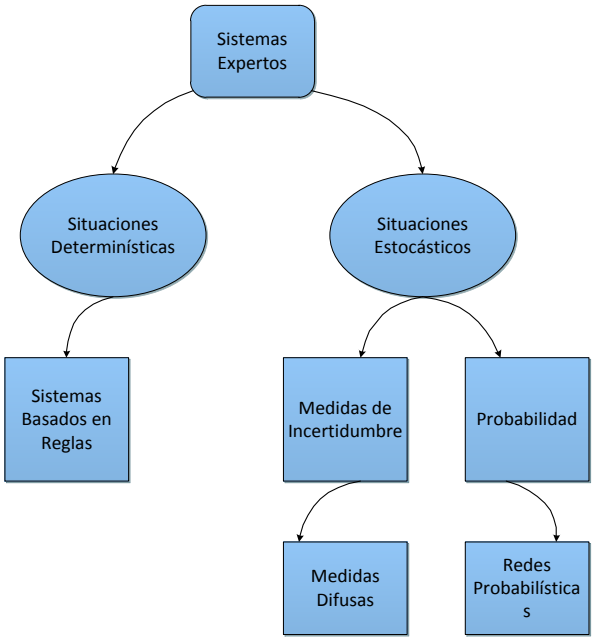


Figura 2.4 Tipos de Sistemas Expertos

En la figura 2.4, se puede apreciar que los sistemas expertos se pueden clasificar en 2: determinísticos y estocásticos (sistemas inciertos).

Los **sistemas deterministas** se definen por estados (el estado actual hacia un estado anterior y las acciones sobre el entorno). Son sistemas expertos basados en reglas predefinidas, y que usan un mecanismo de razonamiento lógico para sacar sus conclusiones[36]. Pueden ser formulados usando un conjunto de reglas de objetos bien definidos.

[36] César, Matilde. *Sistemas Experto*. Facultad Regional de Mendoza, Laboratorio Dharma, Argentina, p 1 – 19. 2012

Los **sistemas estocásticos** son en los que existe una incertidumbre, por lo que necesita ser tratada con probabilidades y/o estrategias de razonamiento (también conocido como *razonamiento probabilístico*)[37]. Aquí es necesario introducir algunos medios para tratar la incertidumbre. Podría tener también un sistema de reglas predefinidas, pero también introducen una medida asociada a la incertidumbre de las reglas y a la de sus premisas. Para ello se usan *fórmulas de propagación para calcular la incertidumbre asociada a las conclusiones* (Césari, 2012). En los últimos años ha ido aumentado las medidas de incertidumbre, como los factores de certeza, la lógica difusa, etc.

Otra medida intuitiva de incertidumbre es la probabilidad, en la que la distribución conjunta de un conjunto de variables se usa para describir las relaciones de dependencia entre ellas, y se sacan conclusiones usando fórmulas muy conocidas por la teoría de la probabilidad (Césari, 2012). Estos sistemas que usan la probabilidad como medida de incertidumbre, se conocen como *sistemas experto probabilísticos*, y la estrategia de razonamiento que usan se conoce como *razonamiento probabilístico* o *inferencia probabilística*.

Los anteriores modelos se incluyen en las redes Bayesianas y de Markov (Césari, 2012), y se basan en una representación gráfica de las relaciones entre las variables.

Como conclusión, se puede afirmar que para solucionar problemas que posean los dos tipos de sistemas anteriores, aunque es posible la existencia de una

[37] Césari, Matilde. *Sistemas Experto*. Facultad Regional de Mendoza, Laboratorio Dharma, Argentina, p 1 – 19. 2012

o más soluciones para los mismos, “la posible solución no está previamente fijada” (Césari, 2012).

2.2.7. Tipos de Sistemas Expertos

Se podría clasificar a los tipos de sistemas expertos de acuerdo a la función que realizan[38]:

- Interpretación
- Predicción
- Diagnóstico
- Diseño
- Plantación
- Monitoreo
- Depuración
- Reparación
- Instrucción
- Control
- Enseñanza

[38] Césari, Matilde. *Sistemas Experto*. Facultad Regional de Mendoza, Laboratorio Dharma, Argentina, p 1 – 19. 2012

2.2.8. Técnicas de Razonamiento Hacia Adelante y Hacia Atrás

Las técnicas de razonamiento (o encadenamiento) hacia adelante y hacia atrás son parte de los sistemas expertos basados en reglas (SEBR)[39]. Se los clasifica dentro de los procesos de razonamiento (una progresión de un conjunto de datos de partida hacia una solución o conclusión de un problema predefinido y determinista). Se caracterizan por el manejo de árboles de decisión o listas, y de permitir obtener un nuevo conocimiento a partir de algo ya existente, por lo mismo se los utiliza en procesos de razonamiento[40].

2.2.8.1. Encadenamiento Hacia Adelante

Parte de pocos datos y/o muchas posibles conclusiones, tiene la característica de ser poco específico (dispara "todas las reglas" posibles) y de ser capaz de crear sistemas expertos (Universidad de Oviedo, 2012).

Esta técnica consiste en construir un árbol de secuencias de movimientos que se pueden presentar como soluciones, empezando por las configuraciones iniciales en la raíz del árbol[41]. Se generará el siguiente nivel del árbol encontrando todas las reglas cuyos lados izquierdos se relacionen con el nodo raíz y que utilicen sus lados derechos para crear nuevas configuraciones.

---

[39] Universidad de Oviedo, "Sistemas Inteligentes", Sistemas Basados en reglas, 2012

[40] El proceso de razonamiento es una progresión de un conjunto de datos de partida hacia una solución o conclusión.

[41] Árbol de decisión es una técnica que permite analizar decisiones secuenciales basadas en el uso de resultados y probabilidades asociadas. Se puede usar para generar sistemas expertos, búsquedas binarias y árboles de juegos.

La dirección del razonamiento no tiene nada que ver con la dirección en que se ejecuta una regla. Siempre se disparan “hacia adelante”, lo que quiere decir que se da por hecho o se ejecuta el consecuente cuando se confirma el antecedente[42].

Es un método muy útil cuando los datos iniciales son pocos, y/o cuando existen muchas posibles conclusiones.

¿Cómo usar el razonamiento hacia adelante?

Para realizar un encadenamiento hacia adelante, se debe seguir los siguientes pasos:

1. **Matching. –** O coincidencia en inglés. Es una búsqueda de las reglas para las que es cierto su antecedente; esto quiere decir que el antecedente debe satisfacer todas las reglas para que sea tomado en cuenta.
2. **Resolución de Conflictos. –** Es una selección de entre las reglas satisfechas para que sea la única que se ejecute. Para poder elegir a una de las reglas, se debe tomar en cuenta algunos *criterios de selección*, o también llamados estrategia de búsqueda; las cuales se listan como:
    - **a.** ¿Cuál es el mayor número de premisas en el antecedente?
    - **b.** ¿Cuál es la regla con la mayor prioridad?
    - **c.** Si la búsqueda es hecha a profundidad.
    - **d.** Si la búsqueda es hecha por anchura.

[42] Antecedente: conjunciones de atributos de un mismo dominio. Consecuente: atributos que pasarán a ser conocidos para el sistema. Universidad de Oviedo, 2012

3. **Ejecución. –** Es el momento en que dispara la regla seleccionada, por la que se amplía los datos conocidos.

Para tener una mejor comprensión de esto, se puede pensar en las posiciones que posee cada ficha del 15-puzzle.

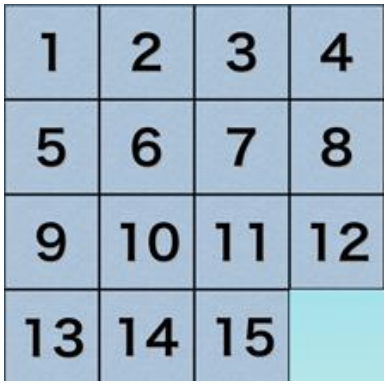


Figura 2.5 Posiciones de cada ficha en el 15-Puzzle

Como se vio en el punto 2.1.7, el 15-puzzle tiene 15 fichas pegadas en pares de 4, formando un “tablero” de 4x4 (4 espacios horizontales por 4 verticales más un espacio vacío) como se nota en la figura 2.5. Cada espacio está posicionado desde el 1 hasta el 15, excepto el espacio vacío, al que podría llamar el espacio “16”. La posición 1 tiene a sus dos lados la posición 2 y la posición 5; a su vez la posición 2 tiene a sus tres lados la posición 1, la posición 6 y la posición 3; la posición 6 a su vez tiene a sus cuatro lados la posición 2, 5, 7 y 10. Con las anteriores posiciones, se podría crear un árbol de decisión, el cual tendría la forma que se muestra en la figura 2.6.

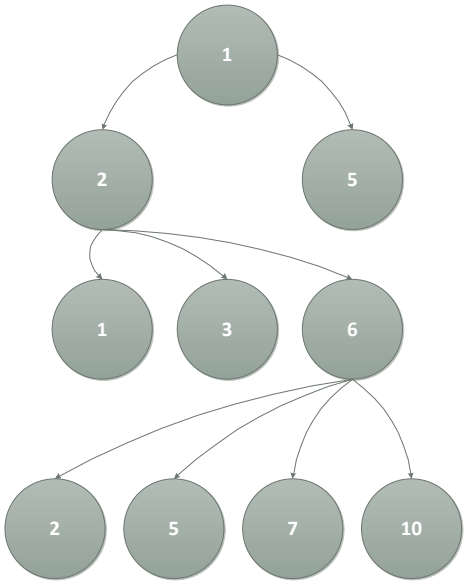


Figura 2.6 Árbol de decisión de movimientos de las posiciones 1, 2 y 6 de un 15-Puzzle

Esto se traduce que si se quiere mover ficha que ahora se encuentra en la posición 1, esta solamente se pueden mover hacia las posiciones 2 y 5. Si se toma la decisión de llevarla hacia la posición 2, esta se podrá mover solamente hacia las posiciones 1, 3 y 6. Y finalmente, si se toma le decisión de mover la ficha a la posición 6, esta solamente podrá moverse a las posiciones 2, 5, 7 y 10.

2.2.8.2. Encadenamiento Hacia Atrás

Esta, a diferencia del encadenamiento hacia adelante, se basa en mucha información que está disponible, pero poca es relevante para su desarrollo (Universidad de Oviedo, 2012); se podría poner como por ejemplo, una visita al médico, donde éste hace muchas preguntas, pero de todas ellas, la información que sirve para obtener un diagnostico válido. Es mucho más específico y generalmente más eficaz que un razonamiento hacia adelante.

Esta técnica consiste en construir un árbol de secuencias de movimientos que ofrezcan soluciones empezando con las configuraciones objetivo en la raíz del árbol. Se generará el siguiente nivel del árbol encontrando todas las reglas cuyos lados derechos estén ligados con el nodo raíz. Éstas serán todas las reglas que, si son las únicas que se aplican, generarán el estado que se desea. Se utilizará el lado izquierdo de las reglas para generar los nodos en este segundo nivel del árbol.

Cuando se habla de razonamiento hacia atrás, se está hablando solamente al proceso de búsqueda y selección de reglas.

Es un método muy útil en aplicaciones con muchos datos disponibles de partida, de los que solo una pequeña parte son relevantes. Si este se encuentra en un sistema interactivo, solo se debe preguntar lo estrictamente necesario, a diferencia del encadenamiento hacia adelante que no pregunta nada.

¿Cómo usar el razonamiento hacia atrás?

Para poder usar este tipo de razonamiento, se puede seguir los siguientes pasos (Universidad de Oviedo, 2012):

1. Se debe formar una **pila inicial** compuesta por todos los objetivos iniciales.
2. Se toma el primer objetivo de la pila. De acuerdo a este, se debe localizar todas las reglas que lo satisfagan.
3. Se debe examinar las premisas de dichas reglas, en orden:
    a. **Si todas las premisas se satisfacen.** Se ejecutan las reglas y se derivan sus conclusiones. Si se derivó un valor para el objetivo actual, entonces se elimina de la pila y se vuelve al paso 2.

b. **Si una premisa de una regla no se satisface** o tiene un valor desconocido en la base de conocimientos, se debe buscar si existen reglas que concluyan un valor para ella. Si existe, se inserta en el tope de la pila de objetivos y se vuelve al paso 2.

c. **Si no se encontró ninguna regla que concluya un valor para la premisa actual.** Entonces se pregunta al usuario por dicho valor y se añade a la base de conocimientos. Si el valor satisface la premisa actual, se continúa examinando el resto del antecedente; si no, se considera la siguiente regla que concluya un valor para el objetivo actual.

4. Si se han examinado todas las reglas que concluyen un valor para el objetivo actual y todas fallaron, entonces se marca el objetivo como indeterminado, se extra de la pila y se vuelve al paso 2, si la pila está vacía, el proceso finaliza.

Para poder comprender mejor esto, se podría volver al ejemplo sobre el 15-puzzle del punto 2.2.9.1. Si el rompecabezas está armado, para desarmarlo solo se podría mover la ficha 15 o bien la ficha 12, como se muestra en la figura 2.7.

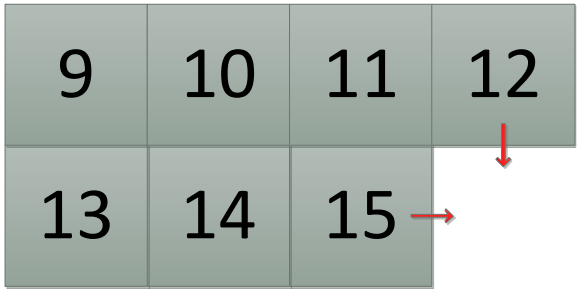


Figura 2.7 Movimientos posibles en un 15-puzzle basados en el razonamiento hacia adelante

Suponiendo que se ha movido la ficha 15, luego la 14 (otra posible por espacio vacío a parte de la 11 y la misma 15), y luego la ficha 10 (así mismo la única posible a parte de la 13 y la misma 14), el rompecabezas quedará como en la figura 2.8.

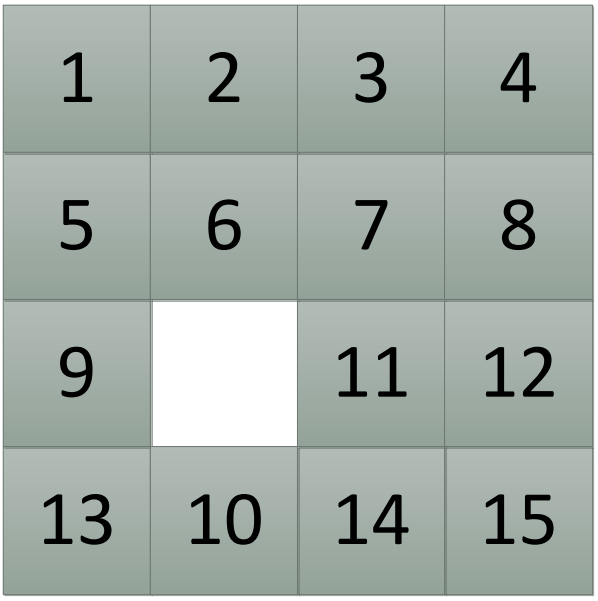


Figura 2.8 15-puzzle después de moverse la ficha 15, 14 y 10 de su estado inicial

Con solo estos 3 movimientos de fichas, se puede considerar que el rompecabezas esta desarmado. La lista de movimientos hecha es {15, 14, 10}, y el espacio vacío es el 10[43]. Teniendo estos datos la lista de movimientos hechos para desarmar el rompecabezas (la pila inicial), y como reglas que todas las fichas coincidan con su posición (regla 1) y que el espacio vacío quede en la posición 16

[43] Se debe recalcar que el valor de una ficha no es la misma que la posición de la misma. El juego de n-puzzle solo termina cuando el valor de todas las fichas coinciden con el valor de la posición.

(regla 2), entonces se puede armar el rompecabezas invirtiendo la lista (encadenamiento hacia atrás) y moviendo las fichas de acuerdo a la nueva lista (cuyo valor ahora es {10, 14, 15}) y de esa manera se llega al objetivo inicial del juego.

Se puede observar que los razonamientos hacia atrás y hacia adelante son complementarios entre sí, pero solamente bajo ciertas condiciones.

2.2.8.3. Combinación de los razonamientos hacia adelante y hacia atrás

Como se pudo apreciar en el ejemplo del punto 2.2.9.2, al momento de desarmar el rompecabezas, se utilizó el razonamiento hacia adelante, mientras que para armarlo, se utilizó el razonamiento hacia atrás. La técnica de combinación de ambos razonamientos consiste en utilizar las mismas reglas tanto para el razonamiento hacia delante, como para el razonamiento hacia atrás.

También depende de la propia forma de las reglas. Si las partes derechas como las izquierdas contienen aserciones puras, entonces el encadenamiento hacia delante puede emparejar las aserciones en el lado izquierdo de una regla y añadir a la descripción del estado las aserciones del lado derecho. Pero si se permiten los procedimientos arbitrarios como partes derechas de las reglas, entonces las reglas no serán reversibles.

2.3. WIIMOTE

El número de Wiimote, que no es más que el control remoto de la consola de juegos Wii, de la empresa Nintendo excedió en el año 2008 en más de la mitad de la cantidad de tabletas vendidas hasta esa fecha[44]. Debido a esto, este dispositivo se convirtió en uno de los más rápidamente difundidos en la historia de la electrónica. Todo esto debido a sus variados componentes revolucionarios.

Considerando el hecho que Nintendo ofrece un API de conexión con la plataforma Windows, los cuales permiten el acceso a los estados internos del Wiimote, a través de un módulo de conexión con una interfaz Bluetooth, permite a los desarrolladores hacer experimentos interesantes con la interfaz HCI[45].

A parte de la tecnología Bluetooth que posee un Wiimote, también viene equipado con u una cámara sensor infrarroja. Esta permite otro tipo de procesos interesantes de conexión y captación de movimientos apuntados hacia una pantalla que emane alguna señal infrarroja.

John Chung Lee experimento con este dispositivo, generando una gran cantidad de recursos y experimentos a seguir[46], todos ellos enfocados a una conexión y mejor manejo de un Wiimote en un sistema operativo tipo Windows. Como por ejemplo se tiene el lápiz óptico, capaz de enviar señales al control, el cual interpreta (con un software específico) las señales como si fueran clics encima de

---

[44] Selver Softic, "*Using Nintendo Wii Remote Control from Finger Tracking, Gesture Detection and as HCI Device*", Instituto de Sistemas de Información y Medios Computacionales de la Universidad Tecnológica de Graz, Austria, 2008

[45] Del inglés *Host Controller Interface* que permite la conexión de dispositivos con una interfaz USB.

[46] J.C. Lee. Wii project site. http://www.cs.cmu.edu/~johnny/projects/wii/, 2008

una pantalla o proyección; o el detector de gestos como un módulo a varios aplicativos iterativos. Todas estas investigaciones se han enfocado, casi exclusivamente en la educación y capacitación e-learning[47].

2.3.1. Descripción

Figura 2.9 Wiimote

El control remoto de Wii (en japonés Wiiリモコン, romanizado Ui Rimokon[48]), o mejor conocido como Wiimote, es un dispositivo creado por la Empresa Nintendo para la consola de videojuegos Wii. Ha creado un gran impacto, pero sobre todo, es una de las más importantes innovaciones tecnológicas que se han hecho dentro del área de videojuegos que se desarrollan en la 3ra Dimensión. Esto se fundamenta en que no solamente es un dispositivo que actúa como un gamepad[49], sino que también permite una accesibilidad a nivel de movimiento 3D. En otras palabras, es

[47] Selver Softic, "Using Nintendo Wii Remote Control from Finger Tracking, Gesture Detection and as HCI Device", Instituto de Sistemas de Información y Medios Computacionales de la Universidad Tecnológica de Graz, Austria, 2008
[48] Referenciado en la página web http://www.nintendo.co.jp/wii/index.html
[49] Un *gamepad* (control de mando) es un dispositivo de entrada en las consolas de juego.

capaz de sentir el movimiento espacial que haga un jugador con el control, y lo transmite al juego. Esto permite a usuarios que no usan o no juegan mucho en consolas de juego (Figura 2.10).

Figura 2.10 Usuarios usando el Wiimote

Al tener esta capacidad, el Wiimote se convierte en un instrumento para mejorar la investigación a personas que no poseen los recursos necesarios para hacer estudios con o sobre este tipo de sensores.

2.3.2. Historia

El desarrollo de control empezó en el año 2001, y fue mostrado en el Espectáculo de Juegos de Tokio en septiembre del año 2005[50]. Aún ahora, cuando

[50] Referenciando de la página web http://en.wikipedia.org/wiki/Wiimote

en el mercado la consola Wii ya tiene un sucesor (el Wii U), el control tiene soporte y es aceptado por la nueva consola.

2.3.3. Reseña Tecnológica

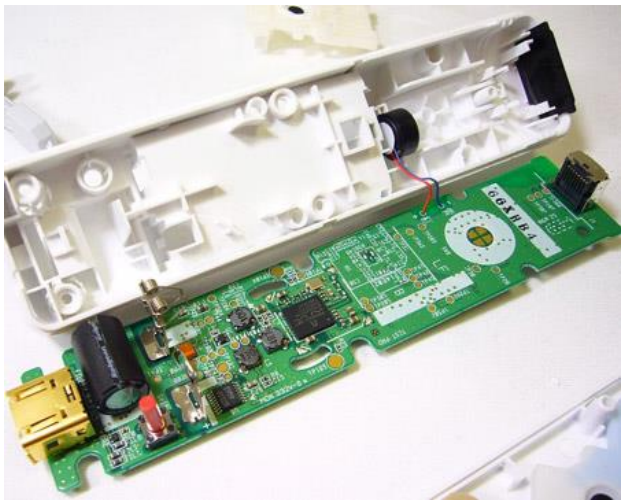


Figura 2.11 El interior de un control remoto de Wii

La estructura del control (figura 2.11) está construida alrededor del chip de Bluetooth Broadcom BCM2042 (Figura 2.12), y contiene múltiples periféricos que permiten acceso de datos hacia el control, además de contener puertos de expansión para agregados externos[51].

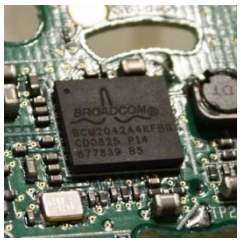


Figura 2.12 El Chip Broadcom BCM2042, la base de construcción de un Wiimote

[51] Referenciado de la página http://wiibrew.org/wiki/Wiimote

En el siguiente cuadro se puede apreciar las características que posee un control estándar Wiimote[52]:

Tabla 2.2: Características de Wiimote

Característica	Estado	Descripción
Comunicación Bluetooth	Perfecto	Conexión con Wiimote y en escucha
Botones Principales	Perfecto	Todos funcionales
Acelerómetro	Perfecto	Todos funcionales
Cámara tipo IR	Perfecto	Todos funcionales
Botón de Encendido	Perfecto	Todos funcionales
Speaker	Perfecto	Todos funcionales
LED de jugador	Perfecto	Todos funcionales
Información de estado	Perfecto	Batería e información de Estado
Controles con Extensión	Usable	Extensiones de 3ros no se entiende

[52] Referenciado de la página http://wiibrew.org/wiki/Wiimote

Cámara Infrarroja

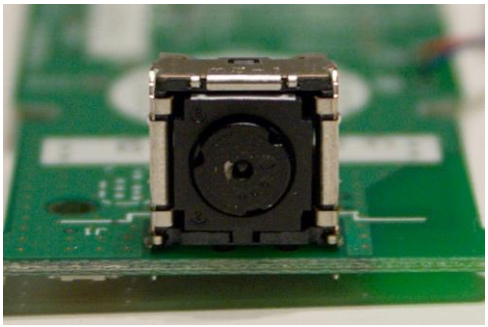


Figura 2.13 Cámara IR de Wiimote

La cámara monocromática contiene un filtro infrarrojo (IR), como se puede apreciar en la figura 2.13, con un procesador capaz de sentir hasta 4 objetos en movimiento a una frecuencia de 100 Hz. Esto convierte al Wiimote en un dispositivo muy sensible para proyecciones en el plano infrarrojo[53].

Acelerómetro

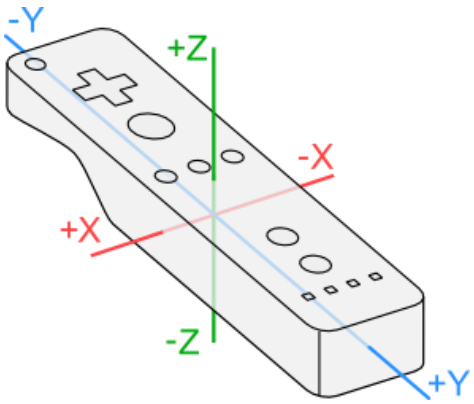


Figura 2.14 El sensor de movimiento detecta cualquier cambio de posición del control en los 3 ejes (x, y, z)

[53] Selver Softic, "Using Nintendo Wii Remote Control from Finger Tracking, Gesture Detection and as HCI Device", Instituto de Sistemas de Información y Medios Computacionales de la Universidad Tecnológica de Graz, Austria, 2008

Así mismo, el control posee un acelerómetro de 8 bits, capaz de sentir los movimientos en los 3 ejes de coordenadas, conocido como ADXL330 (figura 2.15). Este acelerómetro mide con casi exactitud y a una velocidad 3G en las tres direcciones. Esta idea fue concebida gracias a la observación de las bolsas de aire (airbags) de los automóviles, y demuestra en una de las mejores maneras el uso de MEMS (Sistemas electrónicos micro-mecánicos, por sus siglas en inglés)[54] [55].

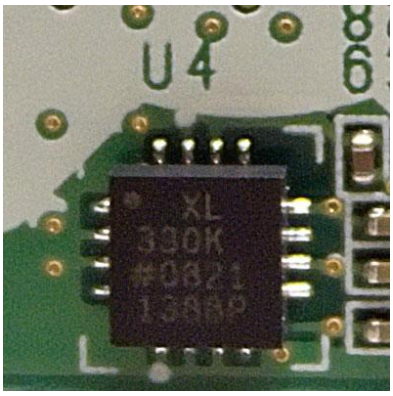


Figura 2.15 Acelerómetro ADLX330

Comunicaciones del Control

Las comunicaciones Wiimote se hacen por medio de un protocolo Bluetooth estándar. El dispositivo puede ponerse en modo de detección por 20 segundos, solamente con presionar el botón de sincronización, el cual se encuentra en la parte posterior, oculto por el protector de la batería (figura 2.18). Presionando el botón 1 o 2 continuamente provocará que el control se mantenga en modo de detección sin apagarse (lo anterior no pasa con el botón de sincronización). Cuando se activa este

[54] WiiLi Wiki. Wii Linux - Wiimote. http://www.wiili.org/index.php/Wiimote, 2008.
[55] J. C. Lee. Hacking the nintendo wii remote. Pervasive Computing, IEEE, (3):3945, 2009.

modo, los LED de jugador (y dependiendo del nivel de la batería) comenzarán a parpadear.

El control usa el protocolo estándar Bluetooth HID para comunicarse con la consola host, este protocolo se basa directamente con el estándar USB HID. En otras palabras, el dispositivo Wiimote puede conectarse como cualquier otro en una consola cualquiera que maneje Bluetooth. Sin embargo, este dispositivo no usa los tipos de datos estándar ni el HID que lo describe, solamente describe el ancho de un formato de reporte, dejando el contenido como tal indefinido, con lo que se hace innecesario el uso de controladores (drives) del estándar HID, aunque no está por más decir que existen controladores tipo Wiimote en el mercado. El control Wiimote actualmente usa un conjunto complejo de operaciones, que se transmiten por puertos de salida tipo HID, y retorna un número diferente de paquete de datos a través de puertos de entrada, los cuales contienen los datos de los periféricos.

Botones

Figura 2.16 Botones de adelante y atrás

El Wiimote posee 11 botones al frente y 1 atrás tipo disparador. De todos estos, el control de poder (Power) es especial y es tratado diferente que al resto; esto se debe a que si se presiona, el control intentará prender la consola Wii y también intentará sincronizarlo. Lamentablemente, el mecanismo para esto no ha sido revelado por Nintendo.

Los principales botones se pueden apreciar en la lista:

Tabla 2.3. Botones de Wiimote

Botón	Símbolo
D-pad Up, Down, Left Right	
A	
B	
1	
2	
Plus	
Minus	
Home	
Power	

Botón Sync

Además de estos botones, existe el botón de sincronización conocido como Sync. Este se encuentra oculto bajo el protector de la batería (figura 2.18).

Figura 2.17 Botón “Sync”

Cuando este botón es presionado, el Wiimote se desconecta de cualquier otro dispositivo que esté conectado en ese momento, volviéndose “detectable” para otros dispositivos Bluetooth, aceptando cualquier emparejamiento o petición de conexión por los próximos 20 segundos (esto sin importar por cuánto tiempo se mantuvo presionado el botón).

Una vez que el Wiimote se encuentre sincronizado con un dispositivo, cuando cualquier otro botón es presionado, automáticamente buscará a ese dispositivo e intentará conectarse con él.

Otras Características

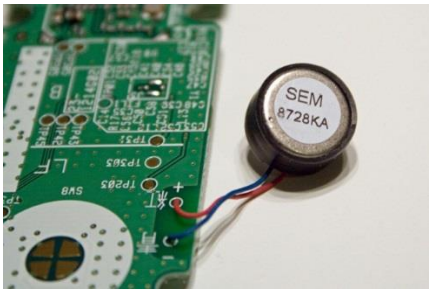


Figura 2.18 Dispositivo de Vibración

Además de las anteriores características, el Wiimote posee un traductor de audio en el tablero, que puede transformar la voz humana en cadenas de datos[56]. Este micrófono/parlante es usado principalmente para reproducir pequeños efectos de sonidos enviados desde la consola durante el tiempo de juego.

Este dispositivo posee una memoria EEPROM[57], como se ve en la figura 2.20.

Figura 2.19 Memoria EEPROM de Wiimote

También tiene integrado 4 LEDS, que le permiten al control indicar cuanto de energía queda en las pilas, y en qué posición se encuentra el jugar para jugar su

[56] Referenciando de http://money.cnn.com/magazines/fortune/storysupplement/wiiremote/index.htm

[57] Una memoria EEPROM es un tipo no volátil de memoria usada en computadoras u otros dispositivos electrónicos, para almacenar pequeñas cantidades de datos, los cuales deben ser guardados cuando la fuente de energía es retirada.

turno. Otra característica, es su puerto de conexión de periféricos, el cual permite conectar otros dispositivos al control. Finalmente, el control tiene un dispositivo que genera vibraciones en el control, con este se puede enviar desde el juego al usuario, movimientos (como si hubiera sufrido un choque), haciendo más interactiva la experiencia del juego.

2.4. METODOLOGÍA DE DISEÑO DE HIPERMEDIA ORIENTADA A OBJETOS

Desarrollada por Daniel Schwabe y Gustavo Rossi en el año 1996, está básicamente basada en cuatro etapas:

- Diseño Conceptual
- Diseño Navegacional
- Diseño Abstracto de Interface
- Implementación

Cada una de las etapas implementa un diseño abstracto específico, los cuales introducen nuevos elementos y/o clases.

2.4.1. Características Fundamentales

- *Apropiada para un comportamiento complejo.*

  OOHDM provee las mejores herramientas para controlar el desarrollo de una aplicación que tiene un complejo comportamiento.

- *Separación del diseño con respecto al desarrollo.*

  Permite que la complejidad del desarrollo de software sea menor ya que ésta ocurre a diferentes niveles: “dominios de aplicación sofisticados (financieros,

médicos, geográficos, etc.); la necesidad de proveer acceso de navegación simple a grandes cantidades de datos, y por último la aparición de nuevos dispositivos para los cuales se deben construir interfaces *Web* fáciles de usar".[58]

2.4.2. Etapas de OOHDM

La tabla 2.3 muestra las etapas de OOHDM.

Tabla 2.4: Etapas de la Metodología OOHDM

Etapas	Productos	Formalismos	Mecanismos	Descripción
Obtención de Requerimientos	Casos de Uso (actores, escenarios)	Plantillas del formato del documento, Diagramas de Interacción de Usuario (UIDs)	Técnicas de Observación, Entrevistas	**Se crea un documento que describe actividades y requerimientos de los usuarios**
Diseño Conceptual	Clases, subsistemas, relaciones, atributos	Modelos Orientados a Objetos	Clasificación, agregación, generalización y especialización	**Se modela la semántica del dominio de la aplicación**
Diseño Navegacional	Nodos, enlaces, estructuras de acceso, contextos navegacionales, transformaciones de navegación	Vistas Orientadas a Objetos, Cartas de navegación orientadas a objetos, Clases de Contexto	Clasificación, agregación, generalización y especialización	**Se tiene en cuenta el perfil del usuario y las tareas. Se enfatiza en los aspectos cognitivos. Se crea la estructura de navegación de la aplicación**
Diseño de Interfaz Abstracta	Objetos de la interfaz abstracta, respuestas a eventos externos, transformaciones de la interfaz	Vistas Abstractas de Datos (ADV), Diagramas de Configuración, Cartas de navegación de los ADVs	Mapeado entre la navegación y los objetos visibles	**Se modelan los objetos visibles. Se describe la interfaz para los objetos de navegación. Se define el aspecto de los objetos de la interfaz**
Implementación	**Aplicación en funcionamiento**	**Los soportados por el entorno**	**Los que provea el entorno**	**Se realiza la puesta en producción del sistema**

[58]Tomado de: Darío Andrés Silva, Construyendo Aplicaciones Web con una Metodología de Diseño Orientado a Objetos, 2002, pp 2

2.4.2.1. Diseño Conceptual

Aquí se construye todo lo relacionado a los esquemas conceptuales, los cuales definen los objetos de dominio, las clases u objetos y las relaciones entre los mismos.

2.4.2.2. Diseño Navegacional

En esta segunda etapa, se define la estructura de navegación que tendrá la aplicación. Todo esto a través de un hiperdocumento, el cual se realiza por medio de modelos navegacionales, que representan diferentes vistas del esquema conceptual definidos en la etapa anterior.

Este diseño se expresa con un enfoque orientado a objetos, lo que viene a decir que se debe representar con modelos o esquemas:

- Esquema de las clases navegacionales. Estas deben mostrar las posibles vistas de un hiperdocumento. Las vistas se representan a su vez con unos tipos predefinidos de clases navegacionales conocidos como “nodos” o “enlaces”, además de otras clases que expresen estructuras o formas alternativas de acceso a nodos, como “índices” o “recorridos guiados”.
- Esquema de contexto navegacional. Estas permiten la estructuración de hiperespacio de navegación entre “subespacios”, en ellos se mostrará la información que se quiere mostrar al usuario, y los enlaces que estarán disponibles cuando se acceda a un objeto o nodo de un contexto determinado.

2.4.2.3. Diseño Abstracto de Interface

Esta fase está dedicada a la especificación de una interfaz abstracta. De esta manera, se puede especificar la forma en la que aparecen los contextos navegacionales.

Así mismo, se incluye en esta fase el modo en que los objetos de la interfaz activarán la interfaz, y el resto de las funcionalidades de la aplicación; esto quiere decir, se escribirán los objetos de interfaz y se los asociará con los objetos de navegación.

Esta separación entre diseño navegacional y de interfaz abstracta permite construir interfaces para el mismo modelo navegacional.

2.4.2.4. Implementación

En esta etapa, simplemente se implementa el hiperdocumento desarrollado en las anteriores etapas de la metodología (lo que quiere decir que se implementa los modelos navegacionales y de interfaz.

“Puzzlemote” será implementado con componentes de hipermedia para escritorio, el cual podrá correr en un sistema operativo Windows. OOHDM se aplicará en conjunto con Ingeniería de Software y Reingeniería, además de documentar el proyecto con UML.

2.4.3. Ventajas y Desventajas

Ventajas

- Clara identificación de los tres diferentes niveles de diseño en forma independiente de la implementación.
- Su forma de representación gráfica es bastante completa y permite representar en forma precisa elementos propios de las aplicaciones hipermedia, tales como nodos, anclas, vínculos, imágenes, estructuras de acceso y contextos.
- En la etapa de diseño navegacional se pueden crear enlaces entre nodos cualesquiera que permiten una verdadera interoperabilidad entre los mismos.
- El desarrollador puede entender y lograr en cada etapa lo que el usuario realmente necesita, gracias a que en el análisis y diseño, el usuario es parte fundamental en la validación del producto obtenido.
- Al generar una cantidad considerable de documentación a través de sus distintas etapas, permite llevar un control del desarrollo de las mismas y tener la posibilidad de realizar una rápida detección, corrección de errores y mantención.
- La utilización de UIDs permite representar en forma clara, rápida y precisa los casos de uso obtenidos.

Desventajas

- Requiere de cierto conocimiento e investigación para aprender la metodología, debido a los modelos que utiliza.

- El diseño pierde un poco de continuidad del modelo navegacional al diseño de interfaz, dado que se pasa a utilizar otro tipo de modelo.
- En ciertos casos OOHDM podría exagerar la cantidad de reglas y pasos (a veces complicados de seguir) para realizar distintos mapeos entre un diagrama y otro por lo cual el desarrollador podría perderse y olvidar detalles fundamentales a ser especificados.
- El diseño navegacional posee una gran cantidad de diagramas que muchas veces entregan información similar a la entregada por los UIDs y las ADVs.

2.4.4. Criterios de Selección de OOHDM

OOHDM es una metodología de diseño de hipermedia, que utiliza el enfoque orientado a objetos, extendiéndolo e integrándolo con técnicas de representación gráfica de relaciones entre objetos y de contextos navegacionales que son ricos en representación estructural y semántica.

Por ello, la metodología OOHDM ha sido escogida ya que reúne las características necesarias para no mezclar aspectos conceptuales (modelo del dominio) con presentación (construcción de la interfaz de usuario) gracias a que se encuentra basada en objetos para la creación de aplicaciones Web y analiza tanto el diseño como la implementación que inevitablemente influyen en todo el proceso de desarrollo.

Además establece que es tan importante el análisis de las tecnologías que pueden limitar la funcionalidad de la aplicación, como las decisiones de diseño equivocadas que pueden reducir la capacidad de extensión y reusabilidad.

2.5. VISUAL STUDIO Y C SHARP

2.5.1. Microsoft Visual Studio

Microsoft Visual Studio es un ambiente de integración de desarrollo (IDE por sus siglas en inglés) creado por la empresa Microsoft. Sus principales aplicaciones son el desarrollo de aplicativos de consola, como programas para usuarios con interfaces de gráficos (conocidos en ambiente de desarrollo como GUID), así como el uso de aplicaciones Windows Forms, sitios, aplicaciones y servicios web[59]. En este ambiente de desarrollo, se puede usar su código nativo junto a código de administración (lenguaje que maneja el sistema operativo), para mezclarlos con todas las plataformas desarrolladas para la familia de Microsoft Windows, como son: Windows Mobile, Windows CE, .NET Framework, Microsoft Silverlight, etc. (Selver Softic, 2008)

Las principales características que posee este ambiente son sin duda alguna los siguientes:

[59] Selver Softic, "Using Nintendo Wii Remote Control from Finger Tracking, Gesture Detection and as HCI Device", Instituto de Sistemas de Información y Medios Computacionales de la Universidad Tecnológica de Graz, Austria, 2008

- Un editor de código que soporta IntellisSense[60], refactorización de código e integración de un depurador para corrección de errores a nivel de código fuente y nivel de máquina.
- Herramientas para diseño de formularios tipo GUID en las aplicaciones.
- Diseñadores Web
- Diseñadores de esquemas de bases de datos.
- A pesar de tener soporte en un gran número de extensiones de archivos, también permite el ingreso de más plug-ins[61] genéricos y de terceros desarrolladores.

Así mismo, este IDE soporta varios lenguajes de programación, siendo los siguientes:

- C / C++
- VB.NET
- C# (C-Sharp)
- J#
- F#
- M
- Python
- Ruby
- XML/XSLT

[60] En programación computacional, el "*IntelligiSense*" se refiere a un ambiente de programación que permite el aumento de velocidad a la hora de codificar o crear código en las aplicaciones, reduciendo los errores de escritura, sintaxis, asignación de variables y otros errores comunes.
[61] Un *plug-in* o conector, es un pequeño programa que permite agregar una específica característica de software y/o de un dispositivo a otro software y/o dispositivo

- HTML/XHTML
- JavaScript
- CSS
- Linq

Microsoft también ha creado versiones “Express”, versiones que poseen solo uno de los componentes de ambiente, lo que viene a significar que no tienen costo, pero no tiene todas las características del IDE[62].

2.5.2. C Sharp

C Número o C# (C Sharp en inglés), proviene de la mezcla de varios lenguajes de programación, de entre ellos, los 3 más importantes son:

- C/C++
- Visual Basic
- Java

Como C++, es orientado a objetos. De Visual Basic aprovecha el poder del diseño gráfico para la creación fácil de interfaces de usuario. Como Java, C# compila a nivel de código de byte, el cual confía en los servicios que mantiene el Sistema Operativo en un tiempo de ejecución[63].

[62] Wikipedia. Microsoft Visual Studio. http://en.wikipedia.org/wiki/Visual_Studio, 2008.
[63] Simon, Robinson and Others, “*Professional C#*”, Wrox Press, 2001

C# es un lenguaje de programación orientado a objetos en multi-paradigmas, que contiene una escritura fuerte[64], imperativo, declarativo, funcional, procesal y genérico (Wikipedia, C#). Fue desarrollado por Microsoft en su iniciativa .NET en el año de 1999, el cual fue aprobado por el estándar ECMA (ECMA-334) e ISO (ISO/IEC 23270:2006). C# es uno de los lenguajes de programación diseñado a partir de la Infraestructura de Lenguaje Común[65].

2.5.2.1. Usos de C#

Los usos del C# varían en casi todas las posibilidades de soluciones que proporciona .NET. Algunas de las más importantes son:

- Aplicaciones ASP.NET
- Librería de Clases
- Servicios Web
- Aplicaciones de Escritorio
- Controles Específicos de Windows
- Aplicaciones de Consola

2.5.2.1. Características de C#

C# tiene todos los tipos de controles de flujos (if, else, etc.) y de bucles (while, for, etc.), además que soporte la declaración *for each*, el que permite la iteración de

[64] En programación computacional, la escritura *fuerte* y *débil* se utiliza coloquial y peyorativamente para clasificar a los lenguajes de programación.

[65] El CLI por sus siglas en inglés, es una especificación abierta creada por Microsoft y estandarizada por la ISO y ECMA que describe un código ejecutable dentro del framework .NET, el cual también puede ser utilizado en software libre e en implementaciones como el proyecto Mono y Portable.NET

cada elemento que se encuentre en una lista, arreglo, colección, u otro contenedor de objetos.

De la misma manera, C# facilita el uso de clases y objetos, al entrar de una manera más fácil y obvia al usuario a sus propiedades, atributos y métodos. El manejo de polimorfismo se encuentra en un gran nivel de detalle, pudiendo evitar las ambigüedades que suelen encontrarse en el C o C++. Comparado con Java, el cual no soporta sobrecarga de funciones, C# tiene la habilidad de especificar como las instancias de las clases se deben comportar cuando un operador del lenguaje es aplicado a esas instancias.

Las clases pueden parametrizar multitud de constructores, aunque C# no implementa destructores de la misma manera que C++. En vez de eso, las clases de C# finalizan los métodos que están realizando, los cuales se aplican cuando el recolector de basura que tiene el motor del Framework hace su aparición en tiempo de ejecución. Aun así, esto se presenta como una falencia a la hora de manejar los recursos que posee el sistema.

2.5.3. Conexión del Control Wimote por medio de C#

El cómo conectar el Wiimote y usarlo con la Librería de control en C#, se encuentra en el Anexo 1.

2.6. WINDOWS FORM Y LIBERÍA GRÁFICA GDI+

Visual Studio.NET es totalmente integrada a un ambiente de desarrollo de interfaces. Como se resaltó en el punto anterior, .NET está diseñado para que el proceso de escritura de código, control de errores y compilamiento sea mucho más sencillo para el programador[66]. Las características que permite esto son:

- Un editor de texto en el que se escribe código en C# (o también en VB.NET o C++), el que es sofisticado, permitiendo chequeos de sintaxis, comprobación de código que puede ser erróneo, ayuda para el uso de código y clases ya creadas, etc.
- Un editor de código de la vista del diseño, el que permite colocar y manejar visualmente las interfaces de usuario y el acceso a controles del proyecto. Cuando se hace esto .NET automáticamente agregará el necesario código en C# en los archivos fuentes, instanciando los controles en cada clase correspondiente a la interfaz. Todos los controles de interfaz son solamente instanciaciones de clases base particulares.
- Soporte a ventanas que permite ver y modificar aspectos del proyecto; en otras palabras, muestra visualmente las clases, propiedades y otros aspectos del código, de una manera más gráfica.
- Compilaciones dentro del mismo ambiente de desarrollo.
- Posee un depurador integrado, que facilita de manera gráfica el control y corrección de errores de código hecho.

[66] Simon, Robinson and Others, "*Professional C#*", Wrox Press, 2001

- Está integrado con una herramienta de rendimiento, el que permite medir cuánto dura cada método que tiene el código.
- Integrado a ayudas gráficas por parte de MSDN[67].
- Permite el acceso a otros programas que permiten configuraciones del Sistema Operativo.

Todo esto permite a un usuario Desarrollador el mejor manejo de la interfaz, y simplificar de manera extraordinaria la creación de las interfaces de usuario en su proyecto de desarrollo.

2.6.1. Clase Form

Cualquier ventana en una aplicación de .NET es creada como un Formulario (*Form* en inglés). Una ventana estándar, de herramientas, sin bordes y ventanas flotantes pueden ser creadas usando la clase Form. Esta clase también permite la creación de ventanas de diálogo u otras de tipo modal.

Usando las diferentes propiedades que posee la clase Form, se puede determinar la apariencia, tamaño, color y administración de la ventana que se está creando. Un ejemplo de ello es la propiedad *Text* (texto en inglés), permite que se pueda poner el título en la barra de títulos de la ventana.

[67] MSDN es un proyecto de Microsoft que viene del acrónimo inglés (Microsoft Developer Network *Red de Desarrollo para Microsoft*), y es responsable de la administración las firmas de relación entre desarrolladores y probadores con la empresa.

Se puede establecer del formulario desde la ventana de propiedades de Visual Studio, como se ve en la figura 2.29

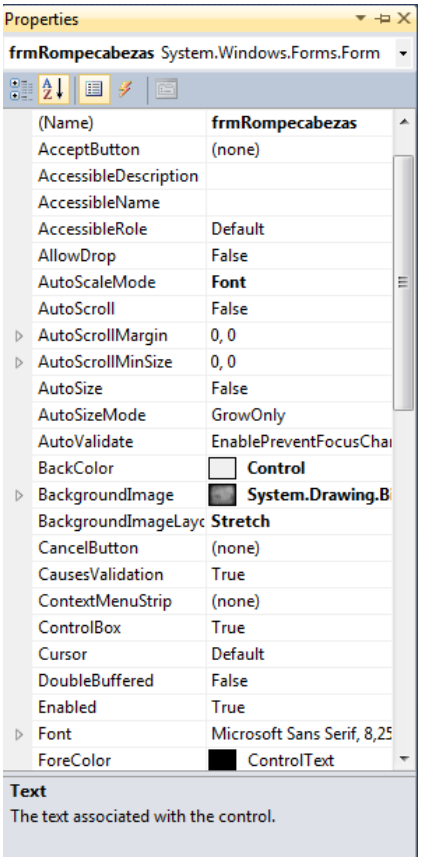


Figura 2.20 Ventana de Propiedades de una clase Form

El código se genera automáticamente al momento de hacer algún cambio en la ventana de propiedades (figura 2.20). Si se desea ver el código, lo que se debe hacer es clic derecho encima del formulario, y seleccionar ver código.

Además de las propiedades, también se puede usar los métodos de la clase para manipular el formulario. Por ejemplo, el método *SetDesktopLocation* para posicionar la ventana en el escritorio.

Los eventos de la clase formulario permiten responder a acciones que se han hecho dentro del formulario. Se puede activar un evento[68] para realizar una operación, como por ejemplo, actualizar los datos que se muestran en los controles del formulario, cuando este está activo.

2.6.2. Controles

Los controles son necesario para el desarrollo, ya que sin ellos, el formulario por sí solo no tiene mucha utilidad. En ellos se puede mostrar y manipular los datos y permite interactuar la interfaz de usuario (Simon, 2001). Windows Forms tiene 2 clases base para estos controles:

La librería *System.Windows.Forms.Control* define el mínimo conjunto de características y código de cada uno de los controles de un formulario. Un control derivado o heredado de la clase *Control* tiene todas las propiedades básicas, y solamente tiene los eventos básicos, tales como eventos de teclado o de mouse.

Los controles que se encuentran en esta librería son los típicos que un programador esperaría de una interfaz de usuario gráfico, como son:

- Etiquetas (Label)
- Botones (Button)
- Casillas de verificación (CheckBox)

[68] Un *evento*, en computación, es una acción u ocurrencia detectada por un programa, que puede ser manejada por dicho programa. Típicos eventos son manejados sincronizadamente con el flujo del programa. Ejemplos de eventos son: clic del mouse, presionar una tecla, otro programa en ejecución, una acción de una función dentro o fuera de la aplicación, etc.

- Menús
- Casillas de selección (RadioButtons)
- Cajas de combinaciones
- Contenedores de listas (ListBox)
- Contenedores de texto (TextBox)
- Controles de tablas
- Barras de herramientas
- Vista de árboles

Controles más específicos incluyen

- Controladores de tiempo (DataTimerPicker y Timer)
- Diálogos para cambiar fuentes (FontDialog)
- Diálogos de archivos (FileDialog)
- Barras de progreso (ProgressBar)

Una mención especial es el contenedor de imágenes (PictureBox), el que posee características pertenecientes tanto a la librería *System.Windows.Forms* como a *System.Drawing*, mejor conocida como el componente GDI/GDI+.

2.6.3. Gráficos con GDI+

El uso de la librería de los formularios, aunque suficiente en muchos casos para desarrollar aplicaciones de escritorio, no permite la flexibilidad necesaria para desarrollar interfaces de usuario más complejas. Por ejemplo, su se necesita dibujar un texto con una fuente distinta en una posición específica de la ventana, o si es

necesario mostrar imágenes sin usar el control PictureBox, simplemente figuras y formas gráficas, entonces usar los controles del formulario o bien es un desperdicio de recursos, o no son suficientes para cumplir una obligación.

Para cumplir con los requerimientos de los ejemplos mencionados anteriormente, es necesario dibujar directamente en la interfaz, o llamar a una imagen externa que satisfaga la necesidad.

Para dibujar o cargar imágenes directamente en un formulario, se debe hacer uso de una tecnología que posee .NET llamada GDI+. GDI+ consiste en un conjunto de clases base cuyo propósito es llevar a cabo dibujos personalizados en la pantalla. Estas clases pueden manejar una amplia gama de instrucciones, los cuales se envían a dispositivos de salida que muestran información gráfica al usuario, asegurándose que el dibujo deseado sea colocado correctamente en pantalla (o en su defecto, impreso en un dispositivo adecuado o guardado en el formato correcto en memoria).

Aunque la estructura de GDI+ es muy sencilla, es necesario entender las principales directivas que se encuentran detrás de la programación, así como la colección de elementos que son dibujados en pantalla, para así poder dibujar de una manera correcta con esta tecnología[69]. En la siguiente tabla (2.4), está la lista de *namespaces*[70] que se puede encontrar en las fuentes de GDI+ (Simon, 2001).

---

[69] Simon, Robinson and Others, "*Professional C#*", Wrox Press, 2001

[70] Un *namespace* es un término usado en C# (y en programación en general) para referirse al contexto o plano en el que se encuentra una clase u otro elemento, así por ejemplo, si la clase *Casa* y *Edificacion* se encuentran en el mismo contexto, no es necesario llamar o referenciar en código la segunda clase en la primera, para así poder usar sus métodos (pero si es necesario hacer una instanciación de la clase *Edificacion*).

Tabla 2.5. Librerías de la Tecnología GDI+

Namespace	Contiene
System.Drawing	La mayoría de clases, estructuras, enumeradores y delegados se encuentran aquí. Maneja la funcionalidad básica de dibujo.
System.Drawing.Drawing2D	Es una clase más especializada de clases, estructuras, etc. Da más efectos avanzados de dibujo en pantalla.
System.Drawing.Imaging	Varias clases que ayudan in la manipulación de imágenes (Bitmaps, archivos GIF, etc.).
System.Drawing.Printing	Son clases que ayudan cuando un objetivo específico se desea imprimir, o imprimir a un dispositivo diferente.
System.Drawing.Desing	Son algunos diálogos predefinidos, propiedades de hojas y otros elementos de interfaces de usuario.
System.Drawing.Text	Son clases que desempeñan manipulación más avanzada de fuentes y sus familias.

2.6.3.1. Definición de GDI+

El Dispositivo de Interfaces Gráficas (GDI por sus siglas en inglés) es una herramienta inherente en Windows, que permite al Sistema Operativo la construcción de gráficos a partir de lo que un programa o el usuario pide, y sin importar la marca de la tarjeta de video (en el mercado existen una gran variedad de tarjetas, las cuales tienen su propio lenguaje e instrucciones).

GDI está basado en la compilación antigua de Windows (hecha en C), por lo que Microsoft desarrollo en los últimos años GDI+, la cual simplemente agrega una capa a GDI, que permite un mejor manejo de objetos y clases heredadas.

2.6.3.2. Cómo dibujar en GDI+

Los elementos que permiten un típico dibujo de las librerías de GDI+ son casi siempre: color, lápiz y la posición.

La estructura ***Color*** se encuentra dentro de *System.Drawing.Color*, la cual permite instanciar la estructura, y con ella usarla como un objeto en donde se necesite asignar un color específico. El total de colores que posee la estructura es más de 16 millones (Simons, 2001), ya que utiliza la teoría de valores RGB (rojo, verde y azul). También se puede llamar a enumeradores que ya tiene asignados colores como el blanco, azul, verdad-amarillo, etc.

La clase ***Pen*** (lápiz en inglés) es la que específica como se debe dibujar una línea (continua, entrecortada, etc.). Así mismo, la clase ***Brush*** (del inglés pincel), es la que expresa como se debe llenar las figuras (rellenar, dejar espacios, etc.). Las clases Pen y Brush están íntimamente relacionadas con la estructura Color, ya que con la estructura, se define qué color se usará para trazar el borde y rellenar el objeto.

La posición es simplemente la representación del lugar en la pantalla o en la impresión del objeto a ser dibujado. Para esto, se utiliza los ejes de coordenadas, siendo la posición (0, 0) la parte superior izquierda de la pantalla.

En la tabla 2.5, se tiene el listado de métodos para dibujar líneas y figuras que posee la librería *System.Drawing*.

Tabla 2.6. Métodos comunes de la librería System.Drawing

Método	Parámetros	Qué dibuja
DrawLine	Grosor y color, posiciones de inicio y de fin	Una simple línea recta.
DrawRectangle	Grosor de borde, color de borde, posición y tamaño	Un rectángulo sin relleno.
DrawEllipse	Grosor de borde, color de borde, posición y tamaño	Una elipse sin relleno.
FillRectangle	Grosor de borde, color de borde, color de relleno, posición y tamaño	Un rectángulo sólido.
FillEllipse	Grosor de borde, color de borde, color de relleno, posición y tamaño	Una elipse sólida.
DrawLines	Grosor y color, colección de posiciones de inicio y de fin	Una serie de líneas, cada una conectada con cada punto dado en la colección de posiciones.
DrawBezier	Grosor y color, 4 puntos	Una curva suave que va hacia los dos puntos finales, los dos otros puntos sirven para controlar la forma de la curva.
DrawCurve	Grosor y color, colección de puntos	Una curva suave que pasa a través de los puntos.
DrawArc	Grosor y color, rectángulo, 2 ángulos	Parte de un circulo que se encuentra entre el rectángulo, el cual está definido por los 2 ángulos.

2.6.4. Mostar Imágenes con GDI+

Una de las cosas más comunes que se hace con las librerías de GDI+, es el cargado de imágenes ya existentes en archivos. Lo cual es más sencillo que dibujar toda la interfaz de usuario, ya que obviamente ya existe las imágenes. Es posible hacer cierta manipulación a las imágenes y hasta a los mismos archivos, como por ejemplo: recortarla, estrecharla, rotarla, etc.

Para esto, GDI+ contiene la librería *System.Drawing.Image*, la cual contiene a su vez la clase *Image*. Una instancia de la clase *Image* representa una imagen. Para cargar una imagen a la clase, lo único que se hace es:

```
Image miImagen = Image.FromFile("NombreArchivo");
```

CAPÍTULO 3: ANÁLISIS Y DISEÑO

3.1. INTRODUCCIÓN AL CASO PUZZLEMOTE

Como se ha considerado anteriormente, el juego “Puzzlemote” se basa en un n-puzzle, el cual consiste en obtener una matriz tipo n x n, donde cada celda contiene un número desde el 1 hasta el valor n – 1, lo que viene a significar que existe un espacio representado por una celda vacía, llamada la celda en blanco. Dependiendo de la posición, la celda en blanco puede tener dos, tres, o cuatro celdas adyacentes ocupadas. Cada movimiento reubica a una de estas celdas dentro de la celda vacía, moviendo dicho espacio con la celda que tenía la posición inicial ocupada. Este procedimiento continúa para mover todas las fichas, donde se debe considerar siempre las celdas adyacentes y no las oblicuas, ni tampoco las celdas alejadas, siendo el objetivo final colocar todas las fichas en orden desde 1 hasta n-1. La Figura 3.1, muestra un ejemplo del 15-puzzle en su estado inicial y estado final objetivo.

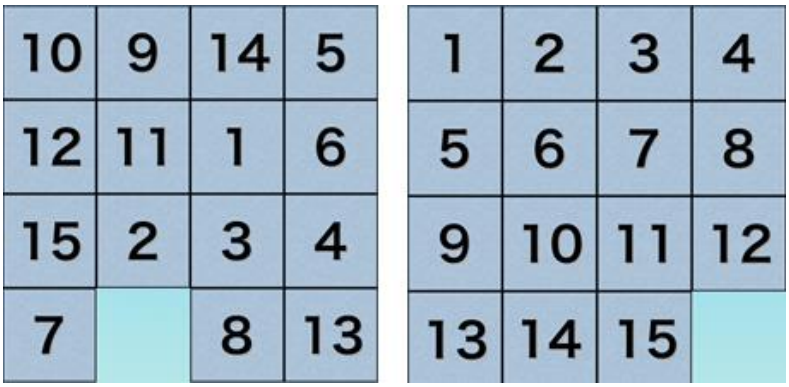


Figura 3.1 Un rompecabezas armado y otro revuelto

3.1.1. Características del Producto "Puzzlemote"

Como se dijo en la sección del alcance del Capítulo I, se debe desarrollar un juego N-Puzzle, el cual se llamará "Puzzlemote", que podrá conectarse con un Wiimote vía Bluetooth. Este juego tendrá la capacidad de mostrar al usuario 3 tipos de rompecabezas: 8-puzzle, 15-puzzle y 24-puzzle; los cuales mostrarán 6 imágenes distintas de animales autóctonos del Ecuador.

Así mismo, para desarmar y armar los rompecabezas, se usará un modelo de Inteligencia Artificial "Hacia atrás y hacia Adelante". También poseerá un módulo de seguridades, el cual permitirá la activación o no del producto resultante en una computadora.

Todas las configuraciones necesarias que tenga la aplicación, serán tomadas y guardadas en archivos planos que vendrán adjuntos a la misma.

3.1.2. Diseño Del Modelo De Inteligencia Artificial En La Aplicación

Para el modelo de IA de la aplicación se ha considerado aplicar la combinación de las técnicas de razonamiento hacia delante y hacia atrás, con lo cual se utiliza grafos dirigidos, quienes son representados por listas enlazadas. Ellas se encargan de almacenar los diferentes movimientos hechos por la misma aplicación y el usuario. Luego de haber llegado a un estado inicial al desordenar el rompecabezas, se puede volver al estado objetivo, donde la aplicación puede reproducir todas las jugadas al manejar los nodos de la lista enlazada para almacenar los movimientos del estado inicial y luego invertir la lista para llegar u obtener el estado objetivo. En la Figura 3.2, muestra tanto los valores iniciales y finales de la lista de fichas y la lista de valores que guardan los movimientos.

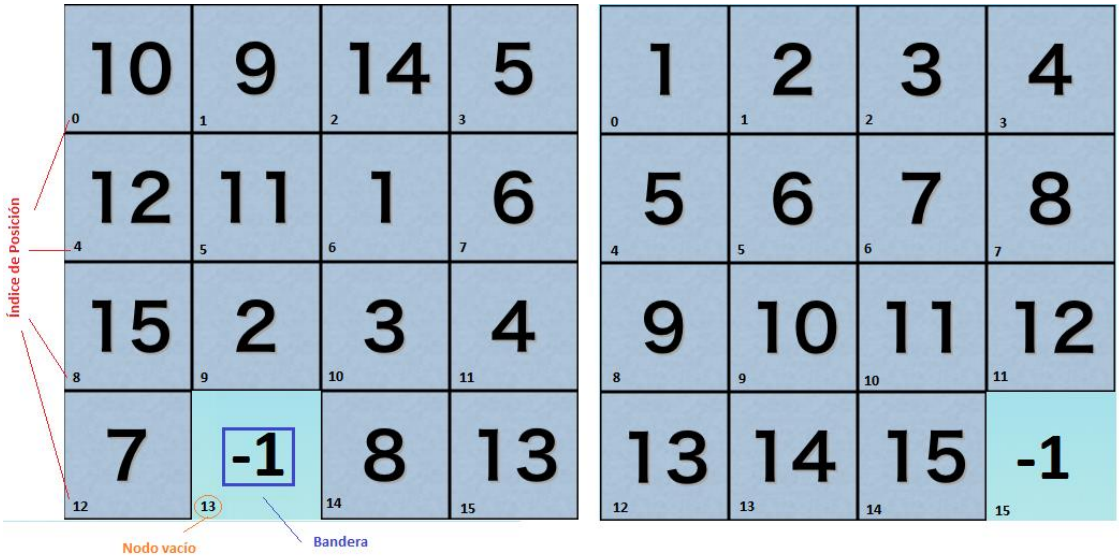


Figura 3.2 Índice de Posiciones en un Puzzle

El modelo anterior muestra con detalle cómo se encuentra estructurado un rompecabezas para 15 fichas. Se compone de 15 grafos dirigidos, los cuales

representan un grupo de movimientos válidos que se pueden realizar dentro de la aplicación; así por ejemplo, el nodo vacío, que en la figura 3.2 es el 13, solo puede relacionarse con los nodos (los índices de posición) 9, 12 y 14, en otras palabras, sus nodos vecinos, de tal manera que se obtiene un grafo que se puede representar como: $G_{13} = \{(13,9), (13,12), (13,14)\}$

Se utiliza una bandera con el valor de -1, la cual indica la ubicación del espacio de la ficha vacía. Cuando se mueve una ficha en el rompecabezas, sucede que los contenidos de ambas fichas (la vacía y la ocupada) intercambian el dato de su número, convirtiendo la ficha vacía en visible y la ocupada en invisible; de tal manera que al final del juego, la ficha vacía se debe encontrar en la posición 15 (la última ficha que está en el tablero de juego). En el caso del juego, la posición 15 corresponde a un botón o controlador de imagen.

La información que se muestra en la tabla 3.1, 3.2 y 3.3, refiere los posibles movimientos válidos de las fichas dentro de los posibles rompecabezas de la aplicación (8,15 y 24 respectivamente), representados por grafos dirigidos:

Tabla 3.1. Grafos Rompecabezas 8

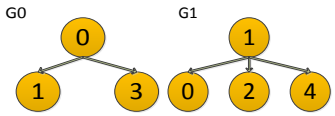
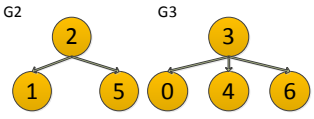
Representación Analítica Grafo Dirigido	Gráfico del Grafo Dirigido
$\boldsymbol{G_0 = \{(0,1), (0,3)\}}$ $\boldsymbol{G_1 = \{(1,0), (1,2), (1,4)\}}$	
$\boldsymbol{G_2 = \{(2,1), (2,5)\}}$ $\boldsymbol{G_3 = \{(3,0), (3,4), (3,6)\}}$	

Tabla 3.1. Grafos Rompecabezas 8

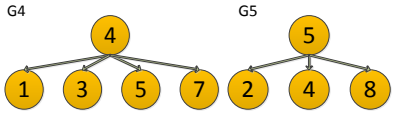
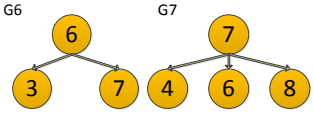
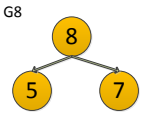
Representación Analítica Grafo Dirigido	Gráfico del Grafo Dirigido
$G_4 = \{(4,1),(4,3),(4,5),(4,7)\}$ $G_5 = \{(5,2),(5,4),(5,8)\}$	
$G_6 = \{(6,3),(6,7)\}$ $G_7 = \{(7,4),(7,6),(7,8)\}$	
$G_8 = \{(8,5),(8,7)\}$	

Tabla 3.2. Grafos Rompecabezas 15

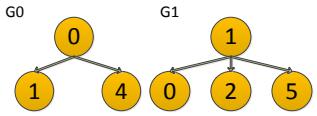
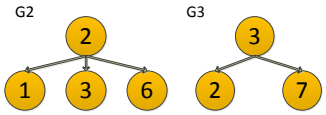
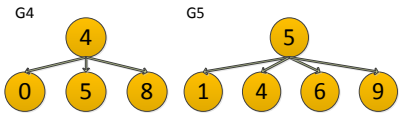
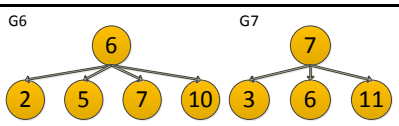
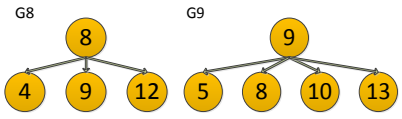
Representación Analítica Grafo Dirigido	Gráfico del Grafo Dirigido
$G_0 = \{(0,1),(0,4)\}$ $G_1 = \{(1,0),(1,2),(1,5)\}$	
$G_2 = \{(2,1),(2,3),(2,6)\}$ $G_3 = \{(3,2),(3,7)\}$	
$G_4 = \{(4,0),(4,5),(4,8)\}$ $G_5 = \{(5,1),(5,4),(5,6),(5,9)\}$	
$G_6 = \{(6,2),(6,5),(6,7),(6,10)\}$ $G_7 = \{(7,3),(7,6),(7,11)\}$	
$G_8 = \{(8,4),(8,9),(8,12)\}$ $G_9 = \{(9,5),(9,8),(9,10),(9,13)\}$	

Tabla 3.2. Grafos Rompecabezas 15

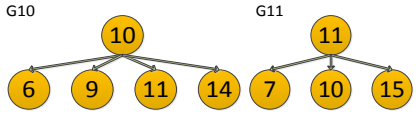
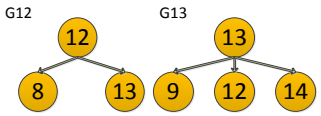
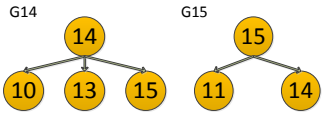
Representación Analítica Grafo Dirigido	Gráfico del Grafo Dirigido
$G_{10} = \{(10,6),(10,9),(10,11),(10,14)\}$ $G_{11} = \{(11,7),(11,10),(11,15)\}$	
$G_{12} = \{(12,8),(12,13)\}$ $G_{13} = \{(13,9),(13,12),(13,14)\}$	
$G_{14} = \{(14,10),(14,13),(14,15)\}$ $G_{15} = \{(15,11),(15,14)\}$	

Tabla 3.3 Grafos Rompecabezas de 24

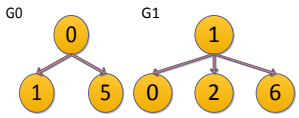
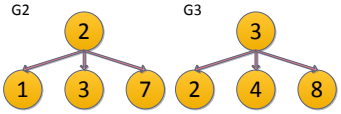
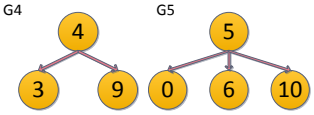
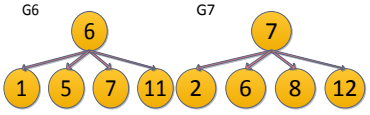
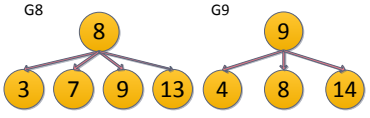
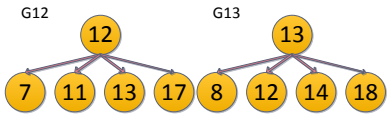
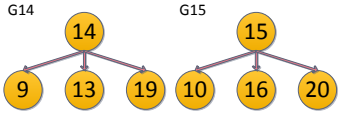
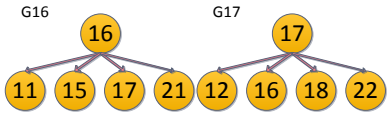
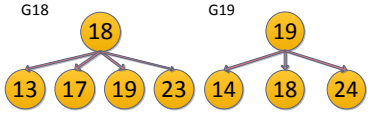
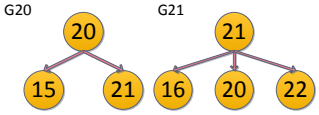
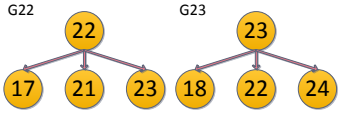
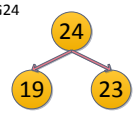
Representación Analítica Grafo Dirigido	Gráfico del Grafo Dirigido
$G_0 = \{(0,1),(0,5)\}$ $G_1 = \{(0,1),(1,2),(1,6)\}$	
$G_2 = \{(1,2),(2,3),(2,7)\}$ $G_3 = \{(2,3),(3,4),(3,8)\}$	
$G_4 = \{(3,4),(4,9)\}$ $G_5 = \{(0,5),(5,6),(5,10)\}$	
$G_6 = \{(1,6),(5,6),(6,7),(6,11)\}$ $G_7 = \{(2,7),(6,7),(7,8),(7,12)\}$	
$G_8 = \{(3,8),(7,8),(8,9),(8,13)\}$ $G_9 = \{(4,9),(8,9),(9,14)\}$	
$G_{10} = \{(5,10),(10,11),(10,15)\}$ $G_{11} = \{(6,11),(10,11),(11,12),(11,16)\}$	

Tabla 3.3 Grafos Rompecabezas de 24

Representación Analítica Grafo Dirigido	Gráfico del Grafo Dirigido
$G_{12} = \{(7,12),(11,12),(12,13),(12,17)\}$ $G_{13} = \{(8,13),(12,13),(13,14),(13,18)\}$	
$G_{14} = \{(9,14),(13,14),(14,19)\}$ $G_{15} = \{(10,15),(15,16),(15,20)\}$	
$G_{16} = \{(11,16),(15,16),(16,17),(16,21)\}$ $G_{17} = \{(12,17),(16,17),(17,18),(17,22)\}$	
$G_{18} = \{(13,18),(17,18),(18,19),(18,23)\}$ $G_{19} = \{(14,19),(18,19),(18,24)\}$	
$G_{20} = \{(15,20),(20,21)\}$ $G_{21} = \{(16,21),(20,21),(21,22)\}$	
$G_{22} = \{(17,22),(21,22),(22,23)\}$ $G_{23} = \{(18,23),(22,23),(23,24)\}$	
$G_{24} = \{(19,24),(23,24)\}$	

3.2. ESPECIFICACIÓN DE REQUERIMIENTOS

Basado en lo anterior, se puede concluir que se tendrá como actor a un usuario Jugador, el cual interactuará con el sistema Puzzlemote, por medio directo (teclado y mouse del computador) como por medio indirecto (Wiimote).

3.2.1 Identificación de Roles y Tareas

Roles

Existe 2 roles en la aplicación:

- Jugador

  Es el usuario que tiene acceso al juego, y el cual interactúa con la aplicación. Puede cambiar el nombre de usuario que se encuentra en el sistema, configurar el juego de acuerdo a sus necesidades, seleccionar un nuevo juego a partir de una nueva imagen y pide si es que no puede armar el rompecabezas, que se arme solo. También activa el producto y puede salir de la aplicación.

- Sistema

  Es todo lo que el sistema hace automáticamente.

Tareas

- Jugador

  1. Inicia programa
  2. Activa el producto
  3. Configura el juego
  4. Mueve una ficha en una partida del juego

5. Ve el puntaje de los juegos
6. Selecciona nueva imagen
7. Pide que se arme solo el rompecabezas
8. Cambia el nombre del jugador
9. Sale del juego

- Sistema

1. Conecta el Wiimote
2. Comprueba la activación del Producto
3. Carga la Configuración Guardada
4. Comprueba si el jugador gano la partida
5. Inicia una partida
6. Guarda el Puntaje

3.2.1. Especificación de Escenarios

Rol de Jugador

- **Inicia programa. –** El jugador puede iniciar el juego desde el sistema operativo.
- **Activa el programa. –** El jugador puede activar el programa para poder usar el programa después del período de prueba del mismo.
- **Configura el juego. –** El jugador podrá cambiar los movimientos iniciales que realice el sistema para desarmar el rompecabezas y cambiar el tipo de rompecabezas (de 8, 15 y 24 piezas). Finalmente podrá guardar la configuración seleccionada.

- **Mueve una ficha en una partida del juego. –** El jugador podrá mover una ficha en el rompecabezas, la cual se moverá automáticamente a donde está el espacio en blanco.
- **Ve el puntaje de los juegos. –** El jugador podrá ver los puntajes que se encuentran almacenados en sistema de todos los juegos anteriores.
- **Selecciona nueva imagen. –** El jugador podrá seleccionar una imagen para una nueva partida.
- **Pide que se arme solo el rompecabezas. –** El jugador podrá pedir al sistema que arme el rompecabezas actual si lo cree conveniente.
- **Cambia el nombre del jugador. –** El jugador podrá cambiar el nombre de usuario que se encuentra almacenado en el sistema.
- **Sale del juego. –** El jugador podrá salir del sistema en el momento que desee.

Rol del Sistema

- **Conecta el Wiimote. –** Al iniciar el usuario el programa, el sistema automáticamente intentará hacer una conexión con un dispositivo Wiimote; en caso de no haber uno, el sistema avisará al jugador que no existe un dispositivo, y continuará con el inicio del programa.
- **Comprueba la activación del Producto. –** Al iniciar el usuario el programa, este automáticamente comprobará que el producto este activado. Si no es así, se lo hace saber al usuario.

- **Carga la Configuración Guardada. –** Al iniciar el usuario el programa, el sistema automáticamente cargara la información guardada de la configuración de la misma.
- **Comprueba si el jugador gano la partida. –** Al mover una ficha el jugador, el sistema hará automáticamente una comprobación de que si la última jugada provocó que haya ganado la partida; si es así, el sistema informará al usuario que ha ganado.
- **Inicia una partida. –** Al iniciar el usuario el programa, el sistema iniciará, de acuerdo a la configuración, el armado de una nueva partida de juego para el jugador.
- **Guarda el Puntaje. –** Al informar al jugador que la partida ha sido ganada, el sistema automáticamente guardará el puntaje obtenido por el usuario.

3.2.3. Especificación de Casos de Uso por Actor

De lo anterior, se puede deducir que el único actor del sistema es del rol *Jugador*, ya que las tareas que realiza el rol de *Sistema* se aplican automáticamente al hacer alguna de las tareas que realiza el Jugador.

En la figura 3.3 se puede apreciar todos los casos de uso del sistema Puzzlemote.

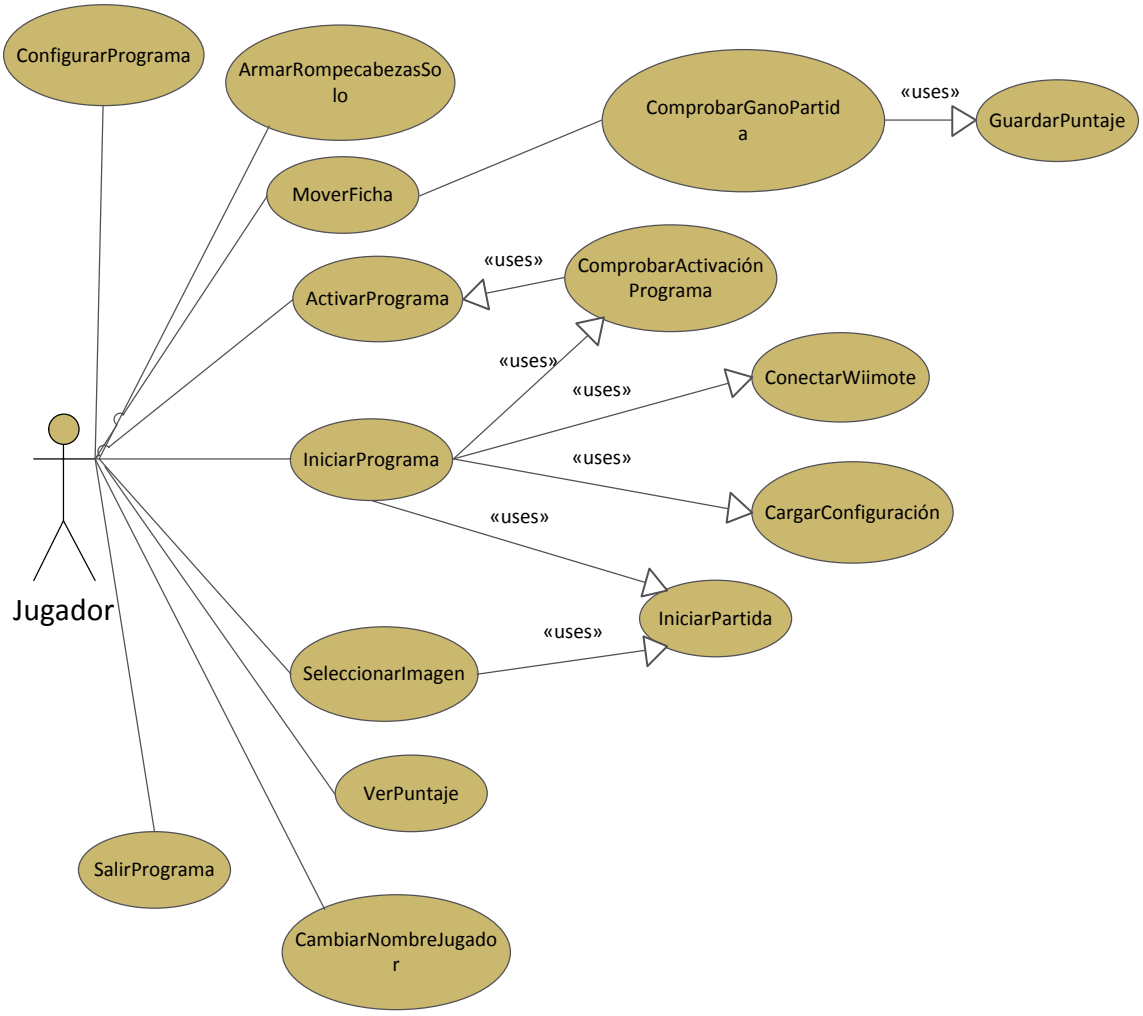


Figura 3.3 Casos de Uso del actor *Jugador* – Puzzlemote

3.2.3.1. Actor: Jugador

1. USR-JUG-PUZZLEMOTE-01: Iniciar Programa
2. USR-JUG-PUZZLEMOTE-02: Activar Programa
3. USR-JUG- PUZZLEMOTE-03: Configurar Juego
4. USR-JUG- PUZZLEMOTE-04: Mover Ficha
5. USR-JUG- PUZZLEMOTE-05: Ver Puntaje
6. USR-JUG- PUZZLEMOTE-06: Seleccionar Imagen
7. USR-JUG- PUZZLEMOTE-07: Armar Rompecabezas Solo
8. USR-JUG- PUZZLEMOTE-08: Cambiar Nombre Jugador
9. USR-JUG- PUZZLEMOTE-09: Salir Programa

En la figura 3.4 se puede ver los casos de uso del actor Jugador.

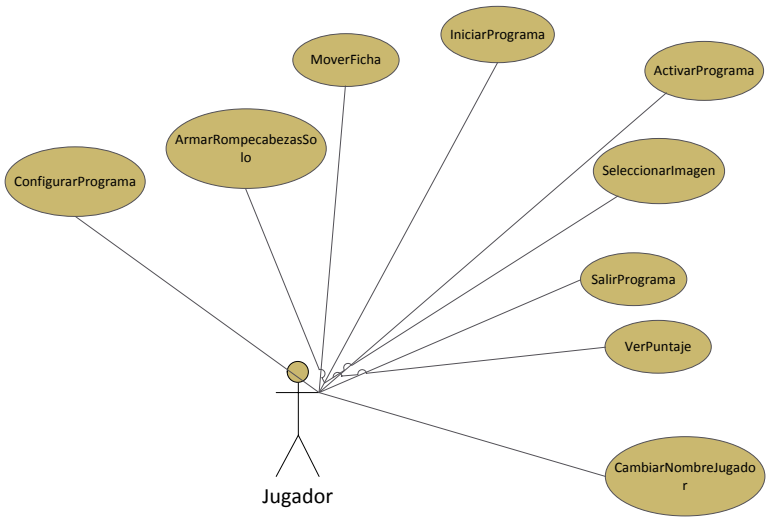


Figura 3.4 Casos de Uso Jugador

3.2.3.1.1. Especificación de Casos de Uso

En las tablas 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 3.10 y 3.11 se ven los detalles de cada caso de uso del actor Jugador.

Tabla 3.4 Iniciar Programa

USR-JUG- PUZZLEMOTE-01**: Iniciar Programa**	
Resumen:	Proceso en el cual el usuario jugador inicia el sistema.
Prioridad:	Esencial
Actores Directos:	Jugador

Escenarios

Tipo de Escenario	**Descripción**
Principal	El usuario inicia la aplicación desde su máquina PC.

Pre-condiciones
Ninguna

Tabla 3.5 Activar Programa

USR-JUG- PUZZLEMOTE-02**: Activar Programa**	
Resumen:	Proceso en el cual el usuario jugador activa el programa.
Prioridad:	Esencial
Actores Directos:	Jugador, Comprobar Activación Programa

Escenarios

Tipo de Escenario	**Descripción**
Principal	El sistema comprueba si el producto ya ha sido activado, si es así, comienza a cargar las configuraciones.

Tabla 3.5 Activar Programa

Tipo de Escenario	**Descripción**
Secundario	El sistema comprueba si el producto ya ha sido activado, si no es así, muestra la ventana de período de prueba, si el usuario tiene aún tiempo o ha ingresado menos del número máximo de ingresos gratuitos, comienza a cagar las configuraciones.
Tipo de Escenario	**Descripción**
Secundario	El sistema comprueba si el producto ya ha sido activado, si no es así, muestra la ventana de período de prueba, si el usuario ya no tiene tiempo o ha ingresado igual al número máximo de ingresos gratuitos, impide la carga de las configuraciones.

Pre-condiciones
Iniciar Programa, Comprobar Activación del Programa

Tabla 3.6 Configurar Juego

USR-JUG- PUZZLEMOTE-03**: Configurar Juego**	
Resumen:	Proceso en el cual el usuario configura las características del juego. Selecciona el número de movimientos de mezcla de fichas (16, 32 o 64) y el nivel del Juego (8-puzzle, 15-puzzle o 24-puzzle). También se guarda la configuración del juego para siguientes.
Prioridad:	Opcional
Actores Directos:	Jugador
Tipo de Escenario	**Descripción**
Principal	1. El usuario hace clic en el botón de configuraciones, el cual mostrará las configuraciones cargadas al inicio de la aplicación. 2. El usuario puede cambiar los valores configurados, y guardar los nuevos.

Pre-condiciones
Cargar Configuración

Tabla 3.7 Mover Ficha

USR-JUG- PUZZLEMOTE-04**: Mover Ficha**	
Resumen:	Proceso en el cual el usuario mueve una ficha en la partida actual.
Prioridad:	Esencial
Actores Directos:	Jugador
Tipo de Escenario	**Descripción**
Principal	El usuario mueve una ficha presionando el D-Pad de Wiimote, o haciendo clic encima de la ficha que desea mover. Solo se moverá las fichas que se encuentran alrededor del espacio en blanco.
Pre-condiciones	
Iniciar Partida, Conectar Wiimote	

Tabla 3.8 Ver Puntaje

USR-JUG- PUZZLEMOTE-05**: Ver Puntaje**	
Resumen:	Proceso en el cual el usuario obtiene la lista de puntajes que tiene el juego.
Prioridad:	Opcional
Actores Directos:	Jugador
Tipo de Escenario	**Descripción**
Principal	El usuario selecciona la opción de Ver Puntaje en la ventana de Configuración.
Tipo de Escenario	**Descripción**
Secundario	El usuario selecciona la opción de Ver Puntaje antes de haber jugado alguna vez en la ventana de Configuración, pero no se cargará ningún puntaje.
Pre-condiciones	
Cargar Configuración	

Tabla 3.9 Seleccionar Imagen

USR-JUG- PUZZLEMOTE-06**: Seleccionar Imagen**	
Resumen:	Proceso en el cual el usuario selecciona una imagen para una nueva partida.
Prioridad:	Esencial
Actores Directos:	Jugador
Escenarios	
Tipo de Escenario	**Descripción**
Principal	1. El usuario selecciona nuevo juego, el cual cargará una ventana con todas las imágenes que posee el rompecabezas para jugar. 2. El usuario selecciona una de las imágenes, e inicia el nuevo juego con la imagen seleccionada.
Pre-condiciones	
Iniciar Programa	

Tabla 3.10 Armar Rompecabezas Solo

USR-JUG- PUZZLEMOTE-07**: Armar Rompecabezas Solo**	
Resumen:	Proceso en el cual el usuario pide al sistema armar el rompecabezas.
Prioridad:	Esencial
Actores Directos:	Jugador, Sistema
Escenarios	
Tipo de Escenario	**Descripción**
Principal	1. El usuario selecciona la opción de armar rompecabezas, que se encuentra en el menú de configuraciones. 2. El sistema usa el método hacia atrás de IA, y arma el rompecabezas solo.
Pre-condiciones	
Iniciar Partida, Cargar Configuraciones	

Tabla 3.11 Cambiar Nombre Jugador

USR-JUG- PUZZLEMOTE-08**: Cambiar Nombre Jugador**	
Resumen:	Proceso en el cual el usuario cambia el nombre del Jugador ya guardado en la configuración.
Prioridad:	Opcional
Actores Directos:	Jugador

Escenarios

Tipo de Escenario	**Descripción**
Principal	1. El usuario selecciona cambiar nombre de jugador. 2. El usuario ingresa el nuevo valor. 3. El sistema guarda el nuevo valor en configuraciones.

Pre-condiciones
Cargar Configuración

Tabla 3.12 Salir Programa

USR-JUG- PUZZLEMOTE-09**: Salir Programa**	
Resumen:	Proceso en el cual el usuario sale del sistema.
Prioridad:	Esencial
Actores Directos:	Jugador

Escenarios

Tipo de Escenario	**Descripción**
Principal	1. El usuario selecciona salir del sistema 2. Se le pregunta al usuario si está seguro de continuar. 3. La aplicación se termina.

Pre-condiciones
Iniciar Programa

3.2.3.2. Actor: Sistema

1. USR-SIS-PUZZLEMOTE-01: Comprobar Activación Programa
2. USR-SIS-PUZZLEMOTE-02: Cargar Configuración
3. USR-SIS-PUZZLEMOTE-03: Conectar Wiimote
4. USR-SIS-PUZZLEMOTE-04: Iniciar Partida
5. USR-SIS-PUZZLEMOTE-05: Comprobar Gano Partida
6. USR-SIS-PUZZLEMOTE-06: Guardar Puntaje

En la figura 3.5 se puede ver al actor Sistema con sus casos de uso.

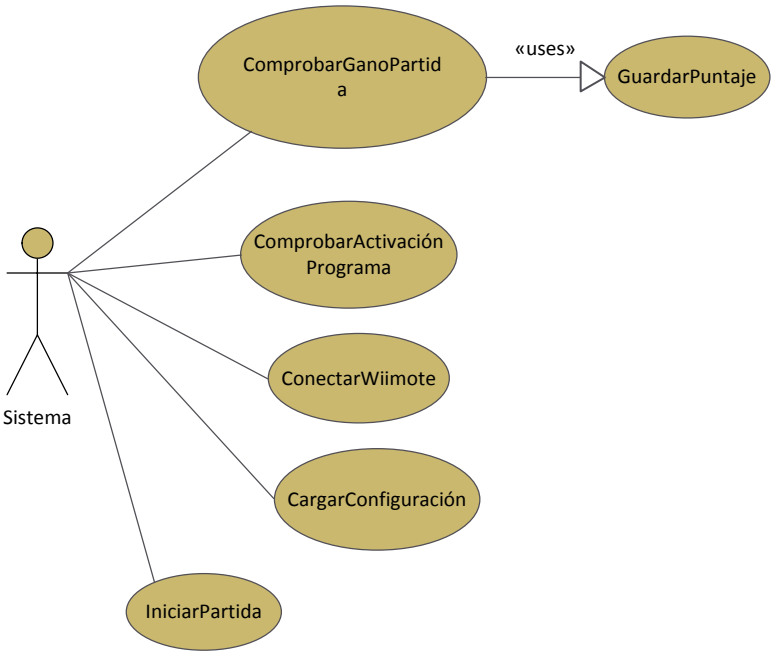


Figura 3.5 Casos de uso del actor *Sistema*.

3.2.3.1.2. Especificación de Casos de Uso

En las tablas 3.12, 3.13, 3.14, 3.15, 3.16 y 3.17 se ven los detalles de cada caso de uso del actor Sistema.

Tabla 3.13 Comprobar Activación Sistema

USR-SIS- PUZZLEMOTE-01**: Comprobar Activación Sistema**	
Resumen:	Proceso en el cual el sistema comprueba si la aplicación ya fue activada.
Prioridad:	Esencial
Actores Directos:	Sistema

Escenarios

Tipo de Escenario	**Descripción**
Principal	El usuario inicia la aplicación desde su máquina PC. Al hacerlo, el sistema comprobará si el usuario tiene activa su copia del programa. Si la tiene, se cargará la configuración guardada.

Tipo de Escenario	**Descripción**
Secundario	El usuario inicia la aplicación desde su máquina PC. Al hacerlo, el sistema comprobará si el usuario tiene activa su copia del programa. Si no la tiene, se cargará la ventana de activación del producto.

Pre-condiciones
Iniciar Programa

Tabla 3.14 Cargar Configuración

USR-SIS- PUZZLEMOTE-02**: Cargar Configuración**	
Resumen:	Proceso en el cual el sistema carga las configuraciones guardadas.
Prioridad:	Esencial
Actores Directos:	Sistema, Comprobar Activación Programa

Tabla 3.14 Cargar Configuración

Tipo de Escenario	Descripción
Principal	Si el producto ya ha sido activado, el sistema cargará los datos guardados de configuración e iniciará automáticamente una nueva partida.

Pre-condiciones
Iniciar Programa, Comprobar Activación del Programa

Tabla 3.15 Conectar Wiimote

USR-SIS- PUZZLEMOTE-03**: Conectar Wiimote**	
Resumen:	Proceso en el cual el sistema se conecta con un Wiimote desde un dispositivo Bluetooth.
Prioridad:	Opcional
Actores Directos:	Sistema, Wiimote

Escenarios

Tipo de Escenario	Descripción
Principal	1. El sistema comprueba si existe algún control que desee conectarse con el sistema. 2. Si es así, se hace la conexión con el control.

Tipo de Escenario	Descripción
Secundario	1. El sistema comprueba si existe algún control que desee conectarse con el sistema. 2. Si no es así, se informa al usuario que no existe un control que desee conectarse.

Pre-condiciones
Iniciar Programa, Wiimote conectado a la PC

Tabla 3.16 Iniciar Partida

USR-SIS- PUZZLEMOTE-04**: Iniciar Partida**	
Resumen:	Proceso en el cual el sistema inicia una nueva partida con la imagen ya seleccionada con anterioridad.
Prioridad:	Esencial
Actores Directos:	Sistema

Escenarios

Tipo de Escenario	**Descripción**
Principal	El sistema inicia con la configuración guardada, de acuerdo a eso y a una imagen seleccionada aleatoriamente, se carga el entorno del juego, y se hacer el desarme del rompecabezas con movimientos de IA “hacia adelante”.

Tipo de Escenario	**Descripción**
Secundario	Cuando el usuario selecciona una nueva imagen, el sistema carga el entorno de acuerdo a la configuración guardada y a la imagen seleccionada, y se hacer el desarme del rompecabezas con movimientos de IA “hacia adelante”.

Pre-condiciones
Cargar Configuración, Seleccionar Imagen

Tabla 3.17 Comprobar Gano Partida

USR-SIS- PUZZLEMOTE-05**: Comprobar Gano Partida**	
Resumen:	Proceso en el cual el sistema comprueba si el último movimiento de fichas provoco que se haya armado el rompecabezas.
Prioridad:	Esencial
Actores Directos:	Sistema

Tabla 3.17 Comprobar Gano Partida

Escenarios	
Tipo de Escenario	**Descripción**
Principal	Al hacer el usuario un movimiento de ficha, se comprueba si con ello el rompecabezas se ha ordenado; en caso de serlo, se guarda el puntaje obtenido, se hace visible la última ficha y se informa al usuario que acaba de ganar.
Tipo de Escenario	**Descripción**
Secundario	Al hacer el usuario un movimiento de ficha, se comprueba si con ello el rompecabezas se ha ordenado; en caso de no serlo, se sigue esperando al siguiente movimiento de ficha.
Pre-condiciones	
Mover Ficha	

Tabla 3.18 Guardar Puntaje

USR-SIS- PUZZLEMOTE-06**: Guardar Puntaje**	
Resumen:	Proceso en el cual el sistema guarda el puntaje obtenido en una partida acabada.
Prioridad:	Esencial
Actores Directos:	Sistema
Escenarios	
Tipo de Escenario	**Descripción**
Principal	1. Si se ganó la partida, se comprueba si el tiempo obtenido es menor al décimo puntaje anterior. 2. Si es así, se guarda el tiempo obtenido en segundos y en el nivel realizado.

Tabla 3.18 Guardar Puntaje

Tipo de Escenario	Descripción
Secundario	1. Si se ganó la partida, se comprueba si el tiempo obtenido es menor al décimo puntaje anterior. 2. Si es así, no se guarda.
Pre-condiciones Comprobar Gano Partida	

3.2.4. Requerimientos no Funcionales

Los requisitos no funcionales se pueden enumerar como:

- El sistema debe poseer un esquema para el manejo de archivos planos, el cual permitirá el manejo de guardado de configuraciones y puntajes que se vayan adquiriendo por parte de los usuarios.
- El sistema debe poseer sonido que relaje los sentidos y sea acorde a la edad a la que se está apuntando.
- Se debe usar imágenes de animales en vez de números, ya que con ello se mejora la capacidad educativa del proyecto.
- El usuario no puede subir más imágenes al sistema.

3.2.5. Diagramas de Secuencia

Se encuentran en el Anexo 2 en la sección Diagramas de Secuencia.

3.3. DISEÑO CONCEPTUAL

Como ya se ha discutido, el diseño conceptual consta de los pasos detallados a continuación.

3.3.1. Diseño de Archivos Planos

El diagrama de la figura 3.6 muestra como es la arquitectura de archivos planos que se utilizan para leer y escribir información de configuraciones y puntaje del producto Puzzlemote.

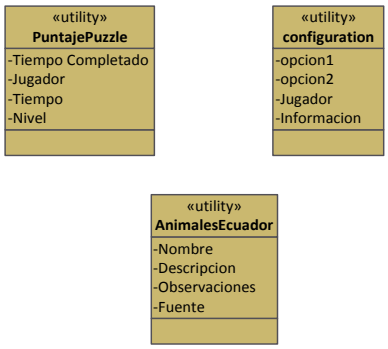


Figura 3.6 Archivos Planos usados en Puzzlemote

El archivo "***PuntajePuzzle****.xml*" es un archivo en formato XML, el cual almacenará el puntaje obtenido por los usuarios de la aplicación.

El archivo "***AnimalesEcuador****.xml*" es un archivo en formato XML, el cual contendrá algunos datos de información que se muestran sobre cada imagen que se puede armar en el rompecabezas.

El archivo "***configuration***" es un archivo de texto sin extensión, el cual contendrá la configuración inicial de la aplicación, el cual se actualizará acorde al usuario.

3.3.2. Diagrama de Clases

Considerando que tipo de arquitectura se usará para el desarrollo, se ha tomado en cuenta el uso del modelo vista-controlador (MVC)[71], es necesario dividir las clases de acuerdo a los componentes que presente cada módulo de desarrollo.

En este caso se generará 5 capaz de control, las cuales son:

- Controlador de Juego (figura 3.7).
- Controlador de Registro de Sistema (figura 3.8).
- Controlador de Seguridad (relacionado con el controlador de Registro de sistema) (figura 3.9).
- Controlador de la Librería Wiimote (figura 3.10).
- Vista del Sistema (figura 3.11).

[71] Para una mejor comprensión de qué es arquitectura y MVC, se recomienda ver estos términos en el Glosario de Términos.

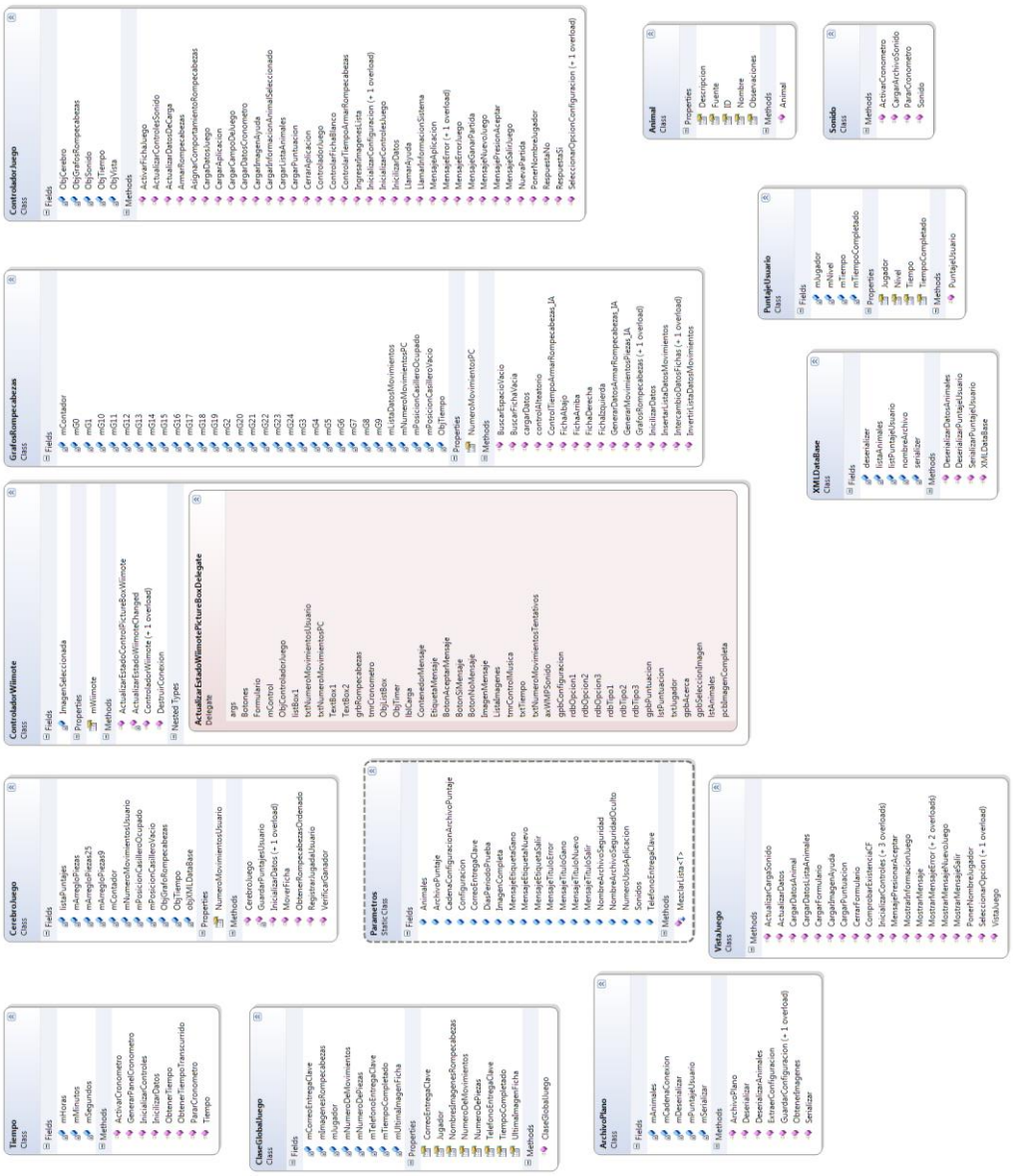


Figura 3.7 Diagrama de Clases de Puzzlemote, Proyecto ComponenteJuego

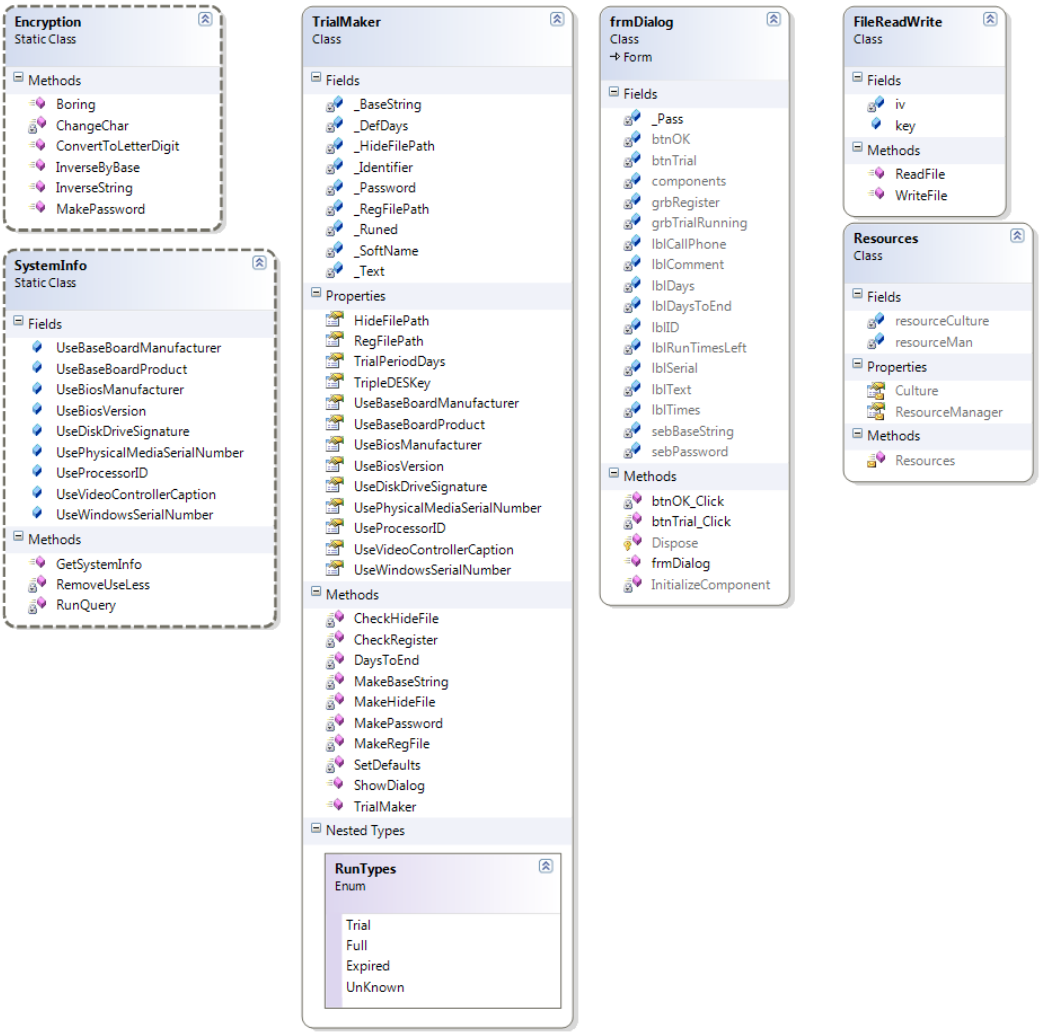


Figura 3.8 Diagrama de Clases Puzzlemote, Componente ControladorRegistroSistema

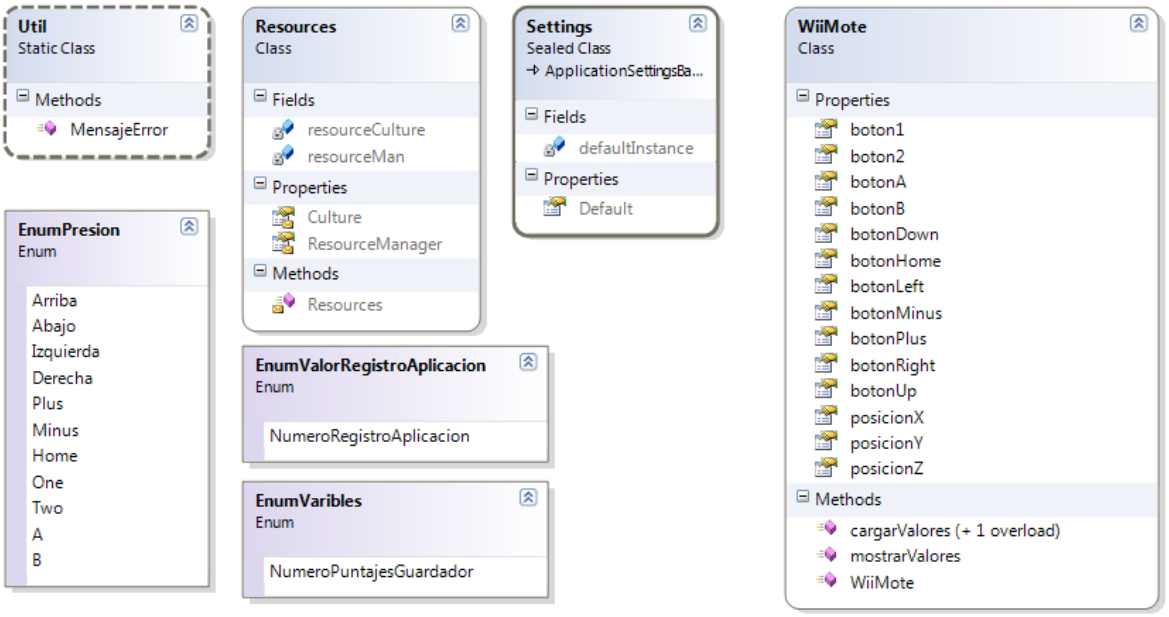


Figura 3.9 Diagrama de Clases Puzzlemote, Componente ControladorLibreriaWiimote

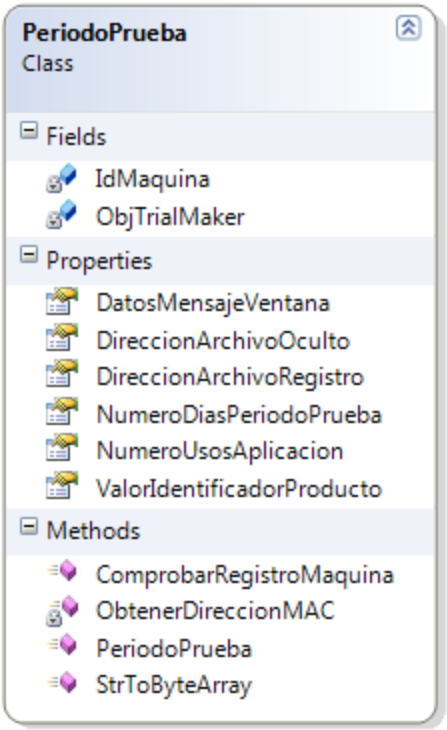


Figura 3.10 Diagrama de Clases Puzzlemote, Proyecto ComponenteSeguridades

Figura 3.11 Diagrama de Clases Puzzlemote, Controlador de Vista WinAppRompecabezas

En total existen 28 clases y 4 estructuras que controlan los estados numerales de algunos componentes.

3.4. MODELO NAVEGACIONAL

El desarrollo de las interfaces estará marcado por el uso de formularios (los controles Form), los que permiten una rápida y adecuada para desarrollar vistas, además de ser muy fáciles de adecuar y ser estéticamente correctas.

Los objetos Navegacionales son:

- Control de Programa
- Formulario de Rompecabezas 8-puzzle
- Formulario de Rompecabezas 15-puzzle
- Formulario de Rompecabezas 24-puzzle
- Formulario de Activación de Producto

Los contextos Navegacionales son:

- Seleccionar Imagen
- Configurar Juego
- Acerca del Juego
- Ayuda del Juego
- Puntajes

3.4.1. Esquema Navegacional

En las siguientes tablas (3.18, 3.19, 3.20, 3.21, 3.22, 3.23, 3.24, 3.25, 3.26 y 3.27) se resume las principales características de las clases navegacionales anteriormente mencionadas.

Tabla 3.19 Control de Programa

Nombre: Nodo Control de Programa
Clase Conceptual (CC): Program
Atributos: ➢ Ninguno
Descripción: Es la que se encarga de cargar el formulario adecuado de acuerdo a los datos obtenidos del archivo de configuración y a la activación del producto.
Enlaces: ➢ Al nodo de Formulario de Rompecabezas 8-puzzle ➢ Al nodo de Formulario de Rompecabezas 15-puzzle ➢ Al nodo de Formulario de Rompecabezas 24-puzzle ➢ Al nodo de Activación de Producto

Tabla 3.20 Formulario de Rompecabezas 8-Puzzle

Nombre: Nodo Formulario de Rompecabezas 8-Puzzle
Clase Conceptual (CC): frmRompecabezas9
Atributos: ➢ **Controlador de Juego** ➢ **Controlador de Wiimote** ➢ **Lista de Fichas** ➢ **Lista de Imágenes** ➢ **Lista de Movimientos**
Descripción: Es la que se encarga de cargar el formulario para jugar un rompecabezas de 8 piezas.
Enlaces: ➢ Al nodo de Configuración de Juego ➢ Al nodo de Seleccionar Imagen

Tabla 3.21 Formulario de Rompecabezas de 15-Puzzle

Nombre: Nodo Formulario de Rompecabezas 15-Puzzle
Clase Conceptual (CC): frmRompecabezas
Atributos: ➢ **Controlador de Juego** ➢ **Controlador de Wiimote** ➢ **Lista de Fichas** ➢ **Lista de Imágenes** ➢ **Lista de Movimientos**
Descripción: Es la que se encarga de cargar el formulario para jugar un rompecabezas de 15 piezas.
Enlaces: ➢ Al nodo de Configuración de Juego ➢ Al nodo de Seleccionar Imagen

Tabla 3.22 Formulario de Rompecabezas de 24-Puzzle

Nombre: Nodo Formulario de Rompecabezas 24-Puzzle
Clase Conceptual (CC): frmRompecabezas25
Atributos: ➢ **Controlador de Juego** ➢ **Controlador de Wiimote** ➢ **Lista de Fichas** ➢ **Lista de Imágenes** ➢ **Lista de Movimientos**
Descripción: Es la que se encarga de cargar el formulario para jugar un rompecabezas de 24 piezas.
Enlaces: ➢ Al nodo de Configuración de Juego ➢ Al nodo de Seleccionar Imagen

Tabla 3.23 Formulario de Activación de Producto

Nombre: Nodo Formulario de Activación de Producto
Clase Conceptual (CC): frmDialog
Atributos: ➢ **Controlador de Juego** ➢ **Controlador de Wiimote** ➢ **Lista de Fichas** ➢ **Lista de Imágenes** ➢ **Lista de Movimientos**
Descripción: Es la que se encarga de cargar el formulario activar el producto.
Enlaces: ➢ Al nodo de Activar Producto ➢ Al nodo de Formulario de Rompecabezas de 8-puzzle ➢ Al nodo de Formulario de Rompecabezas de 15-puzzle ➢ Al nodo de Formulario de Rompecabezas de 24-puzzle

Tabla 3.24 Seleccionar Imagen

Nombre: Seleccionar Imagen
Clase Conceptual (CC): gpbSeleccionarImagen
Atributos: ➢ **Imagen seleccionada**
Descripción: Es la que se encarga de seleccionar una nueva imagen para cargar en el formulario.
Enlaces: ➢ Al nodo de Formulario de Rompecabezas de 8-puzzle ➢ Al nodo de Formulario de Rompecabezas de 15-puzzle ➢ Al nodo de Formulario de Rompecabezas de 24-puzzle

Tabla 3.25 Configurar Juego

Nombre: Configurar Juego
Clase Conceptual (CC): gpbConfiguracion
Atributos: ➢ **Opción de movimientos** ➢ **Tipo de rompecabezas**
Descripción: Es la que se encarga de configurar las opciones del juego.
Enlaces: ➢ Al nodo de Formulario de Rompecabezas de 8-puzzle ➢ Al nodo de Formulario de Rompecabezas de 15-puzzle ➢ Al nodo de Formulario de Rompecabezas de 24-puzzle ➢ Al Acerca del Juego ➢ Al Puntajes

Tabla 3.26 Acerca del Juego

Nombre: Acerca del Juego
Clase Conceptual (CC): gpbAcerca
Atributos: ➢ ninguno
Descripción: Es el que se encarga de mostrar la ventana Acerca del Juego.
Enlaces: ➢ Al nodo de Formulario de Rompecabezas de 8-puzzle ➢ Al nodo de Formulario de Rompecabezas de 15-puzzle ➢ Al nodo de Formulario de Rompecabezas de 24-puzzle ➢ Al Ayuda

Tabla 3.27 Ayuda del Juego

Nombre: Ayuda del Juego
Clase Conceptual (CC): Ejecutable Distinto
Atributos: ➢ Ninguno
Descripción: Es una aplicación distinta que muestra la ayuda del juego.
Enlaces: ➢ ninguno

Tabla 3.28 Puntajes

Nombre: Puntajes
Clase Conceptual (CC): gpbPuntajes
Atributos: ➢ Lista de Puntajes
Descripción: Muestra los puntajes que se han obtenido durante todas las partidas del juego.
Enlaces: ➢ Al nodo de Formulario de Rompecabezas de 8-puzzle ➢ Al nodo de Formulario de Rompecabezas de 15-puzzle ➢ Al nodo de Formulario de Rompecabezas de 24-puzzle

En la figura 3.12 se puede ver el modelo de Clases Navegacionales del sistema, en el cual, a través de la clase Control de Programa (o *Program*, como lo conoce y utiliza el lenguaje C#) se acceden a los demás nodos.

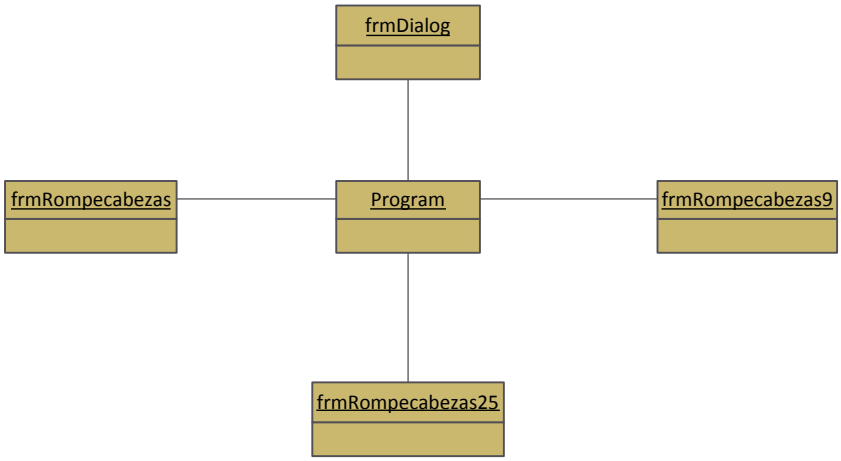


Figura 3.12 Modelo de Clases Navegacionales de Puzzlemote

3.4.2. Esquema de Contextos Navegacionales

La estructura entonces que presentará el sistema está basada en un diseño arquitectónico Vista – Controlador. La figura 3.13 muestra el esquema final del proyecto.

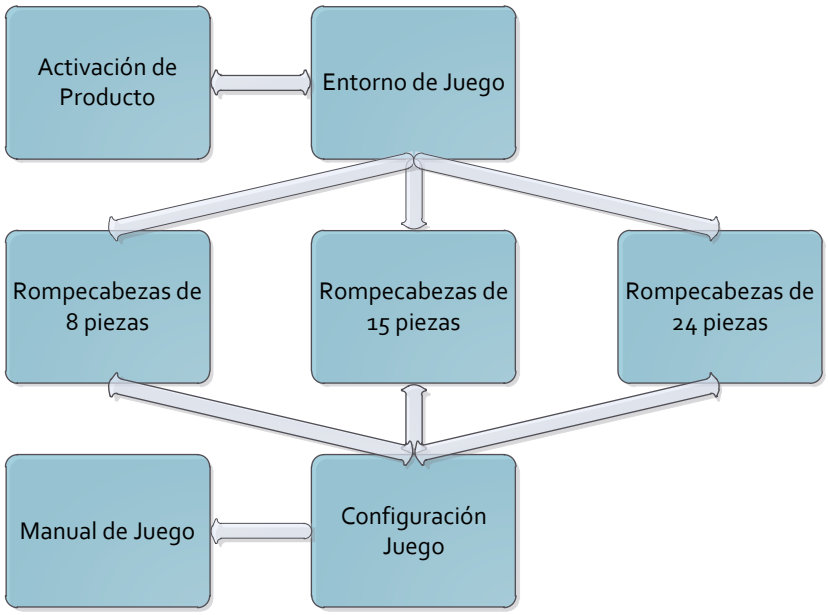


Figura 3.13 Esquema de Contexto Puzzlemote

3.2.3. Arquitectura del Sistema

Como se dijo anteriormente, la aplicación se generará en el modelo vista controlador. La figura 3.14 muestra la arquitectura que poseerá Puzzlemote.

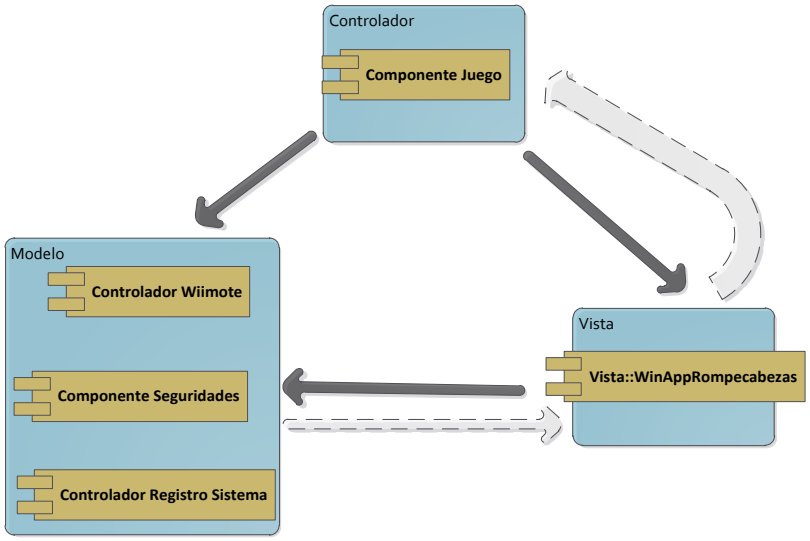


Figura 3.14 Arquitectura de Puzzlemote

3.5. DISEÑO DE INTERFAZ ABSTRACTA

La interfaz de usuario está basada íntegramente en la arquitectura de contenido. Esto quiere decir que se maneja apuntando a una aplicación de escritorio de Windows.

3.5.1 Vista de Datos Abstractos

Aquí se detalla cada uno de los datos que se presentarán en cada formulario. Los formularios de los rompecabezas poseen las mismas características, la única

diferencia sería el número de fichas que posee cada una (9, 16, 25 para el 8-puzzle, 15-puzzle y 24-puzzle respectivamente).

- Control de Juego

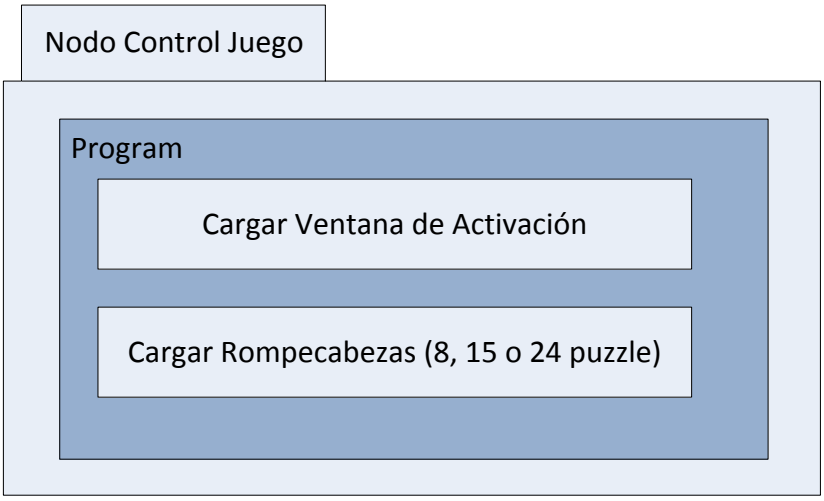


Figura 3.15 Vista Abstracta del Nodo Control Juego "Program" (no visible para el usuario)

- Formularios Rompecabezas (8, 15 y 24)

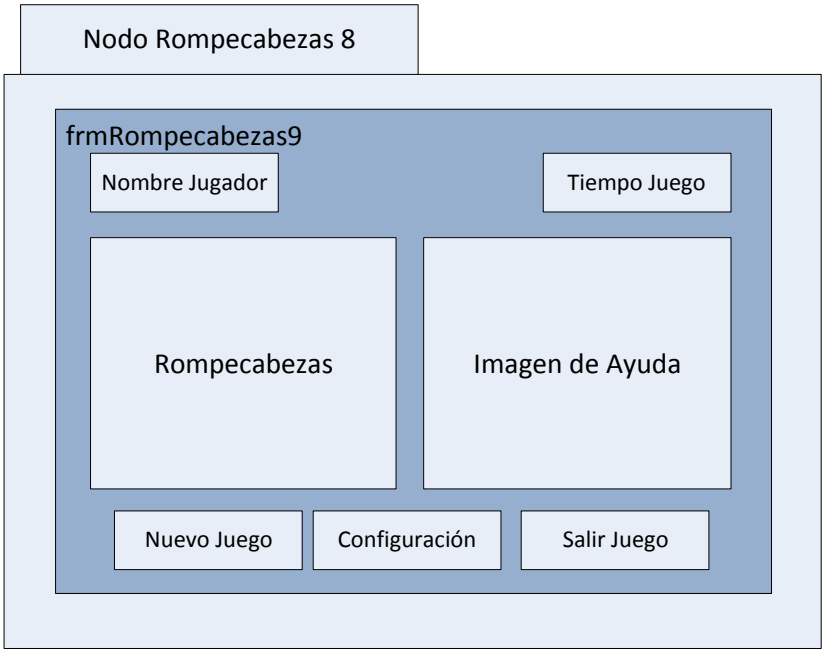


Figura 3.16 Vista Abstracta de Nodo Rompecabezas 8 "frmRompecabezas9"

Figura 3.17 Vista Abstracta de Nodo Rompecabezas 15 "frmRompecabezas"

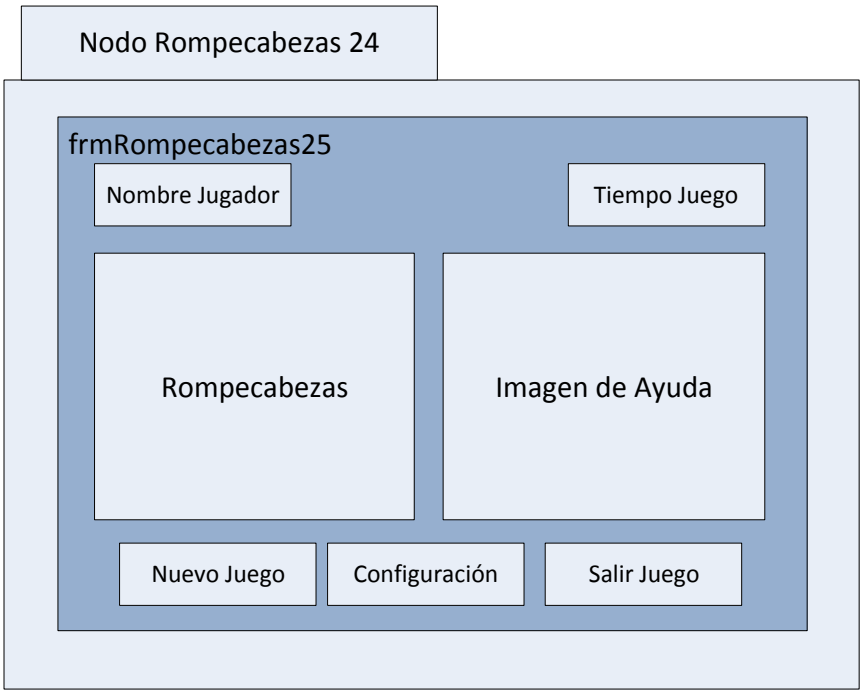


Figura 3.18 Vista Abstracta de Nodo Rompecabezas 24 "frmRompecabezas25"

- Configuración del Juego

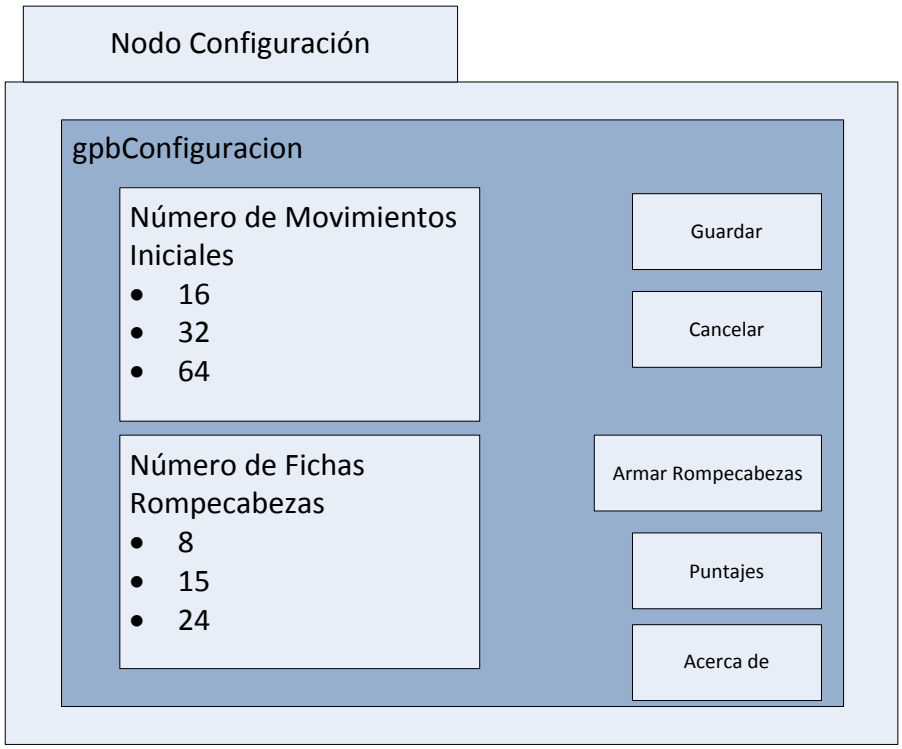


Figura 3.19 Vista Abstracta de Nodo Configuración "gpbConfiguracion"

- Seleccionar Imagen

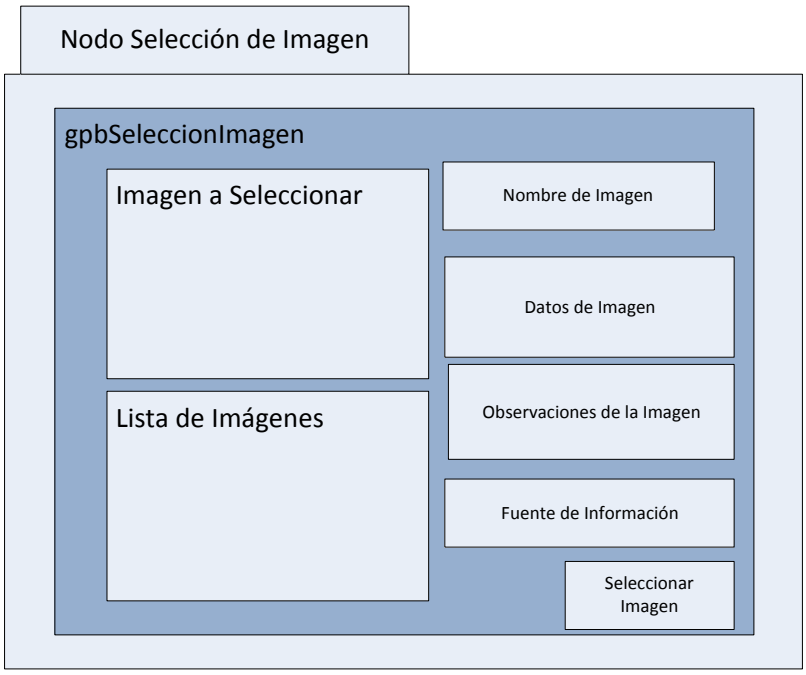


Figura 3.20 Vista Abstracta del Nodo Selección de Imagen "gpbSeleccionImagen"

- Acerca del Juego

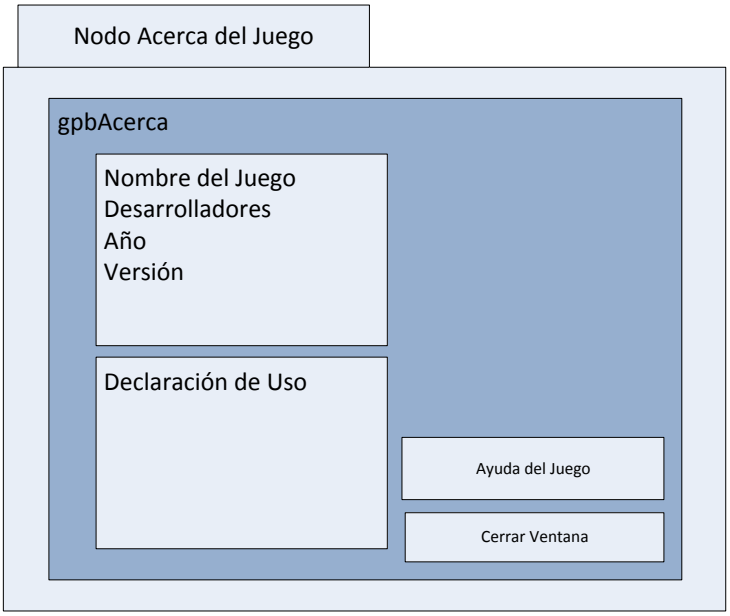


Figura 3.21 Vista Abstracta del Nodo Acerca del Juego "gpbAcerca"

- Ayuda del Juego (se llama a una presentación externa al juego, hecha en un presentador tipo flash llamado *Prezi*)
- Registrar Producto

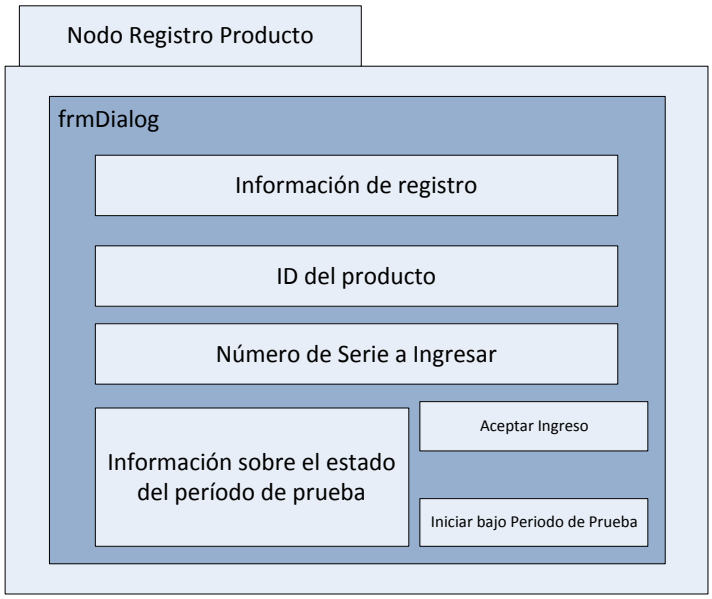


Figura 3.22 Vista Abstracta del nodo Registro Producto "frmDialog"

- Puntuación

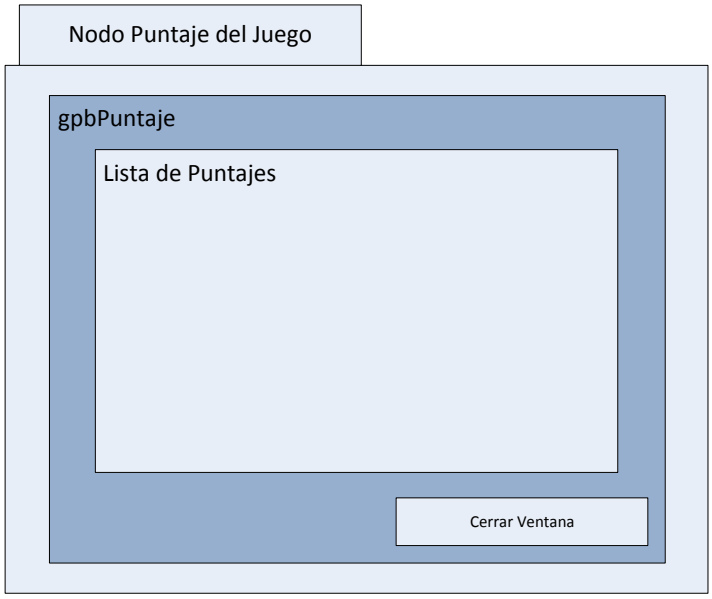


Figura 3.23 Vista Abstracta del Nodo Puntaje del Juego "gpbPuntaje"

3.5.2. Diagrama de Configuración

Se comportan de la misma manera que las vistas abstractas, pero también tienen los eventos clic del mouse u otros eventos de otros dispositivos de entrada, lo que permite acceder a las configuraciones de una manera adecuada para cada uno.

3.6. DISEÑO ESTÉTICO DE PUZZLEMOTE

La siguiente sección detalla los valores de las plantillas, fuentes, colores y tamaños de imágenes que posee el producto Puzzlemote.

3.6.1. Características del Diseño

El diseño se basa sobre todo en el uso de imágenes dentro de cada uno de los controles que se usarán en sistema. Los colores aplicados para los textos son:

- Blanco
- Negro

El fondo principal pasa por una gama de tonalidades (de negro a blanco en el centro), lo que le da un efecto centrista a cada formulario.

Los formularios de rompecabezas aparecerán en el centro de la pantalla, sin la posibilidad de cambiar de tamaño o maximizarse.

El tamaño de todas las ventanas de rompecabezas será de 882 x 604 pixeles.

3.6.2. Consideraciones de Diseño Gráfico

La única fuente que se usa es Microsoft Sans Serif, de tamaño entre 8 a 12. Así mismo, los colores de las fuentes son blancos o negros.

3.7. DISEÑO DE COMPONENTES

El diseño de componentes se muestra en la figura 3.24.

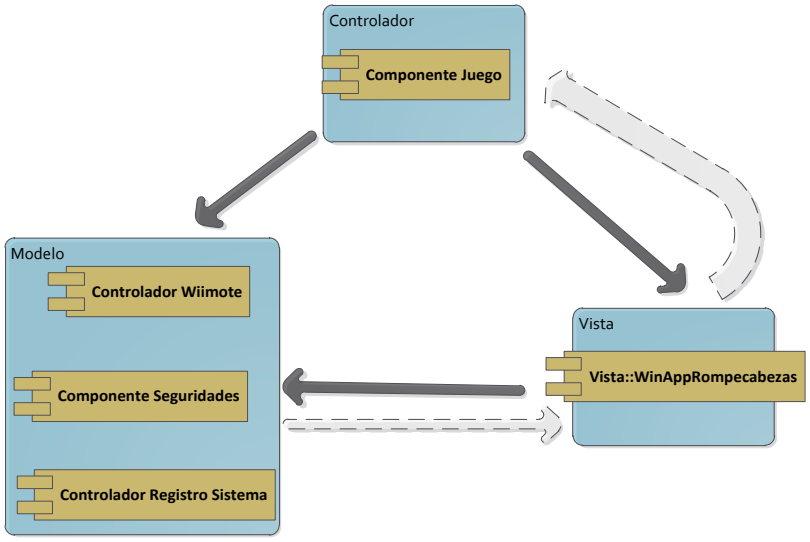


Figura 3.24 Diagrama de Componentes de Puzzlemote

3.8. DIAGRAMA DE DESPLIEGUE

El diagrama de despliegue se puede apreciar en la figura 3.25

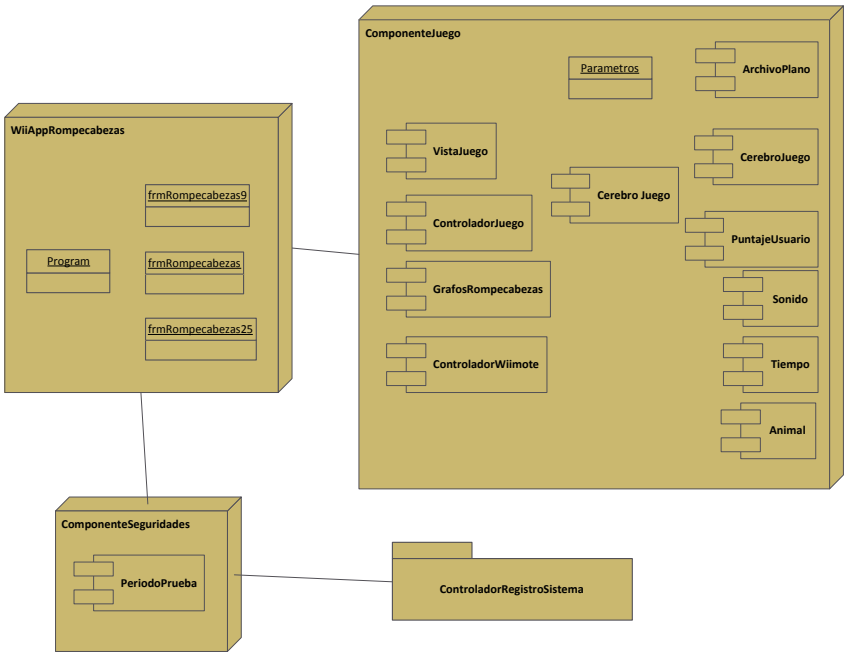


Figura 3.25 Diagrama de Despliegue de Puzzlemote

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBAS DE PUZZLEMOTE

Para realizar la implementación del proyecto, se usará el IDE Visual Studio 2010, en el cual se aplicará la arquitectura MVC.

Lo primero que se hará para la construcción, es la implementación de los componentes que posee el juego y luego seguirán las interfaces de usuario.

Finalmente, se harán las pruebas lógicas y de campo. Las pruebas de campo se realizarán con un grupo de niños de la edad de 6 años.

4.1. CONSTRUCCIÓN DE COMPONENTES PUZZLEMOTE

Los componentes que posee la aplicación se dividen:

- **ComponenteLibreriaWiimote:** Es el componente que se encargará de la contención de datos de Wiimote disponible y conectado con la PC.
- **ComponenteJuego**: Es el componente “motor” o “corazón” de la aplicación, posee el cerebro del juego, el comportamiento de los formularios, el control de sonido, de cronómetros y movimientos, el manejo de los archivos planos y finalmente conecta y controla el comportamiento de un Wiimote conectado a la PC por medio del componente.

- **ComponenteSeguridades**: Se encarga de manejar la activación o no del producto Puzzlemote instalado. Hace uso de la una librería de control libre basada en C# llamada "*TrialMaker*", el cual posee los componentes necesarios para el manejo de ventanas y archivos en el registro de seguridad de activación.

4.1.1. Creación del Componente Librería Wii

El componente se encarga de la contención de los datos que genera el Wiimote al cambiar de estado.

Clase WiiMote

Contiene los estados que presenta un Wiimote conectado al aplicativo. En la tabla 4.1 se muestra los atributos y métodos que posee la clase.

Tabla 4.1 Clase WiiMote

Name	Type	Modifier	Summary
Methods			
cargarValores	void	public	Asigna los valores de los estados del control al objeto
cargarValores	void	public	Asigna los valores de los estados del control al objeto
mostrarValores	List<Label>	public	Carga los valores de control Wii en una lista de etiquetas tipo label
WiiMote		public	Contiene los estados que tiene el Wiimote
<add method>			
Properties			
boton1	bool	public	Presión en el botón Uno (1)
boton2	bool	public	Presión en el botón Dos (2)
botonA	bool	public	Presión en el botón A
botonB	bool	public	Presión en el botón B
botonDown	bool	public	Presión en el botón Abajo
botonHome	bool	public	Presión en el botón Casa (Home)
botonLeft	bool	public	Presión en el botón Izquierda
botonMinus	bool	public	Presión en el botón Más (-)
botonPlus	bool	public	Presión en el botón Más (+)
botonRight	bool	public	Presión en el botón Derecha
botonUp	bool	public	Presión en el botón Arriba
posicionX	float	public	Posición del control en el plano X
posicionY	float	public	Posición del control en el plano Y
posicionZ	float	public	Posición del control en el plano Z

Enumeradores

Los enumeradores son estructuras de datos, permiten almacenar números estáticos que se repiten constantemente durante el desarrollo del software, facilitando su uso dentro del código. Las tablas 4.2, 4.3 y 4.4 muestran los enumeradores de este componente.

Tabla 4.2 EnumPresion

Name	Value
Arriba	1
Abajo	2
Izquierda	3
Derecha	4
Plus	5
Minus	6
Home	7
One	8
Two	9
A	10
B	11

Tabla 4.3 EnumValorRegistroAplicación

Name	Value
NumeroRegistroAplicacion	986

Tabla 4.4 EnumVariables

Name	Value
NumeroPuntajesGuardador	10

4.1.2. Creación del Componente Juego

Este es el componente que mayor actividad tiene en la aplicación, ya que posee el cerebro del juego, controladores de cada una de las facetas del juego (vista, tiempo, sonido, movimiento del control remoto Wii), manejo de archivos planos y control de información (puntajes y datos sobre las imágenes mostradas en la aplicación).

Clase VistaJuego

Controla la vista del juego. La tabla 4.5 muestra sus métodos.

Tabla 4.5 Clase VistaJuego

Name	Type	Modifier	Summary
Methods			
ActualizarCargaSonido	void	public	Actualiza la carga del sonido en el sistema
ActualizarDatos	void	public	Actualiza los datos por cada movimiento hecho por el jugador o el juego
CargarDatosAnimal	void	public	Carga la información del animal en los contenedores adecuados
CargarDatosListaAnimales	void	public	Carga una Lista con los datos que se encuentran en el archivo de animales
CargarFormulario	void	public	Carga un objeto tipo formulario
CargarImagenAyuda	void	public	Carga la Imagen de Ayuda para armar el Rompecabezas
CargarPuntuacion	void	public	Carga la puntuación en una lista
CerrarFormulario	void	public	Cierra el formulario y la aplicación
ComprobarExistenciaCF	bool	public	Comprueba la existencia del Archivo de Configuración
InicializarControles	void	public	Inicializa las fichas del rompecabezas con imágenes
InicializarControles	void	public	Inicializa los controles del sistema para las opciones del juego
InicializarControles	void	public	Inicializa los controles de configuración del sistema al inicio del juego
InicializarControles	void	public	Inicializa las fichas con imágenes
MensajePresionarAceptar	void	public	Cierra la ventana del mensaje al presionar aceptar
MostrarInformacionJuego	void	public	Muestra la ventana de información del sistema
MostrarMensaje	DialogResult	public	Muestra un mensaje por parte del juego
MostrarMensajeError	void	public	Muestra el mensaje de error del sistema
MostrarMensajeError	void	public	Muestra un mensaje de error del sistema
MostrarMensajeError	DialogResult	public	Muestra un mensaje de error específico del sistema
MostrarMensajeNuevoJuego	void	public	Muestra el mensaje de nuevo juego del sistema
MostrarMensajeSalir	void	public	Muestra el mensaje de salida del sistema
PonerNombreJugador	void	public	Pone el nombre del jugador guardado en el contenedor
SeleccionarOpcion	void	public	Selecciona las opciones para la ventana de configuración
SeleccionarOpcion	void	public	Selecciona la carga original del juego
VistaJuego		public	Controla la vista del juego

Clase Tiempo

Controla los movimientos y el tiempo del sistema. La tabla 4.6 muestra sus características.

Tabla 4.6 Clase Tiempo

Name	Type	Modifier	Summary
Methods			
ActivarCronometro	void	public	Activa el cronómetro del tiempo
GenerarPanelCronometro	void	public	Asigna en el contenedor los valores del cronómetro
InicializarControles	void	public	Activa los controles que cambian el tiempo
InicilizarDatos	void	public	Inicializa los datos de la clase
ObtenerTiempo	string	public	Obtiene el tiempo transcurrido durante una partida
ObtenerTiempoTranscurrido	int	public	Transforma el tiempo transcurrido en segundos
PararCronometro	void	public	Para el cronómetro
Tiempo		public	Controla los movimientos y cronómetro del sistema
<add method>			
Properties			
<add property>			
Fields			
mHoras	int	private	Horas recorridas
mMinutos	int	private	Minutos recorridos
mSegundos	int	private	Segundos recorridos

Clase Sonido

Controla el sonido del sistema. La tabla 4.7 muestra sus características.

Tabla 4.7 Clase Sonido

Name	Type	Modifier	Summary
Methods			
ActivarCronometro	void	public	Activa el cronómetro que activa el sonido
CargarArchivoSonido	void	public	reproduce los sonidos para el juego
PararCronometro	void	public	para el sonido
Sonido		public	Controla el sonido en el sistema

Clase GrafosRompecabezas

Controla los movimientos que maneja el sistema. La tabla 4.8 muestra los métodos que posee y la tabla 4.9 muestra sus atributos.

Tabla 4.8 Métodos de la Clase GrafosRompecabezas

Name	Type	Modifier	Summary
Methods			
BuscarEspacioVacio	PictureBox	public	Busca el espacio en blanco
BuscarFichaVacia	int	public	Busca cual es la ficha en blanco
cargarDatos	void	public	Prepara las listas de los movimientos
controlAlteatorio	bool	public	Verifica iteraciones no deseadas
ControlTiempoArmarRompecabezas_IA	void	public	Controla el tiempo y los movimientos de las fichas al armarse solas
FichaAbajo	PictureBox	public	Encuentra la ficha que se puede mover de acuerdo al espacio vacío si se presiona "Down" en el Wiimote
FichaArriba	PictureBox	public	Encuentra la ficha que se puede mover de acuerdo al espacio vacío si se presiona "Up" en el Wiimote
FichaDerecha	PictureBox	public	Encuentra la ficha que se puede mover de acuerdo al espacio vacío si se presiona "Right" en el Wiimote
FichaIzquierda	PictureBox	public	Encuentra la ficha que se puede mover de acuerdo al espacio vacío si se presiona "Left" en el Wiimote
GenerarDatosArmarRompecabezas_IA	void	public	Mueve las fichas para que se arme el rompecabezas con el método "Hacia atrás"
GenerarMovimientosPiezas_IA	void	public	Revuelve las fichas aleatoriamente de acuerdo a el método "Hacia adelante"
GrafosRompecabezas		public	Controla los movimientos que maneja el sistema
GrafosRompecabezas		public	Controla los movimientos que maneja el sistema
InicilizarDatos	void	public	Inicializa los datos
InsertarListaDatosMovimientos	void	public	Inserta en la lista el movimiento hecho
IntercambioDatosFichas	void	public	Intercambia los datos entre la ficha en blanco y la ficha que se va a mover
IntercambioDatosFichas	void	public	Intercambia los datos entre la ficha en blanco y la ficha que se va a mover
InvertirListaDatosMovimientos	void	public	Invierte la lista de movimientos hechos

Tabla 4.9 Atributos de la Clase GrafosRompecabezas

Name	Type	Modifier	Summary
Fields			
mContador	int	private	Contenedor que cuenta cuantas veces se ha hecho algún movimiento
mG0	List<int>	private	Grafo de la posición 0
mG1	List<int>	private	Grafo de la posición 1
mG10	List<int>	private	Grafo de la posición 10
mG11	List<int>	private	Grafo de la posición 11
mG12	List<int>	private	Grafo de la posición 12
mG13	List<int>	private	Grafo de la posición 13
mG14	List<int>	private	Grafo de la posición 14
mG15	List<int>	private	Grafo de la posición 15
mG16	List<int>	private	Grafo de la posición 16
mG17	List<int>	private	Grafo de la posición 17
mG18	List<int>	private	Grafo de la posición 18
mG19	List<int>	private	Grafo de la posición 19
mG2	List<int>	private	Grafo de la posición 2
mG20	List<int>	private	Grafo de la posición 20
mG21	List<int>	private	Grafo de la posición 21
mG22	List<int>	private	Grafo de la posición 22
mG23	List<int>	private	Grafo de la posición 23
mG24	List<int>	private	Grafo de la posición 24
mG3	List<int>	private	Grafo de la posición 3
mG4	List<int>	private	Grafo de la posición 4
mG5	List<int>	private	Grafo de la posición 5
mG6	List<int>	private	Grafo de la posición 6
mG7	List<int>	private	Grafo de la posición 7
mG8	List<int>	private	Grafo de la posición 8
mG9	List<int>	private	Grafo de la posición 9
mListaDatosMovimientos	List<int>	private	Lista de movientos hechos
mNumeroMovimientosPC	int	private	Numero de movimientos hechos por el juego
mPosicionCasilleroOcupado	int	private	Contenedor de la posición del casillero que se encuentra ocupado
mPosicionCasilleroVacio	int	private	Contenedor de la posición del casillero que está vacío
ObjTiempo	Tiempo	private	Controla el tiempo transcurrido

Esta clase es la que implementa el razonamiento “hacia adelante” y “hacia atrás” (métodos que realizan estas actividades: GenerarDatosArmarRompecabezas_IA, ControlTiempoArmarRompecabezas_IA, InvertirListaDatosMovimientos), utilizando las 25 tipos de grafos (desde G0 hasta G24), en los cuales se guardan los posibles movimientos de cada posición que se encuentran en el tablero de juego. Con ellos específica cuales son los únicos movimientos que se puede hacer en el tablero, las únicas fichas que se pueden mover son las que están alrededor del espacio en blanco (los métodos que realizan dicha actividad son BuscarEspacioVacio, BuscarFichaVacia, FichaAbajo,

FichaArriba, FichaDerecha, FichaIzquierda); y mover dichos datos a donde se desea (IntercambioDatosFichas).

GenerarDatosArmarRompecabezas_IA utiliza una lista de PictureBox (que representan las fichas que se encuentran en el rompecabezas) y una lista de posiciones globales, los cuales mezcla aleatoriamente el contenido de la lista, pero utilizando el razonamiento hacia adelante. Para ello, busca la posición del espacio vacío, y de acuerdo a las posiciones que se encuentran a lado de la ficha, se selecciona un valor aleatorio obtenido por C#. Esta selección se utiliza como la ficha que se “mueve”, pero en realidad lo que se hace es intercambiar los datos de la ficha “vacía” (se encuentra invisible) con la ficha que se va a “mover”, y se hace invisible la ficha movida, mientras que la vacía se hace visible.

Para una mejor referencia, se puede revisar las tablas 3.1, 3.2 y 3.3 del capítulo III.

Clase ClaseGlobalJuego

Controla los atributos globales que tiene el sistema. La tabla 4.10 muestra las características que posee esta clase.

Tabla 4.10 Clase ClaseGlobalJuego

Name	Type	Modifier	Summary
Methods			
ClaseGlobalJuego		public	Controla los atributos globales que tiene el sistema
<add method>			
Properties			
CorreoEntregaClave	string	public	Campo de encapsulamiento
Jugador	string	public	Campo de encapsulamiento
NombresImagenesRompecabezas	List<string>	public	Campo de encapsulamiento
NumeroDeMovimientos	int	public	Campo de encapsulamiento
NumeroDePiezas	int	public	Campo de encapsulamiento
TelefonoEntregaClave	string	public	Campo de encapsulamiento
TiempoCompletado	float	public	Campo de encapsulamiento
UltimaImagenFicha	string	public	Campo de encapsulamiento
<add property>			
Fields			
mCorreoEntregaClave	string	private	Correo a donde se debe enviar un mail para activar la clave
mImagenesRompecabezas	List<string>	private	Lista de direcciones de imagenes del rompecabezas
mJugador	string	private	Nombre del jugador
mNumeroDeMovimientos	int	private	Número de movimientos hechos por el jugador en una partida
mNumeroDePiezas	int	private	Número de piezas que tiene un juego seleccionado
mTelefonoEntregaClave	string	private	Teléfono a donde se debe llamar para activar la clave
mTiempoCompletado	float	private	Tiempo completado
mUltimaImagenFicha	string	private	Dirección de la imagen de la ficha invisible

Clase Animal

Contiene la información de cada animal del juego. La tabla 4.11 muestras sus características.

Tabla 4.11 Clase Animal

Name	Type	Modifier	Summary
Methods			
Animal		public	Contiene la información de cada animal del juego
<add method>			
Properties			
Descripcion	string	public	Descripción del animal en la imagen
Fuente	string	public	Fuente de la información
ID	int	public	Id de la imagen
Nombre	string	public	Nombre del animal en la imagen
Observaciones	string	public	Observaciones que puede tener el animal de la imagen

Clase ArchivoPlano

Tabla 4.12 Clase ArchivoPlano

Name	Type	Modifier	Summary
Methods			
ArchivoPlano		public	Controla el guardado de datos, como configuraciones y puntajes
Deserializar	List<PuntajeUsuario>	public	Deserializa un archivo en formato XML
DeserializarAnimales	List<Animal>	public	Deserializa un archivo en formato XML de los datos de los animales que hay en el juego
ExtraerConfiguracion	void	public	Extrae la información que posee el archivo "configuration"
GuardarConfiguracion	void	public	Guarda la información en el archivo "configuration"
GuardarConfiguracion	void	public	Guarda la información en el archivo "configuration"
ObtenerImagenes	List<string>	public	Obtiene todas las imágenes que se tiene en los recursos
Serializar	void	public	Serializa un archivo en formato XML
<add method>			
Properties			
<add property>			
Fields			
mAnimales	List<Animal>	private	Lista de datos de los animales que se encuentran en las imágenes
mCadenaConexion	string	private	Direccion del archivo
mDeserializar	XmlSerializer	private	Deserializador de archivos
mPuntajeUsuario	List<PuntajeUsuario>	private	Lista de puntajes de usuario
mSerializar	XmlSerializer	private	Serializador de archivos

Controla el guardado de datos, como configuraciones y puntajes. La tabla 4.12 muestra sus características.

Clase CerebroJuego

Indica cómo se debe armar, como hacer los movimientos y cuando se gana el juego. La tabla 4.13 muestra sus características.

Tabla 4.13 Clase CerebroJuego

Name	Type	Modifier	Summary
Methods			
CerebroJuego		public	Controla el armado, los movimientos y cuando se gana en el juego
GuardarPuntajesUsuario	void	private	Guarda el puntaje obtenido en una partida por el jugador
InicializarDatos	void	public	Inicializa los datos para una partida
InicializarDatos	void	public	Inicializa los datos para la construcción del rompecabezas
MoverFicha	void	public	Mueve la ficha seleccionada de acuerdo al espacio en blanco dispoible
ObtenerRompecabezasOrdenado	void	public	Obtiene el rompecabezas ordenado invirtiendo los movimientos hechos por el juego y el jugador
RegistrarJugadaUsuario	int	public	Registra la jugada del juego en la lista de jugadas hechas durante la partida
VerificarGanador	bool	public	Comprueba si la última jugada ha finalizado la partida
<add method>			
Properties			
NumeroMovimientosUsuario	int	public	Asigna o muestra el número de movimientos hecho por el usuario
<add property>			
Fields			
listaPuntajes	List<PuntajeUsuario>	private	Lista de puntajes que se han guardado o se guardarán
mArregloPiezas	int[]	private	Lista de piezas para rompecabezas de 16
mArregloPiezas25	int[]	private	Lista de piezas para rompecabezas de 25
mArregloPiezas9	int[]	private	Lista de piezas para rompecabezas de 8
mContador	int	private	Número de movimientos hechos durante la partida
mNumeroMovimientosUsuario	int	private	Número de movimientos hechos por el usuario
mPosicionCasilleroOcupado	int	private	Asigna la posición del casillero que se va a mover
mPosicionCasilleroVacio	int	private	Asigna la posición del casillero vacío en el sistema
ObjGrafoRompecabezas	GrafosRompecabezas	private	Controla los posibles movimientos que pueden hacer las fichas
ObjTiempo	Tiempo	private	Controla el movimiento y el cronómetro del juego
objXMLDataBase	XMLDataBase	private	Controla el guardado de datos del juego

Clase ControladorJuego

Controla la vista, el armado, los movimientos y cuando se gana en el sistema. Esta es la clase que aglutina toda la lógica del juego. Controla la clase VistaJuego, utiliza la clase CerebroJuego para determinar cómo armar un rompecabezas y cuando se gana el mismo, etc. Los atributos de la clase se muestran en la tabla 4.14 y sus métodos en la tabla 4.15.

Tabla 4.14 Atributos de la clase ControladorJuego

Name	Type	Modifier	Summary
Fields			
ObjCerebro	CerebroJuego	private	Controlador del cerebro del juego
ObjGrafosRompecabezas	GrafosRompecabezas	private	Controlador del armaje
ObjSonido	Sonido	private	Controlador del Sonido
ObjTiempo	Tiempo	private	Controlador del tiempo y movimientos
ObjVista	VistaJuego	private	Controlador de la vista

Tabla 4.15 Métodos de la Clase ControladorJuego

Name	Type	Modifier	Summary
Methods			
ActivarFichaJuego	void	public	Controla el movimiento de una ficha, y si al hacerlo, el rompecabezas se ha armado
ActualizarControlesSonido	void	public	Controla la actualización del controlador de sonido del juego
ActualizarDatosDeCarga	void	public	Controla la actualización de los datos de cada movimiento hecho por el usuario o el juego
ArmarRompecabezas	void	public	Controla el armado del rompecabezas automático
AsignarComportamientoRompecabezas	void	public	Prepara el comportamiento del rompecabezas
CargaDatosJuego	string	public	Selecciona aleatoriamente una imagen de los recursos,
CargarAplicacion	void	public	Carga un Formulario a la aplicación
CargarCampoDeJuego	void	public	Dependiendo de la configuración, carga los objetos del formulario para una nueva partida
CargarDatosCronometro	void	public	Controla el contenedor del tiempo transcurrido en una partida
CargarImagenAyuda	void	public	Controla el cargado de la imagen de ayuda en el rompecabezas
CargarInformacionAnimalSeleccionado	void	public	Controla la carga de la información de los datos del animal seleccionado
CargarListaAnimales	void	public	Controla la carga de la lista de datos de las imágenes del animal a seleccionar
CargarPuntuacion	void	public	Controla la ventana de Puntajes
CerrarAplicacion	void	public	Cierra todos los formulario abiertos de la aplicación
ControladorJuego		public	Controla la vista, el armado, los movimientos y cuando se gana en el sistema
ControlarFichaBlanco	PictureBox	public	Controla cual es la ficha en blanco de acuerdo al botón presionado en el Wiimote
ControlarTiempoArmarRompecabezas	void	public	Controla el movimiento que deben hacer las fichas durante un armado automático de una partida
IngresarImagenesLista	void	public	Obtiene todos los archivos y sus directorios de la imagen seleccionada
InicializarConfiguracion	void	public	Inicializa una partida del juego de acuerdo a la configuración guardada con anterioridad
InicializarConfiguracion	void	public	Inicializa la configuración de acuerdo a los controles guardados con anterioridad
InicializarControlesJuego	void	public	Inicializa las fichas del rompecabezas con imágenes
InicilizarDatos	void	public	Inicializa los datos para la construcción del rompecabezas
LlamarAyuda	void	public	Muestra la ayuda que tiene el juego
LlamarInformacionSistema	void	public	Llama al controlador de la información del sistema
MensajeAplicacion	DialogResult	public	Controlador que mostrará un mensaje por parte de sistema
MensajeError	void	public	Controlador que mostrará un mensaje simple de error
MensajeError	DialogResult	public	Controlador del un mensaje de error más complejo
MensajeErrorJuego	void	public	Controlador del Mensaje de Error que mostraría el juego
MensajeGanarPartida	void	public	Mensaje que mostraría el juego en caso de que el jugador gane la partida
MensajeNuevoJuego	void	public	Controla el mensaje de "Nuevo Juego" que mostraría el juego para realizar una nueva partida
MensajePresionAceptar	void	public	Controla el evento clic del botón aceptar del mensaje
MensajeSalirJuego	void	public	Controla el mensaje "Salir del Juego" que mostraría el juego
NuevaPartida	void	public	Controla el evento que se genera al iniciar una nueva partida
PonerNombreJugador	void	public	Controla la activación del control para guardar el nombre del usuario jugador
RespuestaNo	bool	public	Controla si el mensaje de cerrado es no
RespuestaSi	bool	public	Controla si el mensaje de cerrado es si
SeleccionarOpcionConfiguracion	void	public	Controla la selección original al cargar la configuración del juego
SeleccionarOpcionConfiguracion	void	public	Controla la selección de las opciones de la ventana de configuración

Clase PuntajeUsuario

Controla el puntaje de los jugadores en el sistema. La tabla 4.16 muestra sus características.

Tabla 4.16 Clase Puntaje

Name	Type	Modifier	Summary
Methods			
PuntajeUsuario		public	Controla el puntaje de los jugadores en el sistema
<add method>			
Properties			
Jugador	string	public	Encapsulamiento del campo
Nivel	int	public	Encapsulamiento del campo
Tiempo	string	public	Encapsulamiento del campo
TiempoCompletado	float	public	Encapsulamiento del campo
<add property>			
Fields			
mJugador	string	private	
mNivel	int	private	
mTiempo	string	private	
mTiempoCompletado	float	private	

Clase ControladorWiimote

Controla el estado y el manejo del control Wiimote en el sistema. La tabla 4.17 muestra sus características.

Tabla 4.17 Clase ControladorWiimote

Name	Type	Modifier	Summary
Methods			
ActualizarEstadoControlPictureBoxWiimote	void	public	Ejecuta el comportamiento del Wiimote durante el juego
ActualizarEstadoWiimoteChanged	void	private	Ejecuta el delegado que se encarga de comprobar los estado del Wiimote en los formularios de los Rompecabezas
ControladorWiimote		public	Controla el estado y manejo del Wiimote en el sistema
ControladorWiimote		public	Controla el estado y manejo del Wiimote en el sistema
DestruirConexion	void	public	Destructor de las conexiones
<add method>			
Properties			
mWiimote	Wiimote	public	Controlador de los estados del Wiimote
<add property>			
Fields			
ImagenSeleccionada	string	private	Imagen de ayuda

Se podría afirmar que esta clase se comporta como un formulario, ya que para poder realizar las mismas tareas que un mouse, el Wiimote necesita controlar los objetos que se encuentran en cada formulario que tenga eventos clic y otros eventos que cambien la estructura del formulario.

El funcionamiento de D-Pad del Wiimote esta íntegramente ligado a la clase GrafosRompecabezas, ya que de acuerdo a la espacio vacío que hay en el rompecabezas, el control enviará a la clase de control la ficha que se debe mover. Para entender esto mejor, de ejemplo se tiene un rompecabezas de 8 fichas, como se muestra en la figura 4.1.

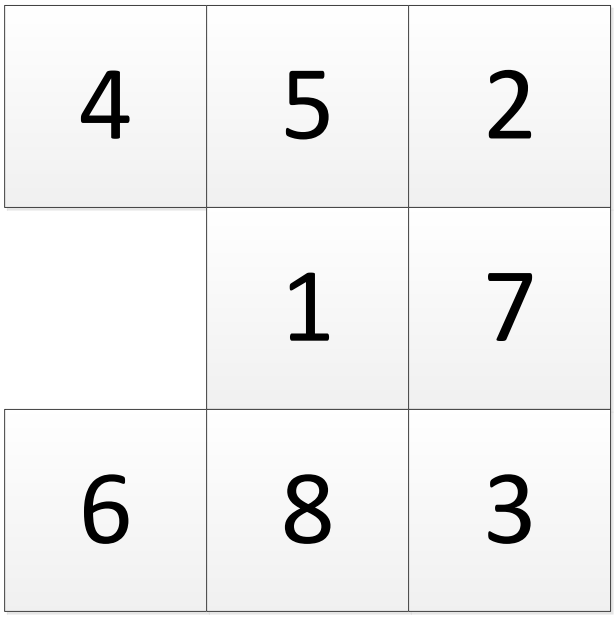


Figura 4.1 Rompecabezas de 3x3

Como se puede observar, el espacio vacío se encuentra en la posición 3 y las únicas fichas que se pueden mover son la 4 (posición 0), la 1 (posición 4) y la 6 (posición 6). Si se desea mover la ficha 6 desde el Wiimote, entonces se debe presionar el botón Up del D-Pad. Cuando se hace presión en ese botón, automáticamente se hace una búsqueda (con el método FichaArriba de la clase GrasRompecabezas) la ficha que se encuentra “debajo” del espacio vacío. Al encontrarla, llama al método ActivarFichaJuego de la clase ControladorJuego, que es la que hace todo el movimiento de las fichas.

Clase Parámetros

Es una clase estática donde se guardan datos estáticos que se usarán durante todo el programa. Se cargan o bien desde el archivo de configuración, o se encuentran quemadas en el código. La tabla 4.18 muestra sus características

Tabla 4.18 Clase estática Parametros

Name	Type	Modifier	Summary
Methods			
MezclarLista<T>	void	public	Mezcla aleatoriamente una lista
<add method>			
Properties			
<add property>			
Fields			
Animales	string	public	Directorio donde se encuentra el archivo de los datos de animales
ArchivoPuntaje	string	public	Nombre del archivo donde se guarda los puntajes del juego
CadenaConfiguracionArchivoPuntaje	string	public	Dirección donde se encuentra el archivo de puntaje del juego
Configuracion	string	public	Dirección donde se encuentra el archivo de configuración
CorreoEntregaClave	string	public	Correo electrónico de contacto
DiasPeriodoPrueba	int	public	Número de días en los que funcionará sin clave el juego
ImagenCompleta	string	public	Dirección donde se encuentra las Imágenes Completas del juego
MensajeEtiquetaGano	string	public	Mensaje de esta etiqueta
MensajeEtiquetaNuevo	string	public	Mensaje de esta etiqueta
MensajeEtiquetaSalir	string	public	Mensaje de esta etiqueta
MensajeTituloError	string	public	Mensaje de esta etiqueta
MensajeTituloGano	string	public	Mensaje de esta etiqueta
MensajeTituloNuevo	string	public	Mensaje de esta etiqueta
MensajeTituloSalir	string	public	Mensaje de esta etiqueta
NombreArchivoSeguridad	string	public	Nombre del Archivo de seguridad
NombreArchivoSeguridadOculto	string	public	Nombre del Archivo de seguridad de respaldo
NumeroUsosAplicacion	int	public	Número de veces en los que funcionará sin clave el juego
Sonidos	string	public	Dirección en donde se encuentra los sonidos del juego
TelefonoEntregaClave	string	public	Teléfono de contacto

4.1.3. Creación del Componente de Seguridades

El componente de Seguridad se encarga de comprobar si el producto Puzzlemote instalado está registrado o no en el sistema. Este componente se basa en 2 aspectos, control para la vista, y control de registro.

4.1.3.1. Control de Registro

El control de registro se encarga de manejar archivos planos y el registro del sistema operativo, para con ello guardar la llave de seguridad de registro. Lo que hace este subcomponente es generar un archivo oculto dentro de una carpeta del sistema, donde se guarde el id del producto, además de guardar la clave de registro si está hecha. Al iniciar el programa, esta control se realizará siempre; así si el sistema no está registrado, entonces mostrará una ventana pidiendo el registro del sistema, o si no que se use el período de prueba del producto. Se ha decidido que tiempo de prueba es de 30 días o 200 usos durante esos 30 días. Si el usuario ha sobrepasado el tiempo o los usos, entonces ya no podrá usar el producto.

Para implementar este subcomponente, se decidió utilizar un código de software basado en C#, llamado TrialMaker. TrialMaker posee todas las características necesarias, permitiendo al desarrollador colocar el código en su fuente, y usarlo sin restricciones. TrialMaker viene con Serial Maker, otro software gratuito y con fuentes basado en C#, el cual produce un ejecutable que al ingresar un ID, genera automáticamente el código serial de un producto relacionado. Las tablas 4.19. 4.20 4.21, 4.22 y 4.23 muestran la estructura que posee los componentes mencionados con anterioridad.

Tabla 4.19 Clase Encryption

Name	Type	Modifier	Summary
Methods			
Boring	string	public	mueve caracteres a una cadena
ChangeChar	char	private	
ConvertToLetterDigit	string	public	
InverseByBase	string	public	
InverseString	string	public	
MakePassword	string	public	

Tabla 4.20 Clase FileReadWrite

Name	Type	Modifier
Methods		
ReadFile	string	public
WriteFile	void	public
<add method>		
Properties		
Fields		
iv	byte[]	private
key	byte[]	public

Tabla 4.21 Clase SystemInfo

Name	Type	Modifier
Methods		
GetSystemInfo	string	public
RemoveUseLess	string	private
RunQuery	string	private
<add method>		
Properties		
Fields		
UseBaseBoardManufac	bool	public
UseBaseBoardProduct	bool	public
UseBiosManufacturer	bool	public
UseBiosVersion	bool	public
UseDiskDriveSignature	bool	public
UsePhysicalMediaSeria	bool	public
UseProcessorID	bool	public
UseVideoControllerCap	bool	public
UseWindowsSerialNum	bool	public

Tabla 4.22 Atributos de la Clase TrialMaker

Name	Type	Modifier
Fields		
_BaseString	string	private
_DefDays	int	private
_HideFilePath	string	private
_Identifier	string	private
_Password	string	private
_RegFilePath	string	private
_Runed	int	private
_SoftName	string	private
_Text	string	private

Tabla 4.23 Métodos y Propiedades de la Clase TrialMaker

Name	Type	Modifier	Summary
Methods			
CheckHideFile	int	private	
CheckRegister	bool	private	
DaysToEnd	int	private	
MakeBaseString	void	private	
MakeHideFile	void	private	
MakePassword	void	private	
MakeRegFile	void	private	
SetDefaults	void	private	
ShowDialog	RunTypes	public	Show registering dialog to user
TrialMaker		public	Make new TrialMaker class to make software trial
<add method>			
Properties			
HideFilePath	string	public	Indicate file path for storing hidden information
RegFilePath	string	public	Indicate File path for storing password
TrialPeriodDays	int	public	Get default number of days for trial period
TripleDESKey	byte[]	public	Get or Set TripleDES key for encrypting files to save
UseBaseBoardManufacturer	bool	public	
UseBaseBoardProduct	bool	public	
UseBiosManufacturer	bool	public	
UseBiosVersion	bool	public	
UseDiskDriveSignature	bool	public	
UsePhysicalMediaSerialNumber	bool	public	
UseProcessorID	bool	public	
UseVideoControllerCaption	bool	public	
UseWindowsSerialNumber	bool	public	

Para mostrar el mensaje de registro, se ha de utilizar un formulario llamado frmDialog, el cual muestra la información de registro del usuario, así como la

facilidad de ingresar el número de serie. También se podrá ver el tiempo que queda del período de prueba y el número de usos restantes que faltan para que se termine dicho periodo. Finalmente, se deberá poder ingresar con o sin registro del producto (en el segundo caso, solamente si existe aún período de prueba vigente). La tabla 4.24 muestra la clase frmFormulario, la figura 4.5 muestra en cambio la interfaz que tendrá esta ventana.

Tabla 4.24 Formulario de Mensaje frmDialog

Name	Type	Modifier
Methods		
btnOK_Click	void	private
btnTrial_Click	void	private
Dispose	void	protected
frmDialog		public
InitializeComponent	void	private
<add method>		
Properties		
Fields		
_Pass	string	private
btnOK	Button	private
btnTrial	Button	private
components	IContainer	private
grbRegister	GroupBox	private
grbTrialRunning	GroupBox	private
lblCallPhone	Label	private
lblComment	Label	private
lblDays	Label	private
lblDaysToEnd	Label	private
lblID	Label	private
lblRunTimesLeft	Label	private
lblSerial	Label	private
lblText	Label	private
lblTimes	Label	private
sebBaseString	SerialBox	private
sebPassword	SerialBox	private

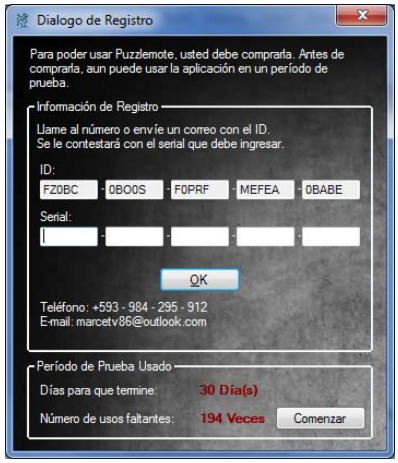


Figura 4.2 Interfaz de período de prueba

4.1.3.1. Control de Registro

El control de registro es el que llama a la funcionalidad del TrialMaker, asegurándose que el sistema tenga o no registrado el producto. Para ellos utiliza la clase PeriodoPrueba. Esta clase inicializa los datos de control de registro (datos del mensaje a mostrar, direcciones de los archivos, número de días que quedan al periodo de prueba y número de usos que quedan del mismo). La tabla 4.25 muestra las características que posee esta clase.

Tabla 4.25 Clase PeriodoPrueba

Name	Type	Modifier	Summary
Methods			
ComprobarRegistroMaquina	bool	public	Comprueba si el sistema está registrado
ObtenerDireccionMAC	byte[]	private	Obtiene una id a base de la MAC de la máquina
PeriodoPrueba		public	Permite comprobar al sistema si está o no registrado
StrToByteArray	byte[]	public	Convierte un cadena de caracteres en un arraglo de bytes
<add method>			
Properties			
DatosMensajeVentana	string	public	Datos de contacto
DireccionArchivoOculto	string	public	Dirección del archivo oculto
DireccionArchivoRegistro	string	public	Dirección donde se encuentra ese archivo
NumeroDiasPeriodoPrueba	int	public	Días que tiene un usuario para activar el producto
NumeroUsosAplicacion	int	public	Número de veces que usuario aún puede abrir en período de prueba la aplicación
<add property>			
Fields			
ObjTrialMaker	TrialMaker	private	Objeto que genera el registro

4.2. CONSTRUCCIÓN DE INTERFAZ PUZZLEMOTE

Como se ha especificado en el modelo navegacional, la interfaz se divide en una clase (Program), que se encarga de mostrar la ventana adecuada de acuerdo a la configuración guardad del sistema; y tres formularios que corresponden a cada nivel que posee el juego (el formulario frmRompecabezas9 de 8-puzzle, el frmRompecabezas de 15-puzzle y el frmRompecabezas25 de 24-puzzle).

Los 3 formularios a su vez contienen “sub-ventanas”, hechas con el Control GroupBox, que muestran la “Configuración del Juego”, la ventana “Acerca del Juego, y dentro de esta, la opción de levantar la animación de ayuda creada en el programa Prezi.

4.2.1. Construcción de los Formularios

Los formularios del juego poseen la misma estructura de diseño (figuras4.3, 4.4 y 4.5), la cual se divide en:

- Control de Nombre de Jugador (el cual posee un control para cambiar el nombre).
- Contenedor del Tiempo Transcurrido en una partida.
- Contenedor del Rompecabezas.
- Imagen de Ayuda para armar el rompecabezas.
- Controles del Juego (Nuevo Juego, Menú de Configuración y Salir del Juego).

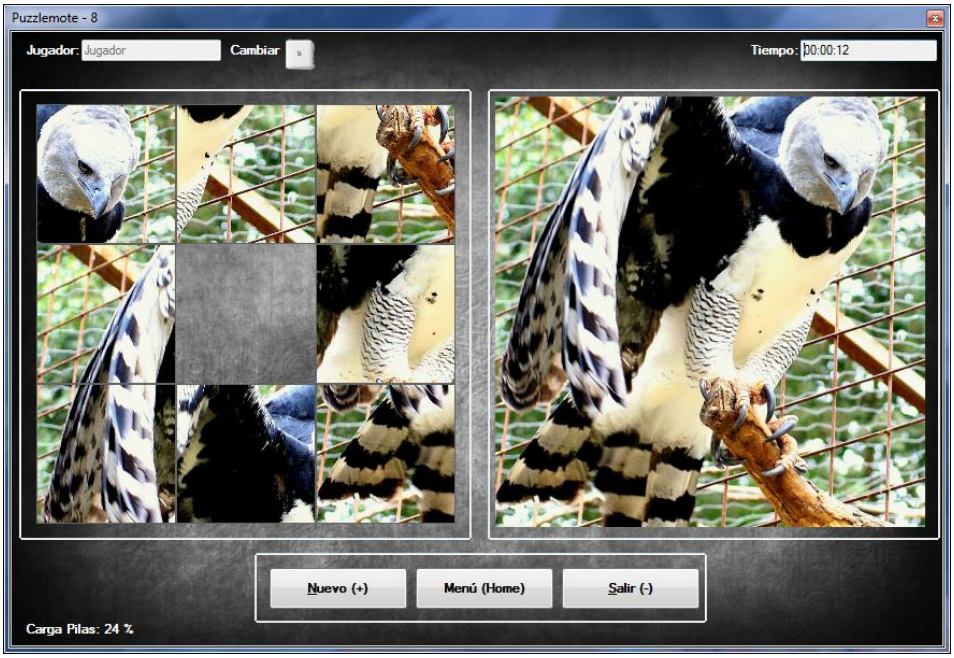


Figura 4.3 Puzzlemote – 8

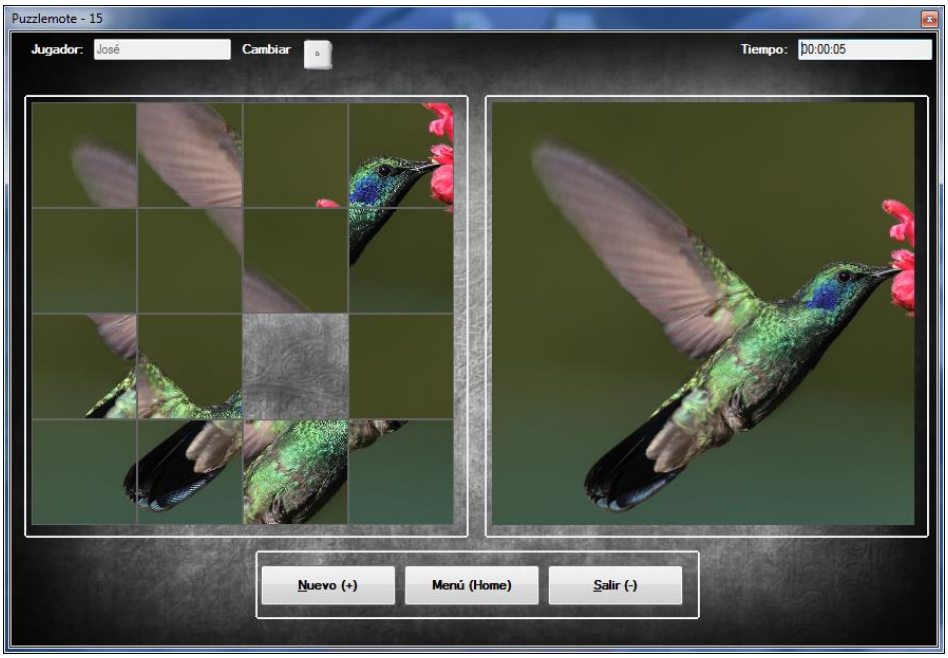


Figura 4.4 Puzzlemote – 15

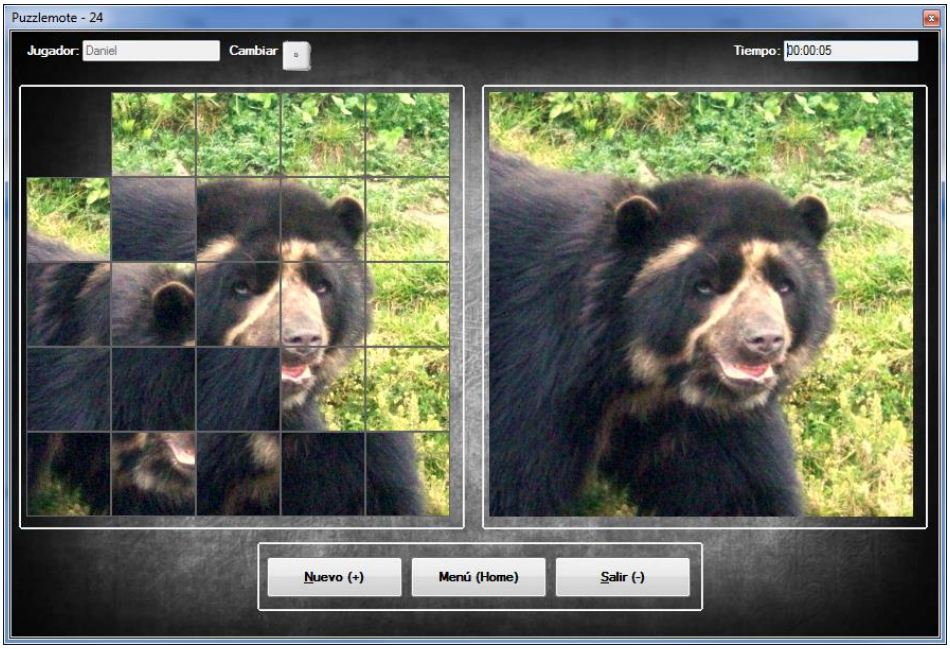


Figura 4.5 Puzzlemote – 24

Si se selecciona Nuevo, el juego mostrará una ventana de Selección de Imagen para cargar en el nuevo juego (figura 4.6). En ella se puede cambiar las configuraciones del juego, así como ver puntajes, armar el rompecabezas solo y ver la ventana de ayuda.

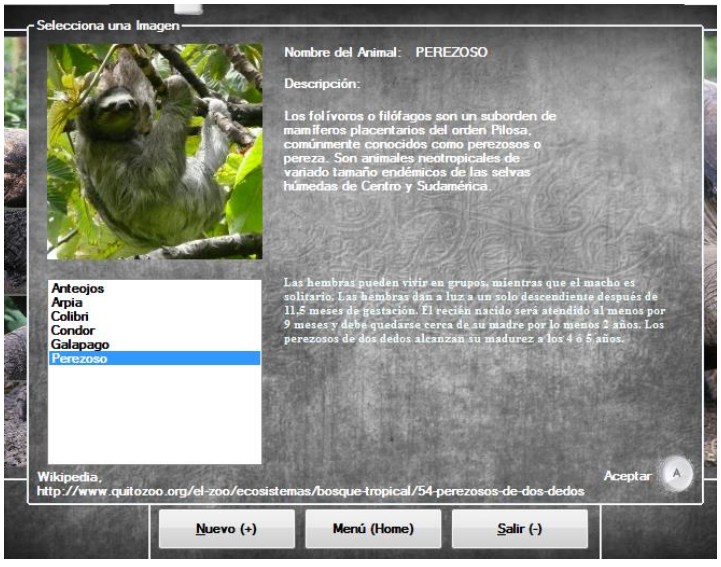


Figura 4.6 Seleccionar Imagen

Si se selecciona Menú, el juego mostrará una ventana de Configuración del Juego (figura 4.7).

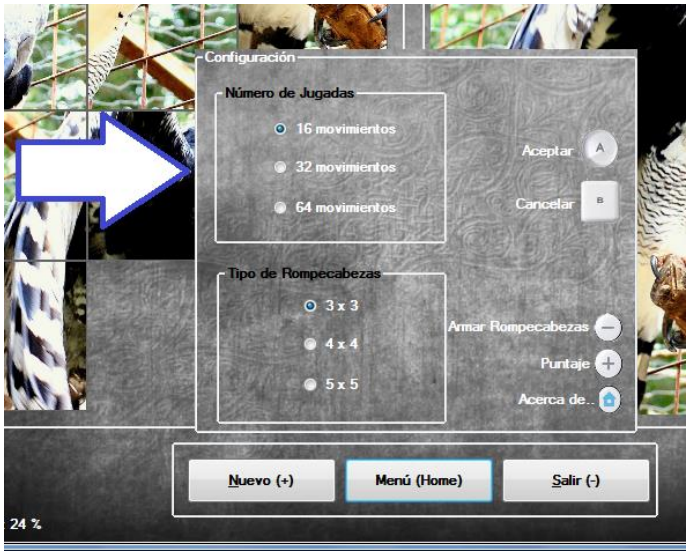


Figura 4.7 Menú de Configuración

Si dentro del anterior Menú, se selecciona Puntaje, aparecerá una ventana con la información de puntajes guardados.

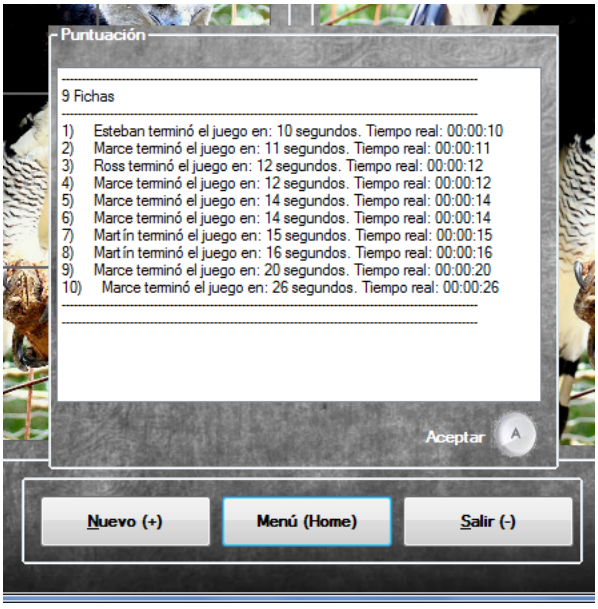


Figura 4.8 Puntaje del Usuario

Si dentro de Configuración se selecciona la opción de “Armar Rompecabezas”, se iniciará la animación de que se arma solo el formulario.

Si dentro de Configuración se selecciona la opción de “Acerca de…”, entonces se mostrará la ventana “Acerca de Puzzlemote” (figura 4.9).

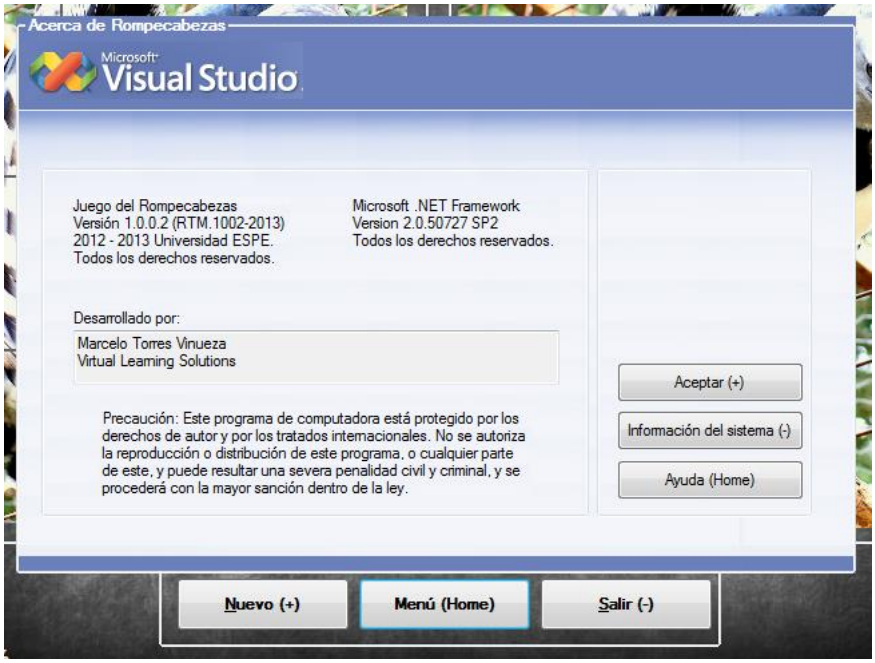


Figura 4.9 Acerca de Puzzlemote

Si dentro de Acerca de Puzzlemote se selecciona la opción “Ayuda”, se cargará la presentación hecha en archivo ejecutable de Flash, diseñado en el programa Prezi.

4.2.2. Desarrollo de la Clase de Control de Formularios

Cada formulario posee una clase que administra los controles y los gráficos. Además, como ya se ha descrito en anteriores ocasiones, existe la clase Program,

la cual posee como función el método de ejecución de los formularios. Si el producto Puzzlemote está activado, se ejecutarán los formularios de acuerdo a las configuraciones del archivo plano "*configuration*". Si no se encuentra activado, entonces se mostrara el formulario frmDialog de Registro de Producto.

En cada formulario se instancia a dos clases: las clases ControladorJuego y ControladorWiimote. Estos objetos permiten al juego comportarse de acuerdo a los controles creados con anterioridad. En la figura 4.10 se puede ver la relación de conexión entre los componentes ControladorJuego, ControladorWiimote y uno de los formularios.

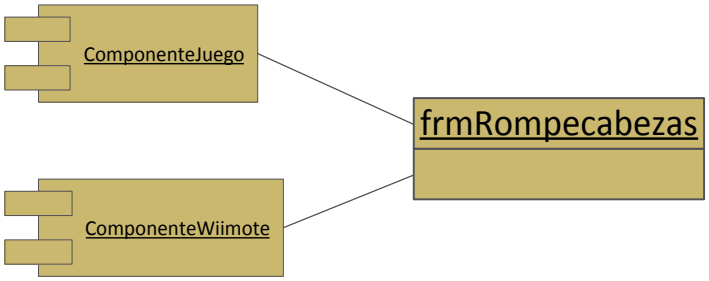


Figura 4.10 Relación entre los componentes Wiimote y Juego con frmRompecabezas

El objeto ControladorJuego hace todo la parte ingreso de imágenes, construcción de grafos de movimientos, iniciar el sonido, inicia el juego, etc. Además de mover la ficha de acuerdo a lo que pide un usuario.

El objeto ControladorWiimote permite hacer lo que hace el ControladorJuego, pero desde el Control remoto del Wii.

En la tabla 4.26 se muestra los métodos que posee el formulario frmRompecabezas9.

Tabla 4.26 Métodos de la clase frmFormulario9

Name	Type	Modifier	Summary
Methods			
btnAceptar_Click	void	private	Evento clic del botón Aceptar
btnAcercaAceptar_Click	void	private	Evento clic del contenedor de imagen Cerrar Acerca de Puzzlemote
btnArmarRompecabezas_Click	void	private	Evento clic del botón Mostrar Menú Configuración
btnAyuda_Click	void	private	Evento clic del contenedor de imagen Ayuda de Puzzlemote
btnInformacionSistema_Click	void	private	Evento clic del contenedor de imagen Información del Sistema
btnNo_Click	void	private	Evento clic del botón No
btnNuevo_Click	void	private	Evento clic del botón Nuevo
btnSalir_Click	void	private	Evento clic del botón Salir
btnSi_Click	void	private	Evento clic del botón Si
CargarImagenAyuda	void	private	Carga la imagen de ayuda del rompecabezas seleccionado
CerrarJuego	void	private	Muestra la ventana anterior a salir de aplicación
DestruirConexiones	void	private	Destruye las conexiones hechas con todos los Wiimote conectados a al aplicación
Dispose	void	protected	Clean up any resources being used.
frmRompecabezas9		public	Formulario que contiene el rompecabezas de 8 fichas
frmRompecabezas9		public	Formulario que contiene el rompecabezas de 8 fichas. Incializa las configuraciones guardadas
frmRompecabezas9_FormClosing	void	private	Evento anterior al cierre de un formulario
frmRompecabezas9_KeyDown	void	private	Evento de presión de teclas Alt + F4
frmRompecabezas9_Load	void	private	Evento de inicio del formulario
IngresarDatosLista	void	private	Inserta en una lista de control las fichas del rompecabezas
IngresarImagenesLista	void	public	Inserta en una lista de control las direcciones de la imagen seleccionada
InicializarControlWiiMote	void	private	Inicia los controles de conexión con un Wiimote
Iniciar	void	private	Activa la animación de entrada del Formulario
InitializeComponent	void	private	Required method for Designer support - do not modify
lstAnimales_SelectedIndexChanged	void	private	Evento cambio de item en la lista de Selección de Imágenes
ManejaWiimote	void	public	Inicia y controla la conexión con el Wiimote
NuevoJuego	void	private	Muestra la ventana anterior al inicio de nuevo juego
pcbCambiar_Click	void	private	Evento clic del contenedor de imagen Cambiar Nombre de Usuario
pcbConfiguracionA_Click	void	private	Evento clic del contenedor de imagen Aceptar de Configuración
pcbConfiguracionB_Click	void	private	Evento clic del contenedor de imagen Cancelar de Configuración
pcbConfiguracionHome_Click	void	private	Evento clic del contenedor de imagen cargar Acerca de Puzzlemote
pcbConfiguracionMinus_Click	void	private	Evento clic del contenedor de imagen Configuración
pcbConfiguracionPlus_Click	void	private	Evento clic del contenedor de imagen Cargar Puntuación
pcbFicha_0_Click	void	private	Evento clic del contenedor de Imagen Ficha 0
pcbFicha_1_Click	void	private	Evento clic del contenedor de Imagen Ficha 1
pcbFicha_2_Click	void	private	Evento clic del contenedor de Imagen Ficha 2
pcbFicha_3_Click	void	private	Evento clic del contenedor de Imagen Ficha 3
pcbFicha_4_Click	void	private	Evento clic del contenedor de Imagen Ficha 4
pcbFicha_5_Click	void	private	Evento clic del contenedor de Imagen Ficha 5
pcbFicha_6_Click	void	private	Evento clic del contenedor de Imagen Ficha 6
pcbFicha_7_Click	void	private	Evento clic del contenedor de Imagen Ficha 7
pcbFicha_8_Click	void	private	Evento clic del contenedor de Imagen Ficha 8
pcbPuntuacionAceptar_Click	void	private	Evento clic del contenedor de imagen Aceptar de Puntuación
pictureBox3_Click	void	private	Evento clic del contenedor de imagen Aceptar de Selección de Imagen
tmrControlMovimiento_Tick_1	void	private	Evento transcurso de intervalo de timer Control de Movimiento
tmrControlMusica_Tick_1	void	private	Evento transcurso de intervalo de timer Música
tmrCronometro_Tick_1	void	private	Evento transcurso de intervalo de timer Cronómetro
tmrEntrada_Tick	void	private	Evento transcurso de intervalo de timer Control de Entrada
vm_WiimoteChanged	void	public	Evento de conexión con un Wiimote. Comprueba cada cierto intervalo de tiempo los cambios hechos en el control

Los otros 2 formularios son iguales, solo cambia el constructor y aumenta el número de fichas de 9 (8-puzzle), 16 (15-puzzle) y 25 (24-puzzle).

4.3. DESARROLLO DE PRUEBAS DE PUZZLEMOTE

Se ha establecido los siguientes parámetros para hacer las pruebas en el producto Puzzlemote:

- Pruebas de Rendimiento. – Se refiere al desempeño físico del producto Puzzlemote, cuántos recursos consume en el sistema, y sí cumple de manera eficaz los requisitos de hardware que se ha pedido del mismo.
- Pruebas de Campo. – Se refiere al desempeño lógico del producto Puzzlemote, cómo es recibido por el usuario final, y sí cumple con los requisitos que la Metodología OOHDM para el desarrollo de software visual.

4.3.1. Pruebas de Rendimiento

Para realizar las pruebas de rendimiento del producto Puzzlemote, se hará uso de medidor de consumo de software que viene integrado con IDE Visual Studio 2010, el cual se basa en el uso que se realiza de cada uno de los métodos y parámetros que utiliza el programa durante su ejecución en tiempo real.

Las pruebas fueron realizadas en una computadora que posee un CPU con procesador Inter Core 2 Duo de 1.80 GHz y de 2 GB de memoria RAM, en un sistema operativo Microsoft Windows 7 Ultimate de 64 bits durante 6 minutos con 20 segundos (380 segundos).

4.3.1.1. Pruebas de Rendimiento de CPU

La figura 4.11 muestra el rendimiento del CPU durante la fase pruebas.

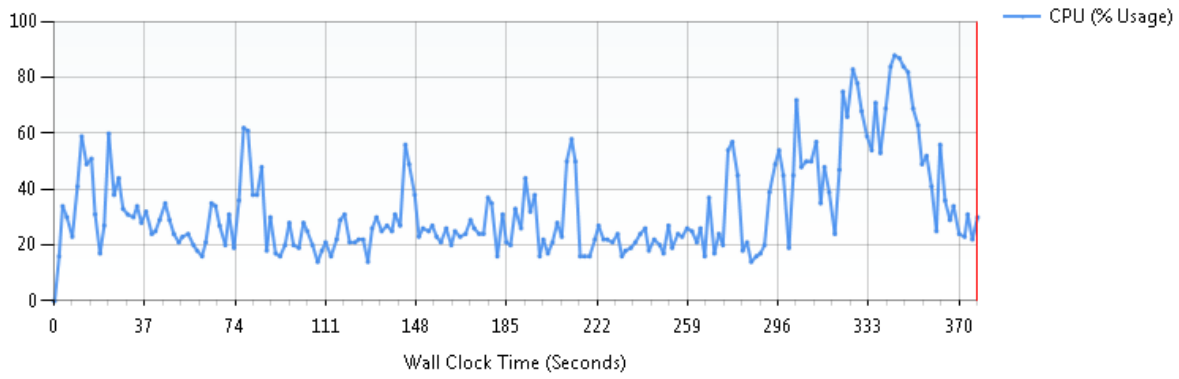


Figura 4.11 Rendimiento del CPU durante la Ejecución de Puzzlemote

Como se puede observar, el CPU mantuvo un promedio de uso de 60%.

4.3.1.2. Funciones más utilizadas

Las funciones más utilizadas durante el proceso fueron método Main de la clase Program en un 80%, seguido por el método del sistema Application.Run. La tabla 4.27 muestra los 10 métodos que más usaron CPU.

Tabla 4.27 Los 10 métodos que más usan el CPU

Nombre de la Función	Datos Inclusivos	Datos Exclusivos	% Inclusivos	% Exclusivos
WinAppRompecabezas.Program.Main()	1.146	0	80,48	0
System.Windows.Forms.Application.Run()	1.096	880	76,97	61,8
WiimoteLib.Wiimote.OnReadData()	277	6	19,45	0,42
ComponenteJuego.ControladorWiimote.ActualizarEstadoWiimoteChanged()	200	6	14,04	0,42
System.IO.FileStream.EndRead()	97	97	6,81	6,81

4.3.1.3. Árboles de Llamado de Paquetes

La tabla 4.28 muestra el árbol de llamadas de los métodos de cada paquete que se han usado durante el tiempo de ejecución.

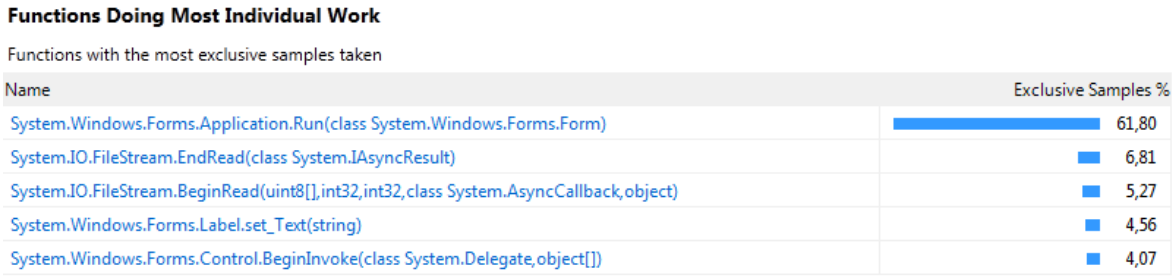
Tabla 4.28 Árbol de Llamado de Métodos durante el tiempo de ejecución

Nombre de Método	Datos Inclusivos	Datos Exclusivos	% Inclusivos	% Exclusivos	Nombre Módulo
- Puzzlemote.exe	1.424	0	100,00	0,00	
- WinAppRompecabezas.Program.Main()	1.146	0	80,48	0,00	Puzzlemote.exe
- System.Windows.Forms.Application.Run()	1.096	880	76,97	61,80	System.Windows.Forms.ni.dll
ComponenteJuego.ControladorWiimoteActualizarEstadoWiimoteChanged()	200	6	14,04	0,42	ComponenteJuego.dll
- WiimoteLib.Wiimote.OnReadData()	277	6	19,45	0,42	WiimoteLib.dll
System.IO.FileStream.EndRead()	97	97	6,81	6,81	mscorlib.ni.dll
- WinAppRompecabezas.frmRompecabezas.vm_WiimoteChanged()	79	4	5,55	0,28	Puzzlemote.exe
ComponenteJuego.ControladorWiimoteActualizarEstadoControlPictureBoxWiimote()	75	7	5,27	0,49	ComponenteJuego.dll
- WiimoteLib.Wiimote.BeginAsyncRead()	78	0	5,48	0,00	WiimoteLib.dll
System.IO.FileStream.BeginRead()	75	75	5,27	5,27	mscorlib.ni.dll

De esto se puede concluir que los módulos que más se usan son: el del Formulario (Vista), el componente del Juego (controlador), el componente de Wiimote (conexión al control). Como dato adicional, el producto utiliza continuamente la lectura de archivos planos.

4.3.1.4. Métodos que Realizan el Mayor Número de Trabajo Individual

La figura 4.12 muestra los métodos que realizan el mayor número de trabajos individuales.



Functions Doing Most Individual Work

Functions with the most exclusive samples taken

Name	Exclusive Samples %
System.Windows.Forms.Application.Run(class System.Windows.Forms.Form)	61,80
System.IO.FileStream.EndRead(class System.IAsyncResult)	6,81
System.IO.FileStream.BeginRead(uint8[],int32,int32,class System.AsyncCallback,object)	5,27
System.Windows.Forms.Label.set_Text(string)	4,56
System.Windows.Forms.Control.BeginInvoke(class System.Delegate,object[])	4,07

Figura 4.12 Métodos que realizan el mayor número de trabajo individual

Del estudio se puede concluir que el método que más trabajo realiza solo, por obvias razones, es el método Run (61.80 %), el cual es el que inicializa la aplicación. Le sigue de cerca el cerrado de archivo planos de la función EndRead (6.81 %), el inicio de lectura de archivos planos de la función BeginRead (5.27 %).

4.3.2. Pruebas de Campo

Para realizar las pruebas de campo, se sigue el proceso de verificación de existencia de una adecuada navegabilidad, colores estandarizados y fuentes correctas. Con esto se puede llegar a seguir el siguiente esquema de verificación.

- Diseño de Interfaz
- Diseño Estético

- Diseño de Contenido
- Diseño de Navegación

4.3.2.1. Diseño de Interfaz

Para esto se tomó en cuenta lo siguiente:

- **Enlaces. –** Lo referente a las conexiones que realiza la aplicación, sean internamente (llamado a otros formularios) o externos (aplicaciones de ayuda externa). En este punto se pudo obtener como resultado que en la aplicación, la llamada de un formulario se hace a partir del cambio de configuración de tipo de juego, solamente cuando se hace ese cambio, la aplicación se cierra y se vuelve a abrir con la nueva configuración. La llamada a aplicaciones externas se da solamente en el momento de selección de ayuda en el producto, llamando a una presentación ejecutable.
- **Formato. –** Como recibe la PC la información y si ésta no se pierde en el proceso de ejecución. En este punto se obtuvo que el producto mostró todos los datos que supone mostrar, sin pérdida alguna.
- **Ventanas Dinámicas. –** Como se maneja la construcción y destrucción de los objetos durante el tiempo de ejecución. Aquí se pudo obtener como resultado que la aplicación construyo todos los objetos al momento de inicio del producto, y al momento de cerrarse los objetos no quedaron guardados en memoria.

4.3.2.2. Diseño de Estético

Para este punto se toma en cuenta el grado de usabilidad del sistema (si es intuitivo o no), facilidad de búsqueda de información, la interacción con el usuario es de su agrado, y el despliegue de la pantalla (que tipo de resolución).

Con estos puntos tomados en cuenta, se pudo obtener de Puzzlemote que:

- El sistema es intuitivo en su gran parte, teniendo las ayudas visuales necesarias según su caso. En las partes que no han sido entendidas, se ha logrado su corrección de acuerdo a las necesidades presentadas en los usuarios de prueba.
- La búsqueda de información se facilita con el manual de usuario.
- El despliegue en pantalla es el adecuado.

4.3.2.3. Diseño de Contenido

El contenido se ha medido basado en las pruebas hechas en usuarios a los que está dirigido el proyecto. Los usuarios no realizaron ninguna observación.

4.3.2.4. Diseño de Navegación

Los usuarios de prueba hicieron la observación del poco entendimiento de la dinámica Wiimote-Juego. Esto se corrigió al mejorar la estética de los controles que contienen las imágenes de las fichas de los rompecabezas.

4.5. DESPLIEGUE DEL PRODUCTO

El despliegue se realizó generando un archivo de instalación con la herramienta de generación de instaladores que posee el IDE Visual Studio 2010.

CAPÍTULO 5: CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

- La inteligencia artificial se puede aplicar en el proceso de enseñanza aprendizaje, permitiendo a profesores y estudiantes fomentar su razonamiento lógico, el que permitirá resolver problemas complejos, sirviendo de guía a futuro, además que organiza el conocimiento disponible en la materia en la que se aplique, lo que permite encontrar soluciones de una manera más rápida y segura.
- El relativo y creciente uso de la computación en la Educación está más relacionado con el impacto que la informática ha tenido en el mundo moderno, y continuará teniendo. Hay que ver a la computadora como un medio complementario, pero este medio debe superar las limitaciones de los medios educativos convencionales, enfrentando el reto que le imponen los últimos avances tecnológicos.
- Limitar el concepto de videojuego a una actividad exclusivamente lúdica supone obviar las potencialidades instructivas o educativas del mismo, estudiadas a partir de numerosas investigaciones. Sin embargo, no se debe olvidar que estos videojuegos didácticos pueden llegar a perder el sentido

propio del juego desde el momento en que quien los utilice lo haga con el objetivo de aprender y no por el simple hecho de jugar.

- No todos los videojuegos son iguales y, lógicamente, no todos presentan el contenido violento de la misma forma; por ejemplo, no es igual un videojuego de violencia fantástica que uno de violencia humana. Así mismo, se puede concluir que, como toda actividad, los videojuegos generan polémica, porque si su contenido no es el apropiado, puede crear daño en el perfil psicológico de un individuo (sin importar la edad, sexo o estructura social). El interés por demostrar los efectos de los videojuegos, ya sean positivos o negativos, ha provocado la mayoría de las investigaciones desarrolladas sobre el tema.
- El Wiimote se convierte en un instrumento que mejora la investigación a nivel educativo, permitiendo a personas que no poseen los recursos económicos necesarios para acceder componentes parecidos, hacer estudios sobre el infrarrojo, sensores de movimiento, reconocimiento de voz conexiones Bluetooth, etc.
- La metodología OOHDM es apropiada para desarrollo de sistemas complejos en un ámbito de diseño de interfaces de usuario, ya que separa el diseño del desarrollo de la aplicación. Es comparable al desarrollo generado por RUP, ya que es tan detallado como él. Las ventajas que ofrece son una clara identificación de los diferentes niveles de diseño en forma independiente de la implementación, la forma de gráfica que se usa para representar los diseños es bastante amigable y fácil de realizar, etc. Se integra fácilmente a la arquitectura Modelo Vista Controlador.

- La tecnología GDI+ es una herramienta inherente en Windows, que permite al Sistema Operativo la construcción de gráficos a partir de lo que un programa o el usuario pide, y sin importar la marca de la tarjeta de video (en el mercado existen una gran variedad de tarjetas, las cuales tienen su propio lenguaje e instrucciones). GDI está basado en la compilación antigua de Windows (hecha en C), por lo que Microsoft desarrollo en los últimos años GDI+, la cual simplemente agrega una capa a GDI, que permite un mejor manejo de objetos y clases heredadas.
- La idiosincrasia de nuestro país, impide que se haga uso de interfaces que hagan una referencia del folklor nacional en la producción de software. De la misma manera, el tema de los videojuegos que se desarrollen en Ecuador, es una idea que no es bien recibida por parte de la gran mayoría de programadores, desarrolladores o arquitectos del país.

5.2. RECOMENDACIONES

5.2.1. Recomendaciones del Proyecto

- El producto Puzzlemote puede llegar a ser utilizado en ventas del campo educativo para instituciones de primer nivel, guarderías y escuelas de hasta 5to año de educación básica. Por lo que se recomienda realizar marketing sobre el producto a estos targets de mercado.
- Se recomienda utilizar pilas recargables para el Wiimote, así se llega a tener un ahorro más con este dispositivo, que es muy dependiente de pilas (si se

juega más de 3 horas al día, las pilas se acabarán en una semana), además de ayudar al reciclaje de energía.

- La conexión de Wiimote con la PC puede llegar a ser un poco difícil si no se tiene la costumbre de hacerlo. Windows no puede realizar de una manera correcta la conexión Bluetooth por problemas instalación de controladores (drivers). Esto se arregla si se saca las pilas del Wiimote, se espera 10 segundos, y de nuevo se intenta hacer la conexión.

5.2.2. Recomendaciones Académicas

- Wiimote es una herramienta que permite investigaciones con los complementos internos que posee, además de tener un precio asequible y de los cuales hay en mercado productos genéricos. Se recomienda que se utilice más este dispositivo para enseñanza de tercer nivel en materia de dispositivos innovadores.
- Se recomienda que se implemente en la malla curricular, investigaciones sobre proyectos de vanguardia, que permitan a los alumnos de la carrera de Ingeniería en Sistemas estar al día con tecnologías que no se encuentran con facilidad en el país.
- Así mismo, se recomienda que se agregue a la malla, tópicos en materia de programación avanzada para videojuegos, dispositivos de entrada y de salida poco comunes (como Wiimote, Kinnetic de Microsoft, etc.), los cuales ayuden a la investigación que debe poseer la carrera.
- Finalmente, se debe agregar en la malla curricular o cambiar las áreas de generación de interfaces de usuario, la conciencia de usar objetos u otros

atributos nacionales, los cuales hagan fácil la identificación de nuestra cultura tanto a usuarios nacionales como internacionales.

BIBLIOGRAFÍA

Autores, V. (2012). *Sistemas Inteligentes. Sistemas Basados En Reglas.* Oviedo: Universidad De Oviedo.

Bello, R. (2002). *Aplicaciones De La Inteligencia Artificial.* Guadalajara: Universidad De Guadalajara.

Blackman, S. (2011). *Beginning 3d Game Development With Unity.* New York: Apress.

Bloom, S. (1982). *Video Invaders.* Nueva York: Arco Publishing.

Brüngger, A., Marzetta, A., Fukuda, K., & Nievergelt, J. (1999). The Parallel Search Bench Zram And Its Applications. *Annals Of Operations Research*, 90.

Césari, M. (2012). *Sistemas Experto.* Mendoza: Facultad Regional De Mendoza, Laboratorio Dharma.

Durkin, J. (1994). *Expert Systems. Design And Development.* México: Prentice Hall Inter-National.

García, R., Quirós, J., González, S., Santos Gallego, R., & Fernández, S. (S.F.). *Diseño Gráfico De Contenidos Para Internet.* Pearson Prentice Hall.

Hannas, L. (1972). The English Jigsaw Puzzle. Wayland.

Johnson, W., Woolsey, S., & William, E. (1879). Notes On The "15" Puzzle. *American Journal Of Mathematics*, 25-30.

Kastner, J., & Hong, S. (1984). A Review Of Expert Systems. *European Journal Of Operational Research*.

Keller, E., Palamar, T., & Honn, A. (2010). *Mastering Autodesk Maya 2011.* New York: Sybex.

Kendall, G., & Parkes, A. S. (2008). *A Survey Of Np-Complete Puzzles.* Obtenido De Uk Campus: Http://Www.Cs.Nott.Ac.Uk/~Gxk/Papers/Icga2008_Preprint.Pdf

Lamarca, M. (2006). *Hipermedia/Multimedia.* Madrid: Universidad Complutense De Madrid.

Lee, J. (2009). *Hacking The Nintendo Wii Remote.* Pervasive Computing, Ieee. Obtenido De Hacking The Nintendo Wii Remote

Lee, J. (14 De Septiembre De 2012). *Wii Project Site*. Obtenido De Http://Www.Cs.Cmu.Edu/~Johnny/Projects/Wii/

Lenat, D. (1990). *Building Large Knowledge Based Systems.* México: Addison-Wesley.

León, M., & García, Z. (2008). La Inteligencia Artificial En La Informática Educativa. *Revista De Informática Educativa Y Medios Audiovisuales Vol. 5*.

Pressman, R. (2002). *Ingeniería De Software Un Enfoque Práctico.* Madrid: Mc. Graw Hill.

Ratner, D. W. (1990). The (N2-1)-Puzzle And Related Relocation Problems. *Journal Of Symbolic Computation*, 5-10.

Ratner, D., & Warmuth, M. (1986). Finding A Shortest Solution For The N × N Extension Of The 15-Puzzle Is Intractable. *Conferencia Nacional De Inteligencia Artificial.*

Rich, E., & Kevin, K. (1994). *Inteligencia Artificial.* México: Mcgraw Hill.

Rusell, S., & Meter, N. (1996). *Inteligencia Artificial: Un Enfoque Moderno.* México: Prentice Hall.

Schneider Electric Instituto. (2008). *Manual De Formación Unity Pro.* Barcelona: Instituto Schneider Electric.

Selver, S. (2008). *Using Nintendo Wii Remote Control From Finger Tracking, Gesture Detection And As Hci Device.* Graz: Instituto De Sistemas De Información Y Medios Computacionales De La Universidad Tecnológica De Graz – Austria.

Shneiderman, B. (2006). *Diseño De Interfaces De Usuario. Estrategias Para Una Interacción Persona-Computadora Efectiva.* México: Addison Wesley.

Silva, D. (2002). *Construyendo Aplicaciones Web Con Una Metodología De Diseño Orientado A Objetos.* Obtenido De Http://Www.Unab.Edu.Co/Editorialunab/Revistas/Rcc/Pdfs/R22_Art5_C.Pdf

Simon, Robinson, & Otros. (2001). *Professional C#.* San Francisco: Wrox Press.

Slocum, J., & Sonneveld, D. (2006). *The 15 Puzzle.* Beverly Hills: Slocum Puzzle Foundation.

Varios. (2006). *C# Language Specification.* Ecma International.

Varios. (14 De Septiembre De 2012). *Programación De Wiimote Con C#*. Obtenido De Wiibrew: Http://Wiibrew.Org/Wiki/Wiimote

Varios. (20 De Abril De 2013). *Generalidades De Wiimote*. Obtenido De Cnn: Http://Money.Cnn.Com/Magazines/Fortune/Storysupplement/Wiiremote/Index.Htm

Varios. (1 De Mayo De 2013). *Historia De Wii*. Obtenido De Nintendo: Http://Www.Nintendo.Co.Jp/Wii/Index.Html

Varios. (25 De Abril De 2013). *Jigsaw Puzzle History*. Obtenido De Http://Www.Jigsaw-Puzzle.Org/Jigsaw-Puzzle-History.Html

Varios. (30 De Abril De 2013). *Microsoft Visual Studio*. Obtenido De Wikipedia: Http://En.Wikipedia.Org/Wiki/Visual_Studio

Varios. (15 De Abril De 2013). *Wii Linux - Wiimote*. Obtenido De Wiili Wiki: Http://Www.Wiili.Org/Index.Php/Wiimote

Varios. (1 De Abril De 2013). *Wiimote*. Obtenido De Wikipedia: Http://En.Wikipedia.Org/Wiki/Wiimote

Wilson Richard, M. (1974). Graph Puzzles, Homotopy, And The Alternating Group. *Journal Of Combinatorial Theory*, Series B 16: 86–96.

GLOSARIO DE TÉRMINOS TÉCNICOS

Actor: Es un objeto que se encuentran fuera del sistema a modelar. Representan entes que tienen necesidad de intercambiar información con el sistema; pueden ser instanciados por usuarios, dispositivos u otros sistemas.

Antecedente: Es las conjunciones de atributos de un mismo dominio.

Árbol de Decisión: Árbol de decisión es una técnica que permite analizar decisiones secuenciales basadas en el uso de resultados y probabilidades asociadas. Se puede usar para generar sistemas expertos, búsquedas binarias y árboles de juegos.

Archivos Planos XML (*XML Files*): Conjunto de información organizada que contiene una colección de registros donde un sistema puede buscar, reescribir, clasificar, borrar, añadir información a los archivos cuyo formato se basa etiquetas de hipertexto.

Bitmap: Es una mapeo de bits, esto quiere decir, valores que son cero o uno. También es llamado arreglo de bits, o índices de bitmap. Cuando se está hablando de graficación en computadoras, cuando el dominio es una rectángulo (indexado con 2 coordenadas), un bitmap permite el guardado de una imagen binaria, lo que quiere decir que es una imagen en la que cada pixel es o bien blanco o bien negro, o cualquier mezcla de dos colores.

Bluetooth: Es una tecnología estándar sin cables (mejor conocida como Wireless), que permite el intercambio de datos en distancias cortas, el cual usa transmisiones de radio corta entre los 2400 y 2480 MHz.

Broadcom BCM2042: Es un chip Bluetooth de bajo costo que permite colocar en dispositivos periféricos como mouse o teclados. Es un chip único que integra todos los perfiles, aplicaciones y protocolos de Bluetooth.

Caso de Uso: Descripción a detalle de las actividades y procesos necesarios para el desarrollo de un sistema o aplicación.

CLI: Infraestructura de Lenguaje Común o CLI por sus siglas en inglés, es una especificación abierta creada por Microsoft y estandarizada por la ISO y ECMA que describe un código ejecutable dentro del framework .NET, el cual también puede ser utilizado en software libre e en implementaciones como el proyecto Mono y Portable.NET

Consecuente: Expresa los atributos que pasarán a ser conocidos para el sistema.

Diagrama de Clases: Es el más utilizado en los modelos de sistemas OO; son los “planos” principales, que muestran los atributos y métodos que manejaran las clases del sistema.

Evento: Un evento, en computación, es una acción u ocurrencia detectada por un programa, que puede ser manejada por dicho programa. Típicos eventos son manejados sincronizadamente con el flujo del programa. Ejemplos de eventos son:

clic del mouse, presionar una tecla, otro programa en ejecución, una acción de una función dentro o fuera de la aplicación, etc.

Front End y Back End: El front-end es la parte del software que interactúa con el usuario y el back-end es la parte que procesa la entrada desde el front-end y es gestionada por el Administrador del software.

Gamepad: Un gamepad (control de mando) es un dispositivo de entrada en las consolas de juego.

Grafo: Viene del griego grafos, el cual significa dibujo o imagen. Es un conjunto de objetos llamados vértices o nodos, unidos por enlaces (llamados a su vez aristas o arcos), que permiten la representación de relaciones binarias entre los elementos de un conjunto. Su uso principal radica en el estudio de las relaciones que existen entre las unidades que contiene.

HID: viene del acrónimo inglés Human Interface Device, en otras palabras Dispositivo de Interfaz Humana. Es un tipo de dispositivo computacional, casi siempre de entrada, que interactúa con un humano. Ejemplos de este es el mouse, el teclado y un Wiimote

Hipermedia: Toma su nombre de la suma de hipertexto y multimedia, una red hipertextual en la que se incluye no sólo texto, sino también otros medios: imágenes, audio, vídeo, etc. (multimedia).

IntelligiSense: En programación computacional, el “IntelligiSense” se refiere a un ambiente de programación que permite el aumento de velocidad a la hora de

codificar o crear código en las aplicaciones, reduciendo los errores de escritura, sintaxis, asignación de variables y otros errores comunes.

Lenguaje Fuerte: En programación computacional, la escritura fuerte y débil se utiliza coloquial y peyorativamente para clasificar a los lenguajes de programación

Memoria EEPROM: Es un tipo no volátil de memoria usada en computadoras u otros dispositivos electrónicos, para almacenar pequeñas cantidades de datos, los cuales deben ser guardados cuando la fuente de energía es retirada.

Metadatos (*Datos sobre los datos):* Información que describe el contenido de los datos. Por ejemplo de un documento los metadatos serían: el título, el nombre del autor, la fecha de creación y modificación, y un conjunto de palabras clave que identifiquen su contenido.

Modelo: Es la conceptualización de un evento, un proyecto, una hipótesis, el estado de una cuestión y se representa como un esquema que posee símbolos descriptivos de características y relaciones más importantes.

Modelo Vista Controlador o MVC: Es un patrón de diseño que permite la separación total entre los datos y la lógica del negocio del diseño de interfaz y el módulo que controla los eventos y las comunicaciones del sistema. Divide ese trabajo en 3 teóricos módulos, conocidos como modelo (los datos), vista (la interfaz) y controlador (las reglas de negocio).

MSDN: MSDN es un proyecto de Microsoft que viene del acrónimo inglés (Microsoft Developer Network Red de Desarrollo para Microsoft), y es responsable de la

administración las firmas de relación entre desarrolladores y probadores con la empresa.

Namespace: Es un término usado en C# (y en programación en general) para referirse al contexto o plano en el que se encuentra una clase u otro elemento, así por ejemplo, si la clase *Casa* y *Edificacion* se encuentran en el mismo contexto, no es necesario llamar o referenciar en código la segunda clase en la primera, para así poder usar sus métodos (pero si es necesario hacer una instanciación de la clase *Edificacion*).

Nodo: Es un elementos físico que representa un recurso con capacidad computacional.

Plug-in: O conector, es un pequeño programa que permite agregar una específica característica de software y/o de un dispositivo a otro software y/o dispositivo.

Proceso de Razonamiento: El proceso de razonamiento es una progresión de un conjunto de datos de partida hacia una solución o conclusión**.**

Usuario: Ente humano que usa al sistema. Un mismo usuario puede actuar como instancias en varios actores diferentes, es decir, puede jugar diferentes roles.

Workflow *(Flujo de trabajo):* Estudio de los aspectos operacionales de una actividad de trabajo: cómo se estructuran las tareas, cómo se realizan, cuál es su orden correlativo, cómo se sincronizan, cómo fluye la información que soporta las tareas y cómo se le hace seguimiento al cumplimiento de las tareas.

Quito, 26 de junio de 2013

Señores (as)

Presente.-

A quien interese

VLBS certifica que el Sr. Egresado Marcelo Daniel Torres Vinueza ha implementado el proyecto de tesis: "Aplicación de la Metodología OOHDM y Técnicas de Inteligencia Artificial en la Solución del Desarrollo de un Videojuego, enfocado a niños de 6 a 10 años, utilizando la tecnología GDI+ basado en C# y Wiimote, cumpliendo con los parámetros solicitados desde el inicio del proyecto.

Por la atención y tiempo prestado al presente documento, anticipamos nuestros sentimientos de gratitud y estima.

V L B S Cia. Ltda.

Firma Autorizada

Sr. Santiago Rivelino Navarrete Carrera
RUC: 1792369134001

Ing. Santiago Navarrete.
Gerente General

Sangolquí, 26 de Junio de 2013.

DE: ING. MARGARITA ZAMBRANO
DIRECTOR DE TESIS

PARA: ING. MAURICIO CAMPAÑA, MsC.
DIRECTOR DE LA CARRERA DE ING. DE SISTEMAS E INFORMATICA

ASUNTO: Informe de Finalización de Tesis

Por medio del presente, tengo a bien informar a Usted Señor Director, que la tesis titulada como: "Aplicación de la Metodología OOHDM y Técnicas de Inteligencia Artificial en la Solución del Desarrollo de un Videojuego, enfocado a niños de 6 a 10 años, utilizando la tecnología GDI+ basado en C# y Wiimote, para su aplicación en la empresa Virtual Learning Solutions", desarrollada por el señor egresado Marcelo Daniel Torres Vinueza, ha sido concluida en su totalidad y cumple con lo establecido en el plan de Tesis aprobado por el H. Consejo de Carrera de Ingeniería de Sistemas, por lo que solicito se continúe con el proceso de graduación.

Atentamente,

Ing. Margarita Zambrano Rivera
Director de Tesis

Sangolquí, 26 de Junio de 2013.

DE: ING. CARLOS PROCEL
CODIRECTOR DE TESIS

PARA: ING. MAURICIO CAMPAÑA, MsC.
DIRECTOR DE LA CARRERA DE ING. DE SISTEMAS E INFORMATICA

ASUNTO: Informe de Finalización de Tesis

Por medio del presente, tengo a bien informar a Usted Señor Director, que la tesis titulada como: "Aplicación de la Metodología OOHDM y Técnicas de Inteligencia Artificial en la Solución del Desarrollo de un Videojuego, enfocado a niños de 6 a 10 años, utilizando la tecnología GDI+ basado en C# y Wiimote, para su aplicación en la empresa Virtual Learning Solutions", desarrollada por el señor egresado Marcelo Daniel Torres Vinueza, ha sido concluida en su totalidad y cumple con lo establecido en el plan de Tesis aprobado por el H. Consejo de Carrera de Ingeniería de Sistemas, por lo que solicito se continúe con el proceso de graduación.

Atentamente,

Ing. Carlos Prócel
Codirector de Tesis

BIOGRAFÍA

Nombres y Apellidos: Marcelo Daniel Torres Vinueza

Lugar y Fecha de Nacimiento: Quito, 10 de febrero de 1986

Formación Académica

***Educación Primaria*:** De Primero a Sexto Grado

Escuela Rosario del Alcázar No.2, Quito **Años:** 1992 - 1994

Unidad Educativa La Salle – Quito **Años:** 1994 – 1998

***Educación Secundaria*:** De Primero a Sexto Curso (Especialidad de Ciencias)

Unidad Educativa La Salle – Quito **Años:** 1998 – 2004

***Educación Superior*:** Carrera de Ingeniería en Sistemas e Informática

Escuela Politécnica del Ejército – Sangolquí **Años:** 2005 – 2013

Títulos Obtenidos

***3D Max y Dark Basic*:** ESPE – Departamento de Ciencias de la Computación

Horas: 40 **Año:** 2008

CISCO – CCNA1 Exploration: Network Fundamentals: ESPE

Horas: 36 **Año:** 2007

CISCO – CCNA2 Exploration: Routing and Concepts: ESPE – Centro de Capacitación

Horas: 36 **Año:** 2007

Suficiencia en el Idioma Inglés: ESPE – Departamento de Idiomas

Nivel: 8vo **Año:** 2011

HOJA DE LEGALIZACIÓN DE FIRMAS

ELABORADA POR

_____

Sr. Marcelo Daniel Torres Vinueza

DIRECTOR DE CARERA

_____

Sr. Ing. Mauricio Campaña

Lugar y Fecha:_____