Universidad de las Ciencias Informáticas

Facultad 3

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Título: Componente *Runner* para el videojuego Isla del Son

Autor: Antonio de Jesús Calvet Rodríguez

Tutores:

MSc. Yarina Amoroso Fernández

Ing. Willian René Zaldívar Pupo

La Habana, 2023

Año 65 de la Revolución

"El software, como el arte, es una expresión de la mente humana creativa. Es difícil de definir, pero generalmente fácil de reconocer cuando lo ves."

Roger S. Pressman

DEDICATORIA

Le dedico este trabajo a mis padres, especialmente a la memoria de mi padre, por ser mi fuente de inspiración, mi guía y ejemplo; a mi hermano, mis abuelos, mis tíos, mis amigos, mis profesores y todas aquellas personas que hicieron posible este sueño.

AGRADECIMIENTOS

Quiero expresar mi sincero agradecimiento a diversas personas que fueron fundamentales en la realización de esta tesis:

En primer lugar, mi gratitud hacia mi familia: mis padres, tíos, hermano y otros seres queridos. Durante estos intensos 5 años, han sido una constante fuente de inspiración y apoyo. Sus consejos, palabras de aliento, y el inquebrantable respaldo que me brindaron son invaluables. Cada consejo y sermón, en su momento, contribuyeron a mi crecimiento. Su ayuda incondicional en todas las circunstancias y su papel como padres ejemplares han sido determinantes en mi camino hacia este logro. Les debo mi gratitud por ser quienes son y por haberme moldeado hasta el día de hoy. Este logro es un tributo a ustedes.

A mis dedicados tutores, Yarina Amoroso Fernández, Willian René Zaldívar Pupo y el profesor Michel Pedrera Suen, les agradezco profundamente por su constante dedicación y apoyo. Sin su guía y orientación este logro no habría sido posible. Valoraré su influencia en mi desarrollo académico y profesional de por vida.

A mis compañeros de aula, tanto los que estuvieron conmigo desde el inicio de mi carrera como aquellos que conocí en el transcurso de la misma, les brindo mi gratitud. Los momentos que compartimos, las risas interminables y el apoyo mutuo forman parte de un precioso capítulo en mi vida. Espero mantener el contacto y seguir compartiendo experiencias en el futuro, recordando nuestros momentos de estudio intenso y camaradería sincera.

A mis amigos que se convirtieron en verdaderos hermanos: Leandro González, Yaniel Sánchez, Adrián Jiménez, Marcos González, Alexander Santacruz y Braydon Rodríguez, les agradezco por la profunda amistad que hemos cultivado. Juntos hemos superado obstáculos y nos hemos apoyado mutuamente a lo largo del camino. Llevaré su amistad siempre conmigo.

En términos generales, quiero agradecer a todas las personas que contribuyeron de alguna manera a hacer realidad este momento. Su apoyo, ánimo y amistad fueron pilares fundamentales. Gracias a cada uno de ustedes por ser parte de este viaje y por haberme enriquecido con su presencia.

DECLARACIÓN DE AUTORÍA

Declaro ser el único autor del presente Trabajo de Diploma y reconozco al Centro de Tecnologías
Interactivas (VERTEX) de la Facultad 4 de la Universidad de las Ciencias Informáticas, los derechos patrimoniales del mismo, con carácter exclusivo, para que hagan el uso que estimen pertinente con el mismo.

Para que así conste se firma la presente a los 29 días del mes de noviembre del año 2023.

Firma del Autor

Antonio de Jesús Calvet Rodríguez

Firma del Tutor

MSc. Yarina Amoroso Fernández

Firma del Tutor

Ing. Willian René Zaldívar Pupo

Tabla de Contenidos

Resumen

El Son cubano, género musical emblemático de Cuba, ha evolucionado a lo largo de los años, pero algunos sostienen que ha perdido sus valores originales, posiblemente debido a cambios sociales y emigración de músicos. El Ministerio de Educación Superior y el Ministerio de Cultura proponen un videojuego llamado "Isla del Son" para revitalizar el género, pero enfrenta problemas de coherencia narrativa e identidad cultural. La investigación busca desarrollar un componente estilo "*runner*" que se integre a este videojuego para abordar estos problemas. El proceso de desarrollo de la propuesta de solución se guía utilizando las buenas prácticas del marco de trabajo ingenieril para el desarrollo de videojuegos, utilizando métodos teóricos y empíricos, como análisis histórico-lógico, revisión documental, observación y entrevistas. Las características y diseño de la solución se describieron mediante la especificación de los mecanismos y la descripción de los requisitos no funcionales. Además, se realizaron pruebas alfa para comprobar el correcto funcionamiento del componente; se elaboró una estrategia de prueba para la ejecución de las pruebas beta una vez integrado el componente al videojuego “Isla del Son”; se aplicó la técnica de *Iadov* para medir el índice de satisfacción grupal y dar validez a la investigación.

Palabras clave: Son cubano, *runner*, videojuegos, *pooling* de objetos

INTRODUCCIÓN

El Son es uno de los géneros musicales más importantes y distintivos de Cuba. Con sus raíces en la música tradicional africana y española, el Son ha evolucionado a lo largo de los siglos para convertirse en un símbolo de la identidad cultural de Cuba. A través del Son, los cubanos han expresado su alegría, su tristeza y su amor por la música y la vida (Roy, 2002). El Son ha sido influenciado por otros géneros musicales, como el jazz y el mambo, y ha sido interpretado por artistas famosos como Benny Moré y Compay Segundo. Hoy en día, el Son sigue siendo una parte integral de la cultura cubana, y se puede escuchar en todo el país en bares, clubes y festivales de música (Rodríguez Ruidíaz, 2019).

La importancia del Son en la cultura cubana no puede ser exagerada. Es una forma única de expresión musical que ha evolucionado a lo largo de los siglos para reflejar los cambios en la sociedad y la política de Cuba. El Son ha sido utilizado para expresar la resistencia y la lucha contra la opresión, así como para celebrar la vida y el amor. A través de su evolución a lo largo de los siglos, el Son ha reflejado los cambios en la sociedad y la cultura de Cuba, y ha sido utilizado para expresar una amplia gama de emociones y sentimientos. El Son es una parte integral de la identidad cultural de Cuba (Lechner, 2013).

La pérdida de los valores del Son como género musical es un tema importante en la cultura cubana. Aunque el Son ha evolucionado a lo largo del tiempo, algunos argumentan que ha perdido algunos de sus valores originales, como la improvisación y la conexión con la cultura afrocubana, al decir del musicólogo cubano Radamés Giro. Estos cambios pueden estar relacionados con los cambios en la sociedad cubana en las últimas décadas, y es importante que los músicos y la sociedad en general continúen valorando y preservando la rica tradición del Son cubano (Giro, 2010).

La pérdida de los valores del Son también puede estar relacionada con los cambios en la sociedad cubana en las últimas décadas. Desde el colapso de la Unión Soviética en 1991, Cuba ha experimentado una serie de cambios económicos y políticos, y algunos argumentan que estos cambios han afectado la cultura y la identidad cubanas. Según el musicólogo Leonardo Acosta, "la música cubana ha sufrido una especie de desplazamiento cultural como resultado de los cambios en la sociedad cubana" (Acosta, 2004).

Otra razón de su decadencia fue el enfoque cultural en promover otros géneros musicales. El Son fue considerado un género "tradicional" y "folklórico", y por lo tanto no recibió la misma atención y apoyo que otros géneros más "modernos" y "revolucionarios". Además,

la emigración de muchos músicos cubanos también contribuyó a esta decadencia. Muchos músicos talentosos abandonaron Cuba en busca de mejores oportunidades en otros países, lo que dejó un vacío en la escena musical del país (Roy, 2002).

A pesar de esta decadencia, el Son sigue siendo valorado y apreciado por muchos cubanos y amantes de la música en todo el mundo. Aun así, es primordial intentar rescatar este género para que se mantenga vigente en todas las generaciones, enfatizando en las edades más jóvenes. Es por ello que el Ministerio de Educación Superior (MES) en conjunto con el Ministerio de Cultura (MINCULT) se ha propuesto la realización de una estrategia que permita dicha tarea, utilizando tecnologías digitales cercanas a la juventud.

Las tecnologías móviles y, por consiguiente, los videojuegos para celular pueden ser elementos educativos valiosos, ya que permiten a los jóvenes aprender de manera interactiva y personalizada. Las tecnologías móviles también pueden ser utilizadas para promover la colaboración y la comunicación. Además, pueden ser utilizados para personalizar el aprendizaje. Los estudiantes pueden aprender a su propio ritmo y nivel de habilidad, y recibir retroalimentación inmediata sobre su desempeño. Los videojuegos también pueden adaptarse a las necesidades y preferencias de cada individuo, lo que puede aumentar su motivación y compromiso (Arcila Ibague et al., 2022; López Raventós, 2016).

Teniendo en cuenta lo anterior, el MES ha decidido desarrollar un videojuego titulado Isla del Son que combina diferentes dinámicas de los videojuegos casuales para móvil. El argumento del mismo es el siguiente: “Luis se propone revitalizar el género del Son cubano. Para tamaña labor les pide ayuda a las leyendas del Son de todos los tiempos. De esa manera se sube a su carro con figuras de la talla de Benny Moré e Ignacio Piñeiro a rescatar las formas de la música tradicional e instaurar una casa del Son en todas las provincias”.

Este es un compendio de minijuegos para celular, los cuales están aglutinados alrededor del Son cubano, su historia, sus principales intérpretes y sus tradiciones en general. Los juegos son variados y de corta duración, que van escalando en dificultad en la medida en que progresa. Esta tarea fue encomendada al Centro de Tecnologías Interactivas (VERTEX) de la Universidad de las Ciencias informáticas (UCI). Sin embargo, atendiendo a este diseño y la propuesta del MES y el MINCULT, debe tenerse en cuenta que Isla del Son carece de elementos que entrelacen de forma lógica las diferentes partes del hilo narrativo del mismo, así como que establezcan interactividad entre los niveles y que

permitan el reconocimiento de elementos culturales de las regiones por las que se transita. Ello conlleva a que Isla del Son sea un videojuego inconexo narrativamente, y carente de elementos que otorguen identidad y cubanía al mismo.

Es por ello que se identifica como **problema** de esta investigación: ¿Cómo contribuir a enlazar lógicamente los componentes que forman parte de la historia del videojuego Isla del Son? Atendiendo a lo antes planteado, se define como **objeto de estudio** los videojuegos con contenidos culturales, basados en componentes. De esta forma se, se plantea como **objetivo general**: Desarrollar un componente que enlace de forma lógica el resto de componentes de Isla del Son. También a partir de ello se identifica como **campo de acción** el desarrollo de videojuego de género *runner*.

De lo anterior se obtienen las siguientes **tareas de investigación**:

1. Elaboración del marco teórico de la investigación a través del estudio del estado del arte.
2. Análisis de soluciones homólogas para definir las características claves para la realización del componente *runner*[1].
3. Identificación de las herramientas y metodologías de desarrollo de software para la realización del componente *runner*.
4. Diseño de un componente de minijuego estilo *runner* para integrar en la propuesta de solución.
5. Implementación del componente *runner*.
6. Integración del componente *runner* a la arquitectura del videojuego propuesto.
7. Pruebas al videojuego propuesto.

Para darle solución a los objetivos trazados en la investigación se emplearon los siguientes **Métodos Científicos**:

- **Métodos Teóricos:**
    - **Histórico-lógico:** se utilizó en este trabajo para analizar los antecedentes y tendencias actuales en el uso de videojuegos para el rescate de la cultura, específicamente en relación al Son cubano. A partir de este análisis se identifican las características necesarias y deseables para la solución que se propone.

[1] *Runner* es un género de videojuegos en el que el personaje corre automáticamente hacia adelante, evitando obstáculos, saltando, deslizándose.

- **Analítico-sintético:** se empleó para realizar un estudio de las teorías y documentos más relevantes sobre el uso e importancia de los videojuegos utilizados como parte de una estrategia para crear interés sobre un tema (en este caso el Son y la cultura cubana), permitiendo así, extraer los elementos más importantes sobre los mismos.
- **Revisión documental:** se usó en este trabajo para recopilar información relevante de diversas fuentes documentales como libros, artículos científicos, tesis, sobre videojuegos que dan soluciones a diferentes problemas de nuestra sociedad. A través de esta técnica se busca identificar los aspectos más importantes y relevantes sobre este tipo de videojuegos, que servirán para fundamentar la investigación y dar soporte al desarrollo de la solución propuesta.
- **Modelación:** en este trabajo se empleó para representar de manera abstracta el diseño del videojuego y sus componentes, con el fin de establecer una guía para su implementación. A través de esta técnica se pueden identificar posibles problemas o limitaciones en el diseño antes de la implementación y corregirlos para mejorar la eficiencia y efectividad de la solución propuesta.

- **Métodos Empíricos:**
  - **Consulta de la información en todo tipo de fuentes confiables:** Se realizó una búsqueda exhaustiva de información en diferentes fuentes, incluyendo bibliografías, bases de datos y otros recursos, para obtener información relevante y actualizada para la elaboración del marco teórico del Componente *Runner* para el videojuego Isla del Son.
  - **Observación:** Este método se utilizó para referirse a la observación y análisis de diversos videojuegos que se han utilizado en el pasado para rescatar culturas, los cuales se tomaron como referencia para establecer los elementos y características esenciales que debía tener el videojuego propuesto en esta investigación.
  - **Entrevista:** Este método permitió una interacción directa con el cliente con el propósito de entender sus necesidades y definir el problema que se quiere solucionar. A partir de esta comprensión y la captura de los requerimientos necesarios, se pueden elaborar las diferentes historias de usuario para desarrollar la solución.

El presente documento está compuesto por tres capítulos en los que se relacionan todo lo referente a la investigación, estructurados de la siguiente forma:

Capítulo 1: Fundamentación y referentes teórico-metodológicos del Componente *Runner* para el videojuego Isla del Son. En este capítulo se establecen los conceptos teóricos necesarios para llevar a cabo la investigación, así como se analizan las soluciones similares existentes en el mercado. También se selecciona la metodología de desarrollo de software y las herramientas y tecnologías a utilizar.

Capítulo 2: Diseño del Componente *Runner* para el videojuego Isla del Son. Este capítulo se enfoca en la descripción técnica de la solución propuesta, detallando los elementos necesarios para su desarrollo, como el diseño del videojuego, los mecanismos necesarios para su funcionamiento, los requisitos no funcionales y la arquitectura necesaria para su implementación.

Capítulo 3: Implementación, pruebas y mantenimiento del Componente *Runner* para el videojuego Isla del Son. En este capítulo se presenta la implementación y las pruebas de la solución. Se detallan los componentes del videojuego, los estándares de codificación utilizados y las pruebas realizadas para validar su funcionamiento y calidad. De la misma forma se detalla la integración del componente *runner* con el videojuego.

CAPÍTULO 1: FUNDAMENTACIÓN Y REFERENTES TEÓRICO-METODOLÓGICOS DEL COMPONENTE *RUNNER* PARA EL VIDEOJUEGO ISLA DEL SON

1.1 INTRODUCCIÓN

En este capítulo se abordan los principales conceptos sobre el dominio de la investigación referente a: el videojuego Isla del Son, los componentes de videojuegos, los *runners*. Se presenta una evaluación del arte de la investigación en relación a este género y las estrategias para rescatarlo. También se detallan la metodología de desarrollo de software, las tecnologías y herramientas más relevantes empleadas en la implementación de la solución propuesta.

1.2 CONCEPTOS GENERALES ASOCIADOS AL DESARROLLO DE VIDEOJUEGOS CON CONTENIDOS CULTURALES, BASADOS EN COMPONENTES

En el presente epígrafe se presentan los conceptos asociados al objeto de estudio de la presente investigación: desarrollo de videojuegos con contenidos culturales, basados en componentes; para un mejor entendimiento de los mismos.

1.2.1 Videojuegos

Según el Diccionario de la Real Academia de la Lengua Española un videojuego es un "dispositivo electrónico que permite, mediante mandos apropiados, simular juegos en las pantallas de un televisor, una computadora u otro dispositivo electrónico" (ASALE & RAE, s. f.).

1.2.2 Videojuegos como enlace narrativo

Un videojuego en sí mismo, un componente de este o los conocidos como minijuegos tienen intrínsecamente una narrativa. Como cualquier medio que pretende proponer una historia, los videojuegos necesitan sostener el hilo narrativo. Es posible distinguir entre dos usos a grandes rasgos de la historia en el videojuego (Curiel Gálvez, 2015):

1. Contextual: La historia sirve como apoyo del juego, es sólo una excusa para poner en marcha la acción y no requiere especial atención.
2. Central: La historia está en el núcleo del juego, éste se propone transmitirla y las mecánicas juegan en favor de la experiencia de ficción.

En este caso, la investigación se centra en el primer uso. De esta manera, un componente de videojuegos, en este caso un minijuego, puede fomentar la narrativa embebida implícita ambiental, lo cual es la forma de narración no tradicional basada en los elementos del entorno y en la propia jugabilidad como conectores o enlaces del hilo narrativo (Curiel Gálvez, 2015).

1.2.3 Uso de los Videojuegos para recuperar la cultura

El uso de videojuegos para recuperar la cultura es un tema cada vez más relevante en el ámbito de la educación y la preservación del patrimonio cultural. Los videojuegos pueden ser una herramienta efectiva para involucrar a los usuarios en la historia, la cultura y el patrimonio, y para promover la recuperación y preservación de tradiciones y prácticas culturales (González-Pérez & García-Ruiz, 2019).

Para garantizar que los videojuegos respeten la cultura y el patrimonio, es importante seguir ciertos principios y mejores prácticas. A continuación, se presentan algunas recomendaciones:

- Investigación y consulta: Es importante que los desarrolladores de videojuegos realicen una investigación exhaustiva sobre la cultura y el patrimonio que se está representando en el juego. También es recomendable trabajar en colaboración con expertos y miembros de la comunidad cultural para garantizar la autenticidad y la precisión de la representación cultural.
- Respeto y sensibilidad: Los desarrolladores de videojuegos deben ser sensibles a las sensibilidades culturales y evitar la representación estereotipada o inexacta de los personajes y la cultura representada en el juego. Es importante prestar atención a los detalles, como la ropa, la música y los objetos culturales representados en el juego.
- Consentimiento y autorización: Los desarrolladores de videojuegos deben obtener el consentimiento y la autorización de las comunidades culturales y los propietarios del patrimonio antes de utilizarlos en el juego. Esto puede incluir el uso de imágenes, música, artefactos y otros elementos culturales.
- Diversidad e inclusión: Los desarrolladores de videojuegos deben trabajar para promover la diversidad y la inclusión en el juego, representando una variedad de perspectivas culturales y de género. Es importante evitar la representación de un solo estereotipo o perspectiva cultural y promover la inclusión de diferentes voces y experiencias.

- Educación y sensibilización: Los desarrolladores de videojuegos pueden utilizar el juego como una herramienta educativa para promover la comprensión y el respeto por la cultura y el patrimonio. Esto puede incluir la inclusión de información y recursos educativos en el juego, así como la promoción de la comprensión y la apreciación de la cultura y el patrimonio en la comunidad de jugadores (González-Pérez & García-Ruiz, 2019).

1.2.4 Componente de videojuegos

Un componente de un videojuego es una parte fundamental del mismo que cumple una función específica dentro del juego. Los componentes pueden ser visibles o invisibles para el jugador, y pueden incluir elementos como personajes, objetos, escenarios, mecánicas de juego, bandas sonoras, efectos de sonido, y otros elementos que contribuyen a la experiencia del juego (Jurado Monroy et al., 2012).

Los componentes de un videojuego Son diseñados y programados por los desarrolladores de juegos para cumplir una función específica dentro del juego. Por ejemplo, los personajes son componentes del juego que permiten al jugador interactuar con el mundo del juego, mientras que las mecánicas de juego son componentes que determinan cómo el jugador interactúa con el juego y cómo se desarrolla la experiencia del juego (Jurado Monroy et al., 2012).

Es importante que los componentes de un videojuego estén diseñados y programados de manera efectiva para garantizar una experiencia de juego satisfactoria para el jugador. Esto puede incluir elementos como una jugabilidad equilibrada, un diseño de niveles interesante y desafiante, una buena integración de la banda sonora y los efectos de sonido, y una historia coherente y atractiva (Jurado Monroy et al., 2012).

1.3 CONCEPTOS GENERALES ASOCIADOS AL DESARROLLO DE VIDEOJUEGO DE GÉNERO *RUNNER*

En el presente epígrafe se presentan los conceptos asociados al campo de acción de la presente investigación: desarrollo de videojuego de género *runner*; para un mejor entendimiento de los mismos.

1.3.1 Los géneros en los videojuegos para dispositivos móviles

Los videojuegos para móviles abarcan una amplia variedad de géneros adaptados específicamente para su juego en dispositivos móviles. Son comunes los videojuegos de

tipo *puzzle* que se centran en desafíos basados en acertijos y problemas que requieren de habilidades lógicas y suelen tener mecánicas simples, ideales para sesiones cortas. Por su parte, los juegos de acción se caracterizan por acciones rápidas y controles simples que exigen de los rápidos reflejos del jugador, pueden estar basados en la supervivencia del personaje o la destrucción de objetos o enemigos. Los juegos de estrategia tienen características heredadas de los juegos del mismo género, propios de computadoras o consolas: se centran en la toma de decisiones y planificación a lo largo de un plazo de tiempo; suelen destacar los de estrategia económica, bélica y de negocios. Como parte de los videojuegos casuales (categoría en la que se podría incluir los *puzzle*) puede incluirse los videojuegos *runner,* caracterizados por que el avance del personaje no es controlado por el jugador, sino que avanza automáticamente y solo se controla la evasión de obstáculos o la adquisición de elementos moviendo al personaje con toques en la pantalla (Dille & Platten, 2015).

1.3.2 El *runner* como género de videojuegos

El *runner* es un género de videojuegos que se ha popularizado en los últimos años. Se considera un subgénero de los videojuegos de plataforma según algunos autores (Polansky, 2013), aunque por sus características únicas, puede definirse como un género por sí mismo (Ramón, 2016; Ryan Whitwam, 2014). En estos juegos, el jugador controla a un personaje que corre automáticamente a través de un nivel o escenario, y debe saltar, esquivar y realizar otras acciones para evitar obstáculos y enemigos (Ryan Whitwam, 2014). En ocasiones es llamado *endless runner*, aunque realmente esa definición corresponde a uno de sus subgéneros.

Entre los subgéneros se encuentra el ya mencionado *endless runner* que se corresponde, como su nombre lo indica, con la infinidad en la duración del nivel siempre que el jugador no alcance la condición de "Perdido". El *runner* clásico, más usado en el pasado por la poca definición que existe en la línea divisoria entre este subgénero y el género de plataformas, pero en el cual, si es posible encontrar un final para cada nivel, ya sea definido por tiempo, o llegando a una meta. Por su parte, el *runner* 2D, a pesar de responder a primera vista a una estética, debido a su linealidad (generalmente horizontal) la jugabilidad cambia respecto a otros subgéneros, están comúnmente caracterizados por la necesidad constante de saltar obstáculos. Es importante no confundir en este último la visualidad con la jugabilidad, las 2 dimensiones se refiere a la forma de avance (solo en una dirección), y no a los gráficos del juego (Polansky, 2013; Ryan Whitwam, 2014).

Dentro de este tipo de videojuegos, lo normal es encontrar que la generación del terreno o mapa sea a través de algoritmos definidos y de forma aleatoria. No es común que el diseño de niveles se realice con mapas predefinidos, aunque dentro de los *runners* plataformas hay excepciones, como es el caso del conocido Geometry Dash.

Uno de estos algoritmos o métodos es la generación procedural, la cual está fundamentada en la creación de datos de forma algorítmica, en oposición a la forma manual. Normalmente combina contenidos generados por una persona y algoritmos unidos a contenido generado por una computadora. Es común ver este método dentro de videojuegos pensados para ser frecuentemente rejugados, por lo que dentro de los videojuegos para móvil es más común en los juegos casuales. Por su capacidad para la generación aleatoria es común encontrarlos en videojuegos con grandes generaciones de terrenos o mundos, por lo que dentro de los *runners* lo normal es encontrarlo en los *endless runner* (Cook, 2016).

Otro método o algoritmo es el *pooling* de objetos. *Pooling* se refiere a juntar o agrupar varios recursos o componentes. Es precisamente esa la forma de funcionamiento de este algoritmo: permite predefinir instancias con varias propiedades diferentes de un recurso en concreto para que puedan ser usadas indistintamente. Dentro de los videojuegos es común encontrarlo, precisamente, en los *runners* con límite de tiempo porque, a pesar de que no se predefine un mapa en concreto, es posible realizar una generación aleatoria eficaz ya sea de terrenos u objetos, que no precisa de algoritmos muy complejos o de inteligencia artificial (Unity Learn, 2022).

Teniendo en cuenta la mencionada clasificación de géneros, el videojuego que se va a desarrollar en la investigación, va a ser de género *runner* dirigido a las plataformas móviles, dígase tableta o celular, según las necesidades planteadas por el MES y el MINCULT. Esto último está basado en que un *runner* permite transitar de forma visual, desde un punto de partida, hasta un punto de llegada definido, ya sea espacialmente o temporalmente (Polansky, 2013; Ryan Whitwam, 2014).

1.4 ANÁLISIS DE VIDEOJUEGOS Y/O COMPONENTES *RUNNER* EXISTENTES

En este epígrafe se realiza un análisis de videojuegos o componentes *runner* seleccionados de los cuales se hace un estudio que permite identificar elementos comunes y destacables en este género, los cuales puedan ser incluidos en el componente *runner* para el videojuego Isla del Son

1.4.1 Temple Run y Temple Run 2

Temple Run es un juego móvil desarrollado y publicado por Imangi Studios, que se puede encontrar en todas las plataformas móviles. El juego es gratuito y utiliza un modelo de negocio basado en micro transacciones y publicidad integrada. La temática del juego es similar a la de las películas del famoso arqueólogo, donde el jugador avanza verticalmente, sorteando obstáculos como minas, giros cerrados y rocas. Además, utiliza la oscilación del dispositivo móvil para mover al personaje por el camino. El juego también incluye misiones diarias y potenciadores para mejorar la experiencia del usuario. Sin embargo, el jugador debe tener cuidado ya que unos monos perseguirán al personaje y lo atraparán si falla demasiadas veces en los obstáculos (Ramón, 2016).

1.4.2 T-Rex *Runner*

La inmersión instantánea que ofrecen este tipo de videojuegos es tan potente que incluso Google decidió lanzar su propio juego llamado T-Rex *Runner* en septiembre de 2014. Este juego aparece automáticamente cuando el usuario tiene problemas de conexión a internet en Google Chrome. También se puede acceder a él a través de chrome://dino. El objetivo del juego es controlar a un dinosaurio que salta cactus y esquiva dinosaurios voladores. El movimiento del personaje es bastante limitado, desplazándose únicamente en la dirección horizontal (X), mientras que la velocidad aumenta progresivamente a medida que avanzamos en el juego. Cada obstáculo que se sortea da puntos (Ramón, 2016; Ryan Whitwam, 2014)

1.4.3 Spider-Man Unlimited

Spider-Man Unlimited es un juego de carrera infinita desarrollado por Gameloft y publicado por Marvel en 2014 para plataformas móviles como Android, iOS y Windows Phone. Fue uno de los primeros juegos de carrera infinita en presentar al popular superhéroe de Marvel y se caracterizó por su estilo de arte de cómic y su jugabilidad adictiva. El juego también contó con un modo de historia con misiones y jefes para derrotar. Sin embargo, en 2019, Marvel decidió retirar el juego de todas las tiendas de aplicaciones móviles (Ramón, 2016; Ryan Whitwam, 2014).

1.4.4 Geometry Dash

Geometry dash es un juego independiente, desarrollado por RobTop. Está característicamente catalogado como un juego de plataformas, pero debido al diseño de niveles y a la continuidad en el movimiento del personaje (el jugador no puede controlar al personaje en su avance, sino que solamente puede saltar presionando la pantalla) es posible incluirlo en el subgénero de *runner* 2D debido a su movimiento en una sola dirección. Posee una curva de dificultad muy inclinada, en la que los niveles "Fácil" son escasos y aun así tienen alta complejidad. Una de las características fundamentales de este juego es su apartado musical, muy halagado por la crítica especializada, pero también su apartado visual con abundantes explosiones de colores con estética neón, pudiendo causar, incluso, reacciones adversas en personas con enfermedades relacionadas con la fotosensibilidad (Noboa, 2017; Ryan Whitwam, 2014).

1.4.5 La chivichana

La Chivichana es un novedoso juego de tipo *Runner*, producido en Cuba y cuenta con la música del popular dúo cubano Buena Fe, entre otros atractivos. Este juego es accesible y es ideal para despejarse sin alejarse del interés educativo. Dirigido principalmente a un público joven, ofrece la posibilidad de adquirir conocimientos en todo momento. El juego se desarrolla en un escenario compuesto por diversas pistas, por las cuales avanza el objeto principal, la Chivichana. Durante el juego, el jugador puede obtener bonificaciones que pueden ser utilizadas para mejorar tanto el personaje como la Chivichana. Uno de los valores más importantes son los elementos visuales propios de la cultura cubana como los conocidos almendrones, los ómnibus urbanos o los contenedores de basura de la Habana. Además de ser un juego de alta calidad estética, la Chivichana también promueve la difusión de valores. El juego fue inicialmente presentado para computadoras y posteriormente para dispositivos móviles y tabletas (Andy Hernández et al., 2017).

1.5 COMPARACIÓN DE VIDEOJUEGOS Y/O COMPONENTES *RUNNER* EXISTENTES

Para realizar un análisis comparativo de los ejemplos citados en el epígrafe anterior, se utilizaron parámetros de comparación con los que (Dille & Platten, 2015) evalúan los videojuegos en general.

Se presenta la siguiente tabla con los parámetros que se describen a continuación.

Tabla 1 Tabla comparativa de videojuegos y/o componentes *runner* existentes (Elaboración propia)

Videojuego	Gráficos	Diseño de niveles	Jugabilidad				Subgénero	Dificultad
			Dirección	Distribución de mapa	Objetivos	Potenciadores		
Temple Run	3D	Repetitivo. Dividido en secciones prediseñadas	Vertical	Un solo carril. Se mueve inclinando el dispositivo de un lado al otro.	Sin objetivos. Basado en recompensas.	Se adquieren durante la partida	Endless *runner*	Aumenta mientras avanza el tiempo de la partida. No tiene niveles.
T-Rex *runner*	2D	Complejidad de gráficos incrementada de acuerdo con la dificultad. Minimalista	Horizontal	Un solo carril. Salta con toques en la pantalla.	Sin objetivos.	No se adquieren potenciadores	Endless *runner* 2D	Aumenta mientras avanza el tiempo de la partida. No tiene niveles.
Spider-Man Unlimited	3D	Variado. Gráficos muy complejos	Vertical	Tres carriles. Se mueve deslizando el dedo por la pantalla.	Objetivos como misiones	Presentes desde el inicio de la partida	Endless *runner*	Aumenta mientras avanza el tiempo de la partida.
Geometry Dash	3D-2D	Variado. Gráficos muy complejos	Horizontal	Un solo carril. Salta con toques en la pantalla.	Debe llegar a la meta sin chocar	No se adquieren potenciadores	*Runner* 2D. Plataformas	Separada por niveles. La dificultad es muy alta
La Chivichana	3D	Repetitivo. Dividido en secciones prediseñadas	Vertical	Tres carriles. Se mueve deslizando el dedo por la pantalla.	Sin objetivos.	Presentes desde el inicio de la partida	Endless *runner*	Aumenta mientras avanza el tiempo de la partida.

Es importante mencionar que algunos elementos propios de los *runners* no fueron analizados en la comparación pues son comunes para todos los homólogos presentes en el estudio. Esto se refiere a la adquisición de monedas y al registro de puntuaciones. A continuación, se listan y caracterizan los parámetros de comparación utilizados (Dille & Platten, 2015):

- **Gráficos** y **Diseño de Niveles**: examina la calidad de los gráficos y la diversidad en la estética en la creación de los niveles.
- **Subgénero**: como su nombre lo indica, enuncia el subgénero al que pertenece el videojuego.
- **Dificultad**: define la progresión de la dificultad dentro del videojuego o entre sus niveles.

- **Jugabilidad**: Evalúa la fluidez de los controles, así como la variedad de movimientos y acciones disponibles para el jugador. Se puede fraccionar en cuatro parámetros para esta comparación:
    - **Dirección**: corresponde a la dirección en la que se mueve el personaje.
    - **Distribución de mapa**: definido por cuantos carriles tiene el personaje para moverse y de qué forma.
    - **Objetivos**: corresponde a los objetivos a alcanzar en los niveles.
    - **Potenciadores**: examina el uso de potenciadores y su adquisición.

1.6 METODOLOGÍA, TECNOLOGÍAS Y HERRAMIENTAS

El proceso de desarrollo de videojuegos difiere en gran medida del proceso de desarrollo de software convencional. Las actividades de ingeniería que se realizan durante el proceso generan resultados distintos a los obtenidos en proyectos de software de otros dominios de aplicación. En el centro VERTEX, se emplea el **Marco de trabajo ingenieril para el desarrollo de videojuegos**, el cual se compone de 5 etapas que se complementan para garantizar un proceso de desarrollo de productos exitoso. Estas etapas son: conceptualización, diseño, implementación, prueba y mantenimiento (Andy Hernández et al., 2017).

- **Etapa 1. Conceptualización:**
    - Definir el género sobre el cual se desarrollará el videojuego.
    - Describir la mecánica del videojuego.
    - Especificar las metas para la experiencia del jugador.
- **Etapa 2. Diseño:**
    - Describir los elementos formales que definen la estructura del videojuego.
    - Describir los elementos dramáticos que definen el entretenimiento del videojuego.

- Diseñar las pantallas gráficas elementales que forman la estructura del videojuego.
- Describir los elementos dinámicos que definen las mecánicas o mecanismos del videojuego.
- Validar los mecanismos teniendo en cuenta criterios técnicos para su implementación.
- Modelar el diagrama de paquetes de mecanismos teniendo en cuenta la distribución arquitectónica.
- Describir la concepción de los mecanismos sobre la distribución arquitectónica diseñada.
- Modelar el comportamiento de los mecanismos mediante diagramas de transición de estado.
- Mantener una trazabilidad bidireccional entre cada elemento del videojuego.
- Describir las características no funcionales del videojuego.

- **Etapa 3. Implementación:**
  - Diseñar los componentes que encapsulan la implementación.
  - Desarrollar las mecánicas especificadas y diseñadas.
- **Etapa 4. Prueba:**
  - Desarrollar pruebas Alfa.
  - Desarrollar pruebas Beta.
  - Registrar defectos durante las pruebas realizadas.
- **Etapa 5. Mantenimiento:**
  - Realizar análisis Postmortem.
  - Retomar la etapa de “Diseño”.

Las tecnologías y herramientas que fueron definidas para este componente fueron determinadas mediante un estudio realizado por el equipo de especialistas VERTEX.

1.6.1 Lenguaje de modelado

El modelado constituye una simplificación de la realidad donde se define lo esencial para la construcción del software con los objetivos de comunicar la estructura de un sistema complejo, especificar el comportamiento deseado del sistema, comprender mejor lo que se está desarrollando y descubrir oportunidades de simplificación y reutilización (Rumbauch & Grady, 2015).

Lenguaje de modelado Unificado (UML)

En sus ediciones más recientes, (Pressman & Maxim, 2020) definen al Lenguaje de Modelado Unificado (UML) como un lenguaje estándar para describir diseños de software, que permite visualizar, especificar, construir y documentar los componentes de un sistema de software intensivo. Este lenguaje es adaptable a todos los métodos de desarrollo, etapas del ciclo de vida de un software, dominios de aplicación y medios, y está basado en conceptos orientados a objetos. El UML es el estándar global que utilizan los desarrolladores, autores y proveedores de herramientas de Ingeniería de Software Asistida por Computación (CASE).

Este lenguaje es lo suficientemente expresivo como para modelar sistemas que no son informáticos, como flujos de trabajo en una empresa, diseño de la estructura de una organización y el diseño del hardware. Un modelo UML está compuesto por tres clases de bloques de construcción :

- Elementos: los elementos son abstracciones de cosas reales o ficticias (objetos, acciones, etcétera).
- Relaciones: relacionan los elementos entre sí.
- Diagramas: son colecciones de elementos con sus relaciones (Hernández Orallo, 2015).

1.6.2 Herramienta de modelado

Las herramientas de Ingeniería de Software Asistida por Computadora (CASE por sus siglas en inglés), son diversas aplicaciones informáticas destinadas a aumentar la

productividad en el desarrollo de software, reduciendo el costo de las misma en términos de tiempo y costo. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida del desarrollo de software, en tareas como el proceso de realizar un diseño de un proyecto, cálculo de costos, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras (*Guión Visual Paradigm for UML*, 2013).

Visual Paradigm 8.0

Es una herramienta CASE que permite el modelado de sistemas mediante UML y brinda apoyo a los analistas, ingenieros de software y desarrolladores en todas las fases del ciclo de vida del desarrollo de un software. La versión 8.0 de esta herramienta permite la creación de diagramas UML de alta calidad, además de proporcionar herramientas para la documentación, generación de código y pruebas de software. Visual Paradigm for UML es ampliamente utilizado en la industria y es reconocido por su interfaz intuitiva y fácil de usar (*Guión Visual Paradigm for UML*, 2013; Hernández Orallo, 2015).

Esta herramienta se selecciona por las facilidades que brinda para capturar los requisitos correctos y transformarlos en diseños precisos, además de su capacidad para diseñar cada uno de los diagramas y/o artefactos que propone UML para el análisis y diseño, implementación, pruebas y despliegue de software.

1.6.3 Motor de Videojuego

Un motor de videojuegos (también conocido como *game engine*) es un software que proporciona una estructura y herramientas para el desarrollo de videojuegos. Puede ser considerado un entorno de desarrollo. Los motores de videojuegos incluyen una variedad de herramientas, como un motor de física, un motor de gráficos, un motor de sonido, un editor de niveles, un sistema de animación, y otros componentes que permiten a los desarrolladores crear y diseñar juegos de manera eficiente. Se utilizan para acelerar el proceso de desarrollo de juegos y reducir los costos de producción. Los desarrolladores de juegos pueden utilizar un motor de videojuegos para crear juegos en diferentes plataformas, como consolas, dispositivos móviles y computadoras (Ruelas, 2017).

Al utilizar un motor de videojuegos, los desarrolladores pueden centrarse en la creación de contenido y la jugabilidad del juego, en lugar de tener que crear cada componente desde

cero. Los motores de videojuegos también pueden incluir herramientas para la optimización del rendimiento, lo que permite a los desarrolladores crear juegos que funcionen sin problemas y sin errores técnicos (Ruelas, 2017).

Unity 3D (2021.3.4f1)

Es un motor multiplataforma de gran alcance que permite la creación de proyectos en 2D y 3D, y es accesible tanto para principiantes como para usuarios avanzados. Con la capacidad de crear fácilmente videojuegos y aplicaciones para una variedad de plataformas, Unity 3D es una herramienta versátil en el desarrollo de software. Además, cuenta con bibliotecas de activos comerciales y gratuitos creados por Unity Technologies y miembros de la comunidad, conocidas como stores (tiendas). A través de estas stores (tiendas), los usuarios pueden acceder a una gran cantidad de recursos, que van desde texturas, modelos y animaciones, hasta tutoriales y extensiones del editor, que se pueden descargar e importar directamente en sus proyectos desde una interfaz simple dentro del Editor Unity (Unity Technologies, 2019).

Algunas de sus características más significativas se enumeran a continuación:

- Soporta oficialmente dos tipos de lenguaje de programación: C#, UnityScript (básicamente Javascript). El más generalizado en la comunidad es C#.
- Facilidad para realizar videojuegos tanto 2D como 3D.
- Curva de aprendizaje sencilla. Posee el Asset Store, que es una gran comunidad de creadores de plugins y recursos, así como muchos manuales oficiales para aprender rápido.
- Las herramientas de edición gráfica son muy buenas y pueden ser mejoradas con plugins.
- Soporta multitud de formatos de recursos y los convierte de manera automática a los formatos más óptimos dependiendo de la plataforma objetivo.
- Despliegue a múltiples plataformas de manera sencilla, tanto móviles como de escritorio y consola.
- Desde Unity 5.0, la licencia gratuita cubre prácticamente todas las necesidades del desarrollador, y algunas de las licencias de pago son bastante asequibles (Unity Technologies, 2019).

1.6.4 Lenguaje de programación

Unity 3D utiliza C# en su versión 7.0 como uno de sus lenguajes de programación para compilar *scripts*. C# es un lenguaje orientado a objetos que simplifica la programación ya que, en Unity, todos los componentes y elementos del videojuego son objetos o clases. Al programar en C#, se pueden definir una o varias clases dentro del mismo espacio de nombres. Además, C# ofrece una amplia gama de tipos de datos definidos, lo que lo hace diferente de otros lenguajes. También admite todas las características de la programación orientada a objetos, como la encapsulación, la herencia y el polimorfismo (González Seco, 2000).

1.6.5 Herramienta de diseño 3D

Una herramienta de diseño 3D es un software que permite a los usuarios crear modelos y escenarios en tres dimensiones. Estas herramientas se utilizan comúnmente en la industria del entretenimiento, como en la producción de videojuegos, películas y animaciones. También se utilizan en la arquitectura, el diseño industrial y la ingeniería para crear prototipos y visualizaciones. Las herramientas de diseño 3D permiten a los usuarios crear objetos y escenarios, aplicar texturas y materiales, ajustar la iluminación y la cámara, animar objetos y personajes, y realizar otras tareas relacionadas con la creación de contenido en tres dimensiones. Estas herramientas a menudo incluyen una amplia variedad de opciones de personalización y configuración, lo que permite a los usuarios crear modelos y escenarios únicos y detallados (Dounas et al., 2023).

Blender 3.5

Blender es un software de creación de contenido 3D que combina una gran variedad de herramientas, desde modelado y renderizado, hasta animación, edición de video, efectos visuales, composición, texturizado y simulaciones. Se trata de una plataforma multiplataforma que cuenta con una interfaz de usuario uniforme en todas las principales plataformas, la cual puede ser personalizada mediante scripts de Python. Gracias a su arquitectura 3D de alta calidad, Blender permite un flujo de trabajo rápido y eficiente para la creación de contenido. Además, la comunidad de Blender es muy activa y cuenta con una amplia lista de sitios web y recursos para el usuario. Por último, Blender tiene un tamaño de archivo ejecutable pequeño, lo que lo hace fácilmente portátil (Dounas et al., 2023).

1.7 CONCLUSIONES DEL CAPÍTULO

Tras un análisis de los conceptos asociados a los componentes en videojuegos y más específicamente, los *runners*, se define que para la propuesta de solución debe seguirse esa línea genérica para lograr un enlace de forma lógica entre los componentes de Isla del Son. La investigación permitió definir el marco de trabajo ingenieril para el proceso de desarrollo de videojuegos como metodología de desarrollo, el cual permite complementar el proceso de desarrollo de la solución, utilizando sus etapas y actividades. En este caso, se utilizaron herramientas y tecnologías ya comprobadas en el desarrollo de videojuegos por el centro VERTEX, como Blender (editor de gráficos), Unity 3D (motor de desarrollo de videojuegos) y Visual Paradigm for UML como herramienta CASE. Tras todo lo anterior se puede definir que la forma en la que este componente está enlazado de forma narrativa con el resto del videojuego es mostrando en el *runner* un viaje que conecte provincias adyacentes. Además, las mecánicas más importantes para este *runner* son el movimiento entre carriles, el puntaje y la administración de potenciadores, respondiendo esto a los videojuegos del mismo género analizados.

CAPÍTULO 2 DISEÑO DEL COMPONENTE *RUNNER* SON DE MÁQUINA PARA EL VIDEOJUEGO ISLA DEL SON

2.1 INTRODUCCIÓN

A partir de este capítulo se inicia el diseño y desarrollo de una propuesta de solución que dé respuesta al problema planteado en el marco teórico, aplicando las tareas y etapas del marco de trabajo ingenieril para el proceso de desarrollo de videojuegos. Además, se presentan diferentes artefactos definidos por la misma. Se expone la interpretación de las necesidades referentes al componente las cuales se muestran en Requisitos no funcionales, Elementos formales y Mecanismos.

2.2 PROPUESTA DE SOLUCIÓN

Atendiendo a lo analizado anteriormente en la comparación de homólogos, así como a las necesidades del MES y el MINCULT respecto a Isla del Son, se define como propuesta de solución para la problemática planteada, un componente *runner* que defina un itinerario a través de todas las provincias del país, lo cual pueda permitir una transición lógica entre los diferentes niveles y minijuegos (componentes) de Isla del Son, así como que aporta identidad y cubanía a la representación de cada provincia respectivamente.

Sujeto a las características antes descritas, así como a la valoración de los mencionados videojuegos de género *runner* expuestos, se determina que, a pesar de que la mayoría de los homólogos analizados son del subgénero *endless runner*, es preciso posicionar el componente *runner* actual en el subgénero *runner* clásico. Esto último debido a la necesidad de que cada nivel tenga un final para no alargar innecesariamente la experiencia del juego; el subgénero seleccionado lo permite, en este caso haciendo uso de objetivos y medición de tiempo.

De los homólogos se toman características concretas, como los elementos visuales, que dan identidad cubana, de La Chivichana, los mecanismos de cambio de carril de Spider-Man Unlimited así como el puntaje y el incremento de la curva de dificultad apreciable en Geometry Dash. La dificultad estará distribuida por niveles: mientras más alto el nivel, más alta la dificultad. Este componente reconocido como un minijuego tomará el nombre de **Son de Máquina** a petición del MES y el MINCULT.

Para el desarrollo de Son de Máquina se seleccionó un algoritmo de *object pooling* o *pooling* de objetos. Dentro de los *runner*, por la necesidad de generación continua y aleatoria del mapa, este es muy usado, sobre todo teniendo en cuenta los *runners* que necesitan de ciertas cantidades de objetos en el mapa para cumplir un objetivo (Unity Learn, 2022). Para ello son necesarias tecnologías apropiadas tanto para el desarrollo de videojuegos, como para la implementación de este algoritmo.

2.3 DISEÑO DE LA PROPUESTA DE SOLUCIÓN

El juego contará con una estética *toon*[2] (personajes y fondos). Se emplea una paleta de colores llamativos semejantes a los que se aprecian en el paisaje campestre cubano. Las ventanas y botones tienen bordes redondeados. Los elementos del menú deben tener un aspecto retro (estética de los años 1930 a los 50) haciendo énfasis en los automóviles de la época y los personajes célebres.

2.3.1 Elementos en pantalla del componente *runner* Son de Máquina

La pantalla de este componente debe contar con los siguientes elementos:

1. **Contador de inicio:** Animación de tres segundos al inicio de la carrera que prepara al jugador. Cuenta de la luz amarilla hasta la verde. (puede ser un gallo que cante tres notas distintas) Se ejecuta después de la pausa.
2. **Temporizador:** Corre de 60 segundos a 0. Los obstáculos le restan tiempo. Si llega a 0 el contador y no ha cumplido todos los objetivos pierde la partida. Abre la ventana emergente de *game over* (texto: “Fin del viaje”, duración 3 segundos). La partida se reinicia automáticamente.
3. **Indicador de objetivos:** Muestra el número de claves de Son a conseguir. a ventana emergente de victoria (texto, “Isla del Son”, duración 3 segundos).
4. **Botón de pausa:** detiene la partida. Abre una ventana emergente. Con los botones reiniciar partida, continuar y salir. Muestra los tres potenciadores a escoger. Estos se recargan cada 24 horas. Solo se puede escoger 1 por carrera. Salir abre la pantalla del mapa.
5. **Potenciadores:**
    - Apicuba (ralentiza la carrera)

[2] Toon o cartoon es el estilo que va buscando una versión cómica e hiperbólica de la realidad. Estilo dibujo animado.

- Cuba Ron (gana el doble de claves de Son)
- EGREM (permite proseguir después de chocar. Abre el contador de inicio)

6. **Obstáculos:**
   - Valla de madera: resta 10 segundos al contador de tiempo.
   - Valla de madera doble: ocupa dos carriles, resta 10 segundos al contador de tiempo.
   - Hueco: resta 5 segundos al contador de tiempo.

2.3.2 Dificultad de los niveles

Este componente tiene la siguiente distribución de dificultades:

Tabla 2 Dificultad para el componente *runner* Son de Máquina (Elaboración propia)

Provincias	Dificultad	Incremento	Velocidad	Obstáculos
Guantánamo	Fácil	0.01	6	4 de cada uno
Santiago	Fácil	0.02	6	4 de cada uno
Holguín	Fácil	0.04	6	4 de cada uno
Granma	Fácil	0.06	6	4 de cada uno
Las Tunas	Medio	0.01	7	4 de cada uno
Camagüey	Medio	0.02	7	4 de cada uno
Ciego de Ávila	Medio	0.04	7	4 de cada uno
Sancti Spíritus	Medio	0.06	7	4 de cada uno
Cienfuegos	Difícil	0.03	7	5 de cada uno
Villa Clara	Difícil	0.05	7	5 de cada uno
Matanzas	Difícil	0.07	7	5 de cada uno
Artemisa	Difícil	0.09	7	5 de cada uno
Mayabeque	Difícil	0.08	8	5 de cada uno
Pinar del Río	Difícil	0.09	8	5 de cada uno
Isla de la Juventud	Difícil	0.1	8	5 de cada uno
La Habana	Difícil	0.15	8	5 de cada uno

Para un mejor entendimiento de la curva de dificultad referente a lo antes planteado, se modela el siguiente diagrama en el cual se puede apreciar una curva estable donde la dificultad inicial pueda diferenciarse de la dificultad final.

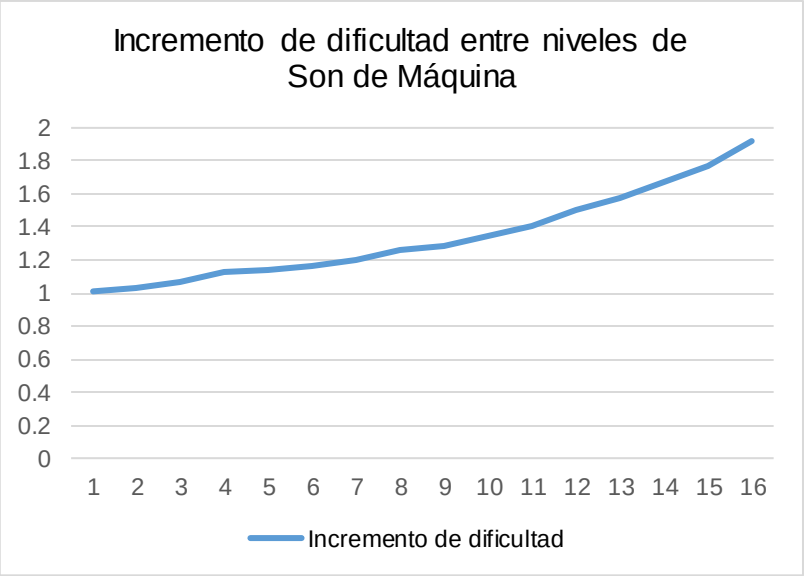


Figura 1 Curva de dificultad definida en el diseño del componente *runner* Son de Máquina (Elaboración propia).

2.4 DISEÑO DEL COMPONENTE *RUNNER* SON DE MÁQUINA

El diseño del videojuego establece la visión y el enfoque que guiará al proyecto hasta el final del proceso (Andy Hernández et al., 2017). En este epígrafe se describe el diseño del videojuego teniendo en cuenta los elementos que lo conforman, presentados en los epígrafes subsiguientes.

2.4.1 Elementos Formales del Componente *runner* Son de Máquina

Los elementos formales describen el videojuego y definen la estructura con que contará el mismo. Para ello se plantean los siguientes elementos (Andy Hernández et al., 2017):

1. **Jugador**: se caracterizan los jugadores según los aspectos definidos (Invitación a jugar, número de jugadores, roles, patrón de interacción entre jugadores, relación con las metas trazadas.)
2. **Objetivos**: definen que es lo que el jugador trata de lograr bajo un conjunto de reglas y tienen la característica de ser retadores, pero siempre se pueden realizar.
3. **Procedimientos**: definen métodos y acciones dentro del videojuego que el jugador puede ejecutar para lograr los objetivos.
4. **Reglas**: definen objetos, limitan el comportamiento dentro del videojuego y determinan efectos.
5. **Recursos**: definen elementos usados por el jugador con fin de cumplir los objetivos propuestos.

6. **Conflictos**: se generan a través de reglas, procedimientos y situaciones que no permiten cumplir los objetivos directamente
7. **Frontera o Límite**: definen todo lo que separa el videojuego de lo que no es el videojuego y está estrechamente relacionado con las metas del mismo.
8. **Resultado**: define la condición de ganado o perdido (en caso de existir).

Jugador y objetivos

De la descripción de la propuesta de solución se presentan los elementos formales referentes a la descripción de los aspectos del jugador y objetivo.

Tabla 3 Elementos Formales del Componente *runner* Son de Máquina. Jugador (Elaboración propia)

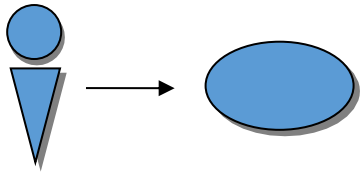
Aspecto	Descripción
Invitación a Jugar	Contador de inicio Botón de comienzo. Botón de pausa: “Pausa”
Cantidad	Un jugador.
Roles	Interacción con las interfaces del componente
Patrón de Interacción	Individual vs Juego 

Tabla 4 Elementos Formales del Componente *runner* Son de Máquina. Objetivo (Elaboración propia)

Objetivo	Descripción
Ganar la mayor cantidad de claves de Son posibles.	Para ganar una carrera es necesario recolectar todas las claves de Son necesarias, antes de que el cronómetro llegue a 0, y evitando todos los obstáculos posibles, moviendo el carro a través de tres sendas.
Tipo	Explícito
Categorías	*Runner*

Procedimientos

Estos son fundamentales para la jugabilidad del juego y determinan cómo el jugador interactúa con el mundo del juego. A continuación, se presentan los procedimientos del Componente *runner* Son de Máquina:

1. **Esquivar**: El jugador debe moverse hacia la izquierda o hacia la derecha para evitar obstáculos que bloquean el camino.
2. **Cambiar de carril**: El jugador puede moverse hacia la izquierda o hacia la derecha para cambiar de carril. Son tres carriles en total.
3. **Recoger objetos**: Estos objetos incluyen los potenciadores que fueron descritos en el epígrafe Elementos en pantalla del componente *runner* Son de Máquina. También debe recolectar las claves de Son para poder ganar el nivel. Cada potenciador se activan manualmente por el usuario pulsándolos en la pantalla.
4. **Pausar**: El jugador tiene la posibilidad de pausar el juego, mediante un botón en pantalla.

Reglas

Este componente tiene las siguientes reglas definidas:

1. El jugador gana la partida cuando colecta todas las claves de Son que se le pide en los objetivos antes que el contador de tiempo llegue a 0.
2. Si el contador de tiempo llega a 0 y no ha recolectado todas las claves de Son el jugador pierde la partida.
3. El jugador cuenta con 60 segundos (el tiempo puede variar según el testeo) para ganar la partida.
4. El jugador solo puede desplazarse por 3 pistas verticales.
5. Las claves de Son a recolectar y la velocidad del carro varían según la dificultad. A mayor dificultad mayor velocidad y más cantidad de claves a recolectar.
6. Para facilitar la partida puede hacer uso de potenciadores. Cada potenciador corresponde a una marca comercial cubana.
7. Si el jugador choca con un obstáculo se le resta tiempo al contador.
8. Cada tipo de obstáculo resta una cantidad de tiempo distinta.
9. En pantalla pueden aparecer elementos sorpresa además de las claves de Son, estos revelan espacios secretos en las casas Llamo al Son.
10. Completar todos los espacios secretos en todos los niveles pueden resultar en premios vía ranking.

Recursos

Para este componente se definen los siguientes recursos:

1. Un carro, el cual manejará el jugador.
2. Botón de pausa, que permite pausar el juego.
3. El componente cuenta con un escenario para cada provincia por la que transita el videojuego Isla del Son.
4. Los potenciadores son: Apicuba (ralentiza la carrera), Cuba Ron (gana el doble de claves de Son), EGREM (garantiza inmunidad para un obstáculo).
5. Los obstáculos son: Vallas, conos, conos, dobles, huecos.
6. Un cronómetro que indica el tiempo de la partida.
7. Contador de Claves de Son que son recogidas
8. Claves de Son, las cuales tienen forma de notas musicales, y deben ser recogidas para ganar el nivel.

Conflictos

El conflicto propio de este componente se basa en que:

Para poder llegar al nivel correspondiente (Provincia) el jugador deberá superar el nivel del minijuego (componente) anterior.

Frontera o límites

Las fronteras de este componente se reducen a haber superado los niveles de minijuegos anteriores.

Resultado

Tras haber recolectado todas las claves de Son que se le pide en los objetivos, antes que el contador de tiempo llegue a 0, el jugador alcanza la condición de ganado. Cualquier caso diferente a este determinará la condición de perdido.

Como parte del resultado esperado se muestra, a continuación, el diseño de unos de los prototipos de interfaz del componente *runner* Son de Máquina.

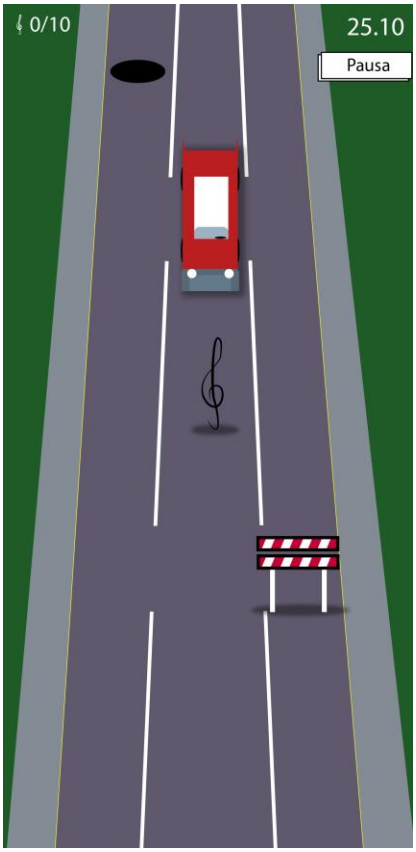


Figura 2 Prototipo de interfaz del componente *runner* Son de Máquina (Elaboración propia)

En el anterior prototipo se muestran varios recursos del componente en la pantalla de juego como son el carro, los obstáculos, las claves, así como el cronómetro, el contador de claves, el mapa y el botón de pausa. El mismo está diseñado con proporciones pensadas para una pantalla de teléfono móvil.

Otras interfaces necesarias para este componente son la interfaz de la pantalla de juego ganado que se activa una vez se alcance la condición de “Ganado”, la cual muestra dicha pantalla con el texto “Has Ganado”, además de un botón de “Salir” y “Continuar”. También se encuentra la interfaz de inicio, que muestra la pantalla de juego con el mapa de fondo y un botón con el texto “Comenzar”; esta interfaz cuenta con el cronómetro establecido en 0. Importante además es la interfaz de pausa, que contiene una pantalla con la misma estética que la interfaz de ganado; en dicha pantalla debe apreciarse el texto “Pausa”, y los botones “Reiniciar”, “Salir” y “Continuar”, además de apreciarse en el fondo el mapa con el contador y cronómetro actuales. Por último, la interfaz de perdido muestra una pantalla con la misma estética de la pantalla de ganado, donde se muestra el texto “Fin del viaje” y los botones “Reiniciar” y “Salir”. Las antes descritas pueden consultarse en los Anexos 1, 2, 3 y 4.

2.4.2 Especificación de mecanismos referentes al componente *runner* Son de Máquina

La especificación de mecanismos tiene como objetivo identificar los mecanismos que forman al videojuego, sí como sus propiedades y organización arquitectónica. Dichos mecanismos se encargan de agrupar por paquetes las funcionalidades del videojuego, permitiendo la interacción del usuario con el sistema e incrementando la jugabilidad (Andy Hernández et al., 2017).

A continuación, se presenta el catálogo de mecanismos que fueron establecidos para el desarrollo del componente *runner* para el videojuego Isla del Son.

Tabla 5 Mecanismos del componente *runner* para el videojuego Isla del Son

No.	Nombre	Descripción	Organización arquitectónica perteneciente
M1	Mecanismo Menú	Objetos: • Botones de música, continuar, juego nuevo y salir. Propiedades: • Música: El jugador puede deshabilitar el sonido. • Continuar: El jugador puede continuar la partida. • Juego nuevo: El jugador puede empezar el nivel desde el inicio • Salir: El jugador puede retirarse del nivel. Comportamientos: • El jugador accede al menú desde la pantalla de juego del componente, luego puede seleccionar las opciones que desea realizar descritas en las Propiedades. • El componente siempre tiene habilitada la opción de ir al menú. Relaciones: • M2	Paquete auxiliar
M2	Mecanismo Pausar	Objetos: • Botón pausa Propiedades: • Botón con el icono característico de pausa (dos barras verticales paralelas) Comportamiento: • El jugador pausa la partida al seleccionar el botón, y se muestra el menú. • Al pausar el mapa deja de generarse (el carro deja de avanzar) y el reloj deja de contar Relaciones • M1	Paquete Auxiliar
M3	Mecanismo Cambiar de carril	Objetos: • Carro, Carriles, Barrera Propiedades: • Carro: Representa al jugador dentro del juego. • Carriles: Tres carriles verticales por los que el carro puede desplazarse.	Paquete núcleo

		• Barrera: Barreras visibles o no, en los lados externos de los carriles no centrales que impiden avanzar más a la izquierda o derecha respectivamente. Comportamiento: • El jugador puede deslizar el dedo por la pantalla de forma horizontal haciendo que el carro cambie de carril en dependencia de la dirección del deslizamiento del dedo • Si el carro se encuentra en el primer o tercer carril (carriles no centrales), las barreras impedirán que el carro siga moviéndose hacia los lados Relaciones: • M4 • M5 • M7	
M4	Mecanismo Obstáculos	Objetos: • Obstáculos Propiedades: • Obstáculos que aparecen en el camino entorpeciendo el avance del carro. Comportamiento: • Si en el avance o en el cambio de carril, el carro intercepta un obstáculo, se descuenta tiempo del cronómetro Relaciones: • M3 • M7 • M10	Paquete núcleo
M5	Mecanismo Generar Mapa Aleatoriamente	Objetos • Porciones de calle Propiedades: • Porciones de calle que aparecen aleatoriamente dando lugar al mapa Comportamiento • Mediante un algoritmo de *pools* la calla va generándose aleatoriamente con porciones que pueden contener obstáculos o Claves de Son, o pueden estar vacías. Esto da la ilusión de que el carro se está moviendo Relaciones: • M2 • M4 • M6 • M8 • M12	
M6	Mecanismo Administrar Potenciadores	Objetos: • Panel de potenciadores Propiedades: • Un panel que expone los potenciadores existentes en el juego, tanto disponibles como no disponibles. Comportamiento: • Si en el avance del carro, el jugador toca uno de los potenciadores disponibles, dicho potenciador se activa causando el efecto descrito en el punto 4 de la lista de recursos. Relaciones: • M4 • M5 • M8	Paquete núcleo

M7	Mecanismo Pantalla de Juego Perdido	Objetos: • Imagen de juego perdido, Salir, Reintentar Propiedades: • Imagen de juego perdido: una imagen que muestra, entre otros ornamentos, el texto “Fin del Viaje” • Reintentar: El jugador puede empezar el nivel desde el inicio. • Salir: El jugador puede retirarse al menú principal. Comportamiento: • El jugar puede reintentar el nivel o retirarse presionando los botones respectivos. • La pantalla se activa si se alcanza la condición de “Perdido”: se acaba el tiempo y no se ha cumplido el objetivo del nivel. Relaciones: • M9 • M10	Paquete núcleo
M8	Mecanismo Claves del Son	Objetos: • Clave del Son Propiedades: • Claves de Son que aparecen distribuidas en los tres carriles Comportamiento: • Si en el avance o en el cambio de carril, el carro toca una Clave de Son, la adquiere y el número de estas se incrementa. Relaciones: • M5 • M9	Paquete núcleo
M9	Mecanismo Puntuaciones	Objetos: • Panel de puntuaciones Propiedades: • Un panel con el que no puede interactuar el jugador, que muestra la cantidad de claves del Son adquiridas. Comportamiento: • Si el jugador obtiene una clave del Son, el número en el panel se incrementa. • Si no obtiene la cantidad de claves indicadas para el nivel, se alcanza la condición de “Perdido” • Al finalizar el nivel y haber obtenido la condición de “Ganado” las puntuaciones son exportadas Relaciones: • M8 • M7 • M11	Paquete núcleo
M10	Mecanismo Cronometro	Objetos: • Reloj Propiedades: • Un reloj que muestra el transcurso del tiempo prefijado para el nivel. Comportamiento: • El tiempo que se visualiza en el reloj va transcurriendo y una vez que se acabe, si no se obtuvieron la cantidad necesaria de claves del Son para superar el nivel, alcanza la condición de “Perdido” Relaciones: • M7 • M11	Paquete núcleo

M11	Mecanismo Pantalla de Ganado	Objetos: • Imagen de juego ganado, puntaje, continuar Propiedades: • Imagen de juego ganado: Una imagen que muestra, entre otros ornamentos, el texto “Has Ganado”. • Puntaje: texto que muestra el puntaje obtenido en el nivel tras superarlo. • Continuar: Botón que permite continuar hacia el siguiente nivel. Comportamiento: • Si el jugador ha obtenido la cantidad necesaria de Claves del Son en el tiempo definido y sin haber chocado con ningún obstáculo, se alcanza la condición de “Ganado” Relaciones: • M9 • M10	Paquete núcleo
M12	Mecanismo Definir Dificultad	Objetos: • Dificultad Propiedades: • Dificultad: archivo proveniente del *Game Manager* de Isla del Son que permite definir la dificultad propia del nivel Comportamiento: • Establece la dificultad propia del nivel (Consultar Tabla 2) Relaciones • M5	Paquete núcleo

2.4.3 Definición de la interacción entre mecanismos

Del catálogo de mecanismos anterior se obtuvo se obtuvo el siguiente diagrama que modela la interacción entre los mecanismos.

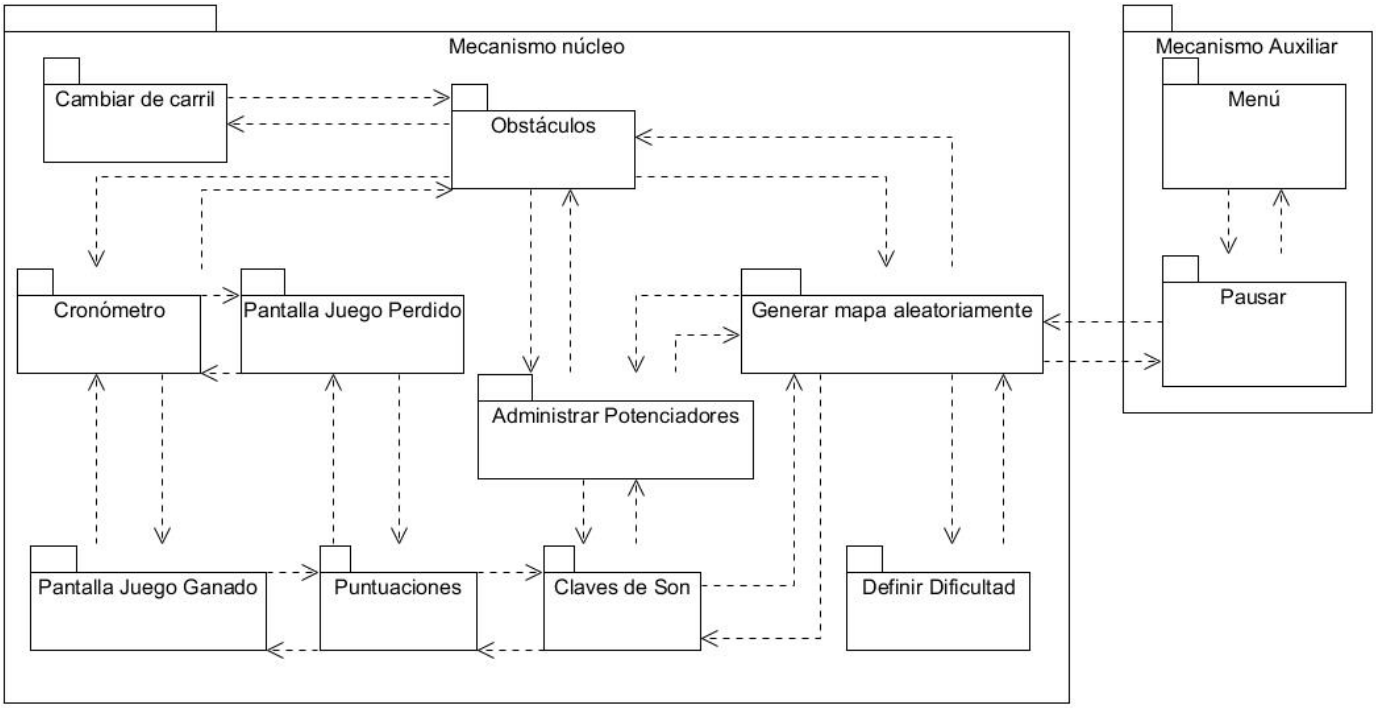


Figura 3 Diagrama de paquetes de los mecanismos del Componente *runner* (Elaboración propia).

De este modelo debe definirse:

- **Mecanismo Núcleo**: Se refiere a los mecanismos que son indispensables para jugar el juego. Desde el punto de vista del jugador, son todos los mecanismos asociados que forman la lógica del mismo (Andy Hernández et al., 2017).
- **Mecanismo Auxiliar**: Son aquellos mecanismos que son identificados como alternativos, los cuales le ofrecen soporte al videojuego. Hacen referencia generalmente a los mecanismos asociados a opciones (Andy Hernández et al., 2017).

2.4.4 Definición arquitectónica para el componente *runner* Son de Máquina

A partir del estudio realizado, se propone una arquitectura basada en la combinación de los estilos arquitectónicos: arquitectura en capas y arquitectura basada en componentes, para estructurar los diferentes elementos necesarios para esta propuesta de solución. Los componentes de cada capa se comunican con otros componentes en otras capas a través de interfaces definidas o instancias de clases (Crnkovic et al., 2011).

Los componentes de cada capa se comunican con otros componentes en otras capas a través de interfaces definidas o instancias de clases (en el caso de las clases se comunican con el Game Manager únicamente dado que provee una fachada para las clases de la Capa Principal para interactuar con las demás). En la siguiente figura se observa la distribución de las capas presentes en la arquitectura propuesta.

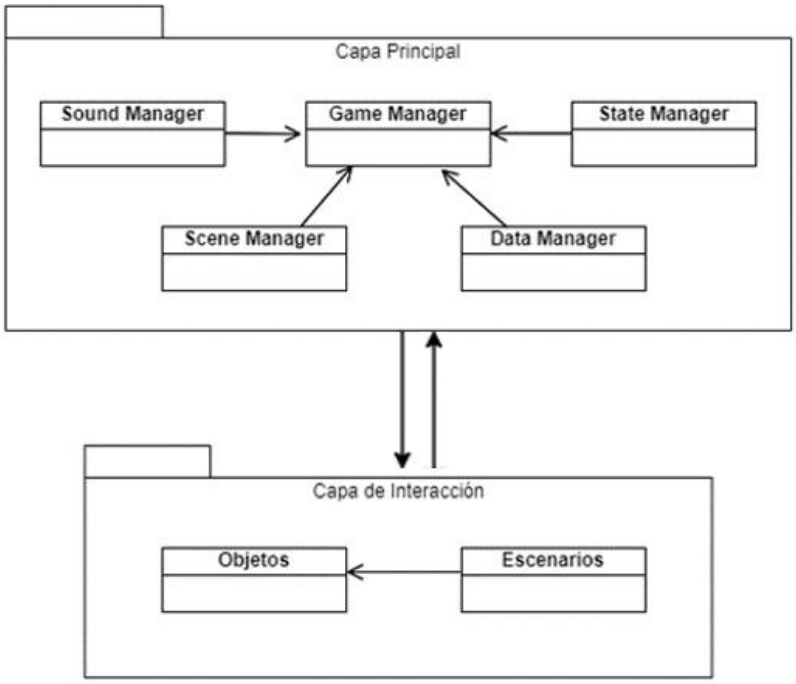


Figura 4 Diagrama arquitectónico del componente *runner* (Modificado por el autor).

Del modelo anterior es necesario definir sus elementos:

Capa de Presentación: esta capa está formada por los principales elementos del videojuego.

- **GameManager**: es el controlador principal del videojuego, se implementa como un GameObject que se encuentra en todas las escenas del videojuego.
- **SoundManager**: controla el sonido del videojuego y particularmente de cada elemento o acción que active un sonido.
- **SceneManager**: maneja las escenas del videojuego y los cambios que ocurren entre ellas.
- **DataManager**: maneja todos los datos del videojuego, además se encarga de cargar y guardar el estado del videojuego.
- **StateMachine**: se encarga de definir los estados del videojuego.

Capa de Interacción: en esta capa se encuentran los diferentes escenarios y los objetos que pertenecen a los mismos.

2.4.5 Diseño de clases del componente *runner* Son de Máquina

Una vez definidos los mecanismos propios del componente *runner* Son de Máquina, así como la relación entre ellos y el patrón arquitectónico a seguir, se modelaron las clases del componente a través de diagramas de clases.

Los diagramas de clases permiten modelar elementos o componentes basados en clases donde cada escenario de uso implica un conjunto de objetos o elementos que puedan ser manipulados y de los cuales puedan definirse atributos y acciones o funcionalidades que puedan realizar (Pressman & Maxim, 2020). A continuación de define el diagrama de clases de los mecanismos Cambiar de Carril y Administrar Potenciadores, presentados como ejemplo. El resto de diagramas de clases podrán ser consultados en los Anexos.

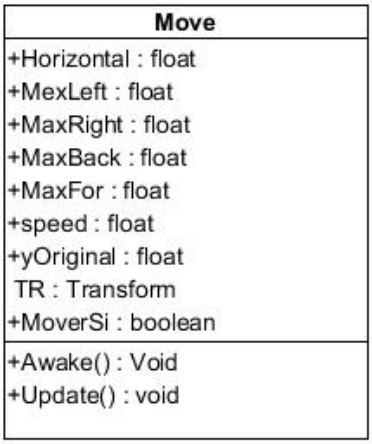


Figura 5 Diagrama de clase del Mecanismo Cambiar de Carril

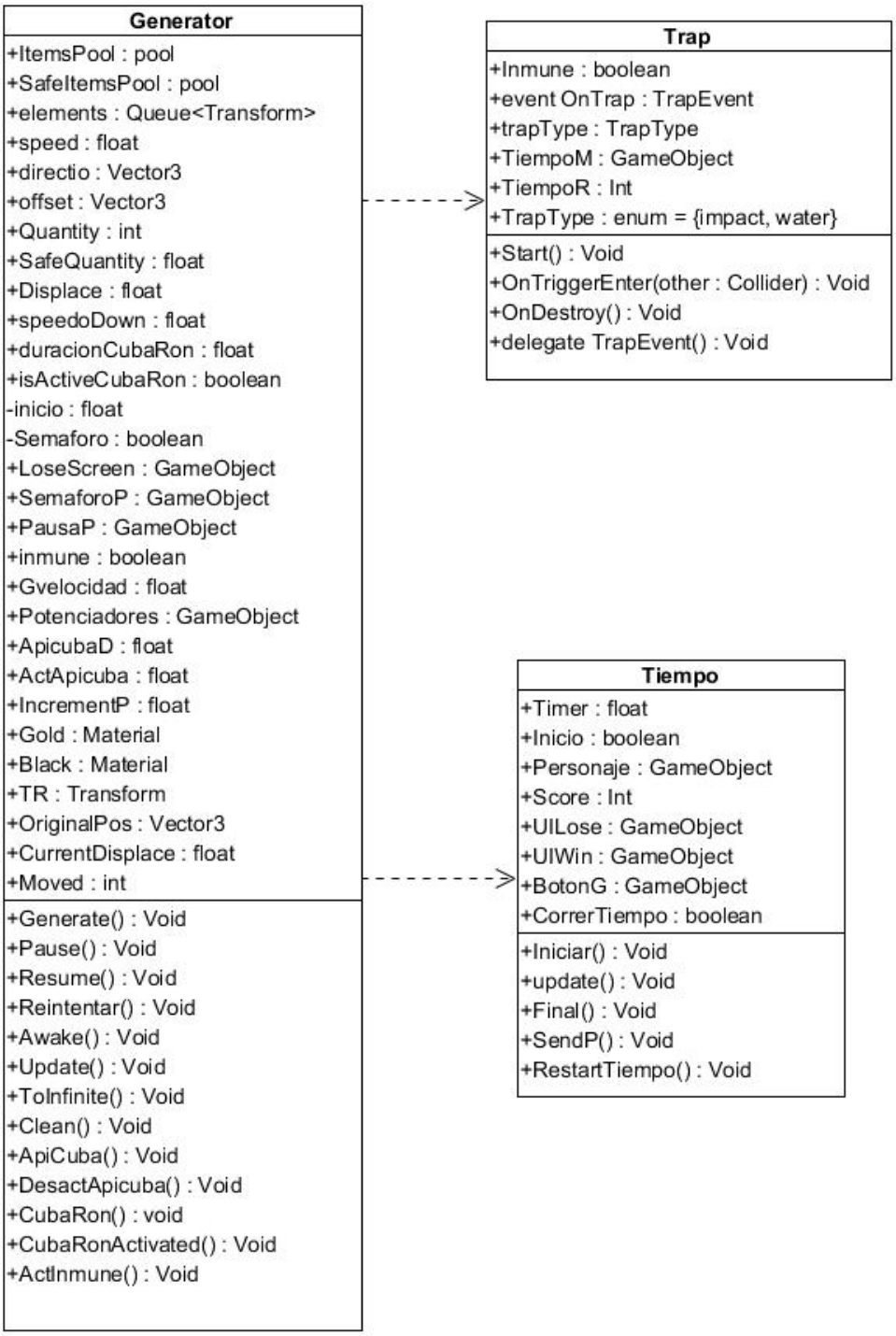


Figura 6 Diagrama de Clases del Mecanismo Administrar Potenciadores

2.5 PATRONES DE DISEÑO

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Contribuyen a reutilizar diseño gráfico identificando aspectos claves de la estructura de un diseño que puede ser aplicado en una gran cantidad de situaciones (Pressman & Maxim,

2020). Por su parte, (Larman, 2003) indica que un patrón es una solución a un problema en un contexto, codifica conocimiento específico acumulado por la experiencia en un dominio.

En el diseño del componente *runner* Son de Máquina se utilizarán algunos de los Patrones Generales de Software de Asignación de Responsabilidad (GRASP, por sus siglas en inglés) y algunos de los patrones Pandilla de cuatro (GOF, por sus siglas en inglés). Además de patrones de diseño propios del desarrollo de este tipo de videojuegos.

Los patrones GRASP usados en la propuesta se enuncian a continuación (Larman, 2003):

- **Experto**: clase que tiene la responsabilidad de ejecutar una acción determinada y cuenta con el acceso a los datos necesarios. Responde a: ¿Quién asumirá la responsabilidad en el caso general?
- **Creador**: se encarga de crear instancias de una clase o de un objeto. Responde a ¿Quién crea?
- **Alta Cohesión**: cada clase realiza solo las funciones que necesita.
- **Bajo Acoplamiento**: una clase no depende de muchas clases.
- **Controlador**: clase que se encarga de manejar los objetos del sistema. Responde a: ¿Quién administra un evento del sistema?

En este caso, la clase **Generator** responde a los patrones Experto y Controlador. Esto está evidenciado en la gestión de los elementos y resto de clases, siendo **Generator** quien asumirá el control de la generación aleatoria del mapa, haciendo uso de la clase **Pool**. De la misma forma, **Generator** puede acceder a los datos referentes a las puntuaciones a través de la clase **Tiempo**, la cual a su vez implementa el patrón Controlador al manejar las clases **Notas** y **Puntos**.

Por su parte, la ya mencionada clase **Pool** implementa el patrón Creador, en interrelación con su clase interna **PoolItem**, la cual, mediante la funcionalidad Instantiate, permite crear instancias que son usadas esencialmente en **Pool**.

Todo ello contribuye al Bajo Acoplamiento y la Alta Cohesión garantizando la responsabilidad única de cada clase, y permitiendo que cada clase dependa de la cantidad estrictamente necesaria de otras. En este último caso, quien tiene mayor cantidad de dependencias es **Generator** pues cumple con la característica de ser la clase generadora de la mayoría de recursos del componente.

También son empleados en el desarrollo del componente, los patrones GoF que pueden dividirse en tres categorías principales (Larman, 2003):

- **Creacionales**: tratan con las formas de crear instancias de objetos. Su objetivo es abstraer el proceso de instanciación y ocultar los detalles de cómo los objetos son creados o inicializados. Ejemplos de estos son los siguientes (Fábrica abstracta, Constructor, Método de fabricación, Prototipo, Instancia única).
- **Estructurales**: describen como las clases y objetos pueden ser combinados para formar grandes estructuras y proporcionar nuevas funcionalidades. Estos objetos adicionados pueden ser incluso objetos simples u objetos compuestos. Ejemplos de estos son (Adaptador, Puente, Compuesto, Decorador, Fachada, Peso ligero).
- **Comportamiento**: ayudan a definir la comunicación e iteración entre los objetos de un sistema. El propósito de este patrón es reducir el acoplamiento entre los objetos. Ejemplo de estos patrones son (Cadena de responsabilidad, Orden, Intérprete, Iterador, Mediador, Observador, Estado, Estrategia, Método plantilla, Visitante).

De los anteriormente mencionados, para el componente Son de Máquina, serán utilizados los que se describen a continuación (Larman, 2003; Unity Learn, 2022):

- **Fachada**: es el que eleva el nivel de abstracción de un determinado sistema para ocultar ciertos detalles de implementación y hacer más sencillo su uso. Evita el exceso de instancias de una clase, dejando a las subclases accesibles para ser usadas directamente. Se evidencia en el método ToInfinite de la clase **Generator** en su relación con la clase **Pool**, en lo cual, esta última funciona como fachada para que el método ToInfinite sea capaz de acceder a la creación de instancias de **PoolItem** (subclase de **Pool**) a través del atributo ItemsPool de la clase **Generator**.
- **Singleton**: se asegura de que una clase solo tenga una instancia, y provee un punto de acceso global a ella. El ejemplo mencionado anteriormente evidencia la implementación de este patrón, pero también puede identificarse el mismo en la implementación de las clases **Puntos**, **Notas** y **Trap** las cuales no permiten las múltiples instancias ya que se necesita un único acceso a las mismas.
- ***Pooling* de Objetos**: es un patrón de diseño creacional, ampliamente implementado en la creación de videojuegos, que consiste en la creación de pre-instancias de los objetos para que puedan ser utilizados con posterioridad. Esto fue empleado en la generación aleatoria del mapa de manera que se pre-instanciaran porciones de calle

con obstáculos, otras con Claves del Son, otras vacías, etcétera. Importante es destacar que este patrón no entra en conflicto con Singleton, ya que dichas pre-instancias no se corresponden con instancias propiamente dichas de una clase, sino que se refiere a *presets*[3], que permiten luego crear dichas instancias.

2.6 REPRESENTACIÓN DEL COMPORTAMIENTO DE LOS MECANISMOS DEL COMPONENTE *RUNNER* SON DE MÁQUINA

Para la representación del comportamiento desde el punto de vista lógico de los mecanismos de Son de Máquina, se emplean los diagramas de estado. Un diagrama de estado son empleados en el modelado de comportamiento, en el cual deben ser consideradas dos caracterizaciones de estado: el estado en el que cada elemento del sistema realiza sus funciones y el estado del sistema observado desde afuera. Un diagrama de estados representa los estados activos de cada elemento (en este caso, de cada mecanismo) y su comportamiento basado en eventos (Pressman & Maxim, 2020).

A continuación, se modelan los diagramas de estado propios de los mecanismos Cambiar de Carril y Administrar potenciadores. El resto de diagramas pueden ser consultados en los Anexos.

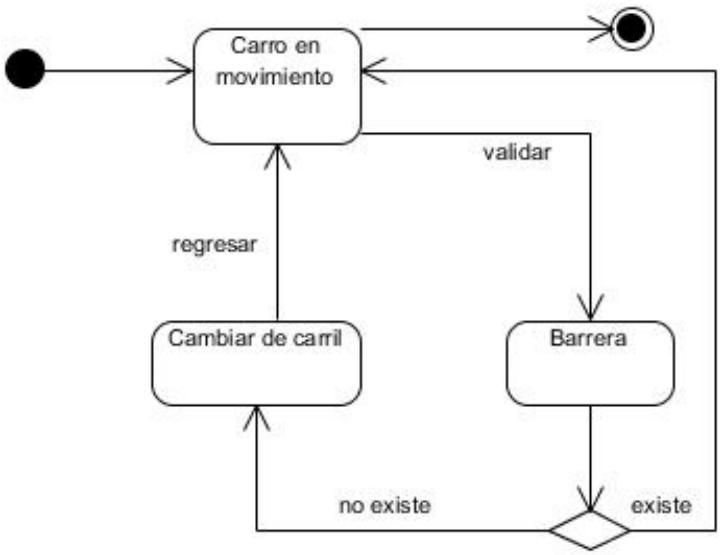


Figura 7 Diagrama de Estado del Mecanismo Cambiar de Carril (Elaboración propia).

[3] Preset se refiere al conjunto de configuraciones y/o ajustes que pueden predeterminarse para aplicar a la vez.

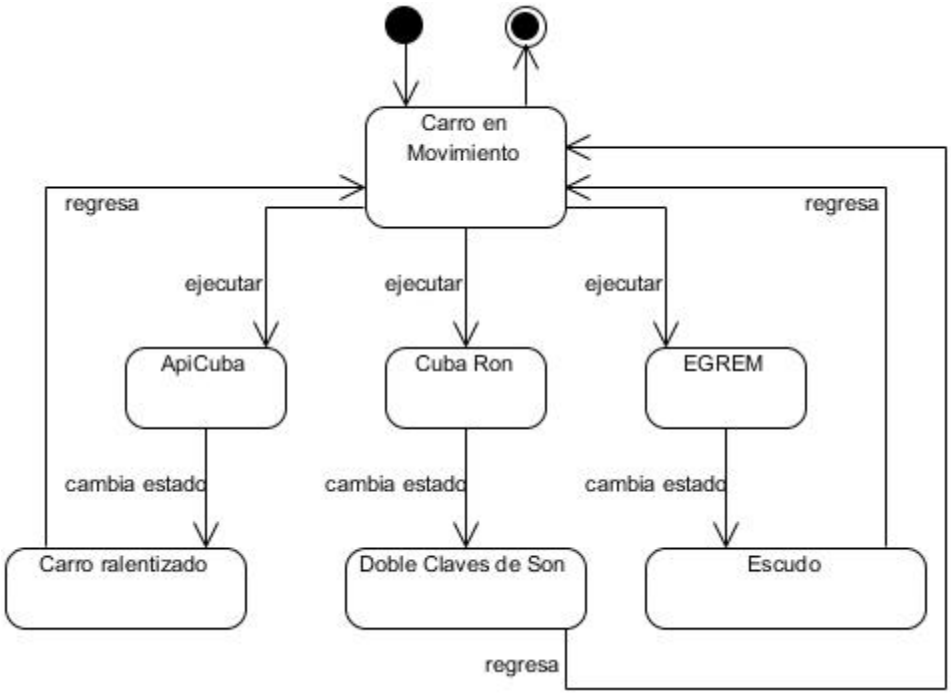


Figura 8 Diagrama de Estado del Mecanismo Administrar Potenciadores (Elaboración propia).

2.7 CONCLUSIONES DEL CAPÍTULO

Tras finalizar lo referente al diseño del componente *runner* Son de Máquina puede concluirse que la definición de los 12 mecanismos del componente, así como el modelado de las clases intervinientes en el mismo permitió obtener un diseño estructurado que permite la implementación guiada y ordenada de las diferentes partes de Son de Máquina. A esto contribuye la definición de la relación entre los mecanismos. Una vez analizado lo anterior es posible concluir que la definición de un modelo arquitectónico mixto (en este caso arquitectura en capas y basada en componentes) es apropiada para el desarrollo de un componente que constituye un minijuego dentro de un juego mayor, debido a que este modelo ofrece una interrelación entre elementos principales (o núcleo) y elementos auxiliares, de manera que puedan crearse componentes (*scripts* en este caso) que pueden contribuir tanto a su propia capa, como al minijuego en desarrollo y a la vez, al videojuego mayor. Todo ello guiado, en el caso específico de Son de Máquina, por el patrón de diseño *Pooling* de Objetos, que puede definirse como el principal dentro del desarrollo de un *runner* con las características actuales.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS DEL COMPONENTE *RUNNER* SON DE MÁQUINA PARA EL VIDEOJUEGO ISLA DEL SON

3.1 INTRODUCCIÓN

En este capítulo se definen los estándares de codificación. Además, se presenta la validación de la propuesta de solución a partir de las pruebas de aceptación y se presenta el nivel de satisfacción de los usuarios aplicando la técnica de Iadov.

3.2 ESTÁNDARES DE CODIFICACIÓN EN EL COMPONENTE *RUNNER* SON DE MÁQUINA

Los estándares de codificación son un conjunto de convenciones, establecidas para la escritura de código. Estos varían en dependencia del lenguaje de programación, lo que posibilitan una mejor lectura e interpretación del software. Su empleo permite que el código sea de fácil comprensión por parte de los programadores (Fernández, 2013).

Para el desarrollo de la solución se especifican estándares para la escritura del código. Ello se presenta a continuación.

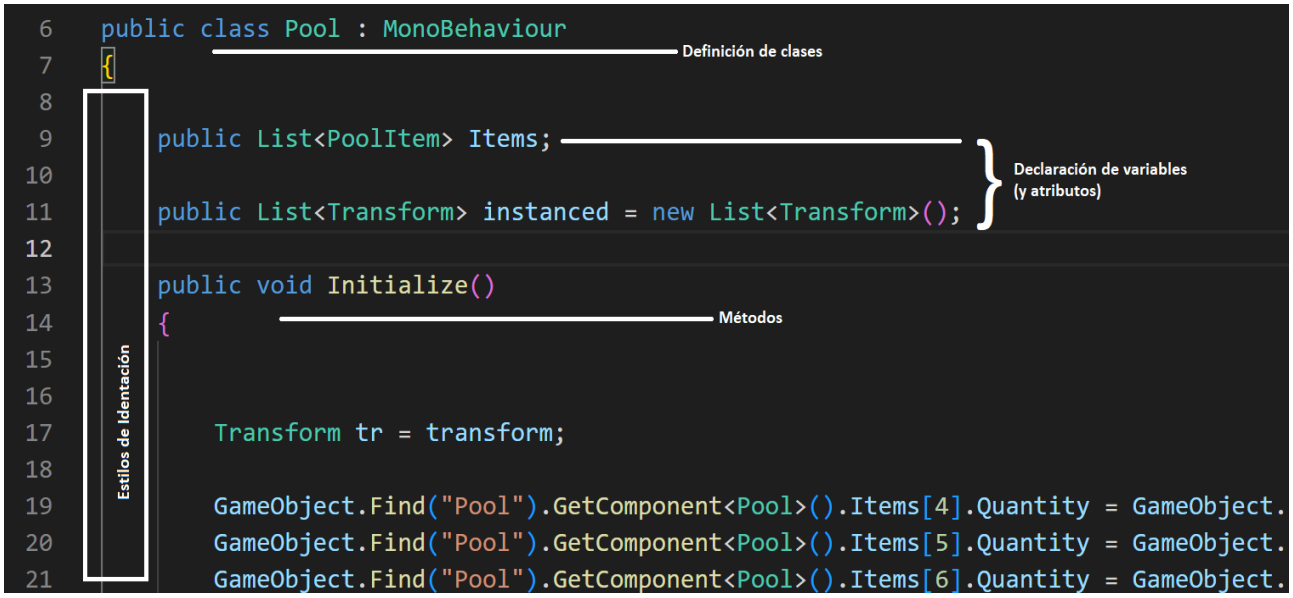


Figura 9 Aplicación de los estándares de codificación en el componente *runner* Son de Máquina (Elaboración Propia).

- **Definición de clases**

Las clases comienzan con mayúscula al inicio de la palabra y en caso de estar conformada por palabras compuestas, la definición debe ser continua y cada palabra debe iniciar con minúscula o mayúscula siguiendo el estilo determinado.

- **Declaración de variables**

Los nombres de las variables comienzan con minúscula y en caso de estar conformada por palabras compuestas, la definición puede estar continua de manera que cada palabra comience con mayúscula.

- **Métodos**

El nombre de los métodos debe comenzar con minúscula y en caso de estar conformada por palabras compuestas puede estar continua y cada palabra debe iniciar con mayúscula.

- **Estilo de indentación**

El estilo utilizado en la implementación de la propuesta de solución es propio para lenguajes de programación que usan llaves para indentar o delimitar bloques lógicos de código y es también un punto clave para hacer el código más legible. Está presente en los ciclos y estructuras de control.

3.3 DIAGRAMA DE COMPONENTES DEL COMPONENTE *RUNNER* SON DE MÁQUINA

Los diagramas de componentes son utilizados para estructurar el modelo de la implementación. Permiten modelar una vista estática del sistema, muestran la organización y las dependencias lógicas entre un conjunto de componentes del software, que pueden ser librerías, binarios, ejecutables y códigos fuentes (Pressman & Maxim, 2020).

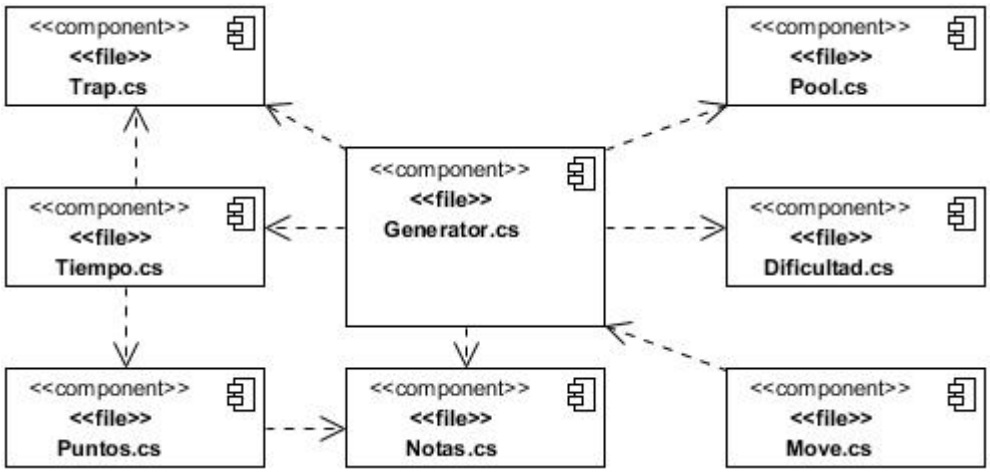


Figura 10 Diagrama de componentes del componente *runner* Son de Máquina (Elaboración propia).

3.4 PRUEBAS DE ACEPTACIÓN DEL COMPONENTE *RUNNER* SON DE MÁQUINA

Las pruebas del software, conocidas también como técnicas de evaluación dinámica, son un elemento crítico para la garantía de la calidad del sistema. Representan una revisión

final de las especificaciones del diseño y de la implementación. Su principal objetivo es diseñar pruebas que, sistemáticamente, saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo (Pressman & Maxim, 2020).

Al construir software a la medida para un cliente, se realiza una serie de pruebas de aceptación a fin de permitir al cliente validar todas las funcionalidades. Realizada por el usuario final en lugar de por los ingenieros de software, una prueba de aceptación puede variar desde una "prueba de conducción" informal hasta una serie de pruebas planificadas y ejecutadas sistemáticamente. De hecho, la prueba de aceptación puede realizarse durante un periodo de semanas o meses, y mediante ella descubrir errores acumulados que con el tiempo puedan degradar el sistema (Caillois, 1986).

Las pruebas de aceptación se usan para evaluar la buena disposición de un software para su despliegue y uso. Para realizar tanto las pruebas Alfa como las pruebas Beta se debe presentar al equipo de pruebas una guía de jugabilidad que recoja los escenarios principales que pueden ejecutarse con la interacción usuario-videojuego. Esto permitirá a los usuarios de pruebas conocer el propósito de las opciones que se diseñan para el producto. Puede considerarse como un manual de usuario, pero no con el mismo nivel de detalles (Andy Hernández et al., 2017).

- **Pruebas Alfa**

Se lleva a cabo en el sitio del desarrollador por un grupo representativo de usuarios finales. El software se usa en un escenario natural con el desarrollador "mirando sobre el hombro" de los usuarios y registrando los errores y problemas de uso. Las pruebas Alfa se realizan en un ambiente controlado (Caillois, 1986).

- **Pruebas Beta**

Se realiza en uno o más sitios del usuario final. A diferencia de la prueba Alfa, por lo general el desarrollador no está presente. Por tanto, la prueba Beta es una aplicación "en vivo" del software en un ambiente que no puede controlar el desarrollador. El cliente registra todos los problemas (reales o imaginarios) que se encuentran durante la prueba Beta y los reporta al desarrollador periódicamente. Como resultado de los problemas reportados durante las pruebas Beta, es posible hacer modificaciones y luego preparar la liberación del producto de software a toda la base de clientes (Caillois, 1986).

3.4.1 Pruebas Alfa

Para la aplicación de las pruebas Alfa se seleccionaron un grupo de personas que se han desempeñado en diferentes áreas del proceso de desarrollo de software. A continuación, se relacionan los involucrados en las pruebas.

Tabla 6 Relación de probadores designados para las pruebas alfa (Elaboración propia)

Nombres y apellidos	Cargo	Años de Experiencia
Ing. William René Zaldivar Pupo	Especialista A en Ciencias Informáticas	3
Ing. Jorge Evelio Valdivia Hernández	Especialista B en Ciencias Informáticas	3
Ing. Yaniel Antonio Sánchez Domínguez	Especialista B en Ciencias Informáticas	4
Ing. Michel Pedrera Suen	Profesor	4
Antonio Jesús Calvet Rodríguez	Estudiante	1

Estas pruebas se desarrollaron en un ambiente para el cual se contó con cinco computadoras con sistema operativo Windows 10 y Ram de 8 GB. Además, fueron ejecutadas en tres iteraciones, dentro de las cuales fueron encontradas 14 no conformidades y 5 no conformidades en la primera y segunda iteraciones respectivamente, y en la última no fueron detectadas no conformidades. Se presenta el listado de las no conformidades detectadas.

Tabla 7 Listado de las no conformidades detectadas (Elaboración propia)

Jugabilidad y mecánicas									
No.	**It.**	**Probador**	**Responsable**	**Descripción del error**	**Comportamiento esperado**	**Tipo de error**	**Impacto**	**Plataforma**	**Estado de la NC**
1	1	Ing. William René Zaldivar Pupo	Antonio C.R.	Al tocar las teclas W A S D, el carro no cambia de carril	El carro debe cambiar de carril	Funcionalidad	Alto	Windows 10	Resuelta
2	1	Ing. William René Zaldivar Pupo	Antonio C.R.	Al tocar un potenciador toma un tiempo antes de lanzarlo	El potenciador debe lanzarse inmediatamente	Funcionalidad	Medio	Windows 10	Resuelta
3	1	Ing. William René Zaldivar Pupo	Antonio C.R.	El botón de pausa no funciona	El botón de pausa debe pausar el juego y abrir el menú	Opción que no funciona	Alto	Windows 10	Resuelta

4	1	Ing. William René Zaldivar Pupo	Antonio C.R.	El carro atraviesa un obstáculo y no pasa nada	Al chocar con un obstáculo, el contador debe decrementarse	Funcionalidad	Alto	Windows 10	Resuelta
5	1	Ing. Yaniel Antonio Sánchez Domínguez	Antonio C.R.	Al acabarse el tiempo, aparece la pantalla de "perdido" pero no aparece el texto de juego perdido	Al acabarse el tiempo debe aparecer la pantalla de juego perdido con una imagen con el texto "Fin del viaje"	Funcionalidad	Bajo	Windows 10	Resuelta
6	2	Ing. Yaniel Antonio Sánchez Domínguez	Antonio C.R.	El botón de pausa, pausa el juego, pero no abre el menú	El botón de pausa debe pausar el juego y abrir el menú	Opción que no funciona	Medio	Windows 10	Resuelta
7	2	Ing. Yaniel Antonio Sánchez Domínguez	Antonio C.R.	Al tocar un potenciador no pasa nada	Al tocar un potenciador, el mismo debe lanzarse inmediatamente	Funcionalidad	Alto	Windows 10	Resuelta
Interfaz									
8	1	Ing. Jorge Evelio Valdivia Hernández	Antonio C.R.	El panel de potenciadores se ve desplazado de la pantalla	El panel de potenciadores debe mostrarse completo	Error de interfaz	Medio	Windows 10	Resuelta
9	1	Ing. Jorge Evelio Valdivia Hernández	Antonio C.R.	El cronómetro se ve desplazado fuera de la pantalla	El cronómetro debe mostrarse completo	Error de interfaz	Medio	Windows 10	Resuelta
10	1	Ing. Jorge Evelio Valdivia Hernández	Antonio C.R.	Error ortográfico en la pantalla de juego ganado "Haz Ganado"	La pantalla debe mostrar el texto "Has Ganado"	Ortografía	Bajo	Windows 10	Resuelta
11	1	Ing. Jorge Evelio Valdivia Hernández	Antonio C.R.	Al quitar la pausa, la interfaz de usuario desaparece	La interfaz de usuario debe permanecer	Error de interfaz	Alto	Windows 10	Resuelta
12	1	Antonio Jesús Calvet Rodríguez	Antonio C.R.	No se ve con claridad la imagen del carro	Debe mostrarse nítido	Error técnico	Bajo	Windows 10	Resuelta
13	1	Antonio Jesús Calvet Rodríguez	Antonio C.R.	No se ve con claridad la pantalla de Juego Perdido	Debe mostrarse nítido	Error técnico	Bajo	Windows 10	Resuelta

14	1	Antonio Jesús Calvet Rodríguez	Antonio C.R.	La pantalla de pausa muestra el texto “pausa”	Debe mostrar el texto “Pausa” con inicial mayúscula	Ortografía	Bajo	Windows 10	Resuelta
15	2	Ing. Michel Pedrera Suen	Antonio C.R.	La interfaz de usuario aparece desplazada fuera de la pantalla	La interfaz de usuario debe mostrarse completa	Error de interfaz	Medio	Windows 10	Resuelta
16	2	Ing. Michel Pedrera Suen	Antonio C.R.	La imagen del carro se ve claramente cuando avanza pero no se ve nítida cuando cambia de carril	Debe mostrarse nítido	Error técnico	Bajo	Windows 10	Resuelta
17	2	Ing. Michel Pedrera Suen	Antonio C.R.	Parte de la pantalla de Juego Perdido no se ve claramente	Debe mostrarse nítido	Error técnico	Bajo	Windows 10	Resuelta
Rendimiento									
19	1	Antonio Jesús Calvet Rodríguez	Antonio C.R.	Al quitar la pausa el juego se demora en responder	Cargar la interfaz inmediatamente	Funcionalidad	Bajo	Windows 10	Resuelta

Análisis de causa y efecto. Diagrama de Ishikawa

El diagrama de Ishikawa o técnica de análisis de causa-efecto (conocido también como diagrama de espina de pescado por su estructura) consiste en una representación gráfica que permite visualizar las causas que explican un determinado problema. A partir del problema o efecto, se enumera un conjunto de causas que potencialmente explican dicho comportamiento. Adicionalmente cada causa se puede desagregar con grado mayor de detalle en subcausas. Esto último resulta útil al momento de tomar acciones correctivas dado que se deberá actuar con precisión sobre el fenómeno que explica el comportamiento no deseado (Rodríguez, 2023). En este contexto, la representación de la técnica de análisis de causa-efecto se muestra a continuación:

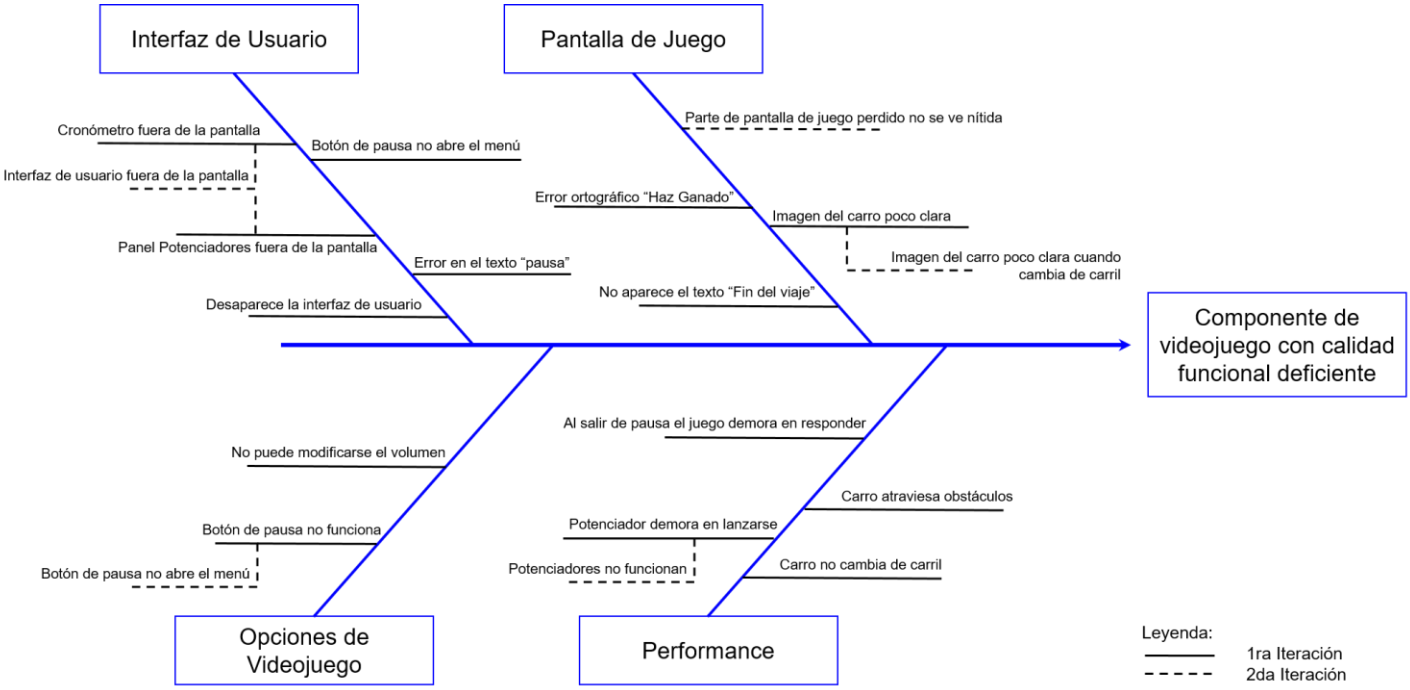


Figura 11 Diagrama de Ishikawa, resultante de la técnica de Causa-Efecto (Elaboración propia).

A partir de lo anterior, se presenta entonces una lista de acciones correctivas usadas para mitigar las no conformidades enumeradas.

Tabla 8 Acciones correctivas aplicadas a las no conformidades detectadas (Elaboración propia)

No. De NC	Acción correctiva
1	Corregir el código para que el cambio de carril se ejecute
2	Eliminar el tiempo de carga antes de lanzar el potenciador
3	Corregir el código para que el botón de pausa sea funcional
4	Corregir las propiedades de los obstáculos para que al ser atravesados, contador se afecte
5	Agregar el texto "Fin del Viaje" a la pantalla de perdido
6	Corregir el código del botón de pausa para que abra el menú
7	Corregir las propiedades de los potenciadores para que se activen al ser tocados
8	Corregir la posición del panel de potenciadores
9	Corregir la posición del cronómetro
10	Corregir el texto de la pantalla de ganado a "Has ganado"
11	Corregir el código de los componentes referentes a Interfaz y Menú pausa
12	Aumentar la calidad del modelo
13	Aumentar la calidad de la imagen
14	Corregir el texto a "Pausa"
15	Modificar las proporciones y posicionamiento de la interfaz de usuario
16	Aumentar la calidad de la imagen en la animación de doblado
17	Aumentar la calidad de la imagen

18	Optimizar las interfaces, imágenes y modelos

Es importante mencionar que no se comenzó una iteración hasta haber resuelto las no conformidades de la iteración anterior. Luego de haber realizado las tres iteraciones mencionadas, y de haber resuelto las no conformidades antes descritas, se puede modelar la evolución de las mismas en el siguiente gráfico:

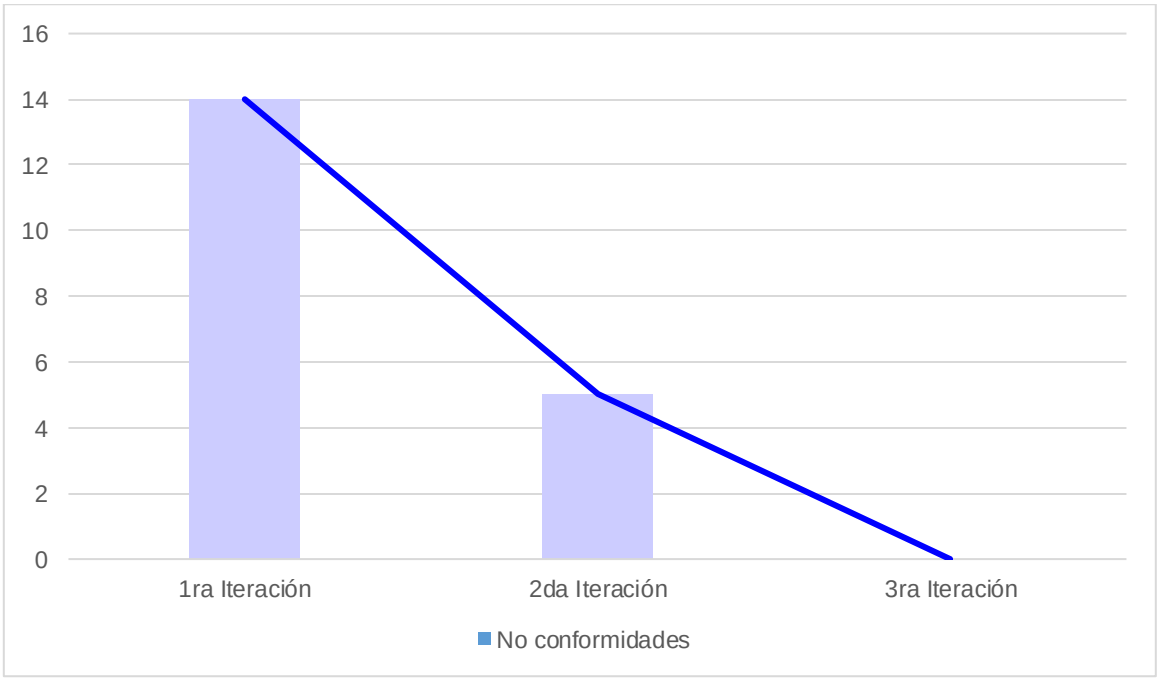


Figura 12 Análisis temporal d la reducción de no conformidades detectadas respecto a las iteraciones realizadas (Elaboración propia).

3.4.2 Pruebas Beta

Según (Ibarra Monteagudo, 2018), la pruebas beta dan por concluidas todas las variaciones de contenido, y lo que fue un proyecto ya se ha convertido en un producto. Es por ello que para la presente investigación se presenta una estrategia de pruebas beta, a la cual se le dará seguimientos una vez culminada la presente. Ello se describe a continuación

Los objetivos de la realización de estas pruebas beta serán:

- Identificar fallos y errores
- Evaluar la jugabilidad
- Recibir retroalimentación sobre la dificultad del juego
- Recopilar opiniones de los jugadores

Para ello se identifica un público objetivo y se determina qué tipo de jugadores formarán parte de esta etapa de pruebas. Para este caso se seleccionan tanto jugadores casuales como jugadores asiduos de videojuegos móviles.

Consecuentemente debe hacerse una selección de participantes. Para la aplicación de estas pruebas beta se selecciona un grupo de 10 estudiantes de la Universidad de las Ciencias Informáticas que pudieran tomar el rol de usuario final, pues se prefiere que estos usuarios sean entes externos al proyecto. Se les solicita compromiso para culminar las pruebas y proporcionar comentarios.

Se definen las métricas y un formulario de retroalimentación. Dichas métricas se deben tener en cuenta individualmente de los resultados textuales de la aplicación del formulario, y deben servir para analizar la objetividad de las opiniones en los formularios. Dichas métricas son:

- Tiempo que dedicó el probador beta al juego
- Tasa de finalización de niveles
- Número de errores o fallos detectados por los probadores beta

Además, se presenta a continuación el formulario

Tabla 9 Formulario de retroalimentación para las pruebas beta (Elaboración propia)

Datos Demográficos					
Edad:	Género:	Experiencia previa en videojuegos móviles:	Alta ___	Media ___	Baja ___
Jugabilidad					
¿Considera que las mecánicas de juego son intuitivas y fáciles de entender?		Si ___	No ___		
¿Qué aspectos de le resultaron más atractivos?					
¿Encuentra alguna mecánica de juego confusa o poco clara?					
Dificultad					
¿Considera que el juego presenta un equilibrio entre desafío y diversión?		Si ___	No ___		

¿Qué ajuste sugeriría en la dificultad del juego?		
Errores y fallos		
¿Encontró algún error o fallo en las sesiones de juego?	Si ___	No ___
Descríbalos		
¿Los errores afectaron negativamente su experiencia de juego?	Si ___	No ___
Retroalimentación general		
¿Qué característica añadiría al juego?		
¿Qué aspectos considera sobresalientes y que no deberían ser cambiados?		
Comentarios adicionales		

Estas pruebas serán aplicadas durante 6 meses, tiempo promedio de pruebas beta para Android según (Google, 2023).

Tras finalizar esta etapa de pruebas beta, es necesario recopilar y analizar los datos y comentarios recopilados, utilizando esta información para identificar patrones y problemas recurrentes, así como áreas del juego a mejorar.

Durante el período de aplicación de estas pruebas se ha detectado una no conformidad correspondiente a la relación visual del componente *runner* respecto al resto del videojuego.

3.4.3 Análisis de los resultados de las pruebas realizadas al componente *runner* Son de Máquina

Las pruebas efectuadas, y documentadas anteriormente corresponden al método de prueba de caja negra, que se lleva a cabo sobre la interfaz del software, para demostrar que las funciones del componente son operativas. En este método de prueba no se ve el código de la aplicación (Pressman & Maxim, 2020). Para desarrollar las pruebas de caja negra se escogió la técnica de análisis de causa-efecto la cual fue modelada mediante un

diagrama de Ishikawa. Dicha técnica arrojó como resultado del análisis que el componente en desarrollo poseía deficiencias funcionales, de las cuales su solución fue factible. Se escogió como tipos de pruebas las de Usabilidad y Funcionalidad.

De las pruebas de usabilidad, las cuales están enfocadas en factores humanos, estéticos, consistencia en la interfaz de usuario y la ayuda sensitiva al contexto, se puede definir que el componente desarrollado es eficiente en ese sentido.

Las pruebas de funcionalidad, que se centran en la validación de las funciones y métodos descritos en los mecanismos, fueron efectuadas en un entorno de desarrollo real con un prospecto de usuario final. De dichas pruebas se obtuvieron resultados satisfactorios.

Además, se efectuaron pruebas de regresión, que consiste en realizar las pruebas antes descritas al completar cada iteración de entrega para todos los casos de prueba diseñados. Al encontrar no conformidades, se debieron corregir las mismas y efectuar otra iteración de pruebas. En la última iteración, como se mostró en datos anteriores, no se encontraron no conformidades.

3.5 INTEGRACIÓN Y DESPLIEGUE DEL COMPONENTE *RUNNER* SON DE MÁQUINA

La integración y despliegue se refiere a todas las actividades que hacen que un producto de software esté disponible para su uso. Provee la vista de implementación del sistema. Los diagramas de despliegue muestran la configuración en funcionamiento del sistema incluyendo su software y su hardware, describen la topología del sistema la estructura de los elementos de hardware y el software que ejecuta cada uno de ellos y representan a los nodos y sus relaciones (Pressman & Maxim, 2020). A continuación, se presenta el diagrama que muestra la integración del componente *runner* Son de máquina con el resto del videojuego Isla del Son.

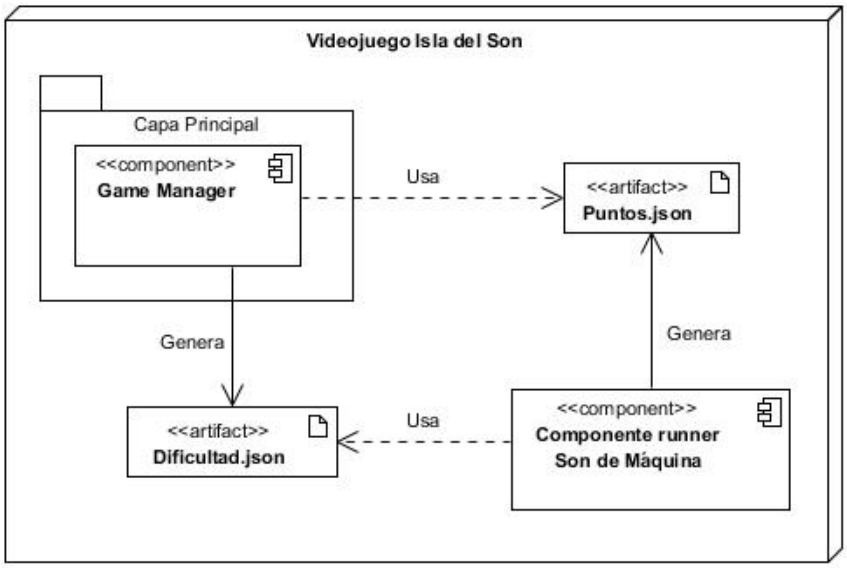


Figura 13 Diagrama de integración del componente Son de Máquina con el videojuego Isla del Son (Elaboración propia)

3.6 VALIDACIÓN DE LA INVESTIGACIÓN E ÍNDICE DE SATISFACCIÓN GRUPAL

Como parte del proceso de evaluación de la satisfacción del usuario y con el objetivo de evaluar el componente de videojuego implementado se utiliza la técnica de Iadov. Para ello se emplearon los postulados teóricos de (Campistrous y Rizo, 2006) que expresan que la técnica de criterio de usuarios, aplicando técnica de IADOV, debe usarse como vía para valorar resultados en aquellos casos en que los evaluadores son usuarios de lo que se propone, es decir, que además de tener dominio del problema en estudio, están inmersos en el contexto en el que se aplica el resultado.

La técnica de IADOV se compone de cinco preguntas claves: tres cerradas y dos abiertas, es utilizada para determinar el nivel de satisfacción individual y grupal de los usuarios a partir de una encuesta elaborada según las exigencias pertinentes. La aplicación de esta técnica constituye una vía indirecta para el estudio de satisfacción, ya que los criterios que se utilizan se fundamentan en las relaciones que se establecen entre las tres preguntas cerradas (Pedrera Suen et al., 2023).

Las preguntas 2,3 y 4 de la encuesta se relacionan a través de lo que se denomina el Cuadro Lógico de IADOV que se muestra en la Tabla 2. Cada encuestado recibe una evaluación individual en dependencia de las respuestas que dé a las preguntas cerradas. Para facilitar el procesamiento posterior, en el diseño de la encuesta se debe tener en cuenta que a estas preguntas deben estar respondidas de la forma prevista en el cuadro lógico de Iadov. La encuesta fue aplicada a 9 personas, de ellas 4 son ingenieros de la

Universidad de Ciencias Informáticas, del cual 2 de ellos pertenecen al centro VERTEX, y 5 estudiantes de dicha universidad. El cuadro lógico se presenta a continuación.

Tabla 10 Matriz de ladov correspondiente a la validación del componente *runner* Son de Máquina (Modificado por el Autor)

4. Luego de haber visualizado los resultados del componente *runner*, refleje en qué medida le satisface la solución diseñada.	2. ¿Considera que sea apropiado mantener al videojuego Isla del Son sin elementos que ofrezcan interactividad e identidad y cubanía entre los niveles del mismo?								
	No			**No sé**			**Sí**		
	3. ¿Considera factible que un componente *runner* sea empleado para dotar de interactividad, identidad y cubanía al videojuego Isla del Son?								
	Sí	**No sé**	**No**	**Sí**	**No sé**	**No**	**Sí**	**No sé**	**No**
Me gusta mucho	1	2	6	2	2	6	6	6	6
Me gusta más de lo que me disgusta	2	2	3	2	3	3	6	3	6
Me da lo mismo	3	3	3	3	3	3	3	3	3
Me disgusta más de lo que me gusta	6	3	6	3	4	4	3	3	4
No me gusta nada	6	6	6	6	4	4	6	4	5
No sé qué decir	2	3	6	3	3	3	6	3	4

Para obtener el índice de satisfacción grupal (ISG) se trabaja con los diferentes niveles de satisfacción que se expresan en la escala numérica que oscila entre +1 y – 1. El número resultante de la interrelación de las tres preguntas que indica la posición de cada encuestado en la siguiente escala de satisfacción (Fernández de Castro Fabre & López Padrón, 2014):

1. Clara satisfacción +1

2. Más satisfecho que insatisfecho 0.5

3. No definido y contradictorio 0

4. Más insatisfecho que satisfecho -0.5

5. Clara insatisfacción -1

El índice de satisfacción grupal (ISG) se expresa en una escala numérica que va desde 1 (máxima satisfacción), hasta -1 (máxima insatisfacción). El ISG se calcula mediante la siguiente fórmula:

$$\mathrm{ISG} = \frac{\mathrm{A}(+1) + \mathrm{B}(+0.5) + \mathrm{C}(0) + \mathrm{D}(-0.5) + \mathrm{E}(-1)}{\mathrm{N}}$$

En esta fórmula A, B, C, D, E, representan la cantidad de encuestados colocados respectivamente en las posiciones de satisfacción y donde N representa la cantidad total de encuestados (Fernández de Castro Fabre & López Padrón, 2014). Los resultados obtenidos de la aplicación de la encuesta se presentan en la tabla siguiente.

Tabla 11 Resultados numéricos de la aplicación de la técnica de Iadov (Elaboración propia)

Categorías grupales de satisfacción	N = 9	Escala
Clara satisfacción	6	A
Más satisfecho que insatisfecho	3	B
No definido	0	C
Más insatisfecho que satisfecho	0	D
Clara insatisfacción	0	E
Contradictorio	0	-

De esta forma se obtiene el cálculo del ISG

$$\mathrm{ISG} = \frac{\mathrm{A}(+1) + \mathrm{B}(+0.5)}{\mathrm{N}}$$

$$\mathrm{ISG} = \frac{6(+1) + 3(+0.5)}{9} = 0.8333$$

El proceso de evaluación del objetivo de la investigación mediante la técnica de IADOV confirmó su factibilidad del componente, expresando cuantitativamente en el alto ISG (0.83) y cualitativamente en los criterios emitidos entre los usuarios a quienes se les aplicó la técnica, lo que refleja la aceptación de la propuesta, y el reconocimiento a su utilidad.

3.7 CONCLUSIONES DEL CAPÍTULO

Tras haber realizado la implementación y pruebas del componente *runner* Son de Máquina para el videojuego Isla del Son puede concluirse que las pruebas de aceptación basadas en Pruebas Alfa y Pruebas Beta son un recurso necesario y suficiente dentro de la etapa 4 del Marco de trabajo ingenieril para el desarrollo de videojuegos, si se está desarrollando un componente que constituye un minijuego dentro de un juego mayor, con las características presentes. Estas pruebas ofrecen como artefacto resultante una lista de no conformidades la cual permite el análisis detallado de dichas no conformidades para posteriormente ser resueltas mediante medidas correctivas. En el caso presente, la aplicación de dichas pruebas separadas en iteraciones, permite que la implementación del componente *runner* Son de Máquina sea efectiva, dejando claro que partes del mismo están correctamente implementadas y qué partes precisan de más tiempo de implementación. Además, el índice de satisfacción grupal, calculado mediante la técnica de Iadov permitió validar la propuesta empleando para ello no el criterio de expertos, sino el criterio de usuarios, tomando como población objetivo a posibles usuarios finales del videojuego Isla del Son.

CONCLUSIONES GENERALES

Tras el desarrollo de la presente investigación se obtuvo un componente *runner* que permite el enlace entre el resto de minijuegos del videojuego Isla del Son, y que su diseño aporta identidad y elementos visuales de cubanía al mencionado videojuego. Acerca de esto puede decirse que:

- A partir de la fundamentación teórica de este componente, guiándose por el Marco de trabajo ingenieril para el proceso de desarrollo de videojuegos, se pueden definir que un minijuego *runner* es una solución apropiada para interrelacionar partes de un videojuego que abstractamente pueden ser vistas como locaciones.
- El componente desarrollado cuenta con una estructura basada en capas y con componentes internos, correspondientes a archivos de *scripts,* lo cual permite, a partir de cada nivel, transitar entre los niveles propios del resto de minijuegos.
- El patrón de diseño *Pooling* de Objetos puede definirse como el patrón principal a la hora de desarrollar videojuegos de género *runner*, pues este permite la creación aleatoria del mapa, en este caso la calle, de manera que contribuye a la sensación de avance del personaje, en este caso el carro, mediante el posicionamiento aleatorio de un *pool* de objetos delante de otro. Por lo que se puede afirmar que es indispensable en el desarrollo de un componente con las características propias de la presente investigación.
- Tras realizar las pruebas y validación correspondientes la etapa 4 del Marco de trabajo ingenieril para el proceso de desarrollo de videojuegos, se logró obtener un componente con alto grado de completitud dentro de la preproducción de un videojuego. Además, gracias a la correcta aplicación de las mismas, es posible conocer las deficiencias actuales, y como pueden ser resueltas o completadas, de acuerdo a los mecanismos definidos en el análisis.

RECOMENDACIONES

Para dar continuidad a la presente investigación se recomienda:

- Continuar con las Pruebas Beta prestando especial atención a la visión de los posibles usuarios finales.
- Completar el apartado visual del componente, haciendo uso de modelos 3D profesionales.
- Tomar la presente como base para iniciar una investigación sobre la estandarización del desarrollo de videojuegos o minijuegos *runner* con las características específicas que pueden ser consultadas en este documento.

RECURSOS BIBLIOGRÁFICOS

Acosta, L. (2004). El desplazamiento cultural en la música cubana. *Latin American Music Review*, *25*(2), 154-166.

Andy Hernández, Karina Pérez, & Omar Correa. (2017). Marco de trabajo ingenieril para el proceso de desarrollo de videojuegos. *Revista Antioqueña de las Ciencias Computacionales y la Ingeniería de Software*, *7*(1), 13-26.

Arcila Ibague, D., Pineda, Z., Bacca-Acosta, J., & Avila-Garzon, C. (2022). *Diseño y desarrollo de un videojuego como herramienta didáctica para aprender o recordar hechos históricos* (pp. 628-642).

ASALE, R.-, & RAE. (s. f.). *Videojuego | Diccionario de la lengua española*. «Diccionario de la lengua española» - Edición del Tricentenario. Recuperado 27 de junio de 2023, de https://dle.rae.es/videojuego

Caillois, R. (1986). *Los juegos y los hombres. Fondo Cultura Económica*. https://dialnet.unirioja.es/descarga/articulo/4895274.pdf

Campistrous, L., & Rizo, C. (2006). *El Criterio de Expertos como Método en la Investigación Educativa* [Doctoral]. Instituto Superior de Cultura Física "Manuel Fajardo".

Cook, M. (2016, agosto). Alien Langueges: How we talk about procedural generation. *Game Developer*. https://www.gamedeveloper.com/design/alien-langueges-how-we-talk-about-procedural-generation

Crnkovic, I., Stafford, J., & Szyperski, C. (2011). Software Components beyond Programming: From Routines to Services. *IEEE Software*, *28*, 22-26. https://doi.org/10.1109/MS.2011.62

Curiel Gálvez, A. (2015). *El Videojuego como Medio Narrativo* [Maestría, Universidad de Sevilla]. https://www.academia.edu/28812924/El_Videojuego_como_Medio_Narrativo

Dille, F., & Platten, J. Z. (2015). *The Ultimate Guide to Video Games Writing and Design*.

Dounas, T., Theodoros, Sigalas, & Alexandros. (2023). Blender, an Open Source Design Tool: Advances and Integration in the Architectural Production Pipeline. *Computation: The New Realm of Architectural Design [27th eCAADe Conference Proceedings / ISBN 978-0-9541183-8-9] Istanbul (Turkey) 16-19 September 2009, pp. 737-744*.

Fernández, A. (2013). *Estándares de codificación en C# y buenas prácticas de programación: Resumen de arquitecturas, metodologías y buenas prácticas*. Canal de Negocios.

Fernández de Castro Fabre, A., & López Padrón, A. (2014). Validación mediante criterio de usuarios del sistema de indicadores para prever, diseñar y medir el impacto en los proyectos de investigación del sector agropecuario. *Revista Ciencias Técnicas Agropecuarias*, *23*(3), 77-82.

Giro, R. (2010). The Roots of Cuban Music. *NPR*.

González Seco, J. A. (2000). INTRODUCCIÓN A C#. En *El lenguaje de programación C#* (pp. 21-34). https://dis.um.es/~bmoros/privado/bibliografia/LibroCsharp.pdf

González-Pérez, C., & García-Ruiz, R. (2019). The use of video games for the promotion of intangible cultural heritage. *Sustainability*, *11*(1), 1-17.

Google. (2023). *Información sobre el programa beta*. google.com

Guión Visual Paradigm for UML. (2013). Visual-Paradigm. http://www.visual-paradigm.com/

Hernández Orallo, E. (2015). *El Lenguaje Unificado de Modelado (UML)*.

Ibarra Monteagudo, Y. (2018). *Proceso de Pruebas en el Desarrollo de Videojuegos* [Maestría, Universidad de las Ciencias informáticas]. https://repositorio.uci.cu/bitstream/123456789/7929/1/Tesis%20de%20Maestr%C3%ADa%20Yeili%20Ibarra%20Monteagudo1.8.pdf

Jurado Monroy, F., Albusac Jiménez, J. A., Castro Sánchez, J. J., Vallejo Fernández, D., Jiménez Linares, L., Villanueva Molina, F. J., Villa Alises, D., Gonzalez Morcillo, C., & Simmross Wattenberg, G. (2012). *Desarrollo de Videojuegos 4 Desarrollo de Componentes.pdf* (Bubok).

Larman, C. (2003). *UML y Patrones: Introducción al análisis y diseño orientado a objetos* (H. Cárdenas, Ed.). Pearson.

Lechner, E. (2013). *Música de Cuba para el mundo, salsa, mambo y más*. AARP. https://www.aarp.org/espanol/entretenimiento/expertos/ernesto-lechner/info-08-2013/foto-musica-cuba.html

López Raventós, C. (2016). El videojuego como herramienta educativa. Posibilidades y problemáticas acerca de los serious games. *Apertura (Guadalajara, Jal.)*, *8*(1), 0-0.

Noboa, G. (2017). Geometry Dash World: A short but esquisite musical adventure (review). *AndroidGuys*. androidguys.com

Pedrera Suen, M., Peña Castellanos, D., & Sánchez Domínguez, Y. A. (2023). *Mecanismos de prevención de intrusos para Nova Servidores*. X Taller de Software Libre y tecnologías Emergentes, Varadero, Cuba.

Polansky, L. (2013, julio 1). The Leaderboard: The Loneliness of the Endless *Runner*. *Paste*. https://www.pastemagazine.com/games/the-leaderboard-the-loneliness-of-the-endless-runn

Pressman, R. S., & Maxim, B. R. (2020). *Software engineering: A practitioner's approach* (Ninth edition). McGraw-Hill Education.

Ramón, M. del C. (2016, octubre 7). *¿Existe una industria de videojuegos en Cuba?* | *Cubadebate*. Cubadebate. http://www.cubadebate.cu/noticias/2016/10/07/existe-una-industria-de-videojuegos-en-cuba/#.WI6jk2WJjIU

Rodríguez, J. (2023, febrero). Qué es el Diagrama de Ishikawa, para qué sirve, cómo crearlo y ejemplos. *Ventas*. https://blog.hubspot.es/sales/diagrama-ishikawa

Rodríguez Ruidíaz, A. (2019). *Los géneros de la música popular cubana. Su origen y evolución*. https://www.academia.edu/48929241/Los_g%C3%A9neros_de_la_m%C3%BAsica_cubana_Su_origen_y_evoluci%C3%B3n

Roy, M. (2002). *Cuban music: From Son and rumba to The Buena Vista Social Club and timba cubana*. Princeton : Markus Wiener Publishers. http://archive.org/details/cubanmusicfromso00roym

Ruelas, U. (2017, julio 13). *¿Qué es un motor de videojuegos (game engine)?* https://codingornot.com/que-es-un-motor-de-videojuegos-game-engine

Rumbauch, J. I., & Grady, B. (2015). *El Lenguaje Unificado de Modelado. Manual de Referencia*.

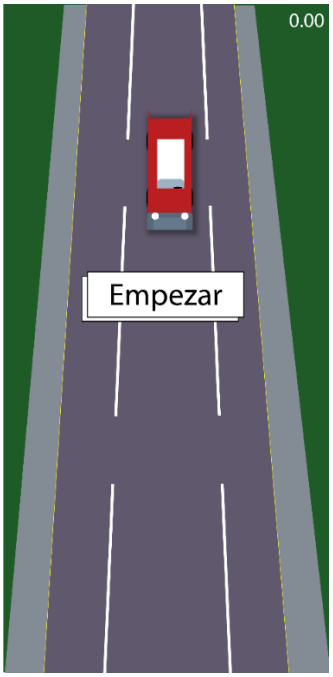
Ryan Whitwam. (2014). Not so fast for Android turns the Endless *runner* on its head. *PC Magazine*.

Unity Learn. (2022, mayo). *Introduction to Object Pooling*. https://learn.unity.com/

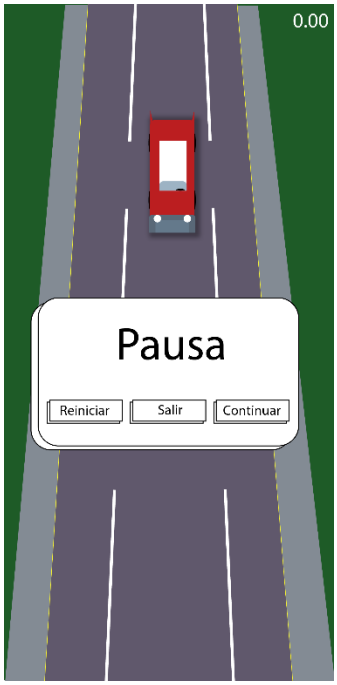
Unity Technologies. (2019). *Unity User Manual (2019.4 LTS)—Unity Manual*. https://docs.unity3d.com/es/2019.4/Manual/UnityManual.html

ANEXOS

Anexo 1 Diseño de interfaz para el componente Son de Máquina (Elaboración propia).

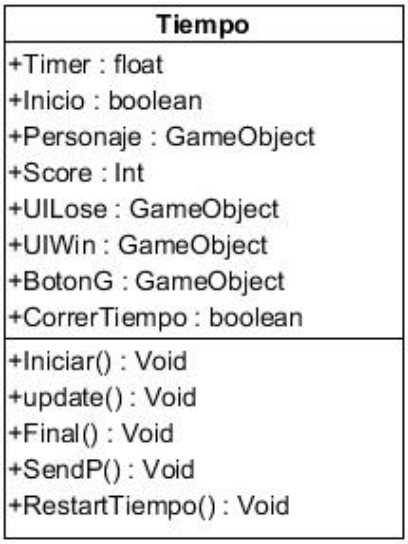


Anexo 2 Diseño de interfaz para el componente Son de Máquina (Elaboración propia).

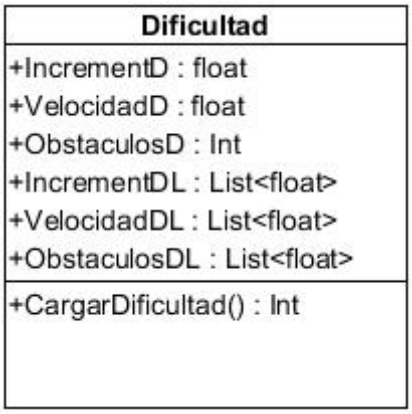


Anexo 3 Diseño de interfaz para el componente Son de Máquina (Elaboración propia).

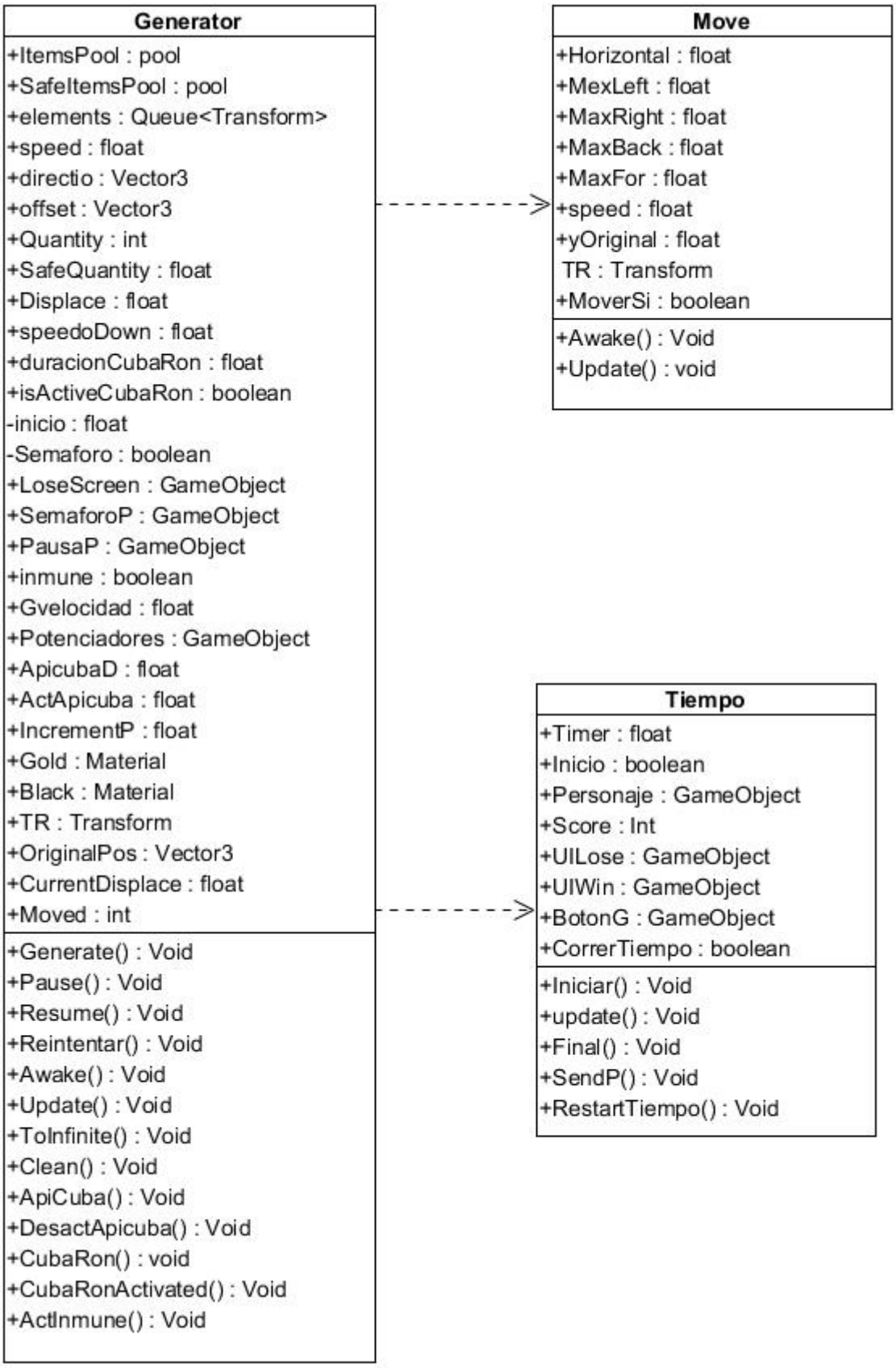
Anexo 4 Diseño de interfaz para el componente Son de Máquina (Elaboración propia).



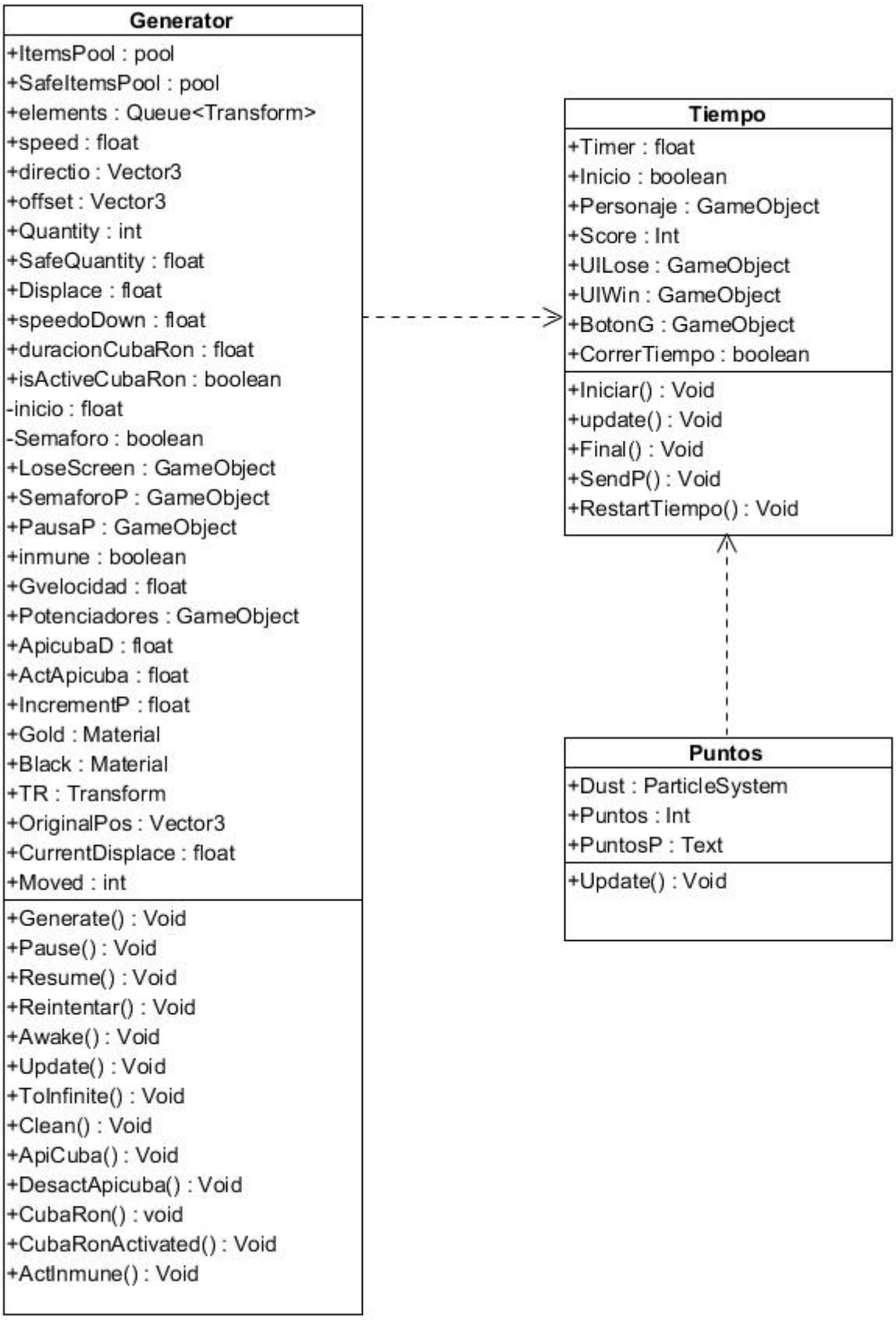
Anexo 5 Diagrama de clases del Mecanismo Puntuaciones (Elaboración propia)



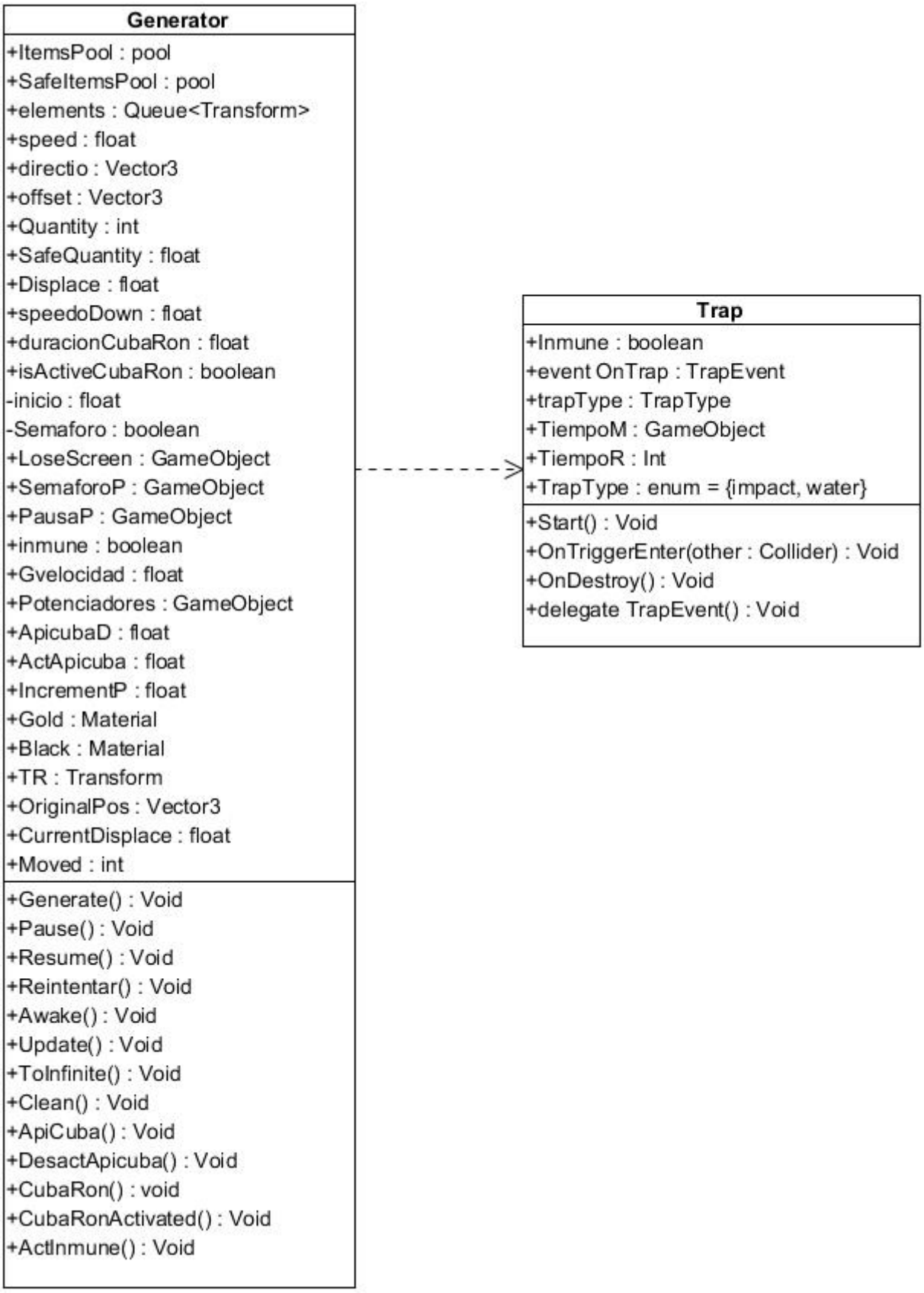
Anexo 6 Diagrama de clases del Mecanismo Dificultad (Elaboración propia)



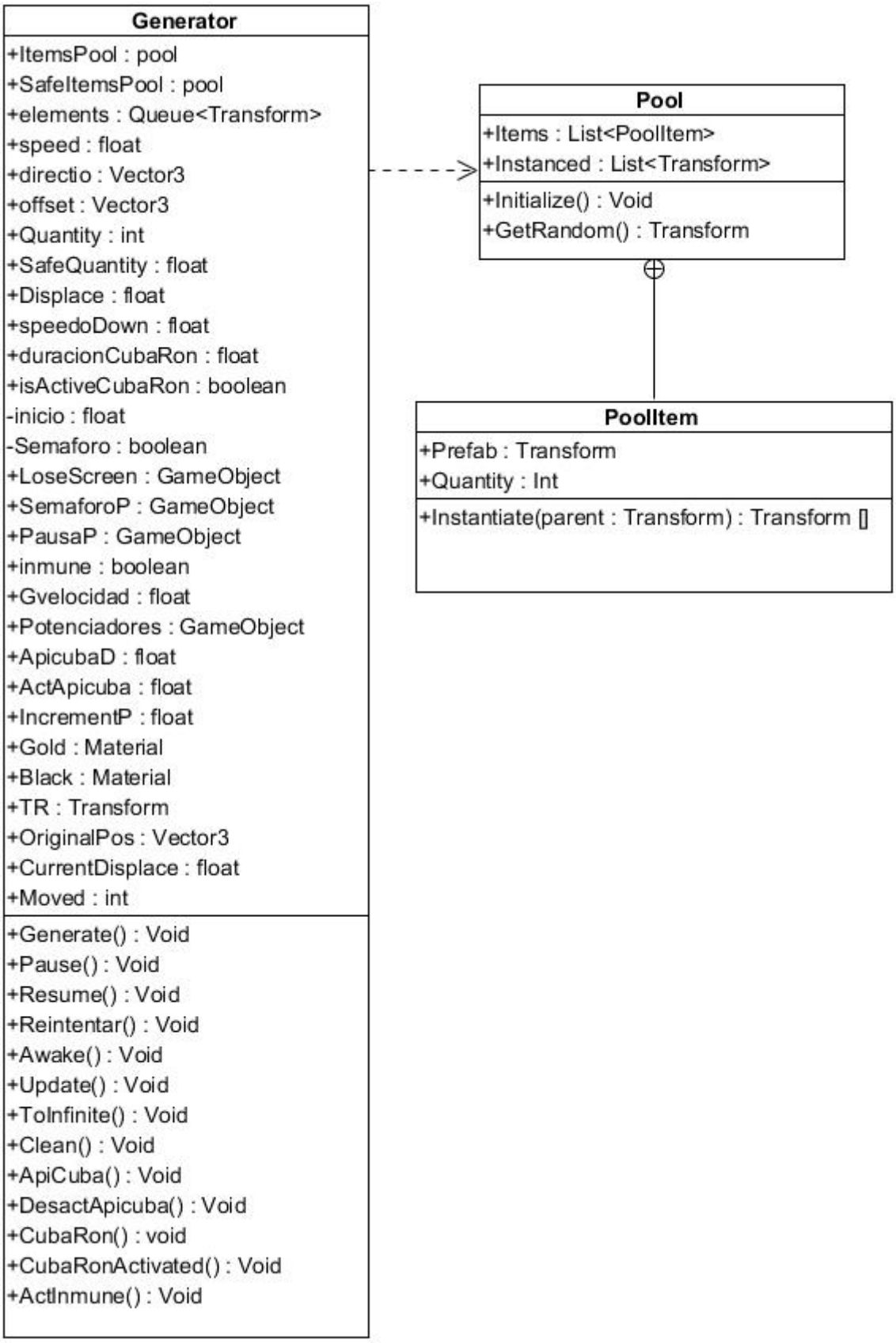
Anexo 7 Diagrama de clases del Mecanismo Pausa (Elaboración propia)



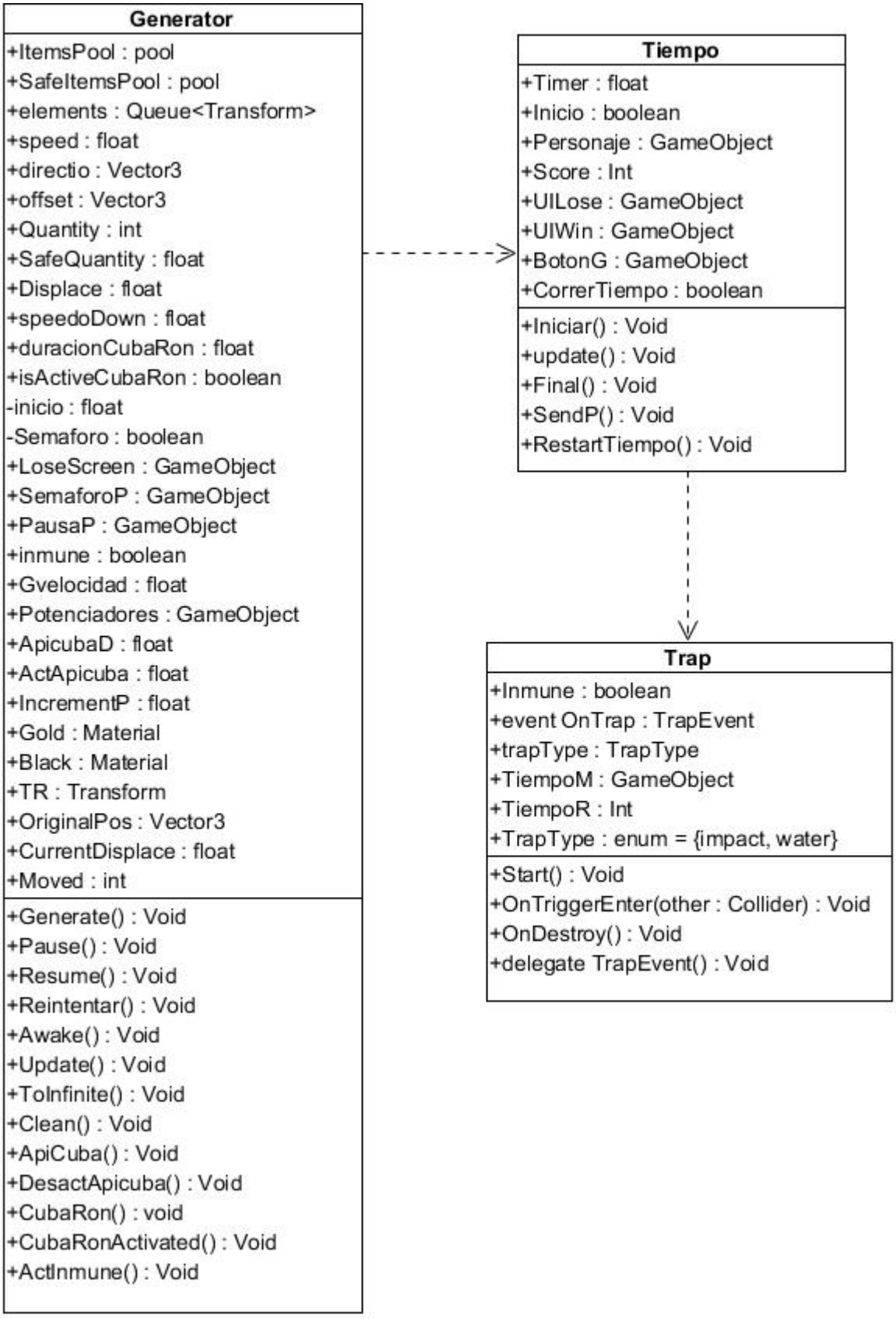
Anexo 8 Diagrama de clases del Mecanismo Pantalla de juego perdido (Elaboración propia)



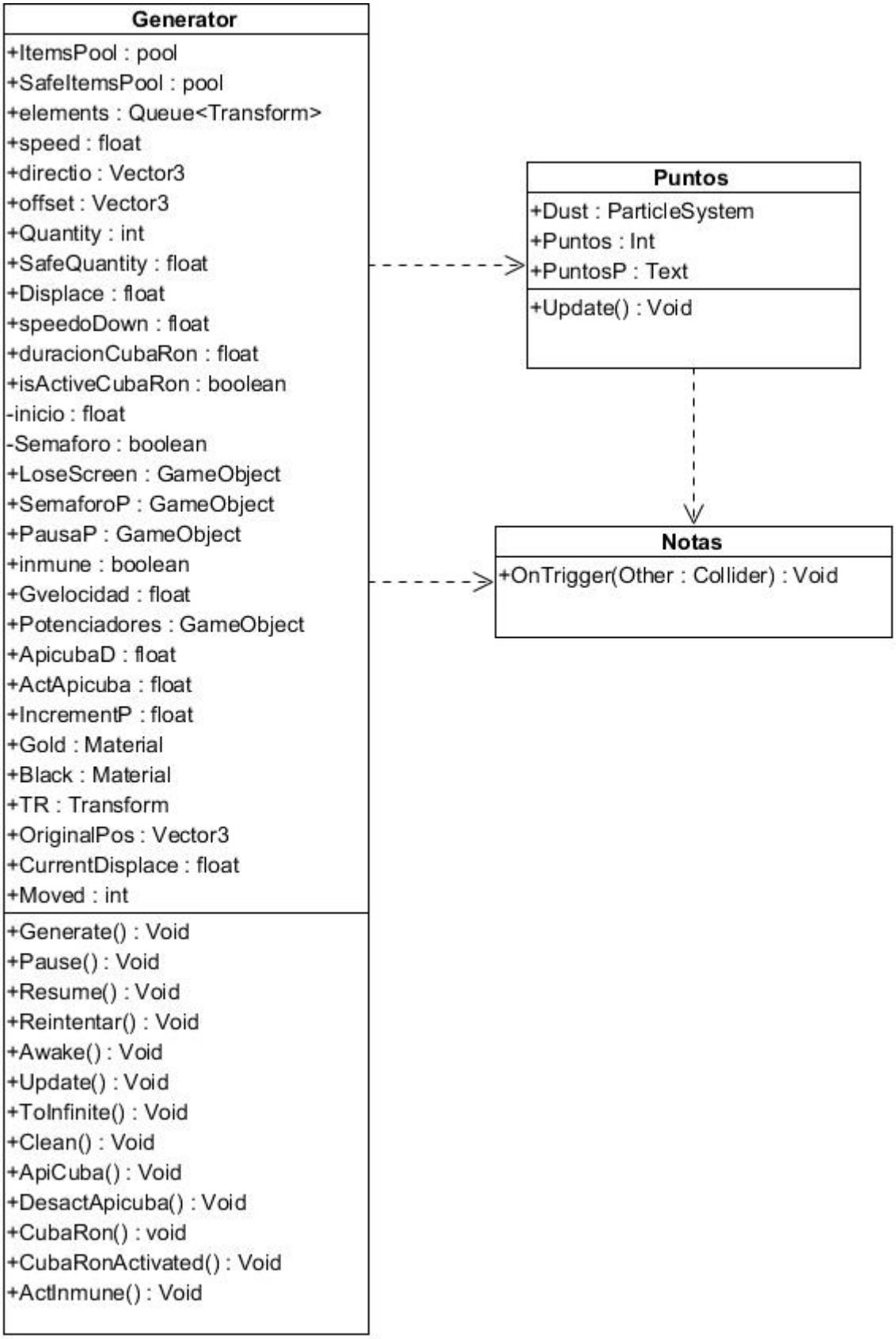
Anexo 9 Diagrama de clases del Mecanismo Obstáculos (Elaboración propia)



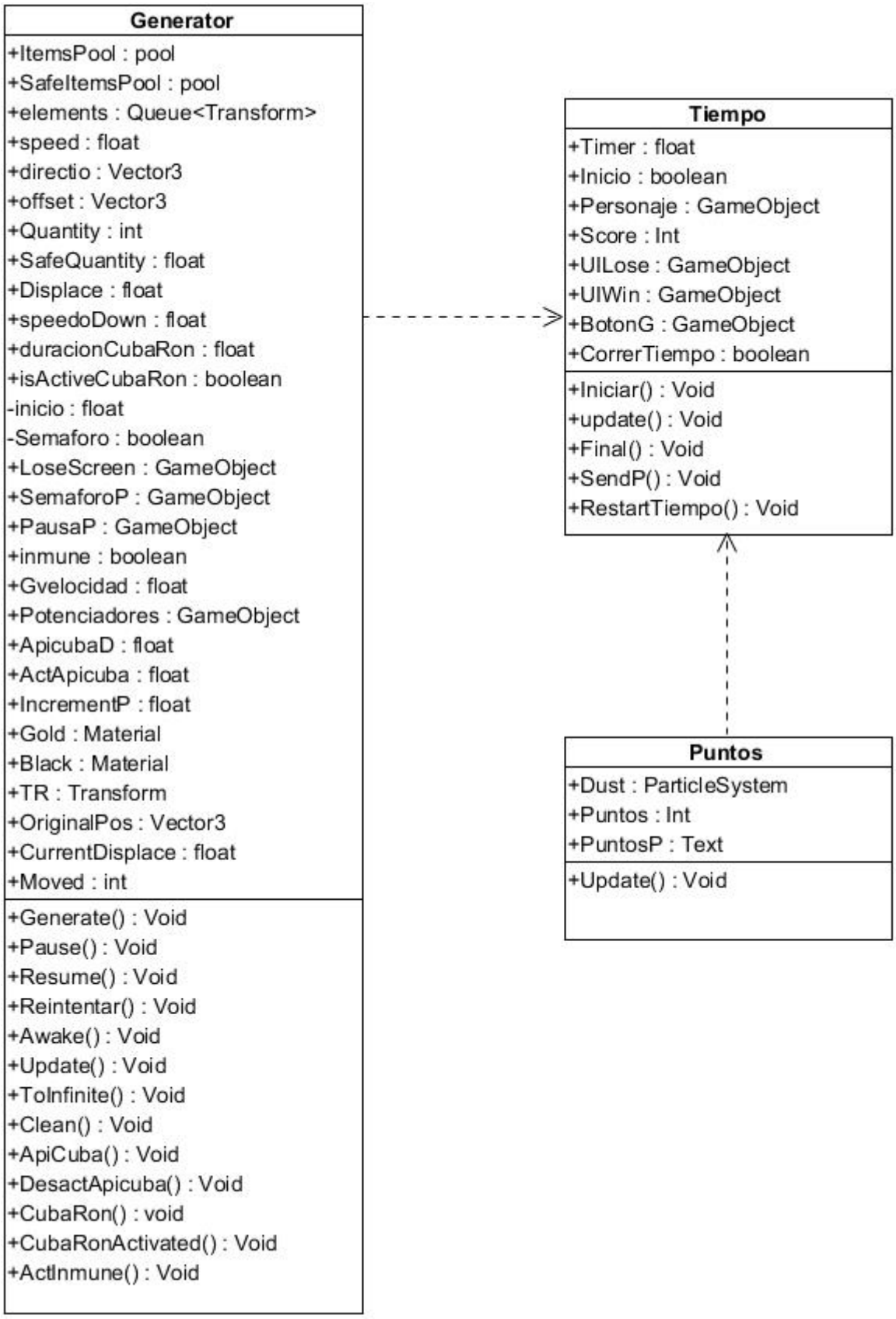
Anexo 10 Diagrama de clases del Mecanismo Generar Mapa (Elaboración propia)



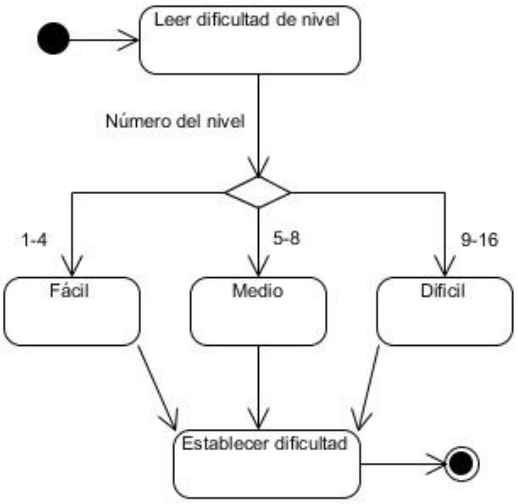
Anexo 11 Diagrama de clases del Mecanismo Cronómetro (Elaboración propia)



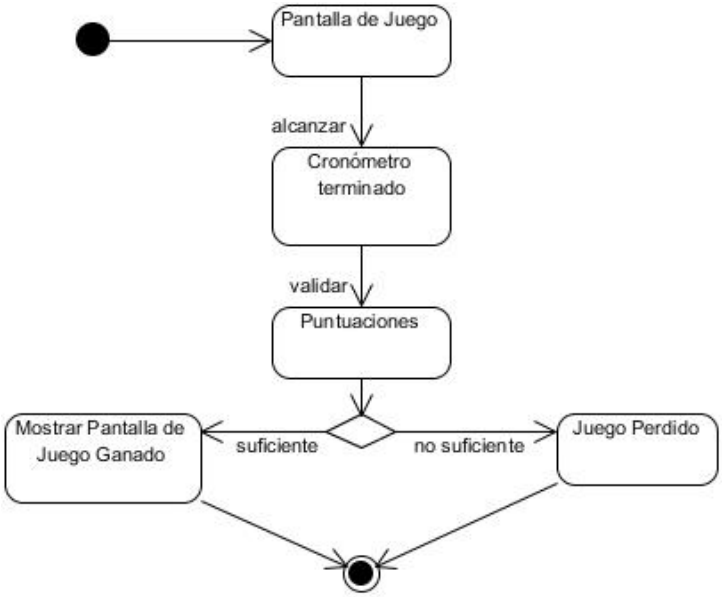
Anexo 12 Diagrama de clases del Mecanismo Claves de Son (Elaboración propia)



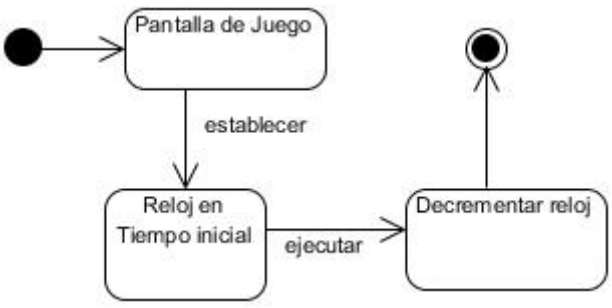
Anexo 13 Diagrama de clases del Mecanismo Juego Ganado (Elaboración propia)



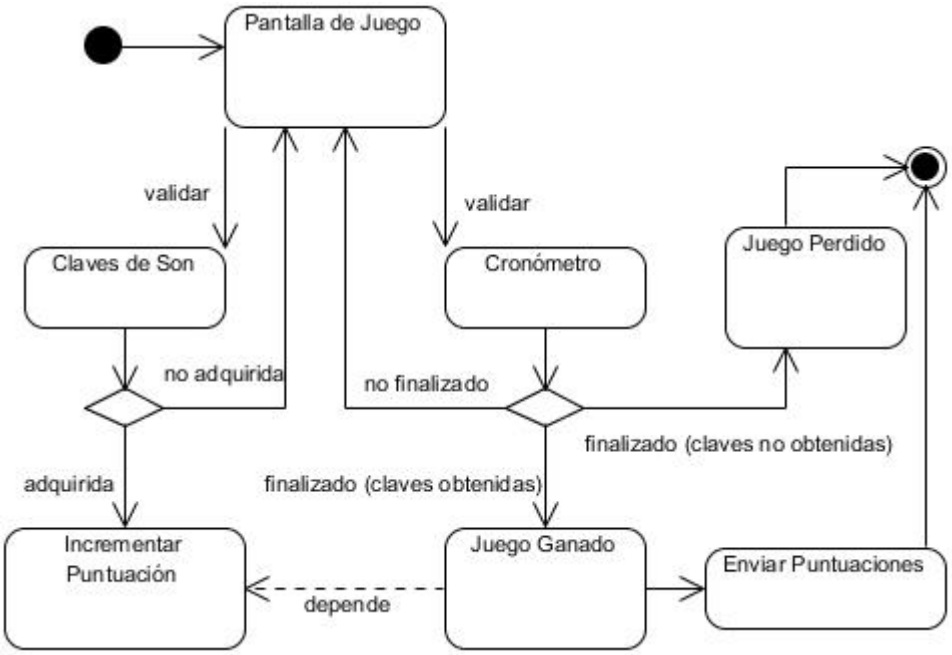
Anexo 14 Diagrama de estado del Mecanismo Dificultad (Elaboración propia)



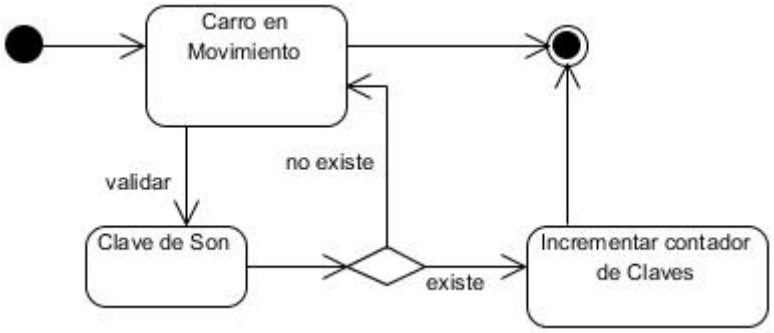
Anexo 15 Diagrama de estado del Mecanismo Juego Ganado (Elaboración propia)



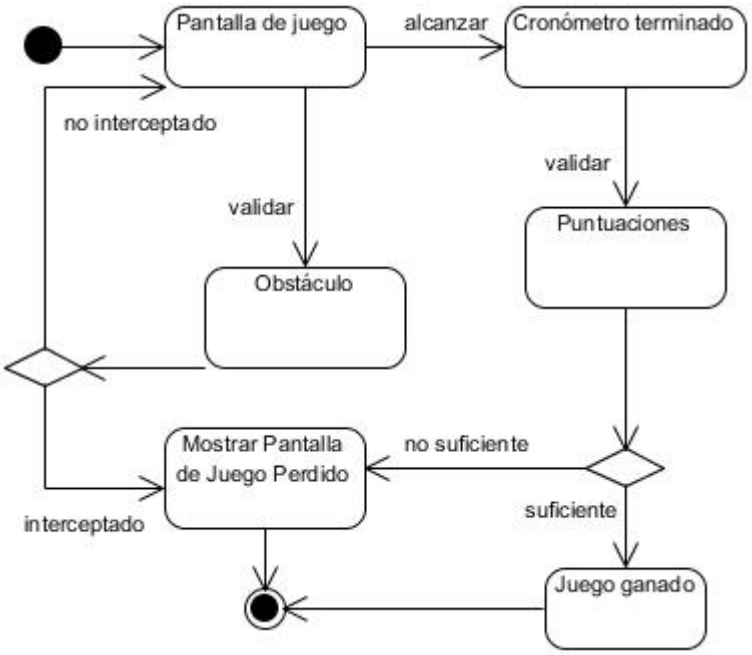
Anexo 16 Diagrama de estado del Mecanismo Cronómetro (Elaboración propia)



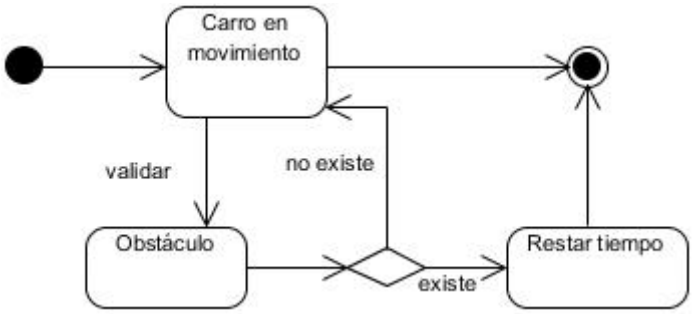
Anexo 17 Diagrama de estado del Mecanismo Puntuaciones (Elaboración propia)



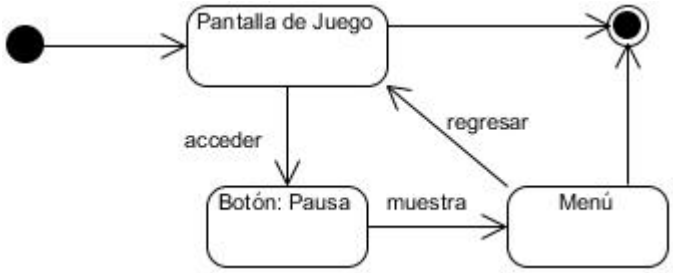
Anexo 18 Diagrama de estado del Mecanismo Claves de Son (Elaboración propia)



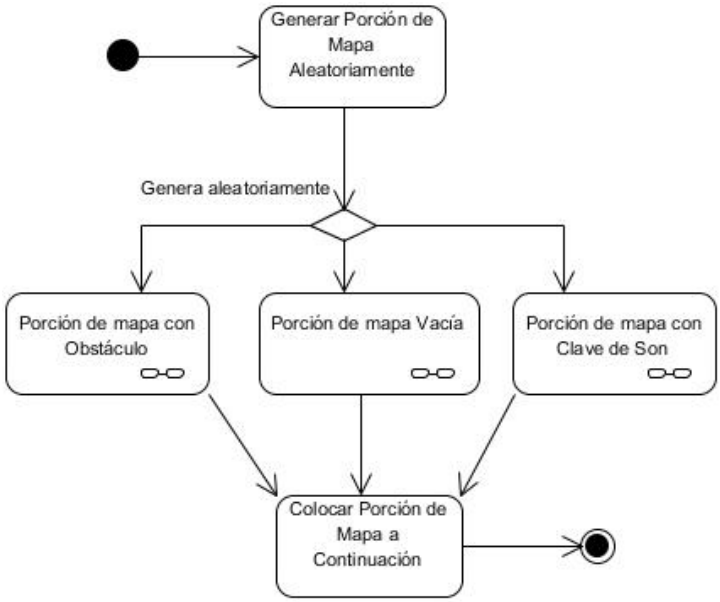
Anexo 19 Diagrama de estado del Mecanismo Juego Perdido (Elaboración propia)



Anexo 20 Diagrama de estado del Mecanismo Obstáculos (Elaboración propia)



Anexo 21 Diagrama de estado del Mecanismo Pausar (Elaboración propia)



Anexo 22 Diagrama de estado del Mecanismo Generar Mapa (Elaboración propia)