UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Desarrollo de los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas” para la colección de juegos MundoClick

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

Autor:
Yailyn Abreu Barrios

Tutores:
Msc. Licet Gutiérrez Mompié
Ing. Yolanda Mauri Pérez

Junio – 2013
La Habana, Cuba

Declaración de autoría

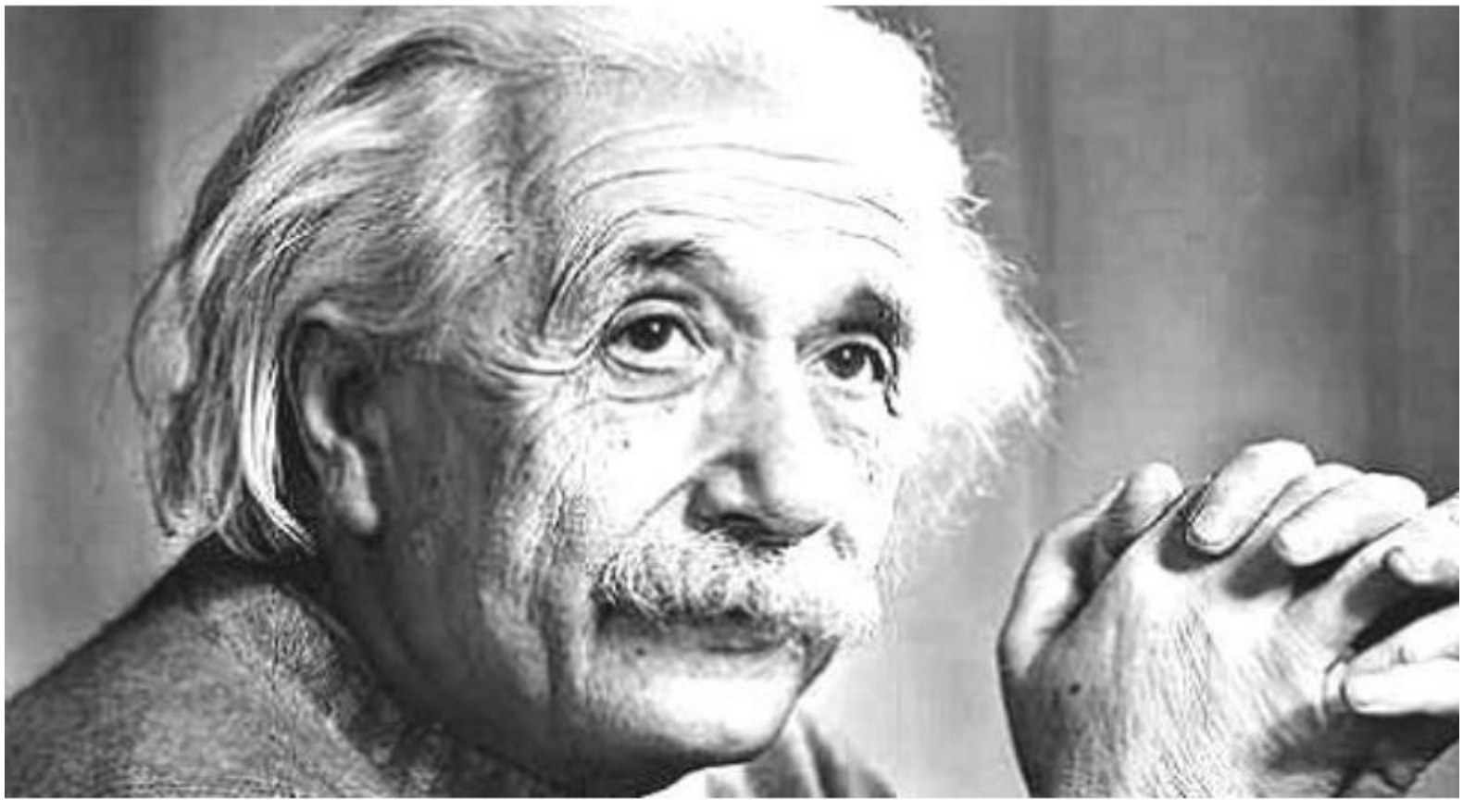
Declaramos ser autores del presente trabajo de diploma y reconocemos al Centro FORTES de la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2013.

_____
Yailyn Abreu Barrios
Firma Autor

_____
Msc. Licet Gutiérrez Mompié
Firma Tutor

_____
Ing. Yolanda Mauri Pérez
Firma Tutor



"Hay una fuerza motriz más poderosa que el vapor,
la electricidad y la energía atómica:
la voluntad."

Albert Einstein.

A mis padres por la educación y el amor que me han brindado todos estos años, a mi hermana por ser la luz de mis ojos y permitirme ser un ejemplo en su vida. A ustedes les dedico este trabajo.

Agradecimientos

A mi tutora Yolanda por su profesionalidad, paciencia, esfuerzo incondicional, apoyo e inmensa dedicación en todos estos meses.

A Wilfredo por toda la preocupación y ayuda. Te agradezco por estar siempre dispuesto cuando necesité de tus conocimientos.

A mis padres por permitirme ser quien soy y hacerme sentir muy orgullosa cada vez que hablo de ellos. A mi madre, por llevarme en su corazón, por cuidarme y estar junto a mí brindándome todo su amor. A mi padre, por ser mi guía y apoyarme en cada momento, por darme tantas lecciones y enseñarme a ser una mejor persona. Gracias a los dos por estar siempre a mi lado y sobre todas las cosas por depositar toda su confianza en mí. A ustedes les debo la vida.

A mi hermanita por ser la luz de mis ojos, por llenar mi vida de felicidad con cada beso, por ser esa personita que siempre está ahí demostrándome su inmenso amor y regalándome todas sus alegrías.

A todos mis amigos y compañeros de grupo con los que he compartido mi vida en esta universidad, siempre fue bueno poder contar con ustedes.

A todos en general que se preocuparon y ayudaron de una forma u otra a la realización de este trabajo.

Resumen

El presente documento aborda los contenidos relacionados al proceso investigativo-productivo referente al desarrollo de los juegos educativos “Recicla”, “Limpiar la pecera” y “Mis maripositas”. Actualmente surge la necesidad en el Centro de Tecnologías para la Formación (FORTES) de la Universidad de las Ciencias Informáticas de proveer juegos educativos a una nueva colección de juegos educativos que se encuentra en desarrollo denominada MundoClick. Para cumplir con la tarea planteada anteriormente es preciso realizar un análisis a otras soluciones existentes, documentar el proceso de desarrollo de los juegos educativos “Recicla”, “Limpiar la pecera” y “Mis maripositas”, implementar e integrar dichos juegos y validarlos mediante la realización de pruebas de *software*. La investigación arrojó como principales resultados la documentación y artefactos de ingeniería de *software* generados durante el desarrollo de las aplicaciones, guiado por la metodología tradicional RUP, así como tres aplicaciones didácticas educativas integradas a la colección MundoClick que cumplen con las especificaciones descritas en los respectivos guiones.

Palabras clave: MundoClick, juego educativo, desarrollo.

Índice

Introducción

El crecimiento indetenible del acceso a las Tecnologías de la Información y las Comunicaciones (TIC) ha beneficiado prácticamente a todos los campos de la sociedad, tanto así que no se concibe el mundo en estos momentos sin ellas. Su uso eficiente permite acceder a mayor cantidad de información de manera rápida y ahorrando recursos, posibilitando perfeccionar la calidad de la educación y generando mejores oportunidades para impulsar el desarrollo de las capacidades humanas.

La aplicación de los adelantos tecnológicos a la educación a nivel mundial ha sido una de las líneas en la que las personas involucradas con procesos de enseñanza-aprendizaje han trabajado en los últimos tiempos. El uso del *software* en los procesos educativos se ha ido convirtiendo en una prioridad de muchos sistemas educacionales del mundo. Junto a estas nuevas concepciones de la educación han surgido los *software* educativos, denominando así a los programas para computadoras creados con la finalidad específica de ser utilizados como medio didáctico, es decir, para facilitar los procesos de enseñanza y de aprendizaje. (1)

El *software* educativo ha propiciado el impulso de la calidad del proceso de enseñanza-aprendizaje, permitiendo la adaptabilidad y atención diferenciada al alumno. Posee además un alto grado de interactividad dado por el empleo de recursos multimedia como sonidos, fotografías y videos. La utilización de estos recursos desarrollan la imaginación, la creatividad, la memoria y facilitan la transmisión de conocimientos de una forma más amena, integradora, diferenciada y activa que el resto de los medios de enseñanza, constituyendo un efectivo instrumento para el desarrollo educacional del hombre. (2)

Dado el espectacular crecimiento del sector de la creación de juegos digitales, se cree firmemente que pueden desempeñar un papel esencial al renovar la percepción del aprendizaje por parte de los estudiantes de todos los sistemas de formación. (3) Los juegos digitales incluyen elementos de fantasía y diversión que permiten captar la atención de los niños, facilitando la construcción de conocimientos sólidos y permanentes a partir de lo aprendido en ellos.

La integración de Cuba a tan importante área del conocimiento ha sido sustentada por la creación de instituciones con el propósito de desarrollar *software* y brindar servicios informáticos, entre las que destaca la Universidad de las Ciencias Informáticas (UCI), creada en el año 2002. (4)

En la actualidad la UCI cuenta con varios centros que se dedican a la producción de *software*, destacándose en las esferas de informática médica, educación, software libre, teleformación, automatización industrial, computación gráfica, sistemas legales, procesamiento de señales digitales, geoinformática, realidad virtual, telecomunicaciones y seguridad informática. Uno de ellos es el Centro de Tecnologías para la Formación (FORTES) que se encarga de satisfacer las demandas en cuanto a *software* educativo se refiere, brindando servicios y productos informáticos que sirven de apoyo al proceso docente-educativo en los diferentes niveles de enseñanza.

El proyecto Multisaber-Navegante perteneciente al centro FORTES se dio a la tarea de crear una nueva colección de juegos educativos denominada MundoClick. Dicha colección agrupará una variedad de juegos didácticos que permitirán a los niños aprender de manera práctica, sencilla e interactiva el uso de los periféricos de la computadora, al mismo tiempo que desarrollan habilidades de índole cognitivas, sociales y conductuales como son: la actividad intelectual, la independencia, la creatividad, la imaginación y la iniciativa.

Con el objetivo de identificar los posibles juegos relacionados con el uso de los periféricos (específicamente el *mouse*) que pudieran formar parte de la colección de juegos educativos MundoClick, se llevó a cabo una investigación sobre este tema en el centro FORTES. La investigación arrojó los siguientes resultados:

- Los juegos se encuentran embebidos en colecciones de *software* educativo que han sido desarrolladas para Cuba y Venezuela.
- La arquitectura sobre la que fueron desarrollados no permite que se integren a MundoClick y en algunos casos tampoco permite la reutilización de código.

Los juegos poseen un funcionamiento diferente al concebido para la colección de juegos educativos MundoClick:

- Su complejidad no es gradual, impidiendo que los jugadores deban aplicar conocimientos más profundos a medida que avancen en ellos y por tanto, atentando contra la motivación y el aprendizaje progresivo.
- Los estados o niveles no son almacenados, por lo que siempre se debe comenzar desde el principio sin considerar los resultados obtenidos en el último acceso al juego.
- Fueron diseñados según las características de un determinado país, dificultando su comercialización al no ser genéricos.

A la hora de enseñar a un niño a interactuar con una computadora uno de los periféricos que requiere más atención es el *mouse* ya que permite controlar los elementos gráficos del sistema operativo. En este proceso el juego se convierte en una herramienta valiosa para que los educadores fomenten en sus estudiantes las habilidades necesarias para el uso correcto del *mouse*. En este sentido se puede decir que los juegos desarrollados en el centro no explotan al máximo las funcionalidades que brinda el periférico antes mencionado.

Producto de la situación existente se plantea el siguiente **problema científico**: ¿Cómo contribuir al desarrollo de las habilidades en el uso del *mouse* en niños de 6 a 8 años a través de juegos educativos?

Como **objetivo general** de la investigación se define: desarrollar los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas” para la colección de juegos educativos MundoClick que contribuyan al desarrollo de las habilidades en el uso del *mouse* en niños de 6 a 8 años.

En concordancia con lo anterior se plantean los siguientes **objetivos específicos**:

- Realizar una valoración crítica a otras soluciones existentes.
- Documentar el proceso de desarrollo de los juegos educativos “Recicla”, “Limpiar la pecera” y “Mis maripositas”.
- Implementar e integrar los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas”.
- Probar los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas”.

El **objeto de estudio** de la investigación es el aprendizaje basado en juegos, teniendo como **campo de acción** el aprendizaje basado en juegos en niños de 6 a 8 años.

En concordancia con los objetivos anteriores fueron identificadas las siguientes **tareas de investigación**:

- Análisis de soluciones similares para identificar las características que se le pueden incorporar a los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas” en MundoClick.
- Estudio valorativo de los conceptos, tecnologías, metodologías y herramientas relacionados con la investigación para elaborar la fundamentación teórica.
- Definición de los requisitos funcionales y no funcionales de acuerdo con las características de la colección de juegos educativos MundoClick.
- Confección del modelo de dominio para un mejor entendimiento de los conceptos asociados a él.
- Modelación de los artefactos correspondientes a cada fase de desarrollo.
- Implementación de los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas”.
- Análisis de la arquitectura de la colección de juegos educativos MundoClick para facilitar la integración de los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas”.
- Integración de los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas” a la colección de juegos educativos MundoClick.
- Ejecución de pruebas a los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas”para detectar posibles errores.

Los métodos de investigación científica proporcionan la orientación y dirección adecuada al trabajo del investigador, ya que se convierte en el camino más corto para alcanzar los resultados esperados y condiciona los nuevos conocimientos. Existen dos tipos de métodos: los teóricos y los empíricos. Los métodos teóricos permiten revelar las relaciones esenciales del objeto de investigación, no observables directamente y participan en la etapa de la construcción del modelo e hipótesis de investigación. Los métodos empíricos revelan y explican las características fenomenológicas del objeto y se emplean en la etapa de acumulación de información empírica y de comprobación experimental de la hipótesis. (5)

A continuación se describen los métodos utilizados en la presente investigación:

Métodos teóricos:

- Análisis histórico-lógico: este método propicia una mejor comprensión del estado actual del *software* educativo porque a través de él se hace un estudio de su trayectoria tanto en el mundo como en Cuba, enfocándose en los juegos educativos para niños y en las ventajas que brindan en el proceso de enseñanza-aprendizaje.
- Analítico-sintético: con este método se pretende identificar la metodología y herramientas que se utilizarán en el desarrollo de los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas”. Para ello se realizó el estudio de una variada bibliografía relacionada con el proceso de desarrollo de juegos educativos.
- Modelación: este método se utiliza para la representación de los diversos modelos durante el desarrollo de la investigación según la metodología empleada.

Métodos Empíricos:

- Observación: fue de gran importancia para la obtención de información relacionada con la problemática tratada en la investigación y para identificar las mejoras que se le hicieron a los juegos implementados.

El presente documento está conformado por cuatro capítulos, estructurados de la siguiente manera:

Capítulo 1: Fundamentación teórica. Este capítulo recoge la fundamentación teórica de la investigación y muestra el estado del arte del tema que se investiga. Se argumenta además la selección de las herramientas y metodologías a utilizar para cumplir con el objetivo de esta investigación.

Capítulo 2: Características del sistema. Se identifican los requisitos funcionales y no funcionales, se definen los casos de usos del sistema y se realiza la descripción de cada uno de ellos.

Capítulo 3: Diseño del sistema. Se realiza el diseño del sistema a desarrollar con el objetivo de definir la arquitectura, los componentes y las interfaces del mismo. Se describe el sistema a partir de diferentes diagramas que representan su funcionamiento y estructura interna.

Capítulo 4: Implementación y pruebas. En este capítulo se describe cómo está implementado el sistema, a través de los diagramas de componentes y el diagrama de despliegue. Se documenta el proceso de implementación y la estrategia de prueba a utilizar para verificar la calidad de las funcionalidades implementadas.

Capítulo 1: Fundamentación teórica

Introducción

En el presente capítulo se define qué es el *software* educativo y se enumeran sus principales características. Se analizan los juegos educativos para niños como un poderoso medio de enseñanza y aprendizaje. Conjuntamente se lleva a cabo un estudio de soluciones similares y se describen las tecnologías y herramientas que se definieron por el equipo de desarrollo al cual pertenece la solución.

1.1. *Software educativo*

La adecuada utilización de las nuevas tecnologías de la información y las comunicaciones supone un reto sin precedentes en la organización de la enseñanza y el proceso de aprendizaje, el uso de estas nuevas herramientas aportan grandes beneficios a la educación, tales como (6):

- Interés y motivación: los juegos educativos despiertan en los alumnos el interés y la motivación, lo cual beneficia en gran medida su aprendizaje.
- Interacción: es la rapidez con la que el niño percibe la respuesta del juego ante una acción que él haya hecho.
- Alfabetización digital y audiovisual: estos materiales proporcionan a los alumnos los conocimientos necesarios para la utilización adecuada de las TIC, brindándoles así la posibilidad de aprovechar las diversas herramientas existentes para su aprendizaje.
- Visualización de simulaciones: los programas permiten simular fenómenos físicos, químicos o sociales, de manera que los estudiantes pueden experimentar con ellos y así comprenderlos mejor.

Un *software* educativo es cualquier programa computacional cuyas características estructurales y funcionales sirvan de apoyo a los procesos de enseñar, aprender y administrar. El *software* educativo puede tratar diferentes temáticas y ofrecer un entorno de trabajo más o menos sensible a las circunstancias de los alumnos; pero todos comparten las siguientes características (7):

- Permite la interactividad con los estudiantes, retroalimentándolos y evaluando lo aprendido.
- Facilita las representaciones animadas.
- Incide en el desarrollo de las habilidades a través de la ejercitación.
- Permite simular procesos complejos.
- Reduce el tiempo disponible para impartir gran cantidad de conocimientos facilitando un trabajo diferenciado, introduciendo al estudiante en el trabajo con los medios computarizados.
- Facilita el trabajo independiente y a la vez un tratamiento individual de las diferencias.
- Permite al usuario familiarizarse con las nuevas tecnologías de la información y las comunicaciones.

Existen muchos tipos de *software* educativo, como son: los programas tutoriales, las herramientas, las bases de datos, los constructores y los juegos. Los programas tutoriales tienen como objetivo que los estudiantes aprendan o refuercen los conocimientos y las habilidades mediante la realización de actividades. Las herramientas brindan un entorno instrumental con el cual se facilita el tratamiento de la información. Las bases de datos ofrecen información más o menos organizada al alumno en un entorno estático. Los constructores proporcionan a los usuarios elementos simples que pueden ser utilizados para construir elementos más complejos. Por último y no menos importante se encuentran los juegos educativos, cuyo objetivo es aumentar o promover la motivación de los alumnos a través de actividades lúdicas que integran actividades educativas. (1)

En su conjunto todas estas variantes de *software* educativo contribuyen a desarrollar en los estudiantes habilidades en el uso correcto de los periféricos de una computadora. La presente investigación se enfoca en la tipología de juegos educativos, específicamente en aquellos que familiarizan a los alumnos con el manejo del *mouse*. En epígrafes posteriores se detallan las principales características de esta variante de juegos.

1.2. *Juegos educativos*

El juego es una de las actividades más relevantes en el proceso de desarrollo de la persona, es necesario para el perfeccionamiento y adquisición de habilidades de índole cognitivas, sociales y conductuales. Sus principales características son (8):

- Es una actividad libre, comenzada y terminada a voluntad del usuario.
- Es improductiva en generación de propiedad o riqueza, cuyo fin último es el juego en sí mismo.
- Está acotada, con límites de tiempo y de espacio.
- Produce incertidumbre en el desarrollo y la finalización, dado que cada partida es distinta y depende de diversas ideas.
- Inhibe las conductas socialmente recriminadas.
- Es expresivo, comunicativo, productivo, explorador y comparativo.

Los juegos educativos presentan además las siguientes características (9):

- Deben partir de una premisa a resolver.
- Combina la puesta en juego de diversas habilidades, destrezas y conocimientos.
- Estimula el razonamiento.
- Debe tener siempre al menos una solución cierta.
- Fomenta la comunicación.
- Su objetivo radica en que los niños se diviertan y aprendan simultáneamente.

Por muchos años el juego fue considerado únicamente como una diversión para el disfrute de los niños, hoy en día es uno de los recursos didácticos más interesantes. Su valor fundamental consiste en el desarrollo de la independencia y la actividad del pensamiento y del lenguaje. El índice del nivel alcanzado en la asimilación de los conocimientos y en el desarrollo de la actividad intelectual, la creatividad, la imaginación, la iniciativa y los hábitos de razonamiento, es el resultado de la utilización de estos.

1.3. *Análisis de soluciones similares*

Atrapar mariposas con burbujas

El juego consiste en encerrar cada mariposa en una burbuja con el *mouse*. Muestra una interfaz en forma de paisaje natural donde aparecen las mariposas volando de izquierda a derecha. Las especies de mariposas se diferencian en cuanto a forma, tamaño y color, por tanto la cantidad de puntos acumulados al capturar cada una de ellas varía entre 10 y 50 dependiendo de la especie capturada. El juego brinda un sistema de bonificaciones que le permite al jugador llenar el frasco de espuma con el que se hacen las burbujas y obtener un poco más de tiempo. En la parte derecha de la pantalla se muestra la cantidad de puntos acumulados, la cantidad de espuma que le queda al frasco y el tiempo que le queda al jugador para finalizar la partida. (10)

¡A Pescar!

El objetivo de este juego es capturar los peces que se mueven en la pantalla. Muestra una interfaz en forma de océano por el cual se mueven una gran diversidad de peces, el *mouse* está representado por un anzuelo el cual se ubica encima del pez y se pesca presionando el clic izquierdo. El juego contribuye al desarrollo de la percepción auditiva, pues al pasar el *mouse* por encima de los peces suenan notas musicales y cada nueva captura el jugador recibe un premio sonoro. (11)

Vamos a reciclar

El objetivo del juego consiste en colocar cada objeto en el cesto que le corresponde según las clasificaciones de papel, plástico, vidrio, aluminio y orgánico. En cada juego se muestra una interfaz con tres de los 5 cestos donde los objetos van apareciendo de uno en uno. Este juego desarrolla las habilidades auditivas ya que cuando se sitúa el *mouse* encima del objeto se escucha su nombre para que el jugador deduzca en cada caso de que material está construido. Para reciclar los objetos se debe dar clic encima del objeto y arrastrarlo hasta el cesto con el *mouse*. (12)

Colección “La caja mágica”

La colección “La caja mágica” está constituida por 14 productos que abarcan contenidos de las asignaturas de Matemática, Lengua Española, Ciencias Naturales e Informática. Cada producto está dividido en 6 módulos específicos (Temas, Ejercicios, Juegos, Mediateca, Maestro y Resultados). (13)

El módulo Juegos permite evaluar el conocimiento adquirido por el usuario a través de los juegos interactivos. (13) La colección presenta varios juegos entre los que se encuentran “Embellecer el patio”, “Las mariposas en el campo” y “Vamos a limpiar la pecera”. El juego “Embellecer el patio” consiste en recoger los objetos que están regados por el mismo para que este quede limpio. Los objetos se recogen al pasarle el cursor del *mouse* por encima, está claro que eso es una dificultad porque a veces el niño no se da cuenta de lo que tiene que hacer y con solo mover el *mouse* inconscientemente ya fueron recogidos la mayoría de los objetos. “Las mariposas en el campo” se juega moviendo el *mouse* sobre las mariposas que vuelan para que se posen y tomen el néctar de las flores, es importante resaltar que la trayectoria de las mariposas no es aleatoria por lo que siempre siguen el mismo recorrido. “Vamos a limpiar la pecera” consiste en extraer los peces para poder limpiar la pecera. Estos juegos poseen tres niveles de complejidad (fácil, medio y avanzado) y un sistema de premios

divididos por categorías que le brinda la posibilidad al usuario de conocer mediante imágenes los sitios más distinguidos de Venezuela. Estas bonificaciones son informativas, no son para ayudar al jugador a vencer próximos niveles.

Conclusiones parciales

Partiendo de las características de los juegos educativos mencionados anteriormente es posible resaltar elementos positivos como la presencia de un alto grado de fantasía y creatividad, que les permite a los niños aprender de forma interactiva las diferentes temáticas que abordan. Dichos juegos presentan muy buen diseño de acuerdo al tema que abordan y al tipo de público al que están dirigidos, pero no cuentan con una historia que motive al jugador a llegar al final del mismo. Otros aspectos relevantes son la utilización de niveles de dificultad para controlar la complejidad del juego y la presencia de mecanismos para premiar a los jugadores.

Evidentemente estas soluciones similares contribuyen al desarrollo de habilidades en el uso del *mouse*, pero en sus respectivas implementaciones pasaron por alto elementos importantes como el uso del clic derecho, la participación de varios usuarios compitiendo por un mismo objetivo, la inclusión de premios y bonificaciones que faciliten el trabajo del jugador en niveles más complejos y almacenar el estado del juego cuando el usuario sale del mismo y no ha concluido. Realizar estas mejoras y adaptar los juegos “Vamos a reciclar”, “¡A Pescar!”, “Atrapar mariposas con burbujas” y los pertenecientes a “La Caja Mágica”, para incluirlos dentro de MundoClick requeriría de mucho esfuerzo por parte de los desarrolladores ya que el código fuente no está correctamente documentado y en algunos casos no se tiene acceso al mismo. Es por ello que se decide implementar desde cero los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas” a partir de las características y deficiencias identificadas en las soluciones similares.

1.4. *Metodologías de desarrollo de software*

Para guiar el desarrollo de las soluciones propuestas es necesario el uso de una metodología que controle estrictamente cada uno de los procesos que lleva a cabo el equipo de desarrollo.

Las metodologías de desarrollo de *software* surgen ante la necesidad de utilizar una serie de procedimientos, técnicas, herramientas y soporte documental a la hora de desarrollar un producto *software*. Las mismas pretenden guiar a los desarrolladores al crear un nuevo *software*, pero los requisitos de uno a otro son tan variados y cambiantes, que ha dado lugar a que exista una gran variedad de metodologías para la creación del *software*. Estas se podrían clasificar en dos grandes grupos (14):

Metodologías ágiles: Están orientadas a la interacción con el cliente y el desarrollo incremental del *software*, mostrando versiones parcialmente funcionales del *software* al cliente en intervalos cortos de tiempo, para que pueda evaluar y sugerir cambios en el producto según se va desarrollando.

Metodologías pesadas: Son las más tradicionales, se centran en la definición detallada de los procesos y tareas a realizar, herramientas a utilizar, y requiere una extensa documentación, ya que pretende prever todo de antemano. Este tipo de metodologías son más eficaces y necesarias cuanto mayor es el proyecto que se pretende realizar respecto a tiempo y recursos que son necesarios emplear, donde una gran organización es requerida.

Debido a la experiencia del proyecto al cual pertenece la solución propuesta se decide utilizar la metodología pesada RUP[1] para que la investigación cuente con una amplia documentación sobre el proceso de desarrollo.

1.4.1 Proceso Unificado de Desarrollo (RUP)

El Proceso Unificado de Desarrollo de *software* está caracterizado por ser uno de los procesos más completos de los existentes. Es un marco genérico que puede especializarse para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos. Unifica todo el equipo de desarrollo de *software* y optimiza la comunicación de cada uno de los miembros del equipo, proporcionándoles una guía para ordenar sus actividades. Especifica los artefactos que deben desarrollarse y ofrece criterios para el control y la medición de los productos y actividades del proyecto. Está basado en componentes y utiliza el Lenguaje de Modelado Unificado (UML[2]), el cual define técnicas de análisis y diseño que permiten elaborar una solución más factible. El ciclo de vida de RUP se caracteriza por estar dirigido por casos de usos, estar centrado en la arquitectura y ser iterativo e incremental. (15)

RUP divide el proceso de desarrollo de un *software* en cuatro fases y nueve flujos de trabajos como se indica en la Figura 1.

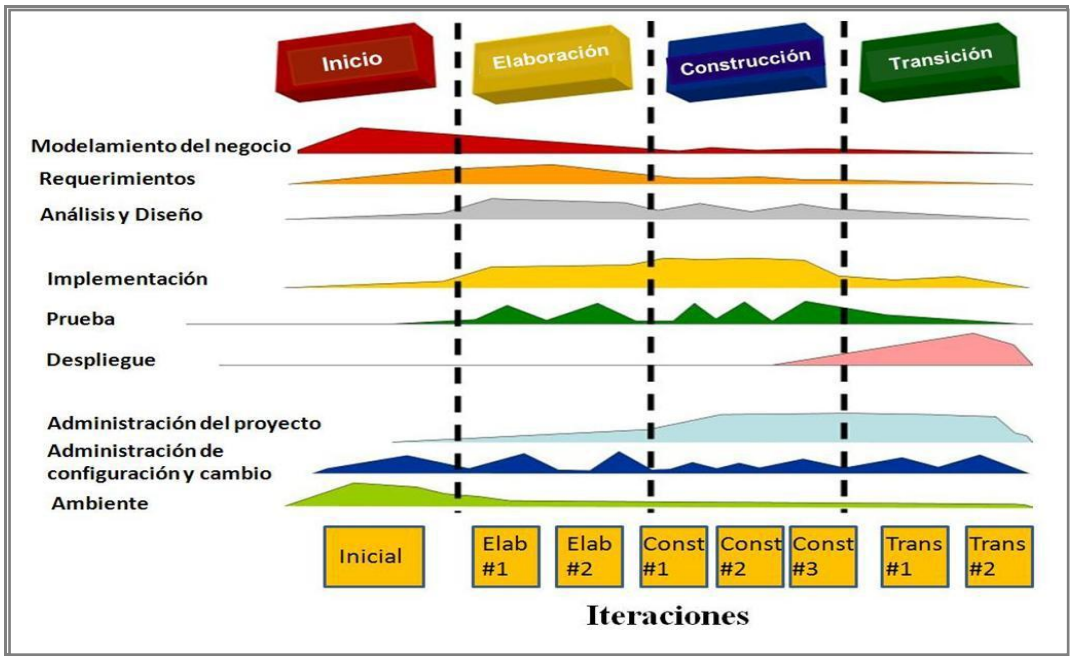


Figura 1: Fases y flujos de trabajo de RUP.

[1]Del inglés Rational Unified Process
[2] Del inglés Unified Modeling Language

1.5. *Lenguaje Unificado de Modelado*

UML es un lenguaje que permite modelar, construir y documentar los elementos que forman un *software* orientado a objetos. Captura decisiones y conocimientos sobre los sistemas que se deben construir. Se usa para entender, diseñar, configurar, mantener y controlar la información sobre tales sistemas. (16)

UML posee una notación gráfica muy expresiva que permite representar en mayor o menor medida todas las fases de un proyecto informático. (17) Este lenguaje permite realizar una verificación y validación del modelo realizado. Es importante señalar que UML no es una metodología de desarrollo de *software* sino un lenguaje para construir modelos, pues éste indica qué debe hacerse y no cómo debe hacerse. Para el modelado de los artefactos en este trabajo se utiliza en su versión 2.1.

1.6. *Herramienta CASE*

Las herramientas CASE[3] son un conjunto de métodos, utilidades y técnicas que ayudan al desarrollo de *software*. Aumentan la productividad y logran una reducción en el costo de tiempo y de dinero. Al utilizar estas herramientas se puede abstraer al código fuente en un nivel donde la arquitectura y el diseño son más fáciles de entender y modificar. (18)

Existen varias herramientas CASE para el modelado visual mediante UML de sistemas *software*, entre estas se encuentran Enterprise Architect, Rational Rose y Visual Paradigm.

1.6.1 Visual Paradigm for UML 8.0

Visual Paradigm for UML es una herramienta CASE profesional que soporta el ciclo de vida completo de desarrollo de *software* definido en RUP: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. (19)

La principal razón por la que fue seleccionada esta herramienta fue por la necesidad del uso de tecnologías libres durante el desarrollo de las aplicaciones. Por otro lado es importante destacar que permite generar la documentación en diferentes formatos como: .jpeg, .html, .pdf.

1.7. *Lenguajes de desarrollo*

Un lenguaje de desarrollo es aquel elemento dentro de la informática que permite crear programas mediante un conjunto de instrucciones, operadores o reglas de sintaxis; que pone a disposición del programador para que este pueda comunicarse con los dispositivos *hardware* y *software* existentes. (20) A continuación se describen los lenguajes que se emplean en la implementación de las soluciones.

[3] Computer Aided Software Engineering o Ingeniería de Software Asistida por Computadora.

1.7.1 HTML 5

HTML son las siglas de *Hyper Text Markup Language* (Lenguaje de Marcación de Hipertexto), es un lenguaje de composición de documento y especificación de ligas de hipertexto que define la sintaxis y coloca instrucciones especiales que no muestra el navegador. (21)

HTML 5 es la actualización de HTML, utilizada para la manipulación y maquetación de los elementos visuales de los juegos. Consta de más de cien especificaciones que se relacionan con la nueva generación de tecnologías web, es un entorno de programación completo para aplicaciones de cualquier plataforma con acceso a las capacidades de los dispositivos, vídeos, animaciones, gráficos, estilo y tipografía. (22)

1.7.2 CSS 3

Las Hojas de Estilo en Cascada (CSS[4]) son un mecanismo creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. Es la mejor forma de separar los contenidos y su presentación ya que obliga a crear documentos bien definidos y con significado completo. Es imprescindible para crear páginas web complejas, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes. El lenguaje CSS brinda la posibilidad de tener el control del diseño en la aplicación. (23)

La ventaja principal de CSS 3 es que incorpora nuevos mecanismos para mantener un mayor control sobre el estilo con el que se muestran los elementos de las páginas, sin tener que recurrir a trucos, que a menudo complicaban el código de las web. CSS 3 incluye una serie de novedades entre las que se encuentran los textos con sombras, bordes con imágenes y con las esquinas redondeadas, la capacidad de asignar múltiples fondos, contiene varias mejoras en cuanto a posicionamiento y tamaño de los objetos, usando condiciones de alineación para cada uno. (24)

1.7.3 JavaScript

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Es interpretado, multiplataforma y orientado a eventos con manejo de objetos, cuyo código se incluye directamente en el mismo documento HTML. (25)

JavaScript responde a eventos en tiempo real y se caracteriza por manejar objetos dentro de una página web sobre la cual se pueden definir diferentes eventos que facilitan la programación de páginas interactivas, a la vez que se evita la posibilidad de ejecutar comandos que puedan ser peligrosos para la máquina del usuario. Este lenguaje proporciona los medios para controlar las ventanas del navegador y el contenido que muestran, además de verificar los datos que el usuario introduce en un formulario antes de enviarlos.

[4] Del inglés Cascading Style Sheets.

1.8. *Framework de desarrollo*

Un *framework* se puede considerar como una aplicación genérica incompleta y configurable a la que podemos añadirle las últimas piezas para construir la aplicación final. Los objetivos principales que persigue son: acelerar el proceso de desarrollo, reutilizar el código ya existente y promover las buenas prácticas de desarrollo como el uso de patrones. (26)

Un *framework* simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. Por último, un *framework* facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas. (27)

Por la amplia documentación que posee se decide utilizar el *framework* Bootstrap con el objetivo de reducir el tiempo de desarrollo de la solución propuesta.

1.8.1 Bootstrap 2.3.0

Bootstrap es una herramienta de código abierto para el desarrollo rápido y correcto de aplicaciones y sitios web. Fomenta las buenas prácticas de diseño y desarrollo web. Permite diseñar webs adaptables y fluidas, visualizables correctamente en múltiples dispositivos. Incluye una base de HTML5, CSS3 y JavaScript, también incluye elementos de diseño, tipografías, tablas, formularios, navegación y alertas. (28)

Bootstrap ofrece una serie de plantillas CSS y ficheros JavaScript que nos permiten integrar el *framework* de forma sencilla y potente en los proyectos webs. Otras características son (29):

- Permite crear interfaces que se adapten a distintas escalas y resoluciones.
- Se integra perfectamente con las principales librerías JavaScript, por ejemplo jQuery.
- Ofrece un diseño sólido usando estándares como CSS3/HTML5.
- Es un *framework* ligero que se integra de forma limpia en nuestro proyecto actual.
- Funciona con todos los navegadores.

1.9. *Otras herramientas*

1.9.1 JQuery 1.9.1

JQuery es una librería JavaScript muy rápida y ligera que simplifica el desarrollo de la parte del cliente de las aplicaciones web. En otras palabras, jQuery incluye muchas utilidades para crear fácilmente las páginas web de las aplicaciones dinámicas complejas. Consiste en un único fichero JavaScript que contiene las funcionalidades comunes de DOM, eventos, efectos y AJAX. La característica principal de la biblioteca es que permite cambiar el contenido de una página web sin necesidad de recargarla, mediante la manipulación del árbol DOM y peticiones AJAX. (30)

1.9.2 NodeJS 0.8

NodeJS es un intérprete JavaScript del lado del servidor que cambia la noción de cómo debería trabajar un servidor. Su meta es permitir a un programador construir aplicaciones altamente escalables y escribir código que maneje decenas de miles de conexiones simultáneas en una máquina física. (31)

NodeJS es un proyecto de código abierto que surge como una nueva forma de aprovechar la experiencia con JavaScript, esta vez, del lado del servidor, permitiendo generar *software* de manera sencilla con recursos asíncronos y orientados a eventos, creando un solo hilo de procesos para todos los clientes, lo que hace que el servidor soporte muchas más conexiones. Presenta una arquitectura orientada a eventos, por lo que el uso de memoria es bajo, el rendimiento es alto y el modelo de programación es más simple.

La filosofía detrás de esta herramienta es hacer programas que no bloqueen la línea de ejecución de código con respecto a entradas y salidas, de modo que los ciclos de procesamiento se queden disponibles cuando se está esperando a que se completen tales acciones.

1.9.3 AppJS 0.20

AppJS es una aplicación multiplataforma, simple y potente que utiliza NodeJS para el desarrollo de aplicaciones de escritorio mediante los lenguajes: HTML, CSS y JavaScript, permitiendo además el uso de tecnologías asociadas a ellos como Bootstrap y jQuery. Dicha herramienta permite utilizar las API de HTML 5 para crear aplicaciones atractivas desde procesadores de texto hasta juegos en tres dimensiones. (32) La utilización de Node.js como la columna vertebral de AppJS facilita el proceso de desarrollo de aplicaciones altamente escalables de redes y una interfaz de programación de aplicaciones agradable.

1.10. *Entorno de desarrollo integrado*

Un entorno de desarrollo integrado (IDE[5]) es un programa informático compuesto por un conjunto de herramientas para programar en uno o varios lenguajes de programación, donde podemos encontrar como mínimo un editor de texto, un compilador, un intérprete y un depurador de código. (33)

Existen diversos tipos de entornos de desarrollo integrado, entre ellos se encuentran Eclipse, Microsoft Visual Studio, ZendStudio y NetBeans, seleccionando este último por las características que se mencionan a continuación.

1.10.1 NetBeans 7.3

NetBeans es un IDE de código abierto, gratuito y sin restricciones de uso, posee una gran comunidad de usuarios y desarrolladores de todo el mundo que proporcionan una amplia documentación y recursos de capacitación a los programadores. Dicha herramienta está pensada para escribir, compilar, depurar y ejecutar programas. (34)

[5] Del inglés Integrated Development Environment.

La plataforma está diseñada para el lenguaje Java pero soporta otros lenguajes como: C/C++, PHP y JavaScript. Entre sus principales características se encuentra la disponibilidad en diferentes sistemas operativos, que posee un navegador web integrado y un sistema de proyectos basado en control de versiones. (34)

Conclusiones del capítulo

- El estudio detallado de los conceptos asociados al objeto de estudio y de aplicaciones similares a la que se desarrolla en la investigación demostró la necesidad de desarrollar los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas” para la colección de juegos educativos MundoClick.
- La selección de RUP como metodología de desarrollo, UML como lenguaje de modelado, JavaScript, CCS 3 y HTML 5 como lenguajes para la programación, Bootstrap como *framework* de desarrollo, jQuery como librería y Visual Paradigm y NetBeans como herramientas CASE e IDE respectivamente, conformó el entorno de desarrollo para dar solución al problema a resolver, garantizando la compatibilidad con la arquitectura del proyecto productivo Multisaber-El Navegante.

Capítulo 2: Características del sistema

Introducción

En el presente capítulo se realiza una caracterización del sistema que se desea implementar. Teniendo en cuenta las características y cualidades que el sistema debe cumplir se identifican los requisitos funcionales y no funcionales del *software*. Con el modelo del dominio se puntualizan los conceptos fundamentales asociados a él. Además se analizan los actores involucrados, se realiza el diagrama de casos de uso del sistema y se describen cada uno de estos.

2.1. *Aspectos generales de los juegos*

MundoClick es una colección de juegos educativos destinada a fomentar el aprendizaje en niños de 5 a 12 años. Los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas” estarán integrados a dicha colección con el objetivo de desarrollar en ellos habilidades que faciliten el manejo del *mouse* como un periférico de la computadora.

2.1.1 Juego: “Mis maripositas”

El juego consiste en recoger las mariposas y llevarlas a un lugar seguro, para capturar las mariposas el jugador debe dar clic encima de ella y ser veloz, pues cada vez que las mariposas prueben el néctar infectado de las flores con fertilizantes irán enfermando poco a poco y no podrá cumplir su meta. Cuenta con 10 niveles que representan un reto para el jugador, en cada uno de ellos se muestra una interfaz en forma de paisaje natural donde aparecen las mariposas volando por toda la pantalla, las cuales se posan aleatoriamente en las flores para alimentarse. El juego brinda un sistema de bonificaciones que le permite al jugador cuando llegue a 15 puntos usar un spray para curar las flores enfermas y cuando llegue a 10 aplicar un jamo más grande y más rápido para rescatar las mariposas. En la parte inferior de la pantalla se muestra la cantidad de puntos acumulados, el total de mariposas de cada especie que tiene el jugador y el tiempo que lleva el jugador en la partida (ver anexo 13).

2.1.2 Juego: “Recicla”

El objetivo del juego es que los niños aprendan sobre el cuidado del medio ambiente a través de la clasificación de objetos reciclables. El juego consiste en colocar cada objeto en el cesto que le corresponde según las clasificaciones de papel, plástico, vidrio y metal. Cuenta con 6 niveles de complejidad, en cada uno se muestra una interfaz que representa diferentes partes de una ciudad. El juego le brinda al jugador la posibilidad de ser premiado con una lupa que sirve para darle pistas en caso de no saber el material de algún elemento. En la parte inferior de la pantalla se muestra la cantidad de puntos acumulados y el tiempo que lleva el jugador en la partida (ver anexo 14).

2.1.3 Juego: "Limpiar la pecera"

En este juego el jugador debe limpiar la pecera, cuenta con 3 niveles de complejidad donde se extraen los peces de la pecera y se depositan en otros recipientes, es importante destacar que los peces solo podrán ser ubicados en el recipiente que le corresponde según el tamaño del pez. Después se le saca el agua sucia a la pecera para depositar la limpia y para finalizar se colocan nuevamente los peces extraídos inicialmente. En la parte inferior de la pantalla se muestra el tiempo que lleva el jugador en la partida (ver anexo 15).

2.2. *Especificación de los requisitos de software*

La captura de requisitos es un flujo de trabajo cuyo propósito esencial es orientar el desarrollo hacia el sistema correcto. Esto se lleva a cabo mediante la descripción de los requisitos del sistema de forma tal que se pueda llegar a un acuerdo entre el cliente (incluyendo los usuarios) y los desarrolladores del sistema, acerca de lo que el sistema debe hacer y lo que no. Se entiende por requisito la condición o capacidad que debe cumplir un sistema. Teniendo en cuenta sus características, los requisitos se clasifican en: funcionales y no funcionales. (35)

Un requisito funcional especifica una acción que debe ser capaz de realizar el sistema, sin tener en cuenta las restricciones físicas. Un requisito no funcional describe las propiedades del sistema, como restricciones del entorno o de implementación, rendimiento, dependencias de la plataforma, mantenibilidad, seguridad, extensibilidad o fiabilidad. (35)

A continuación se muestran los requisitos funcionales y no funcionales de cada uno de los juegos a desarrollar.

2.2.1 Requisitos funcionales

Juego "Mis maripositas"

RF 1: Crear entorno por niveles.

RF 1.1: Mostrar las mariposas volando por toda la pantalla. Las mariposas toman valores aleatorios para moverse por toda la pantalla.

RF 1.2: Modificar las flores aleatoriamente de forma tal que le permita al jugador identificar cuáles están roseadas con el fertilizante y cuáles no. Las flores enfermas estarán en blanco y negro.

RF 1.3: Permitir que las mariposas se posen aleatoriamente en las flores para alimentarse.

RF 1.4: Modificar el color de la mariposa para diferenciar las enfermas de las sanas. Las mariposas enfermas estarán en blanco y negro.

RF 1.5: Mostrar el total de mariposas sanas rescatadas por el jugador en cada momento.

RF 1.6: Mostrar el total de mariposas enfermas rescatadas por el jugador en cada momento.

RF 1.7: Mostrar total de puntos.

RF 1.8: Mostrar contador de tiempo.

RF 2: Bonificar al jugador con un jamo más grande. Este premio brinda la posibilidad al jugador de rescatar las mariposas con mayor facilidad porque se amplía el tamaño del cursor para hacer clic sobre ella.

RF 3: Premiar al jugador con un spray para rociar las flores. Con este premio se curan las flores enfermas presionando el clic izquierdo sobre ellas.

RF 4: Salvar el estado del juego. Este requisito brinda la posibilidad al jugador de guardar el estado del juego al salir del mismo, evitando que cuando vuelva a entrar no tendrá q comenzar desde el primer nivel.

RF 5: Permitir la participación de hasta 4 jugadores uno detrás de otro en la misma computadora.

Juego: “Recicla”

RF 1: Crear entorno por niveles.

RF 1.1: Almacenar los objetos reciclables. Para esto el usuario debe señalar el objeto y hacer clic en el cesto al cual pertenece según las clasificaciones de plástico, papel, cristal y metal.

RF 1.2: Mostrar una moneda de forma aleatoria detrás del objeto reciclado. El usuario acumula 5 puntos cuando hace clic sobre ella para coleccionarla.

RF 1.3: Mostrar algunos objetos más resaltados que otros. Por cada uno de estos objetos reciclados se activa la lupa como bonificación del juego.

RF 1.4: Modificar la posición de los objetos de la matriz. A partir del cuarto nivel la matriz de objetos va a cambiar de posición aleatoriamente cada vez que el jugador haga clic en uno de ellos.

RF 1.5: Mostrar el total de puntos.

RF 1.6: Mostrar tiempo.

RF 2: Premiar al jugador con una lupa. Esta bonificación permitirá al usuario ver los objetos que faltan por reciclar en el nivel.

RF 3: Salvar el estado del juego. Este requisito brinda la posibilidad al jugador de guardar el estado del juego al salir del mismo, evitando que cuando vuelva a entrar no tendrá q comenzar desde el primer nivel.

RF 4: Permitir la participación de hasta 4 jugadores uno detrás de otro en la misma computadora.

Juego: “Limpiar la pecera”

RF 1: Crear entorno por niveles.

RF 1.1: Mostrar los peces moviéndose por toda la pecera.

RF 1.2: Extraer los peces de la pecera. Para extraer los peces de la pecera se dará clic sobre el pez.

RF 1.3: Depositar los peces en otro recipiente. Para depositar los peces en un recipiente determinado se dará clic en el recipiente deseado. Es importante señalar que los peces grandes no podrán ser ubicados en el recipiente pequeño.

RF 1.4: Extraer el agua de la pecera. Para vaciar la pecera se da clic derecho y se selecciona la acción correspondiente.

RF 1.5: Depositar agua limpia en la pecera. Para llenar la pecera se da clic derecho y se selecciona la acción correspondiente.

RF 1.6: Mostrar tiempo.

RF 2: Salvar el estado del juego. Este requisito brinda la posibilidad al jugador de guardar el estado del juego al salir del mismo, evitando que cuando vuelva a entrar no tendrá q comenzar desde el primer nivel.

RF 3: Permitir la participación de hasta 4 jugadores uno detrás de otro en la misma computadora.

2.2.2 Requisitos no funcionales

RNF 1: Requisito de portabilidad

El sistema podrá ser utilizado bajo los sistemas operativos Windows XP o superior y Linux distribución Ubuntu 12.04 o superior.

RNF 2: Requisito de *hardware*

- Procesador Pentium 233 MHz (recomendado 500 MHz o mayor).
- 512 MB de RAM (recomendado 1 GB de RAM o mayor).
- Lector de CD-ROM.
- Soporte de video que admita resolución de al menos 1024x780px y 24 bits.
- Dispositivo de red de al menos 10 Mbits de velocidad de transmisión.

RNF 3: Requisito de interfaz

La interfaz del sistema debe presentar rasgos de diseño infantiles, predominando el uso de colores atractivos para captar la atención del niño e imágenes en lugar de textos para sugerir las acciones.

RNF 4: Requisitos de soporte

Se realizará transferencia tecnológica de la colección a los clientes.

RNF 5: Requisitos de usabilidad

El *software* será usado por los niños en las edades de 5 a 12 años.

RNF 6: Restricciones de diseño e implementación

- Lenguaje de desarrollo: JavaScript
- Lenguaje de marcado: HTML 5
- Hoja de estilo: CSS 3
- Herramientas de desarrollo: jQuery V1.7 y NodeJs v0.8.11
- Entorno de Desarrollo Integrado: NetBeans v7.3
- Herramienta CASE: Visual Paradigm

2.3. *Modelo del dominio*

El modelo del dominio captura los tipos de objetos más importantes en el contexto del sistema. Los conceptos asociados al dominio representan las cosas que existen o los eventos que suceden en el entorno en el que trabaja el sistema. Muchos de los elementos del dominio pueden obtenerse de una especificación de requisitos o mediante la entrevista con los expertos del dominio. Las clases del dominio aparecen en tres formas típicas (15):

- Objetos que representan cosas que se manipulan en el negocio.
- Objetos del mundo real y conceptos de los que el sistema debe hacer un seguimiento.
- Sucesos que ocurrirán o han ocurrido.

El modelo de dominio se describe mediantes diagramas de UML. Estos muestran los conceptos del dominio y cómo se relacionan unos con otros mediante asociaciones. A continuación se definen los elementos principales que intervienen en el dominio de cada uno de los juegos a desarrollar.

Juego “Mis maripositas”

Juego: Conjunto de niveles en los que se recrea el paisaje con las mariposas que el jugador debe rescatar.

Jugador: Usuario que interactúa con el juego “Mis maripositas”.

Bonificación: Premio que recibe el jugador al acumular cierta cantidad de puntos.

Nivel: Cada nivel representa un desafío diferente donde aumenta la complejidad del juego a medida que el jugador va avanzando.

Mariposa: Animal de colores vistosos que se mantiene volando por toda la pantalla para que el jugador lo atrape.

Flor: Es donde las mariposas se posan para alimentarse, varían en cuanto a tamaño, forma y color.

Jamo: Herramienta para cazar mariposas.

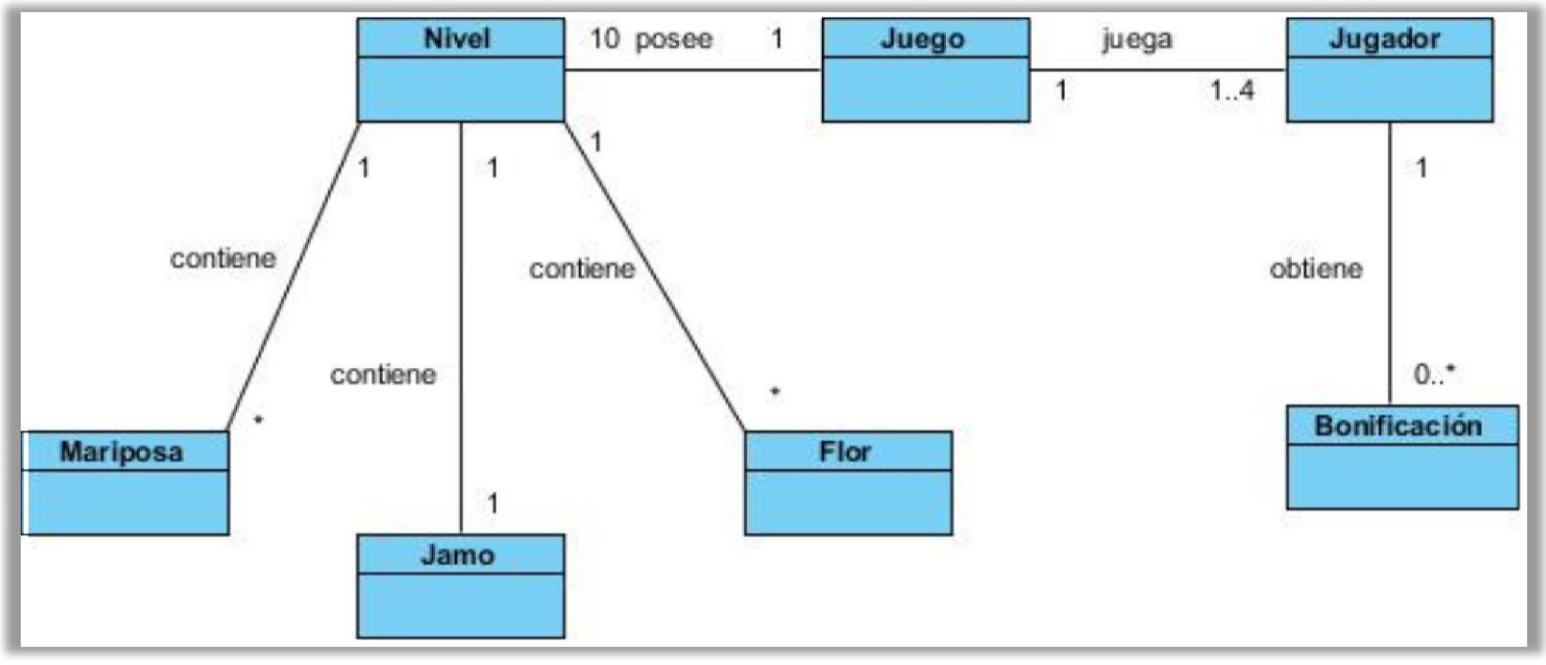


Figura 2: Modelo del dominio del juego “Mis maripositas”.

Juego “Recicla”

Juego: Conjunto de niveles en los que se recrea el paisaje con los objetos que el jugador debe reciclar.

Jugador: Usuario que interactúa con el juego “Recicla”.

Bonificación: Premio que recibe el jugador al reciclar los objetos que resaltan en la pantalla.

Nivel: Cada nivel representa un desafío diferente donde aumenta la complejidad del juego a medida que el jugador va avanzando.

Matriz de objetos: Lugar donde están representados los objetos reciclables.

Objeto: Objetos reciclables clasificados de acuerdo con el material que fueron fabricados: plástico, papel, metal y vidrio.

Cesto: Recipiente donde el jugador recicla los objetos.

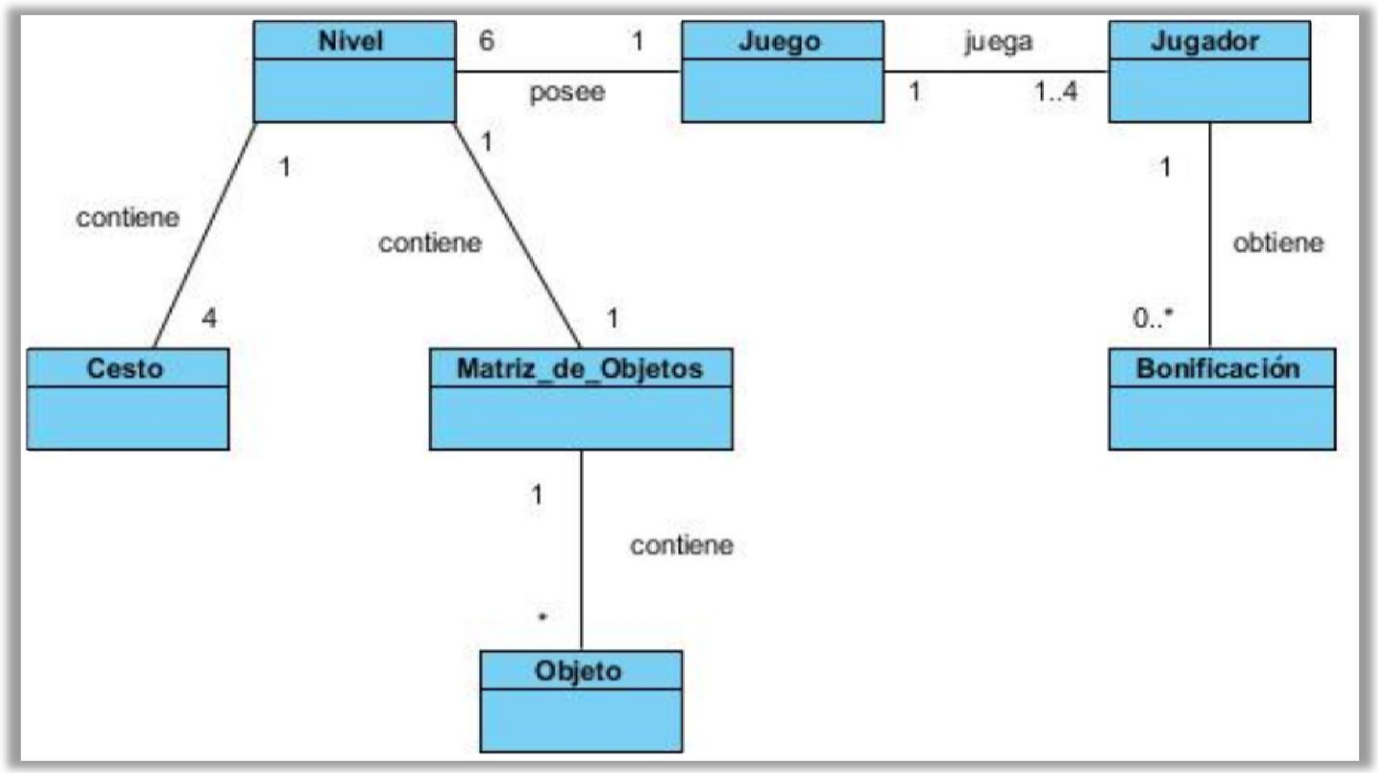


Figura 3: Modelo del dominio del juego “Recicla”

Juego “Limpiar la pecera”

Juego: Conjunto de niveles en los que se recrea la pecera con los peces que el jugador debe extraer para poder limpiarla.

Jugador: Usuario que interactúa con el juego “Limpiar la pecera”.

Nivel: Cada nivel representa un desafío diferente donde aumenta la complejidad del juego a medida que el jugador va avanzando.

Pez: Animal que se mantiene moviéndose por toda la pecera.

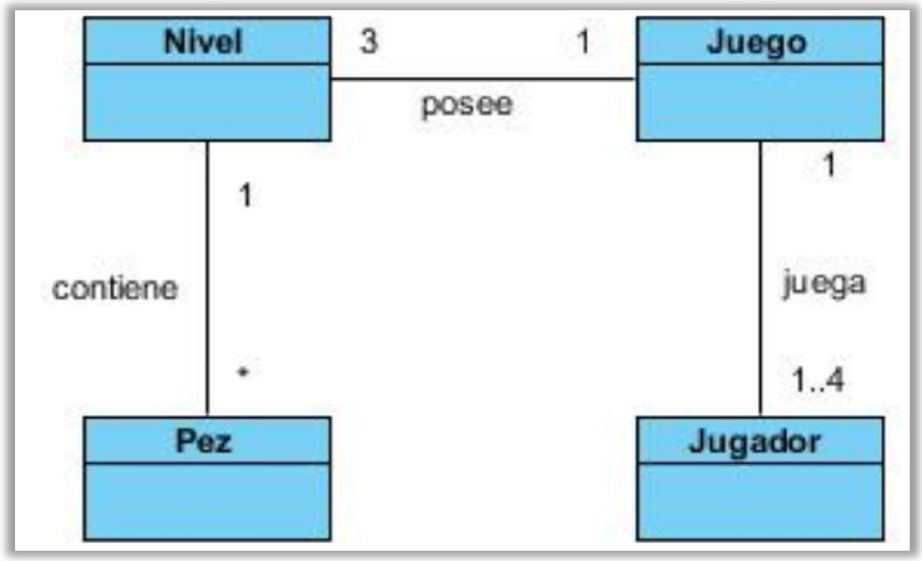


Figura 4: Modelo del dominio del juego "Limpiar la pecera".

2.4. *Modelo de casos de uso del sistema*

El modelo de casos de uso describe los requerimientos funcionales del sistema en forma de casos de uso. Un caso de uso especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto. Un diagrama de casos de uso explica gráficamente un conjunto de casos de uso de un sistema, los actores y la relación entre estos y los casos de uso. Tiene por objeto ofrecer una clase de diagrama contextual que permite conocer rápidamente los actores externos de un sistema y las formas básicas en que lo utilizan. (15)

A continuación se presentan los diagramas de casos de uso del sistema y la descripción de los casos de usos más significativos de cada juego, las restantes descripciones se pueden encontrar en los anexos del presente documento.

Juego "Mis maripositas"

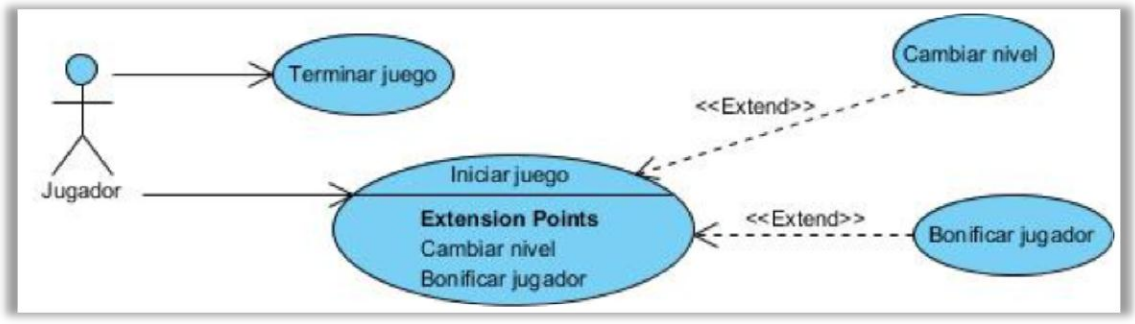


Figura 5: Diagrama de casos de uso del juego "Mis maripositas".

Caso de uso	Iniciar juego
Actores	Jugador

<table>
<tr><td>Resumen</td><td colspan="2">El caso de uso inicia cuando el jugador selecciona el juego y termina cuando recoge todas las mariposas.</td></tr>
<tr><td>Precondiciones</td><td colspan="2">-</td></tr>
<tr><td>Postcondiciones</td><td colspan="2">-</td></tr>
<tr><td>Prioridad</td><td colspan="2">Crítico</td></tr>
<tr><td colspan="3">Flujo normal de eventos</td></tr>
<tr><td>Acción del actor</td><td colspan="2">Respuesta del sistema</td></tr>
<tr><td>1. Selecciona el juego.</td><td colspan="2">2. Muestra una interfaz que permite al jugador seleccionar una de las siguientes opciones:
-Iniciar juego: ir a la sección “Iniciar juego”
-Multijugador: ir a la sección “Multijugador”</td></tr>
<tr><td colspan="3">Sección 1 “Iniciar juego”: Flujo Normal de Eventos</td></tr>
<tr><td>Acción del actor</td><td colspan="2">Respuesta del sistema</td></tr>
<tr><td>1. Selecciona la opción: Iniciar juego.</td><td colspan="2">2. Muestra la interfaz del juego:
• Muestra las mariposas volando por toda la pantalla.
• Modifica las flores de forma tal que le permita al jugador identificar cuáles están rociadas con el fertilizante y cuáles no.
• Permite que las mariposas se posen en las flores aleatoriamente para alimentarse.
• Modifica el color de la mariposa para diferenciar las enfermas de las sanas.
• Muestra el puntero del mouse en forma de jamo.
• Muestra el contador del tiempo.</td></tr>
<tr><td>3. Hace clic en una mariposa sana.</td><td colspan="2">4. La mariposa es eliminada de la pantalla.</td></tr>
</table>

	5. Aumenta el contador de mariposas sanas. 6. Aumenta la puntuación general. 7. Finaliza el caso de uso.
Flujo Alterno: "Mariposa enferma" de la sección 1.3.a	
1. Hace clic en una mariposa enferma.	2. Aumenta el contador de mariposas enfermas.
Flujo Alterno: "Ultima mariposa" de la sección 1.3.b	
1. Hace clic en una mariposa.	2. El juego cambia para el próximo nivel.
Sección 2 "Multijugador": Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema.**
1. Selecciona la opción: Multijugador.	2. El sistema muestra la interfaz del juego: • Muestra las mariposas volando por toda la pantalla. • Modifica las flores de forma tal que le permita al jugador identificar cuáles están rociadas con el fertilizante y cuáles no. • Permite que las mariposas se posen en las flores aleatoriamente para alimentarse. • Modifica el color de la mariposa para diferenciar las enfermas de las sanas. • Muestra el puntero del mouse en forma de jamo. • Muestra el contador del tiempo.
3. Hace clic en una mariposa sana.	4. La mariposa es eliminada de la pantalla. 5. Aumenta el contador de mariposas sanas. 6. Aumenta la puntuación general. 7. Termina el caso de uso.

Flujo Alterno: "Mariposa enferma" de la sección 2.3.a		
1. Hace clic en una mariposa enferma.	2. Aumenta el contador de mariposas enfermas.	
Flujo Alterno: "Ultima mariposa" de la sección 2.3.b		
1. Hace clic en una mariposa.	2. Cambia para el siguiente jugador.	
Requisitos no funcionales		
Relaciones	**CU Incluidos**	
	CU Extendidos	

Tabla 1: Descripción textual del CU Iniciar juego "Mis maripositas".

Juego "Recicla"

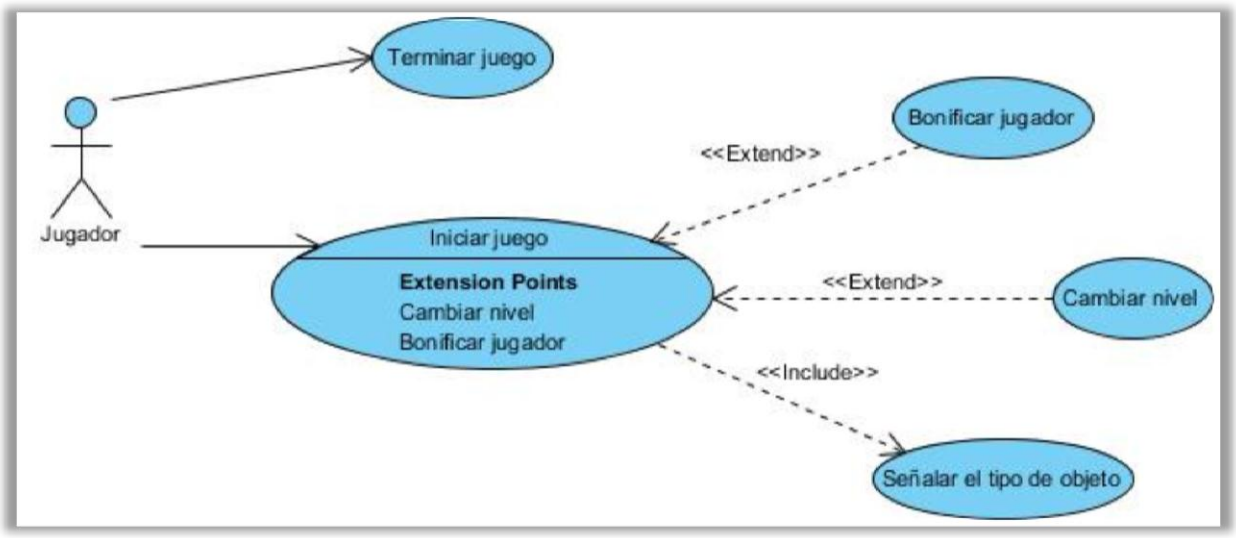


Figura 6: Diagrama de casos de uso del juego "Recicla".

Caso de uso	Iniciar juego
Actores	Jugador
Resumen	El caso de uso inicia cuando el jugador selecciona el juego y termina cuando recicla todos los objetos.
Precondiciones	**-**
Poscondiciones	**-**
Prioridad	Crítico
Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**

1. Selecciona el juego.	2. Muestra una interfaz que le permite al jugador seleccionar una de las siguientes opciones: -Iniciar juego: **ir a la sección "Iniciar juego"** -Multijugador: **ir a la sección "Multijugador"**
Sección 1 "Iniciar juego": Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción: Iniciar juego.	2. Muestra la interfaz del juego: • Muestra una pantalla con la matriz de objetos reciclables y cuatro cestos en los cuales serán almacenados. • Muestra un cartel para que el jugador recicle los objetos que le indica el sistema. • Muestra el contador del tiempo.
3. Hace clic en un objeto.	4. El objeto es resaltado.
5. Hace clic en el cesto para reciclar el objeto.	6. El objeto desaparece. 7. Incrementa el contador de puntos. 8. Termina el caso de uso.
Flujo Alterno: "Cesto equivocado" de la sección 1.5.a	
1. Hace clic en el cesto para reciclar el objeto.	2. Muestra un mensaje indicándole al jugador que el objeto no pertenece al cesto seleccionado.
Flujo Alterno: "Ultima objeto" de la sección 1.7.a	
1. Hace clic en un objeto.	2. El juego cambia para el próximo nivel.
Sección 2 "Multijugador": Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema.**
1. Selecciona la opción: Multijugador.	2. Muestra la interfaz del juego: • Muestra una pantalla con la matriz de objetos reciclables y cuatro cestos en los cuales serán almacenados.

	• Muestra un cartel para que el jugador recicle los objetos que le indica el sistema. • Muestra el contador del tiempo.
3. Hace clic en un objeto.	4. El objeto es resaltado.
5. Hace clic en el cesto para reciclar el objeto.	6. El objeto desaparece. 7. Incrementa el contador de puntos. 8. Termina el caso de uso.
Flujo Alterno: "Cesto equivocado" de la sección 2.5.a	
1. Hace clic en el cesto para reciclar el objeto.	2. Muestra un mensaje indicándole al jugador que el objeto no pertenece al cesto seleccionado.
Flujo Alterno: "Ultima objeto" de la sección 2.7.a	
1. Hace clic en un objeto.	2. Cambia para el siguiente jugador.

Requisitos no funcionales		
Relaciones	**CU Incluidos**	
	CU Extendidos	

Tabla 2: Descripción textual del CU Iniciar juego "Recicla".

Juego "Limpiar la pecera"

Figura 7: Diagrama de casos de uso del juego "Limpiar la Pecera".

Caso de uso	Iniciar juego
Actores	Jugador
Resumen	El caso de uso inicia cuando el jugador selecciona el juego y termina cuando la pecera queda limpia.
Precondiciones	-
Poscondiciones	-
Prioridad	Crítico

Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1. Selecciona el juego.	2. Muestra una interfaz que le permite al jugador seleccionar una de las siguientes opciones: -Iniciar juego: **ir a la sección "Iniciar juego"** -Multijugador: **ir a la sección "Multijugador"**
Sección 1 "Iniciar juego": Flujo Normal de Eventos	
Acción del actor	Respuesta del sistema
1. Selecciona la opción: Iniciar juego.	2. Muestra la interfaz del juego: • Muestra una mesa con varios objetos: una pecera y dos recipientes al lado de la misma. • Muestra los peces moviéndose por toda la pecera. • Muestra el contador del tiempo.
3. Hace clic en el pez.	4. El pez desaparece de la pecera y el cursor del mouse se muestra con una imagen del pez.
5. Hace clic en un recipiente.	6. El pez aparece en el recipiente y el cursor del mouse se restablece.
7. Hace clic derecho en la pecera.	8. Muestra un menú con la opción "Vaciar pecera".
9. Selecciona la opción: "Vaciar pecera".	10. Muestra el agua saliendo de la pecera.
11. Hace clic derecho en la pecera.	12. Muestra un menú con la opción "Llenar pecera".
13. Selecciona la opción: "Llenar pecera".	14. Muestra la pecera llenándose. 15. Los peces se mueven para la pecera. 16. Termina el caso de uso.
Flujo Alterno: "Clic fuera del recipiente" de la sección 1.5.a	
1. Hace clic fuera del recipiente.	2. El pez vuelve a la posición que tenía

	en la pecera.
Flujo Alterno: "Peces en la pecera" de la sección 1.9.a	
1. Selecciona la opción: "Vaciar pecera".	2. Muestra un mensaje: "No se puede vaciar la pecera con peces entro".
Flujo Alterno: "Cambiar nivel" de la sección 1.15.a	
	1. El juego cambia para el próximo nivel.
Sección 2 "Multijugador": Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema.**
1. Selecciona la opción: Multijugador**.**	2. Muestra la interfaz del juego: • Muestra mesa con varios objetos: una pecera y dos recipientes al lado de la misma. • Muestra los peces moviéndose por toda la pecera. • Muestra el contador del tiempo.
3. Hace clic en el pez.	4. El pez desaparece de la pecera y el cursor del mouse se muestra con una imagen del pez.
5. Hace clic en un recipiente.	6. El pez aparece en el recipiente y el cursor del mouse se restablece.
7. Hace clic derecho en la pecera.	8. Muestra un menú con la opción "Vaciar pecera".
9. Selecciona la opción: "Vaciar pecera".	10. Muestra el agua saliendo de la pecera.
11. Hace clic derecho en la pecera.	12. Se muestra un menú con la opción "Llenar pecera".
13. Selecciona la opción: "Llenar pecera".	14. Muestra la pecera llenándose. 15. Los peces se mueven para la pecera. 16. Cambia para el próximo jugador. 17. Termina el caso de uso.
Flujo Alterno: "Clic fuera del recipiente" de la sección 2.5.a	
1. Hace clic fuera del recipiente.	2. El pez vuelve a la posición que tenía en la pecera.
Flujo Alterno: "Peces en la pecera" de la sección 2.9.a	

1. Selecciona la opción: “Vaciar pecera”.	2. Muestra un mensaje: “No se puede vaciar la pecera con peces entro”.
Requisitos no funcionales	

Tabla 3: Descripción textual del CU Iniciar juego "Limpiar la pecera”.

Conclusiones del capítulo

- La realización de la descripción de la propuesta del sistema permitió definir los principales conceptos asociados al dominio del problema y ofreció un mejor entendimiento de las soluciones a implementar.
- La identificación de las características y funcionalidades a incluir en el desarrollo de los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas” propició un correcto levantamiento de requisitos, el que permitirá una adecuada elaboración de los artefactos ingenieriles de las diferentes fases del desarrollo.

Capítulo 3: Diseño del sistema

Introducción

Según la IEEE 610.12-90 el diseño del sistema es el proceso de definición de la arquitectura, componentes, interfaces y otras características de un sistema o componente que resulta de este proceso. En el siguiente capítulo se abordarán los temas mencionados anteriormente a través de los diagramas de interacción y los diagramas de clases del diseño; además, se explica el estilo de codificación, el patrón arquitectónico y los patrones de diseño utilizados en la solución propuesta.

3.1. *Diagramas de clases del diseño*

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de *software* y de las interfaces en una aplicación. (36) Una clase de diseño es una abstracción de una clase o construcción similar en la implementación del sistema. Por ejemplo (15):

- El lenguaje utilizado para especificar una clase del diseño es el mismo que el lenguaje de programación. Consecuentemente, las operaciones, parámetros, atributos, tipos y demás son especificados utilizando la sintaxis del lenguaje de programación elegido.
- La visibilidad de los atributos y las operaciones de una clase del diseño se especifica con frecuencia, por ejemplo: las palabras *public*, *protected* y *private*.
- Las relaciones de aquellas clases del diseño implicadas con otras clases, a menudo tienen un significado directo cuando la clase es implementada. Por ejemplo, la generalización tiene una semántica que se corresponde con el significado de generalización o herencia en el lenguaje de programación.
- Los métodos de una clase del diseño tienen correspondencia directa con el correspondiente método en la implementación de las clases.

A continuación se muestran los diagramas de clases del diseño para cada uno de los juegos “Mis maripositas”, “Recicla” y “Limpiar la pecera” respectivamente.

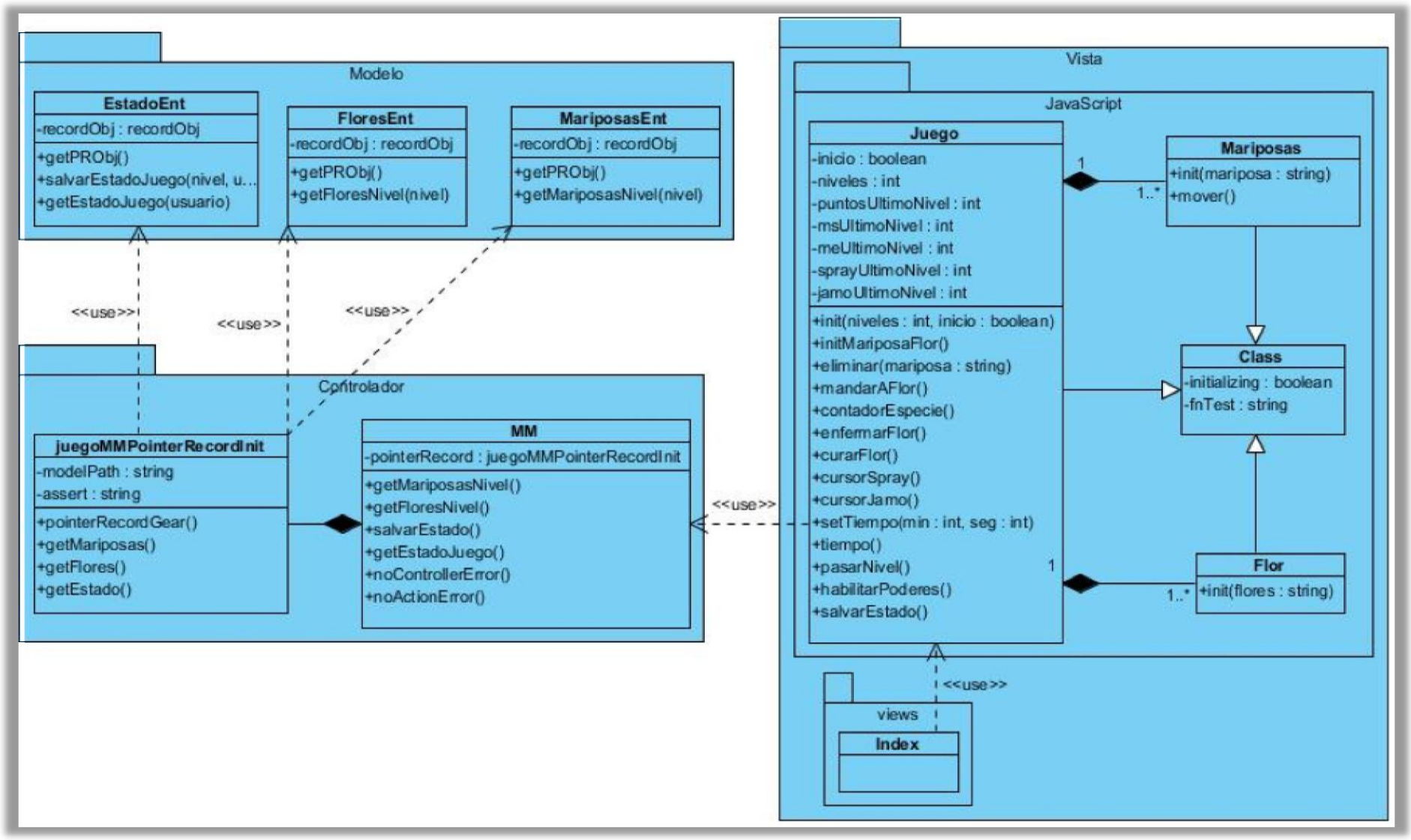


Figura 8: Diagrama de clases del diseño del juego "Mis maripositas".

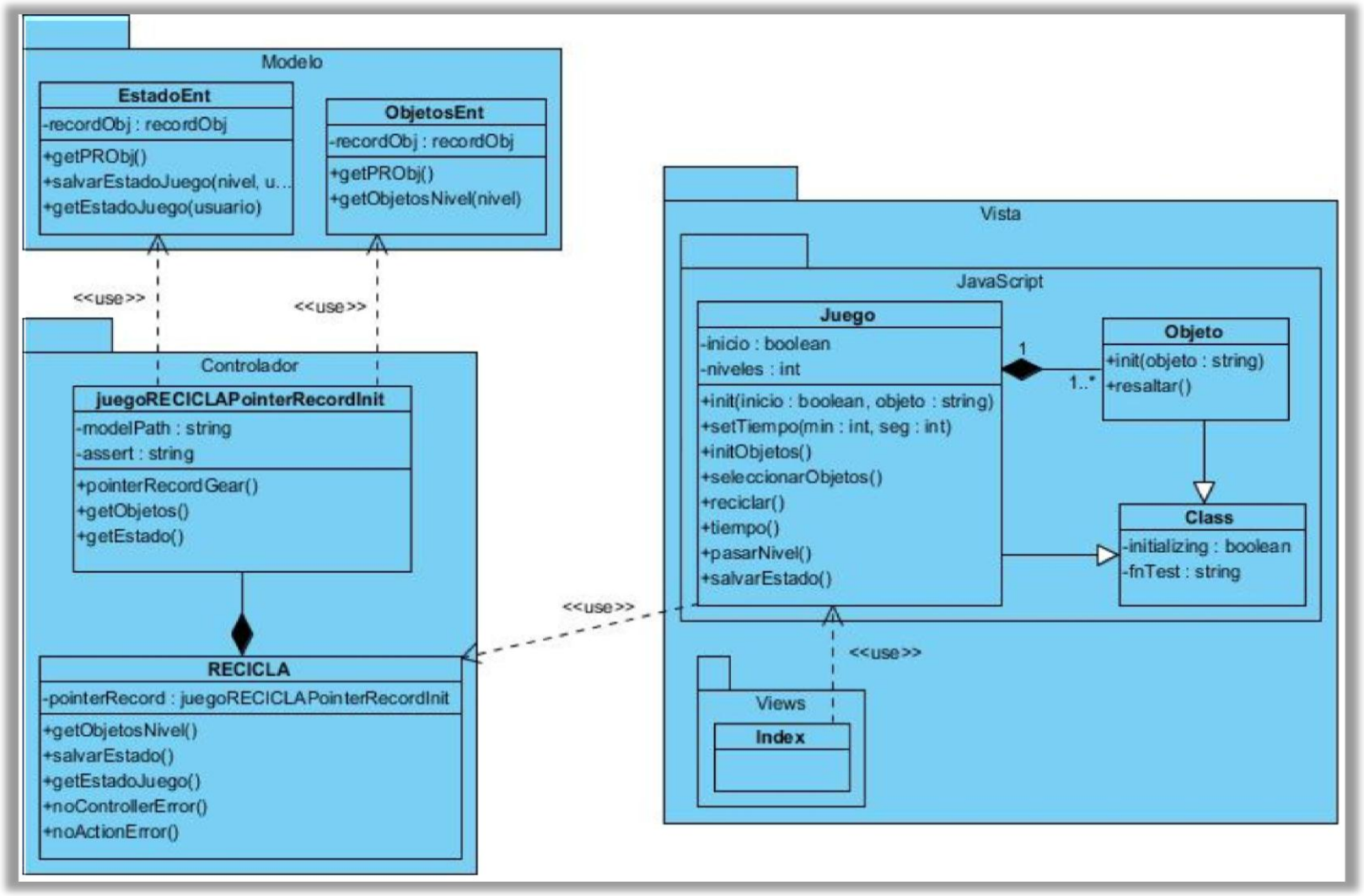


Figura 9: Diagrama de clases del diseño del juego "Recicla".

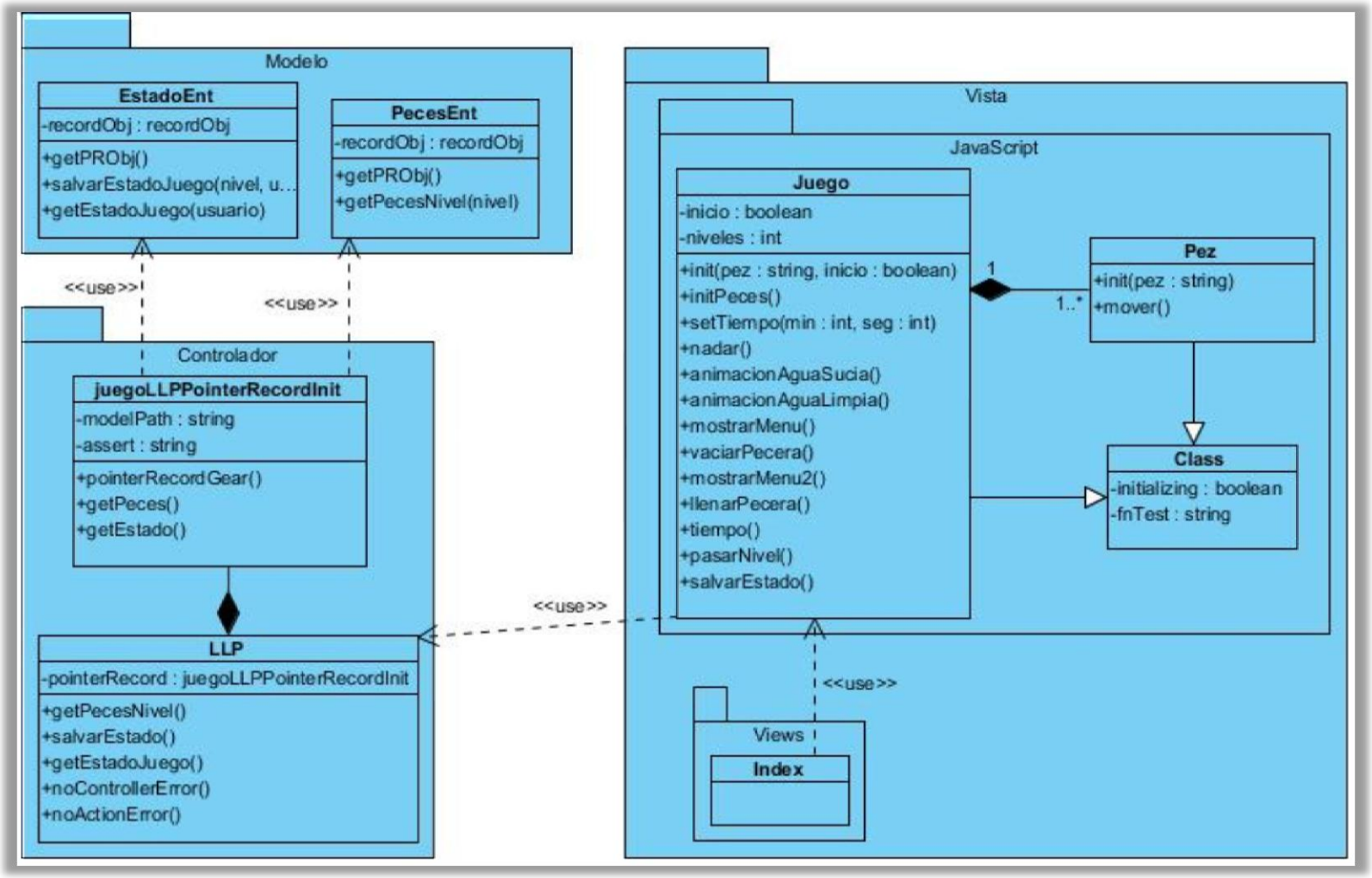


Figura 10: Diagrama de clases del diseño del juego "Limpiar la pecera".

3.2. *Diagramas de interacción*

Un diagrama de interacción explica gráficamente las interacciones existentes entre las instancias de las clases del modelo. El punto de partida de las interacciones es el cumplimiento de las poscondiciones de los contratos de operación. El UML define dos tipos de estos diagramas; ambos sirven para expresar interacciones semejantes o idénticas de mensaje (36):

1. Diagramas de colaboración: que describen las interacciones entre los objetos en un formato de grafo o red.
2. Diagramas de secuencia: que describen las interacciones en una especie de formato de cerca o muro.

En el diseño, es preferible representar la secuencia de acciones con diagramas de secuencias ya que el centro de atención principal es encontrar secuencias de interacciones detalladas y ordenadas en el tiempo (15).

A continuación se muestran los diagramas de secuencias para cada uno de los juegos “Mis maripositas”, “Recicla” y “Limpiar la pecera” respectivamente.

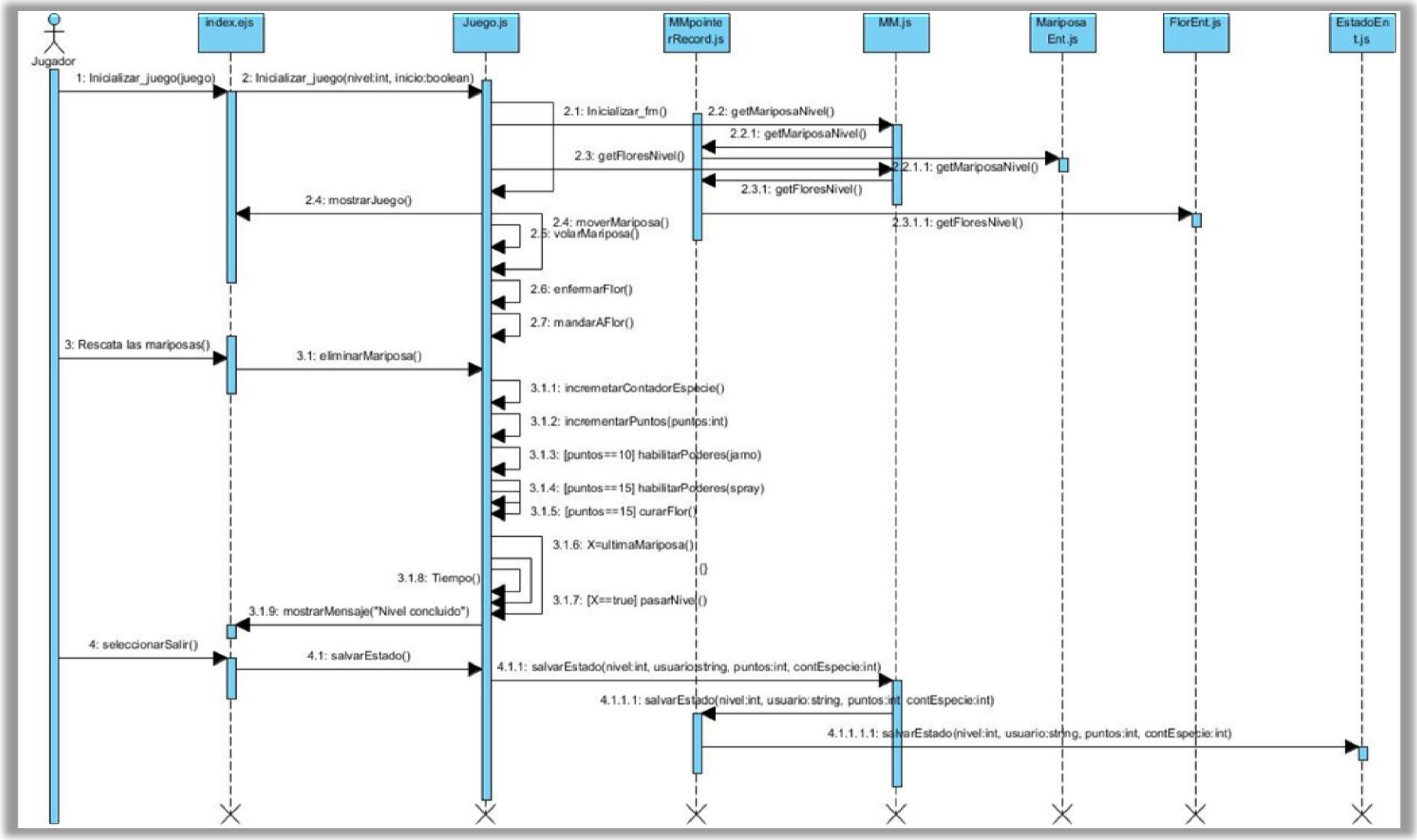


Figura 11: Diagrama de secuencia del juego "Mis maripositas".

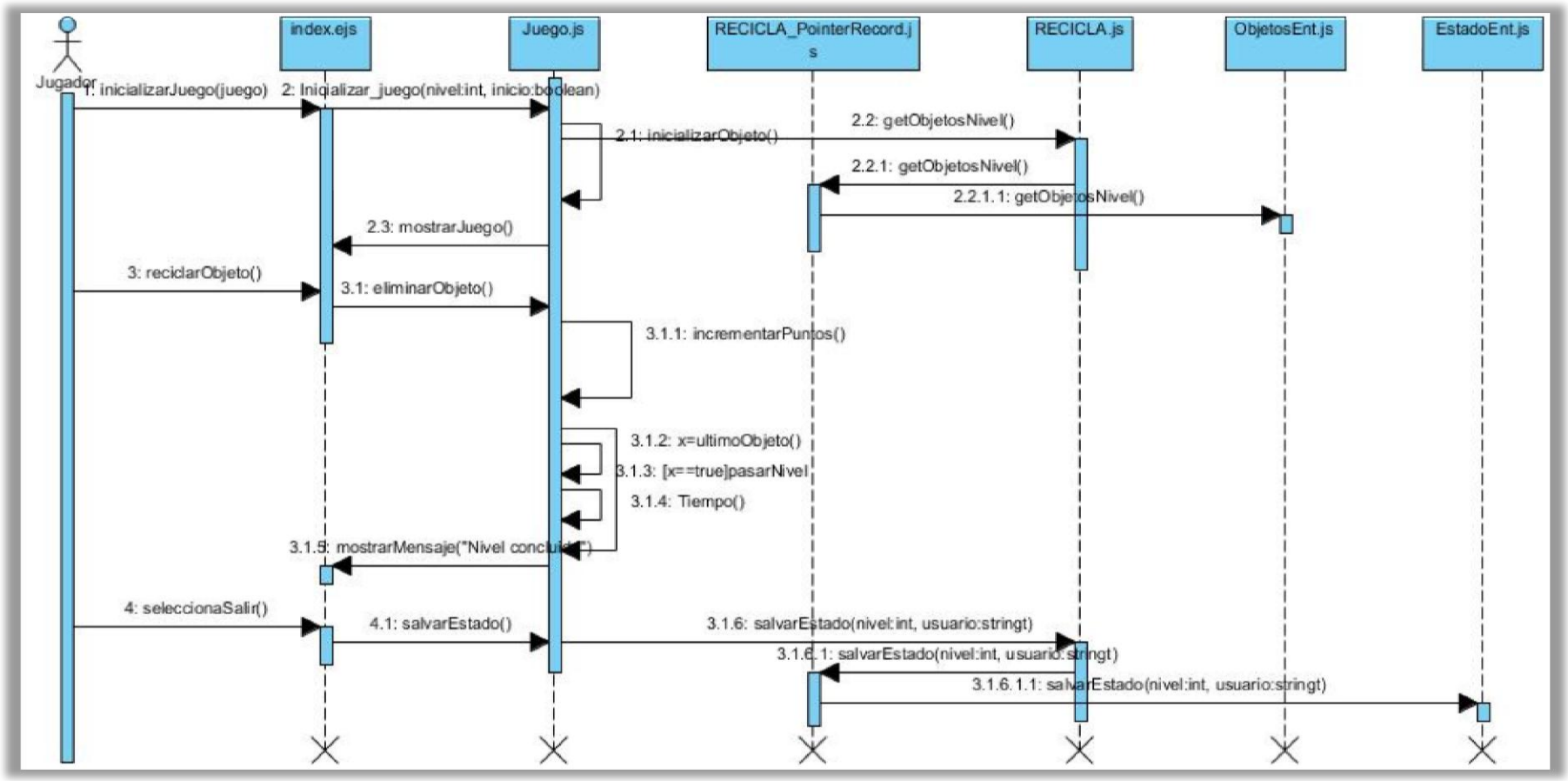


Figura 12: Diagrama de secuencia del juego "Recicla".

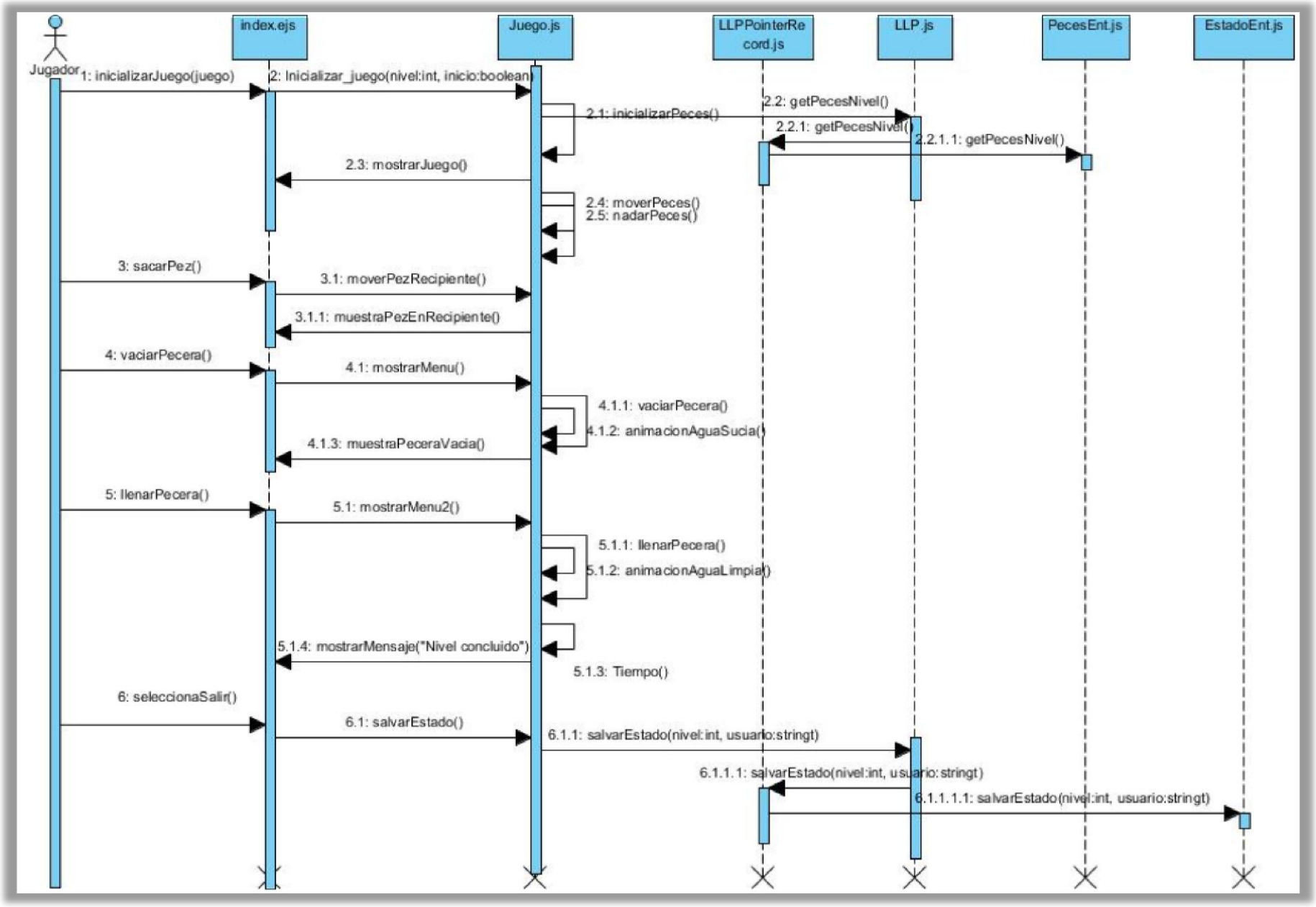


Figura 13: Diagrama de secuencia del juego "Limpiar la pecera".

3.3. *Definiciones de diseño*

3.3.1 Patrón arquitectónico

Los patrones arquitectónicos expresan el esquema de organización estructural fundamental para sistemas de *software*. Provee un conjunto de subsistemas predefinidos, especifica sus responsabilidades e incluye reglas y pautas para la organización de las relaciones entre ellos. Son plantillas para arquitecturas de *software* concretas, que especifican las propiedades estructurales de una aplicación y tienen un impacto en la arquitectura de subsistemas (37).

Modelo Vista Controlador (MVC)

El patrón MVC es un patrón de arquitectura de *software* encargado de separar la lógica de negocio de la interfaz del usuario y es el más utilizado en aplicaciones Web, ya que facilita la funcionalidad, mantenibilidad y escalabilidad del sistema, de forma simple y sencilla, a la vez que permite no mezclar lenguajes de programación en el mismo código. (38)

MVC divide las aplicaciones en tres niveles de abstracción: el modelo, la vista y el controlador. El modelo representa la lógica de negocios, es el encargado de acceder de forma directa a los datos actuando como intermediario con la base de datos. La Vista es la encargada de mostrar la información al usuario de forma gráfica. Y por último el Controlador, que es el intermediario entre la vista y el modelo, este controla las interacciones del usuario solicitando los datos al modelo y entregándolos a la vista para que lo presente al usuario, de forma humanamente legible (ver figura 14). (38)

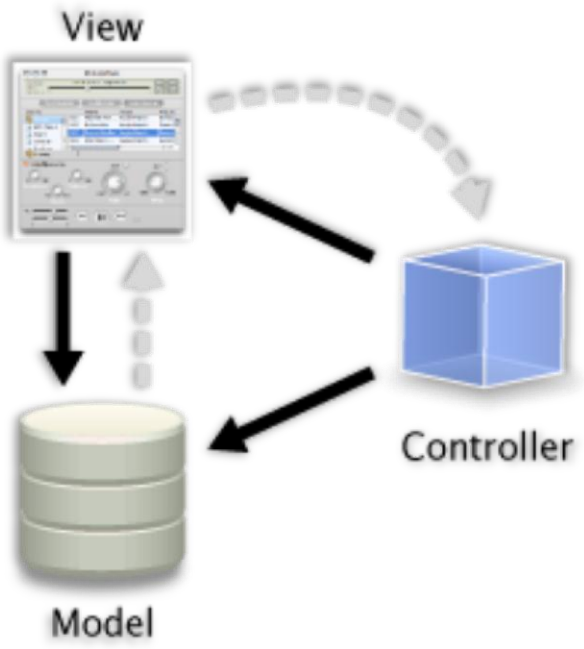


Figura 14: Patrón arquitectónico Modelo Vista Controlador.

Ventajas del patrón arquitectónico MVC (38):

- Clara separación entre interfaz, lógica de negocio y de presentación.
- Sencillez para crear distintas representaciones de los datos.
- Reutilización de los componentes.
- Simplicidad en el mantenimiento de los sistemas.
- Facilidad para desarrollar prototipos rápidos.
- Los desarrollos suelen ser más escalables.

3.3.2 Patrones GRASP

Un patrón es una descripción de un problema y su solución, que recibe un nombre y puede emplearse en otros contextos; en teoría, indica la manera de utilizarlo en circunstancias diversas. Muchos patrones ofrecen orientación sobre como asignar las responsabilidades a los objetos ante determinada categoría de problemas. (36)

GRASP es un acrónimo que significa *General Responsibility Assignment Patterns* (patrones generales de *software* para asignar responsabilidades). El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el *software* orientado a objetos. (36)

- **Experto:** Experto es un patrón que se usa al asignar responsabilidades y contribuye a mantener el encapsulamiento.(36) El patrón se pone de manifiesto en los tres juegos a la hora de modelar los elementos principales de los mismos: mariposas, flores, peces y objetos. Todos ellos fueron implementados en clases independientes que contienen la lógica asociada al movimiento, selección y demás acciones que se pueden realizar sobre ellos. Cada instancia utiliza su propia información para llevar a cabo sus tareas.
- **Creador:** Permite identificar quién debe ser el responsable de la instanciación de nuevos objetos. (36) En el juego “Limpiar la pecera” la clase Juego es la que contiene los datos necesarios para crear los objetos de la clase Pez, utilizándolos para completar la información asociada a cada nivel del juego.
- **Alta cohesión y bajo acoplamiento:** Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas; el bajo acoplamiento promueve la implementación de clases lo menos interdependientes posible para que en caso de cambio en alguna de ellas repercuta lo menos posible en el resto. (36) En el juego “Recicla” la clase Objeto modela los objetos de los diferentes materiales, la misma contiene toda la lógica de las acciones que se pueden realizar sobre ellos. Si se realiza algún cambio en las funcionalidades asociadas a los objetos (seleccionarlos con doble clic, resaltar objetos de un material específico, cambiar la frecuencia con que aparecen los objetos que bonifican) no afecta la implementación del resto de la lógica del juego. Entre las clases Juego y Objeto se evidencia el bajo acoplamiento y en Objeto se evidencia la alta cohesión.
- **Controlador:** Este patrón sugiere dividir los elementos del sistema en el mayor número de controladores para aumentar la cohesión y disminuir el acoplamiento. (36) En el juego “Mis maripositas” es la encargada del manejo de las características asociadas a cada nivel del juego como son: mostrar información de usuario, cambio de los fondos, bonificaciones, tiempo y puntuación. Para completar las funcionalidades del posicionamiento de las flores y el vuelo de las mariposas, se implementaron las clases Mariposa y Flor.

3.3.3 Estilos de codificación

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. La mantenibilidad del código es la facilidad con que el sistema de *software* puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores o mejorar el rendimiento. El mejor método para asegurarse que un equipo de programadores mantenga un código de calidad es establecer un estándar de codificación sobre el que se efectuarán revisiones del código rutinarias. Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad del *software* y para obtener

un buen rendimiento. (39) Teniendo en cuenta lo antes expuesto el equipo de desarrollo al cual pertenece la solución propuesta acordó como estándares de codificación CamelCase e idiomatic.js.

CamelCase es la práctica de escribir frases o palabras compuestas eliminando los espacios y poniendo en mayúscula la primera letra de cada palabra. Existen dos tipos de CamelCase (40):

- *UpperCamelCase*: cuando la primera letra de todas está escrita en mayúscula. Ejemplo: EjemploDeUpperCamelCase.
- *LowerCamelCase*: cuando la primera letra de todas está escrita en minúscula. Ejemplo: ejemploDeLowerCamelCase.

Para los nombres de las clases se utilizará *upperCamelCase* y para los nombres de las variables y los métodos *lowerCamelCase*. Además, no debe haber variables ni funciones globales en el código.

idiomatic.js es un manual escrito por varios autores donde se especifican una serie de principios y buenas prácticas para guiar la programación en el lenguaje JavaScript. A continuación se muestran algunos de estos consejos a tener en cuenta (41):

- Nunca mezclar espacios y tabulaciones.
- Eliminar el espacio en blanco del fin de línea.
- Evitar códigos con nombres de variables pobres o poco descriptivos.
- Hacer *return* tempranos para mejorar la legibilidad del código.

Conclusiones del capítulo

- El modelado de los diagramas de clases del diseño y los diagramas de secuencias proporcionó una mejor comprensión del sistema a la hora de ser implementado.
- El uso de los patrones arquitectónico Modelo Vista Controlador y de diseño GRASP y CamelCase e idiomatic.js como estilos de codificación evidenció la presencia de buenas prácticas de programación en el desarrollo de los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas”.

Capítulo 4: Implementación y pruebas

Introducción

Los artefactos generados en el diseño son la principal entrada en el flujo de trabajo implementación. En esta etapa se describe el *software* en términos de componentes (ficheros de código fuente y binario, ejecutables y similares). En el presente capítulo se describen cada uno de dichos componentes y la estrategia de prueba utilizada para verificar el resultado de la implementación.

4.1. *Implementación*

4.1.1 Modelo de despliegue

El modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. (15) A continuación se representa el dicho modelo correspondiente a la colección de juegos educativos MundoClick.

Figura 15: Diagrama de despliegue para MundoClick.

4.1.2 Diagrama de componentes

Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases en el modelo de diseño. (15) Este diagrama se encarga de relacionar los componentes de software en un sistema. A continuación se muestran los diagramas de componentes para la implementación de cada uno de los juegos “Mis maripositas”, “Recicla” y “Limpiar la pecera” respectivamente:

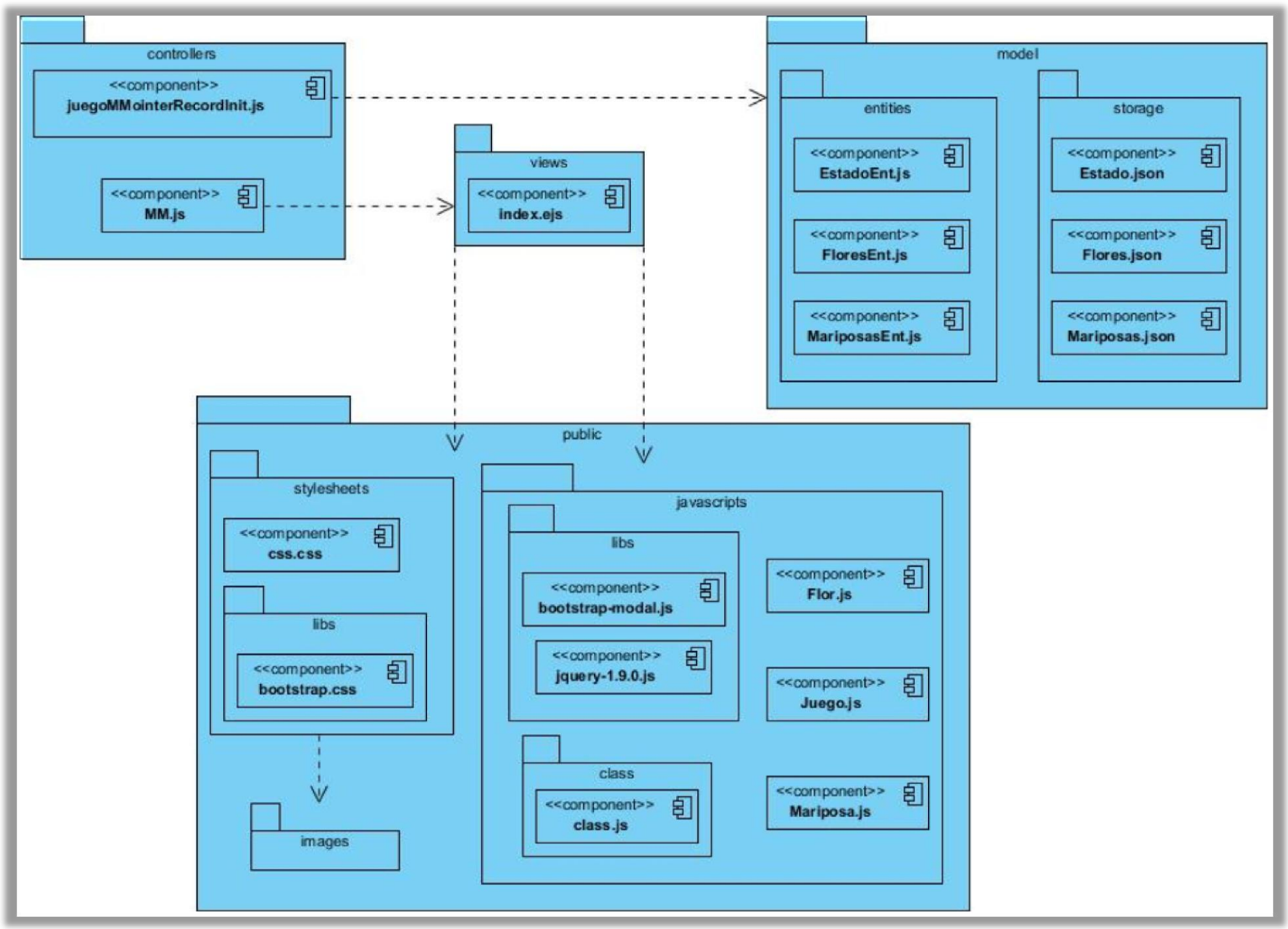


Figura 16: Diagrama de componentes para el juego "Mis maripositas".

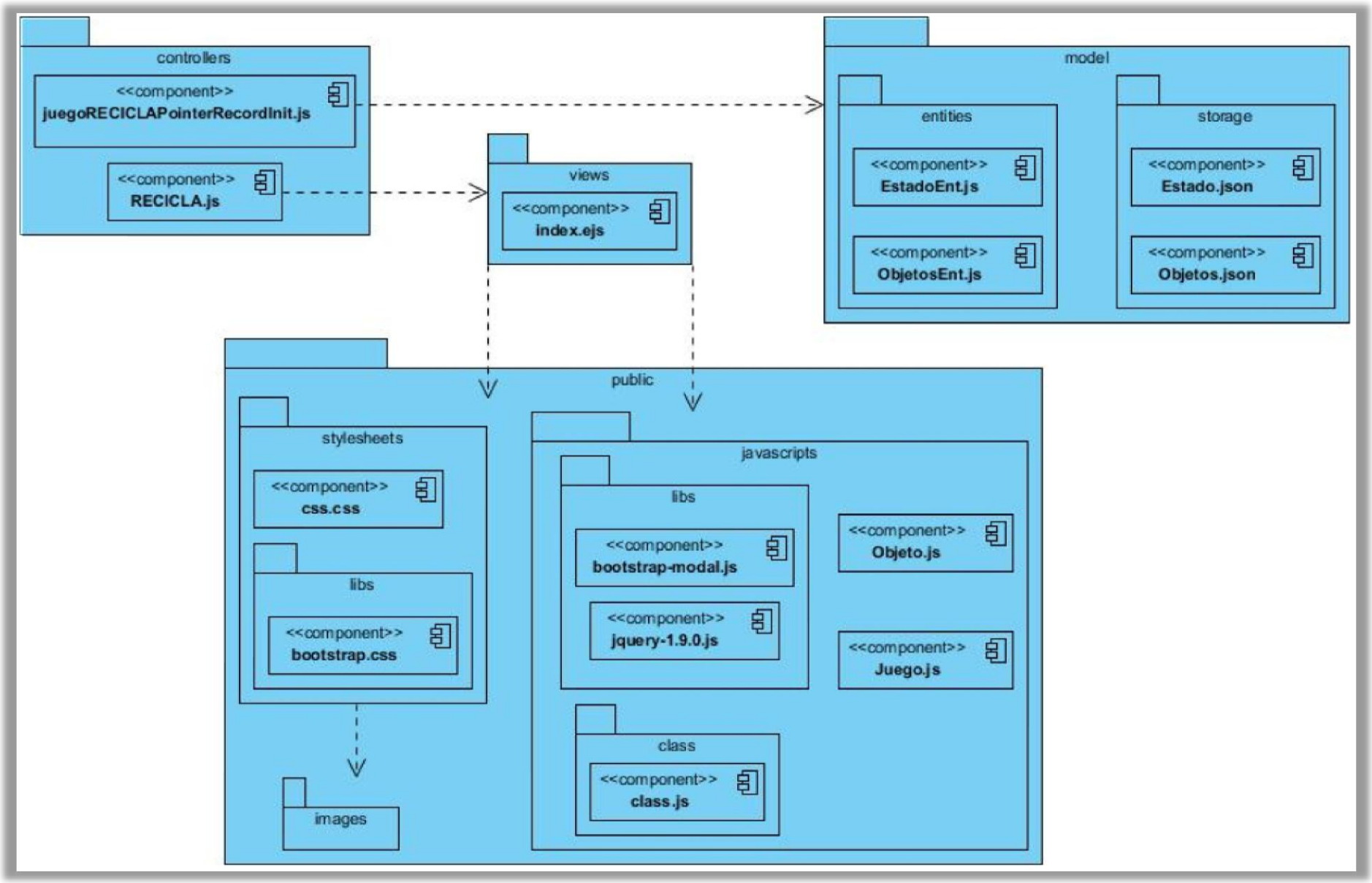


Figura 17: Diagrama de componentes para el juego "Recicla".

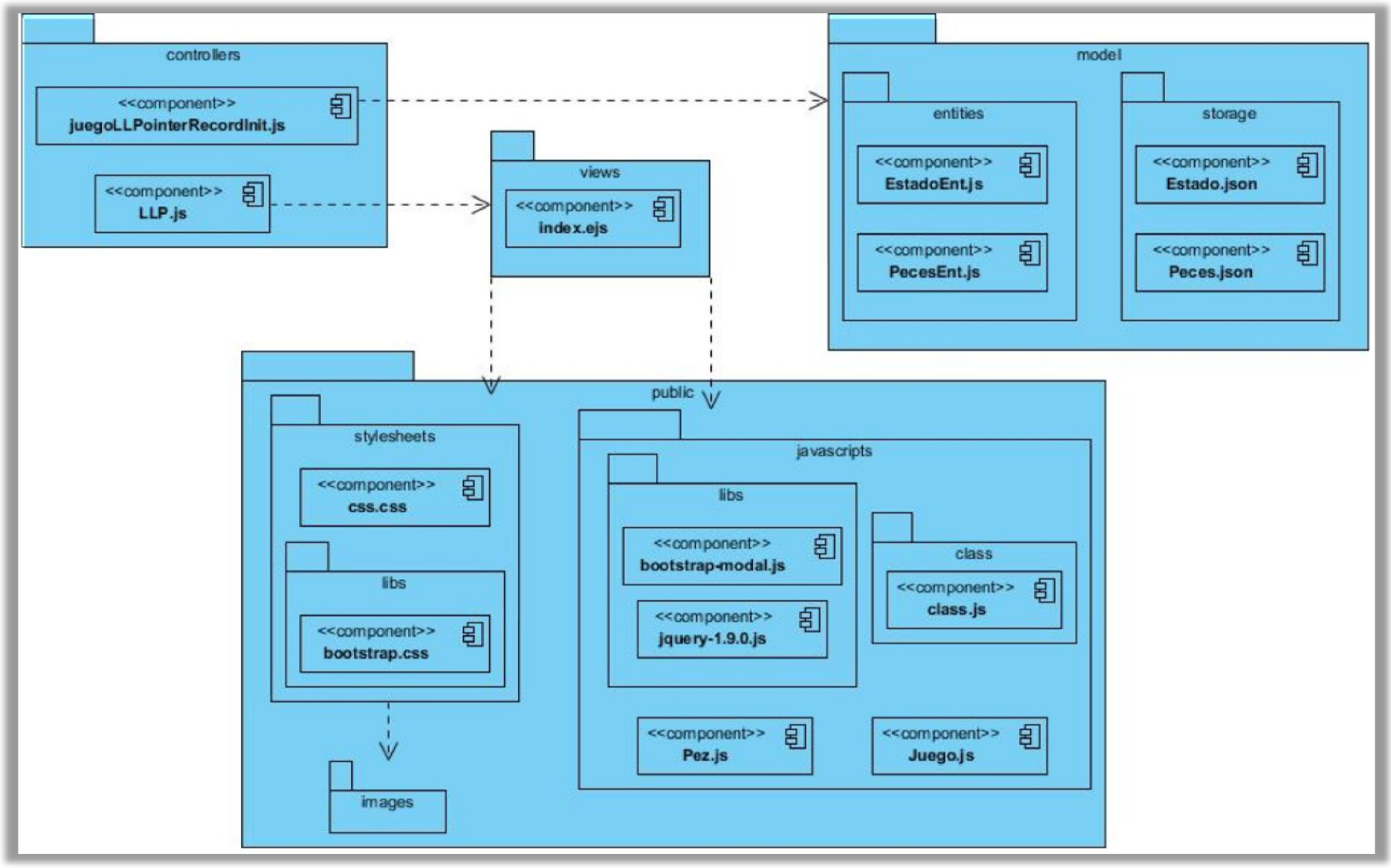


Figura 18: Diagrama de componentes para el juego "Limpiar la pecera".

4.2. *Prueba*

Se puede definir pruebas de *software* como una actividad en la cual un sistema o componente es ejecutado bajo unas condiciones específicas, los resultados son observados, registrados y una evaluación es hecha de algún aspecto del sistema o componente. (15)

Con el objetivo de garantizar la calidad de los productos desarrollados se hizo necesario verificar su funcionamiento. En el proceso de desarrollo de *software* se define una etapa de pruebas con los siguientes objetivos (15):

- Planificar las pruebas necesarias en cada iteración, incluyendo las pruebas de integración y las pruebas de sistema.
- Diseñar e implementar las pruebas creando los casos de pruebas que especifican qué probar, creando los procedimientos de pruebas que especifican cómo realizar las pruebas y creando componentes de prueba ejecutables para automatizar las pruebas.

- Realizar las diferentes pruebas y manejar los resultados de cada prueba sistemáticamente. Las construcciones en las que se detectan defectos son probadas de nuevo y posiblemente devueltas a otro flujo de trabajo, como diseño o implementación, de forma que los defectos importantes puedan ser arreglados.

Para la elaboración de una adecuada estrategia de prueba es necesario realizar un estudio a los distintos conceptos y representaciones de los mismos en esta área de la ingeniería de *software*. A continuación se muestran los resultados del análisis realizado por el equipo de desarrollo con las características específicas de la estrategia trazada.

4.2.1 Niveles de Prueba

La prueba es aplicada para diferentes tipos de objetivos, en diferentes escenarios o niveles de trabajo. Se distinguen los siguientes niveles de pruebas (42):

- **Prueba de integración:** es ejecutada para asegurar que los componentes en el modelo de implementación operen correctamente cuando son combinados para ejecutar un caso de uso. Se prueba un paquete o un conjunto de paquetes del modelo de implementación. Estas pruebas descubren errores en las especificaciones de las interfaces de los paquetes.
- **Prueba de Sistema:** son las pruebas que se hacen cuando el *software* está funcionando como un todo. En este nivel existen una gran variedad de pruebas entre las que se encuentran las de recuperación, seguridad, resistencia y rendimiento.

4.2.2 Tipos de pruebas

Dentro de cada nivel de prueba se engloba una técnica de prueba específica según los atributos de calidad que se deseen verificar con las pruebas al *software*. De acuerdo con el nivel se utilizan varios tipos pruebas, a continuación se exponen varios de ellos:

- **Pruebas de funcionalidad:** se realiza con el propósito de verificar el cumplimiento de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados.
- **Pruebas de regresión**: se realizan para asegurar que cuando una no conformidad encontrada en el sistema ha sido corregida, ninguna de las funcionalidades liberadas previamente falla como resultado de las correcciones, o que las características nuevamente agregadas no han creado conflicto con las versiones anteriores del *software*.

4.2.3 Métodos de pruebas

Las pruebas de **caja negra** o también llamadas pruebas de comportamiento se centran en los requisitos funcionales del *software*, estas tratan de encontrar errores en las siguientes categorías: funciones incorrectas o ausentes, errores de interfaz, errores en estructuras de datos o en accesos a bases de datos externas, errores de rendimiento y errores de inicialización y de terminación. (42)

La **partición equivalente** es una técnica de prueba de caja negra que divide el dominio de entrada de un programa en clases de datos a partir de las cuales pueden derivarse casos de prueba. La partición equivalente se esfuerza por definir un caso de prueba que descubra ciertas clases de errores, reduciendo así el número total de casos de prueba que deben desarrollarse. (42)

4.3. *Resultados de las pruebas*

La estrategia de prueba trazada consistió en abarcar los niveles integración y sistema. Las pruebas a nivel de integración fueron utilizadas para comprobar el intercambio de datos entre componentes, mientas las pruebas del nivel de sistema se realizaron una vez los juegos fueron integrados a la colección de juegos educativos MundoClick. En el nivel de integración fueron realizadas solo pruebas de funcionalidad mientras que en el nivel sistema se realizaron ambos tipos de prueba para verificar el funcionamiento de los juegos integrados a la colección. La tabla número 4 muestra de manera resumida los aspectos esenciales de la estrategia de prueba que guió el proceso de validación de las aplicaciones.

Nivel de prueba	Tipo de prueba	Método de prueba	Técnica
Integración	Funcionalidad	Caja Negra	Partición equivalente
Sistema	Funcionalidad Regresión	Caja Negra	Partición equivalente

Tabla 4: Estrategia de prueba

El seguimiento de esta estrategia permitió reconocer no conformidades de los juegos durante las tres iteraciones de prueba que fueron definidas por el equipo de desarrollo. La tabla número 5 refleja las cantidades de no conformidades detectadas durante las pruebas.

	Cantidad de No Conformidades		
Juego	**Iteración 1**	**Iteración 2**	**Iteración 3**
Mis maripositas	17	8	0
Recicla	15	6	1
Limpiar la pecera	12	4	0

Tabla 5: Resultado de las pruebas. No conformidades según iteraciones de prueba.

Las no conformidades detectadas durante la primera iteración se relacionaban con la activación de las bonificaciones del juego, los cambios de nivel y el posicionamiento de los elementos en las páginas web. En la segunda iteración las pruebas estuvieron enfocadas en la integración de los juegos a la colección MundoClick y, aunque fueron solucionadas algunas de las no conformidades detectadas en la fase anterior surgieron otras tales como: problemas con el diseño y el acceso a datos para el funcionamiento de los juegos. Solucionados los errores detectados se realizó la tercera iteración de pruebas donde se comprobó la solución de la mayoría de las no conformidades anteriormente detectadas.

Conclusiones del capítulo

- La obtención del modelo de implementación posibilitó representar la realización física de los elementos del diseño.
- La implementación de cada caso de uso, componente y procedimiento de prueba estuvo enfocada en los requisitos de mayor impacto en las habilidades que propone desarrollar cada juego en los niños.

- El estudio realizado sobre el área de las pruebas de *software* permitió la elaboración de una estrategia para la validación de las soluciones obtenidas, demostrando el cumplimiento de las especificaciones de cada una de ellas.

Conclusiones generales

El presente documento es resultado de la investigación iniciada con el objetivo de desarrollar los juegos educativos “Recicla”, “Limpiar la pecera” y “Mis maripositas” para la colección de juegos educativos MundoClick. Al término de la implementación de estas soluciones, el equipo de desarrollo definió a las siguientes conclusiones:

- El estudio detallado de aplicaciones similares a las que se desarrollan en la investigación demostró la necesidad de implementar los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas” para el colección de juegos educativos MundoClick.
- La obtención de los artefactos ingenieriles, definidos en cada una de los flujos de trabajo de la metodología RUP, permitió la ejecución de un proceso de desarrollo bien documentado en cada una de sus fases.
- Como principal resultado de la investigación se obtuvo la implementación de los juegos “Recicla”, “Limpiar la pecera” y “Mis maripositas” integrados en su totalidad a la colección de juegos educativos MundoClick.
- La estrategia de pruebas diseñada para los juegos permitió mejorar aspectos relacionados con el diseño y el funcionamiento de los mismos, logrando que el resultado final esté acorde con los guiones que describen cada uno de ellos.

Recomendaciones

A modo de recomendación, se propone generalizar la propuesta desarrollada en los centros del país donde la tecnología disponible lo permita.

Bibliografía

1. **Marquès, Pere.** *El software educativo.* Barcelona, España : s.n.

2. **Ricardo, Carlos Expósito.** *Conceptos generales de Software.* 2009.

3. **Pivec, Maja, Koskinen, Tapio y Tarín, Lluís.** eLearning Papers. [En línea] [Citado el: 16 de Enero de 2013.] http://www.elearningpapers.eu/es/paper/aprendizaje-basado-en-juegos-nuevas-pr-cticas-nuevas-aulas.

4. **Universidad de las Ciencias Informáticas.** Universidad de las Ciencias Informáticas. [En línea] [Citado el: 20 de Novienmbre de 2012.] http://www.uci.cu/mision.

5. **Hernández León, Rolando Alfredo y Coello González, Saida.** *El proceso de la Investigación Científica.* La Habana : Editorial Universitaria del Ministerio de Educación Superior, 2011.

6. *Software educativos.* **Gómez Martínez, Freddy, Vidal Ledo, María y Ruiz Piedra, Alina.** 1, La Habana : Editorial de Ciencias Médicas, 2009, Vol. 24. ISSN 0864-2141.

7. **Ramos Pérez, Lourdes, y otros, y otros.** ¿Software educativo, hipermedia o entorno educativo? [En línea] 28 de Septiembre de 2008. Dispobible en: http://bvs.sld.cu/revistas/aci/vol18_4_08/aci61008.htm.

8. **Huertas, Catalina Ponce.** *El Juego Como Recurso Educativo.* Granada : s.n., 2009. 18005 .

9. *La Importancia del Juego.* **Rodríguez., Esmeralda Jiménez.** 26, Sevilla : s.n., 2006, Vol. III . SE – 3792 - 06 .

10. **Juego "Atrapar Mariposas con Burbujas".** King Online Game. *King Online Game.* [En línea] [Citado el: 6 de Marzo de 2013.] http://www.kingonlinegame.com/jugar-online-39450-Atrapar-Mariposas-con-Burbujas.html.

11. **Juego A Pescar!** Kiddia. *Kiddia.* [En línea] [Citado el: 6 de Marzo de 2013.] http://www.kiddia.org/juego-flash-0-5-anos-pesca.

12. **Juego "Vamos a reciclar".** Tu Discovery Kids. *Tu Discovery Kids.* [En línea] [Citado el: 6 de Marzo de 2013.] http://www.tudiscoverykids.com/juegos/vamos-a-reciclar/.

13. **Lic. Liana Isabel Araujo Pérez1, Lic. Héctor Matías González1, Ing. Osdalme Fuentes Colina1, Ing. Mailyn Cabrera Torres1, MSc. Ismael Armando Nodarse Mora1, Ing. Ana María Álvarez Valdés1, Ing. Osmany Montes De Oca Rodríguez1 y otros.** *Colección de software educativo multiplataforma "La caja mágica".* La Habana, Cuba : s.n., 2011.

14. **Pérez, Isaías Carrillo y Pérez González, Rodrigo.** *Metodología de desarrollo de software.* 2008.

15. **Jacobson, Ivar, Booch, Grady y Rumbagh, James.** *El Proceso Unificado de Desarrollo de Software.* Madrid : Pearson Educación, 2000.

16. **Jacbson, Ivar, Rumbagh, James y Booch, Grady.** *El Lenguaje Unificado de Modelado. .* 1999.

17. **Orallo, Enrique Hernández.** *El Lenguaje Unificado de Modelado(UML).*

18. **Gómez, Ruth Priscila Laderos y del Juncal Huerta, Jorge Luis.** *Herramientas CASE.* 2007.

19. **Sierra, María.** *Trabajando con Visual Paradigm for UML.* Cantabria : Universidad de Cantabria – Facultad de Ciencias.

20. **Definición de lenguaje de programacion.** definicion.org. *definicion.org.* [En línea] [Citado el: 18 de Febrero de 2013.] http://www.definicion.org/lenguaje-de-programacion.

21. **Chuck Musciano, Bill Kemedy.** *HTML La Guía Completa.* Mexico : s.n., 1999. 06450 .

22. **Completada la definición de HTML5, el W3C pasa a las pruebas de interoperabilidad y rendimiento.** World Wide WebConsortium (W3C) . *World Wide WebConsortium (W3C) .* [En línea] 17 de Diciembre de 2012. [Citado el: 29 de Abril de 2013.] http://www.w3c.es/Prensa/2012/nota20121217_html5.

23. **Pérez, Javier Eguíluz.** *Introducción a CCS.* 2008.

24. **Alvarez, Miguel Angel.** *Manual de CSS 3 .* 2007.

25. **Pérez, Javier Eguíluz.** *Introducción a JavaScript JavaScript .* 2008.

26. **Gutiérrez., Javier J.** *¿Qué es un framework web?*

27. **Potencier, Fabien y Zaninotto, François.** *Symfony 1.2, la guía definitiva.* 2008.

28. **Framework, Twitter Bootstrap!** Tutorial Bootstrap. *Tutorial Bootstrap.* [En línea] [Citado el: 1 de Junio de 2013.] http://internoma.github.io/tutorial-bootstrap/.

29. **Bootstrap.** GENBETA:dev Desarrollo y software. *GENBETA:dev Desarrollo y software.* [En línea] 16 de Junio de 2012. [Citado el: 2 de Junio de 2013.] http://www.genbetadev.com/frameworks/bootstrap.

30. **Espinoza, Margareth.** [En línea] [Citado el: 8 de Enero de 2013.] http://margaespinoza.com/?p=315.

31. **Abernethy, Michael.** IBM DeveloperWorks. *IBM DeveloperWorks.* [En línea] 14 de Junio de 2011. [Citado el: 16 de Enero de 2013.] http://www.ibm.com/developerworks/ssa/opensource/library/os-nodejs/.

32. **Brandon Benvie, Morteza Milani.** AppJS. *AppJS.* [En línea] 2012. [Citado el: 16 de Enero de 2013.] http://appjs.org/.

33. **Entorno de Desarrollo Integrado (IDE). Definición .** Desarrollo Móvil Multiplataforma. *Desarrollo Móvil Multiplataforma.* [En línea] 26 de Agosto de 2012. [Citado el: 15 de Enero de 2013.] http://desarrollomovilmultiplataforma.blogspot.com/2012/08/aspectos-teoricos-entorno-de-desarrollo.html.

34. **NetBeans IDE - The Smarter and Faster Way to Code .** NetBeans. *NetBeans.* [En línea] [Citado el: 12 de Febrero de 2013.] http://netbeans.org/features/.

35. **Jacobson, Ivar, Booch, Grady y Rumbagh, James.** *El Proceso Unificado de Desarrollo de Software.* Madrid : Pearson Addison, 2000.

36. **Larman, Craig.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* 1999.

37. **Camacho, Erika, Cardeso, Fabio y Nuñez, Gabriel.** *Arquitectura de software.* 2004.

38. **Bahit, Eugenia.** *El paradigma de la Programación Orientada a Objetos en PHP y el patrón de arquitectura de Software MVC.* 2011.

39. **Revisiones de código y estándares de codificación.** MicroSoft Developer Network. *MicroSoft Developer Network.* [En línea] [Citado el: 29 de Marzo de 2013.] http://msdn.microsoft.com/es-es/library/aa291591%28v=vs.71%29.aspx.

40. **CamelCase definition.** DicLib.com. *DicLib.com.* [En línea] [Citado el: 29 de Marzo de 2013.] http://www.diclib.com/CamelCase%20/show/en/es_wiki_10/C/7724/660/0/12/30673.

41. **Principios para escribir JavaScript consistente e idiomático.** GitHub. *GitHub.* [En línea] [Citado el: 5 de Junio de 2013.] https://github.com/rwldrn/idiomatic.js/tree/master/translations/es_ES.

42. **Pressman, Roger A.** *Ingeniería de software: Un enfoque práctico.* 5. 2005. pág. 404. 970105473.

Anexos

Anexo 1: Descripción del caso de uso Cambiar nivel del juego "Mis maripositas".

Caso de uso	Bonificar jugador	
Actores	Jugador	
Resumen	El caso de uso inicia cuando el jugador obtiene una cantidad de puntos para ser bonificado. Termina cuando se otorga la bonificación.	
Precondiciones	El jugador debe alcanzar la cantidad de puntos necesarios para ser bonificado.	
Postcondiciones	-	
Prioridad	Crítico	
Flujo normal de eventos		
Acción del actor		**Respuesta del sistema**
		1. Premia al jugador con las siguientes bonificaciones: -Jamo: **ir a la sección "Jamo"** -Spray: **ir a la sección "Spray"**
Sección 1 "Jamo": Flujo Normal de Eventos		
Acción del actor		**Respuesta del sistema**
1. Obtiene un total de 10 puntos.		2. Premia al jugador con un jamo más grande.
		3. Muestra el premio obtenido en la parte inferior de la pantalla. 4. Finaliza el caso de uso.
Sección 2 "Spray": Flujo Normal de Eventos		
Acción del actor		**Respuesta del sistema**
1. Obtiene un total de 15 puntos.		2. Premia al jugador con un spray que le permite curar las flores enfermas.
		3. Muestra el premio obtenido en la parte inferior de la pantalla. 4. Finaliza el caso de uso.
Requisitos no funcionales		

Relaciones	**CU Incluidos**	
	CU Extendidos	Bonificar jugador en el CU Iniciar juego.

Tabla 6: Descripción textual del CU Bonificar jugador del juego "Mis maripositas".

Anexo 2: Descripción del caso de uso Cambiar nivel del juego "Mis maripositas".

Caso de uso	Cambiar nivel	
Actores	Jugador	
Resumen	El caso de uso inicia cuando el jugador rescata todas las mariposas. Termina cuando pasa de nivel.	
Precondiciones	**-**	
Postcondiciones	**-**	
Prioridad	Crítico	
Flujo normal de eventos		
Acción del actor	**Respuesta del sistema**	
1. Rescata las mariposas.	2. Muestra una ventana informando al jugador que cambiará para el siguiente nivel. 3. Termina el caso de uso.	
Flujo Alterno: "Tiempo agotado"1.a		
1. Llega al tiempo límite sin rescatar todas las mariposas.	2. Muestra una ventana informando al jugador que su tiempo se ha agotado.	
Flujo Alterno: "Ultimo nivel"1.b		
1. Rescata las mariposas.	2. Muestra una ventana felicitando al jugador por haber ganado el juego.	
Requisitos no funcionales		
Relaciones	**CU Incluidos**	Cambiar nivel en el CU Iniciar juego.
	CU Extendidos	

Tabla 7: Descripción textual del CU Cambiar nivel del juego "Mis maripositas".

Anexo 3: Descripción del caso de uso Terminar juego del juego "Mis maripositas".

Caso de uso	Terminar juego	
Actores	Jugador	
Resumen	El caso de uso inicia cuando el jugador selecciona la opción Salir. Termina cuando se cierra el juego.	
Precondiciones	**-**	
Postcondiciones	Se termina el juego	
Prioridad	Crítico	
Flujo normal de eventos		
Acción del actor		**Respuesta del sistema**
1. Presiona el botón “Salir”.		2. El sistema finaliza el juego. 3. El sistema guarda la partida. 4. Finaliza el caso de uso.
Requisitos no funcionales		
Relaciones	**CU Incluidos**	
	CU Extendidos	

Tabla8: Descripción textual del CU Terminar juego del juego "Mis maripositas”.

Anexo 4: Descripción del caso de uso Bonificar jugador del juego "Recicla”.

Caso de uso	Bonificar jugador
Actores	Jugador
Resumen	El caso de uso inicia cuando el jugador recicla un objeto resaltado y termina cuando se muestra la bonificación en la pantalla.
Precondiciones	El jugador debe de haber reciclado un objeto resaltado.
Poscondiciones	**-**
Prioridad	Crítico
Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1. Recicla un objeto resaltado.	2. Premia al jugador con una lupa que le brinda pistas en el juego. 3. Muestra el premio obtenido en la parte

		inferior de la pantalla. 4. Termina el caso de uso.
Requisitos no funcionales		
Relaciones	**CU Incluidos**	
	CU Extendidos	Bonificar jugador en el CU Crear entorno del juego.

Tabla 9: Descripción textual del CU Bonificar jugador del juego "Recicla".

Anexo 5: Descripción del caso de uso Señalar el tipo de objeto del juego "Recicla".

Caso de uso	Señalar el tipo de objeto	
Actores	Jugador	
Resumen	El caso de uso inicia cuando el jugador señala un objeto. Termina cuando este objeto se resalta.	
Precondiciones	**-**	
Poscondiciones	**-**	
Prioridad	Crítico	
Flujo normal de eventos		
Acción del actor		**Respuesta del sistema**
1. Señala un objeto.		2. Resalta el objeto.
		3. Termina el caso de uso.
Requisitos no funcionales		
Relaciones	**CU Incluidos**	Señalar el tipo de objeto en el CU Iniciar juego.
	CU Extendidos	

Tabla 10: Descripción textual del CU Señalar el tipo de objeto del juego "Recicla".

Anexo 6: Descripción del caso de uso Cambiar nivel del juego "Recicla".

Caso de uso	Cambiar nivel
Actores	Jugador
Resumen	El caso de uso inicia cuando el jugador recicla todos los objetos.

	Termina cuando pasa de nivel.	
Precondiciones	**-**	
Poscondiciones	**-**	
Prioridad	Crítico	
Flujo normal de eventos		
Acción del actor		**Respuesta del sistema**
1. Recicla todos los objetos.		2. Muestra una ventana informando al jugador que cambiará para el siguiente nivel. 3. Termina el caso de uso.
Flujo Alterno: "Tiempo agotado"		
1. Llega al tiempo límite sin reciclar todos los objetos.		2. Muestra una ventana informando al jugador que su tiempo se ha agotado.
Flujo Alterno: "Ultimo nivel"		
1. Recicla todos los objetos.		2. Muestra una ventana felicitando al jugador por haber ganado el juego.
Requisitos no funcionales		
Relaciones	**CU Incluidos**	Cambiar nivel en el CU Iniciar juego.
	CU Extendidos	

Tabla11: Descripción textual del CU Cambiar nivel del juego " Recicla".

Anexo 7: Descripción del caso de uso Terminar juego del juego "Recicla".

Caso de uso	Terminar juego	
Actores	Jugador	
Resumen	El caso de uso inicia cuando el jugador selecciona la opción Salir. Termina cuando se cierra el juego.	
Precondiciones		
Poscondiciones	Se termina el juego.	
Prioridad	Crítico	
Flujo normal de eventos		
Acción del actor		**Respuesta del sistema**

1. Presiona el botón “Salir”.		2. Finaliza el juego. 3. Guarda la partida. 4. Termina el caso de uso.
Requisitos no funcionales		
Relaciones	**CU Incluidos**	
	CU Extendidos	

Tabla12: Descripción textual del CU Terminar juego del juego "Recicla”.

Anexo 8: Descripción del caso de uso Cambiar nivel del juego "Limpiar la pecera”.

Caso de uso	Cambiar nivel	
Actores	Jugador	
Resumen	El caso de uso inicia cuando el jugador limpia la pecera. Termina cuando pasa de nivel.	
Precondiciones	**-**	
Postcondiciones	**-**	
Prioridad	Crítico	
Flujo normal de eventos		
Acción del actor		**Respuesta del sistema**
1. Llena la pecera y los peces vuelven a ella.		2. Muestra una ventana informando al jugador que cambiará para el siguiente nivel. 3. Termina el caso de uso
Flujo Alterno: “Tiempo agotado”1.a		
1. Llega al tiempo límite sin limpiar la pecera.		2. Muestra una ventana informando al jugador que su tiempo se ha agotado.
Flujo Alterno: “Ultimo nivel”1.b		
1. Llena la pecera y los peces vuelven a ella.		2. Muestra una ventana felicitando al jugador por haber ganado el juego.
Requisitos no funcionales		
Relaciones	**CU Incluidos**	Cambiar nivel en el CU Crear entorno del juego.

	CU Extendidos	

Tabla13: Descripción textual del CU Cambiar nivel del juego "Limpiar la pecera".

Anexo 9: Descripción del caso de uso Terminar juego del juego "Limpiar la pecera".

Caso de uso	Terminar juego	
Actores	Jugador	
Resumen	El caso de uso inicia cuando el jugador selecciona la opción Salir. Termina cuando se cierra el juego.	
Precondiciones	**-**	
Poscondiciones	Se termina el juego	
Prioridad	Crítico	
Flujo normal de eventos		
Acción del actor	**Respuesta del sistema**	
1. Presiona el botón “Salir”.	2. Finaliza el juego. 3. Guarda la partida. 4. Finaliza el caso de uso.	
Requisitos no funcionales		
Relaciones	**CU Incluidos**	
	CU Extendidos	

Tabla14: Descripción textual del CU Terminar juego del juego "Limpiar la pecera”.

Anexo 10: Guión del juego “Mis maripositas” para la colección de juegos educativos MundoClick.

1. **Historia**

La historia del juego puede ser representada en forma de historieta (*comic*) y cada una de las imágenes puede relacionarse con la siguiente secuencia de sucesos.

- Rosi ha ido de vacaciones a una casa en el campo. Allí quedó impresionada con la cantidad de especies de mariposas que habían por todos lados.
- Rosi se dio cuenta de que los días que regaban las plantas con fertilizantes se enfermaban muchas maripositas.
- Esta niña tan inteligente decidió recoger las mariposas y llevarlas a un lugar seguro: ¡Un jardín de flores hermosas y limpias de fertilizantes!

- Para capturar las mariposas Rosi debe ser veloz pues cada vez que las mariposas prueben el néctar infectado de las flores con fertilizantes irán muriendo poco a poco y no podrá cumplir su meta: ¡Salvar las maripositas para llevarlas a su jardín!

2. Objetivo

Rosi debe ir seleccionando las mariposas que desea llevar de acuerdo con el las indicaciones que se muestren en pantalla para cumplir las diferentes metas de cada nivel. El juego permite desarrollar en los usuarios sus habilidades en el uso de *mouse*.

3. Descripción

Interfaces

- El juego se muestra a pantalla completa para aprovechar todo el espacio de la pantalla. En todas las interfaces del juego se tiene acceso a los elementos más generales de la configuración del mismo.
- En el momento en que se accede a jugar se mostrará una pantalla representando el campo en donde viven las mariposas y las flores de las que se alimentan.
- Algunas flores deben tener algún elemento que le permita identificar a los jugadores que están roseadas con el fertilizante que hace que se enfermen las maripositas.
- Las maripositas van a estar volando por toda la pantalla y aleatoriamente se posarán en las flores para alimentarse.
- Las mariposas deben ser de colores y tamaños diferentes para representar las diferencias entre las especies. Cuando prueben el néctar con fertilizantes se pondrán en blanco y negro (buscando representar la enfermedad). Los colores pueden ser tan diversos como se quiera, pero los tamaños deben ser solo tres: pequeño, mediano y grande.
- En la pantalla del juego el mouse será representado con un “jamo” y las mariposas se cazan dando clic izquierdo sobre ellas.

Secuencia del juego por niveles

- Cada uno de los 10 niveles representa un reto para el jugador sin perder de vista los objetivos y la secuencia lógica de la historia del juego.
- Los primeros niveles están dedicados a que el jugador se familiarice con el juego, por el límite de puntos a alcanzar será ínfimo. Cuando terminen ambos niveles el jugador habrá recolectado al menos dos especies diferentes de mariposas.
- A partir del octavo nivel se incluye el factor tiempo en el juego ya que el jugador ya tiene la destreza suficiente para jugar contra reloj. Para dar continuidad a la historia, puede mostrarse una imagen (parte del *comic* del inicio) que indique que a Rosi se le terminan las vacaciones en el campo y ya no le queda casi tiempo para salvar a las maripositas que le faltan.
- Al terminar estos últimos niveles se muestra el final feliz de la historia (también en forma de comic) donde la niña marcha hacia un nuevo jardín muy hermoso y libera a las maripositas que ha cuidado durante todo el juego.

Puntos y bonos

Para ayudar al jugador a llegar al final del juego le serán otorgados algunos elementos que le facilitará el trabajo en los niveles contiguos. Los puntos se acumulan de la siguiente manera:

- La cantidad máxima en los diez niveles es 99 puntos.
- Cada mariposa sana acumula 1 punto.
- Cada mariposa opacas o en blanco y negro (enfermas) se resta 1punto.
- Al acumular 15 puntos el jugador será premiado con un "*spray*" para plantas infectadas. El "*spray*" aparecerá de tamaño pequeño en una de las esquinas de la pantalla y cuando el jugador lo seleccione el puntero del *mouse* tomará esa forma y podrán hacer clic izquierdo en la flor que desea desinfectar. Permite ser utilizado 3 veces en cada nivel.
- Al acumular 10 puntos el jugador será premiado nuevamente, esta vez con un "jamo" más grande y más rápido que el anterior que utilizará permanentemente durante el resto del juego.

Anexo 11: Guión del juego "Recicla" para la colección de juegos educativos MundoClick.

1. **Historia**

La historia que fundamenta este juego estará encaminada a mostrarles a los niños la importancia del reciclaje y los beneficios que trae para el entorno. Para ello el juego se introduce con una única imagen que contendrá algo de texto para motivar al jugador en con el tema. Ejemplo:

¿Te gusta disfrutar del aire puro y del verde de las plantas que adornan la ciudad?

Para ello debes mantener limpia la ciudad recogiendo todos los desechos que encuentres. Si no sabes cómo hacerlo este juego te enseñará a clasificar los desechos para que sean reciclados y sirvan de materia prima para elaborar nuevos objetos.

2. **Objetivo**

El objetivo del juego es que los niños aprendan sobre el cuidado del medio ambiente a través de la clasificación de objetos reciclables. Desarrolla habilidades en el uso del *mouse*.

3. **Descripción**

Interfaces

- El juego se muestra a pantalla completa para aprovechar todo el espacio de la pantalla. En todas las interfaces del juego se tiene acceso a los elementos más generales de la configuración del mismo.
- En el momento en que se accede a jugar se mostrará una pantalla con una matriz de objetos reciclables y 4 cestos en los cuales serán almacenados.
- El fondo de cada uno de los niveles puede ser representado imágenes que representen diferentes partes de una ciudad cualquiera.

- Las figuras que representan los objetos deben ser lo suficientemente claras como para que el jugador pueda identificar el material de los objetos.
- Para almacenar los objetos el jugador deberá hacer clic encima de alguno de ellos según las indicaciones del nivel.
- Al terminar cada nivel se mostrará una pantalla con el resultado histórico del jugador.

Secuencia del juego por niveles

- Cada uno de los 6 niveles representa un reto para el jugador sin perder de vista los objetivos del juego.
- Los primeros niveles son muy sencillo, solamente aparecerá una matriz con unos cuantos objetos. La matriz estará fija y cuando se recojan los elementos se irán reemplazando por nuevos elementos.
- En el cuarto nivel los objetos de la matriz van a cambiar de posición aleatoriamente cada vez que el jugador haga clic en uno de ellos.
- Una vez terminado este nivel debe mostrarse una imagen con algo de texto que le indique al jugador que ya tiene algunos conocimientos sobre el reciclaje de objetos y que está listo para salir aplicarlos en la vida real.

Puntos y bonos

El jugador no pude avanzar al próximo nivel sin acumular la cantidad mínima de puntos que exige dicho nivel. Para ayudarle a llegar al final del juego le serán otorgados algunos elementos que le facilitará el trabajo. Los puntos se acumulan de la siguiente manera:

- Cada objeto reciclado vale 1 puntos.
- Aleatoriamente al reciclar un objeto detrás de él pude aparecer una “moneda” que acumula 10 puntos adicionales (dicha “moneda” no se recicla).
- En la matriz aparecerán algunos objetos más resaltados que otros. Esto indica que cuando se recoja ese objeto le dará al jugador una “lupa”. La “lupa” sirve para darle pistas al jugador en caso de no saber el material de algún elemento.
- Al finalizar cada nivel, para dar la puntuación final se tomará en cuenta el factor tiempo.

Anexo 12: Guión del juego “Limpiar la pecera” para la el paquete de juegos MundoClick.

El juego consiste en ayudar a limpiar la pecera a un niño desarrollando los siguientes pasos:

- Extraer los peces de la misma y depositarlo en otros recipientes.
- Extraer el agua de la pecera.
- Depositar agua limpia en la misma.
- Colocar nuevamente los peces extraídos inicialmente.

Objetivo

Desarrollar habilidades en el uso del *mouse*.

Inicio

Se muestra una mesa con varios objetos sobre ella: una pecera con varios peces de múltiples tamaños sobre una mesa, dos recipientes (de tamaños diferentes) al lado de la misma y se escucha una voz solicitando la ayuda para limpiar la pecera.

Intermedio

Para depositar los peces en un recipiente determinado se dará clic en el pez y en el recipiente deseado. Hay que indicar que los peces grandes no podrán ser ubicados en el recipiente pequeño (no deben caber). Importante aclarar que:

- Los recipientes tienen un límite que coincide con la cantidad de peces grandes o pequeños existan en la pecera. Ejemplo: si el recipiente grande se llena no se podrán poner más peces, aunque queden en la pecera.

Anexo 13: Interfaz del juego “Mis maripositas”.

Figura 19: Interfaz del juego "Mis maripositas".

Anexo 14: Interfaz del juego "Recicla".

Figura 20: Interfaz del juego "Recicla".

Anexo 15: Interfaz del juego "Recicla".

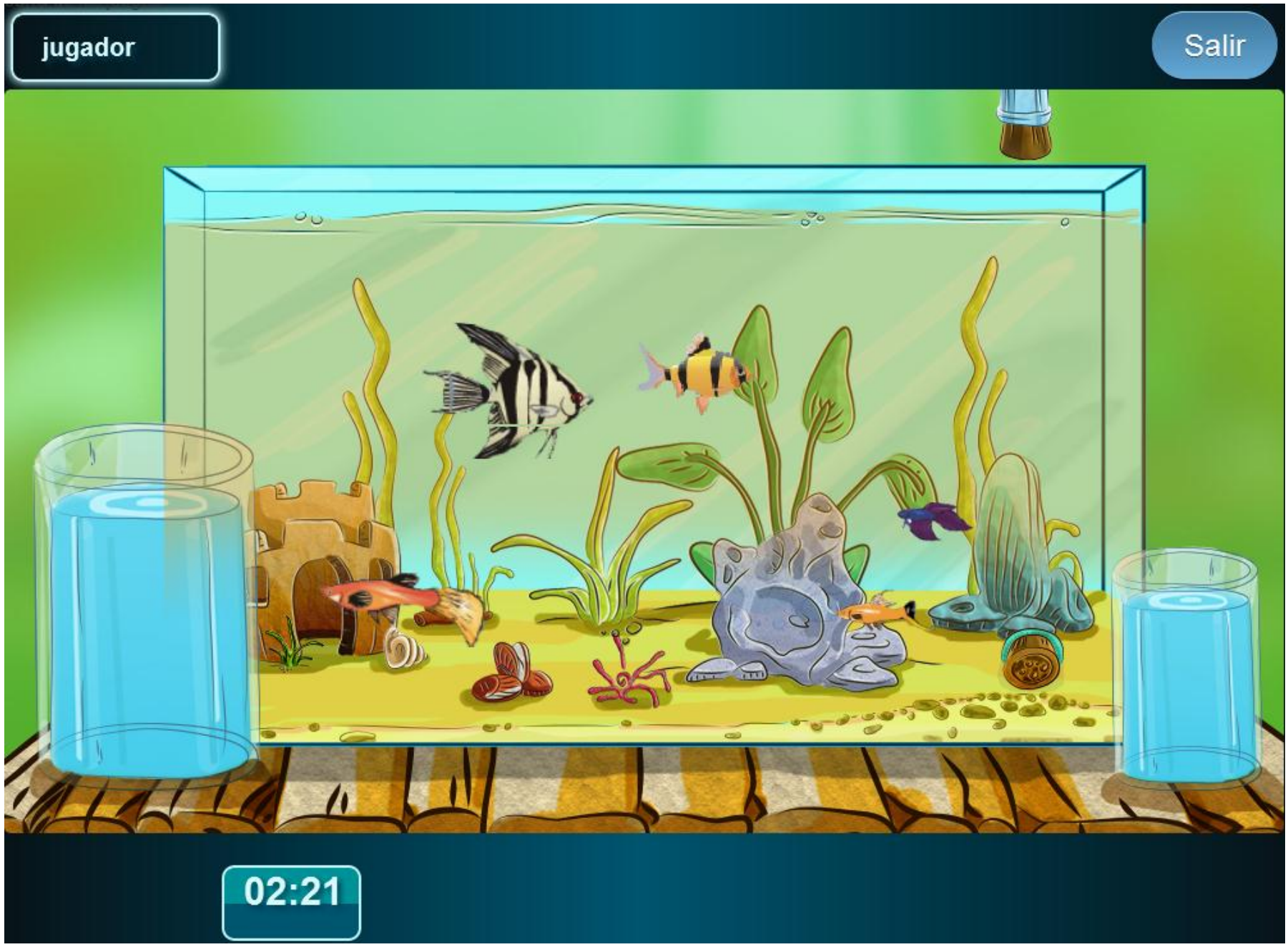


Figura 20: Interfaz del juego "Limpiar la pecera".