Ingeniería de Software

Trabajo Práctico Final

Presentación Final

Proyecto “TBreaker”

|  |  |
| --- | --- |
| Profesor: | Julián Nonino  Martín Miceli |
| Grupo: | Fuplace |
| Integrantes: |  |
|  | Velazquez, Cristian David |

**1 INTRODUCCIÓN**

**1.1** **Propuesta y alcance**

Este documento cubre el Plan de Gestión de las Configuraciones para el proyecto “TBreaker”. El propósito del Plan CM es controlar la configuración de los requisitos, documentos, hardware, software y herramientas utilizados para este proyecto.

SCM es el proceso mediante el cual se identifican métodos y herramientas para controlar el software a lo largo de su desarrollo y uso. Se definirán las distintas tareas que se deben realizar y como estas tareas afectaran al desarrollo del proyecto.

**1.2** **Propósito de la administración de la configuración del software (SCM)**

● Asegurar la consistencia de la información poniendo en práctica la SCM.

● Definir las personas que le dan soporte a las prácticas de SCM.

● Mantener la integridad a los largo de todo el ciclo de vida del producto.

● Informar a grupos y las personas el estado del proyecto.

● Crear un historial del estado anterior y actual del proyecto.

● Mejora de Procesos.

**1.3** **Referencias, abreviaturas y glosario**

|  |  |
| --- | --- |
| **Abreviatura** | **Descripción** |
|  |  |
| SCM | Administración de la configuración del software (Software Configuration Management) |
| GPCM | Administración global de la configuración del proyecto (Global Project Configuration Management) |
| CI | Items de configuración (Configuration items) |
| CM | Adminnistración de la configuración (Configuration Management) |
| CCB | Consejo de control de cambios (Change Control Board) |

**1.4** **Herramientas de Gestión de las Configuraciones**

|  |  |  |
| --- | --- | --- |
| Referencia | Propósito | Link |
| Android Studio | Entorno de desarrollo. | https://developer.android.com/studio/?hl=es-419 |
| Java | Lenguaje de programación. | https[://www.java.com](http://www.java.com/) |
| Git | Software de manejo de versiones. | https[://www.gitscm.com](http://www.gitscm.com/) |
| GitHub | Sistema de control de versiones. | https://github.com/Gabyiacc/UNC-Ing.-de-Software-2018-Fuplace-1-/issues |
| Git Issues | Sistema de seguimiento de errores. | https://github.com/Gabyiacc/UNC-Ing.-de-Software-2018-Fuplace/issues |
| Gradle | Herramienta de automatización. | [http://www.gradle.org](http://www.gradle.org/) |
| Travis | Herramienta de integración. | https://travis-ci.org/Gabyiacc/UNC-Ing.-de-Software-2018-Fuplace |
| Visual Paradigm | Aplicación orientada a la creación de diagramas UML. | https://www.visual-paradigm.com/ |

**2 ROLES Y RESPONSABILIDADES**

**2.1 Administradores de la configuración del proyecto**

La siguiente tabla muestra personas que tomarán el rol de Gerentes de Configuración Global:

|  |  |
| --- | --- |
| Rol | Titular |
| Global PCM | Velazquez, Cristian David  Vignolo, Gabriel Enrique  Losano Quintana, Juan Cruz |

**2.2 Responsabilidades de la administración de la configuración**

La Administración de la configuración del proyecto está a cargo del Global PCM, él es el responsable de actividades como seguimiento de las herramientas, creación de las nuevas ramas, creación del release, etc.

|  |  |
| --- | --- |
| Rol | Responsabilidades |
| GPCM | Posee toda responsabilidad sobre todos los CI. Responsabilidad en la creación de ramas y administración de sus políticas.  Responsabilidad y asistencia sobre el etiquetado y lanzamiento de ramas.  Coordinar actividades del CM en el proyecto. Asegurar la correcta ejecución del esquema del CM. Participación en auditorias.  Analizar todas las novedades relacionadas al CM. |
| TPCM | Asistencia en la creación de etiquetas y ramas. Creación de actividades para el equipo específico en ramas.  Garantizar la integridad del producto y el seguimiento de los elementos de configuración propios del equipo.  Participación en auditorias.  Analizar todas las novedades relacionadas al CM. |
| Team | Ayudar a resolver conflictos durante la actividad de fusión.  Asegurarse que los criterios de calidad de los entregables a la rama principal se cumplan. Seguir todos los procesos asociados, políticas y prácticas definidas por sus roles asignados. |

**3 ADMINISTRACIÓN DE CAMBIOS**

**3.1 Alcance**

La administración de cambios es un proceso que ocurre después de que se identifica y aprueba la línea de base de la documentación, el código fuente o el hardware del producto. Los cambios a gestionar también serán internos, ya sea debido a resultados de simulación o pruebas e incluso por solicitudes externas en busca de cambios de características o funcionalidades.

El alcance del trabajo será aprobar o rechazar cambios necesarios en planes, documentos y código. Se tomarán decisiones sobre las acciones a realizar basadas en el resultado de las actividades de aseguramiento de la calidad del producto y la salud del producto después de cada ciclo de prueba.

**3.2 Versiones principales y módulos para el cliente**

**3.2.1 Consejo de control del cambio (CCB)**

Es un comité que garantiza que cada cambio sea considerado adecuadamente por todas las partes y que esté autorizado antes de su implementación. Es responsable de aprobar, supervisar y controlar las solicitudes de cambio para establecer líneas base de elementos de configuración.

Los elementos a ser revisados por el CCB son aquellos cambios presentados por el Coordinador de Emisión, el presidente de CCB u otros miembros de CCB.

**3.2.1.1 Miembros**

|  |  |
| --- | --- |
| **Integrantes** | **Rol en el CCB** |
| Velazquez, Cristian David | GPCM  Gerente de ingeniería |
| Vignolo, Gabriel Enrique | GPCM  Administrador de versiones – coordinador de problemas |
| Losano Quintana, Juan Cruz | GPCM  Director de ingeniería |

**3.2.1.2 Frecuencia de reuniones**

Tendremos reuniones semanales o en base a la demanda que tengamos. Estas serán generalmente para organizarnos e informarnos, ya que la mayor parte del trabajo se va a realizar por módulos y según la disposición del tiempo de los integrantes.

**3.2.1.3 Herramienta de gestión de cambios**

Se utilizará la herramienta GitHub para realizar el control de versiones. GitHub maneja las versiones del proyecto, registrando los cambios entre ellas y facilitando la comparación e incorporación de modificaciones ya sea la producción de código nuevo o la corrección de errores desde varios colaboradores.

**4 GESTIÓN DE LA CONFIGURACIÓN DEL CÓDIGO FUENTE**

**4.1 Esquema de directorios**

Se utilizará el siguiente esquema de directorios:

* Rama principal
* Rama de guardado
* Lanzamientos
* Documentación

*Rama principal:* Esta es la carpeta principal que estará almacenada en el repositorio en GitHub. Aquí estarán todos los archivos que se utilizaran para crear el código, realizar pruebas y crear las versiones a lanzar.

*Rama de guardado:* En esta carpeta se almacenarán ramas que se han deseado guardar.

*Lanzamientos:* En esta carpeta se almacenarán las versiones de la aplicación que se lanzaron durante el desarrollo del software.

*Documentación:* En esta carpeta estarán todos los documentos necesarios para el proyecto y la documentación de la aplicación.

**4.2 Normas de etiquetado y de nombramiento de los archivos**

El sistema de versionado se implementará en forma de etiquetas adjuntadas a los nombres de los archivos. Estas etiquetas tendrán una notación de una “V” seguido de tres números separados por puntos.

**Primer número**: Se incrementa cuando se produce un cambio que es incompatible con alguna versión anterior, pueden incluir cambios en el segundo y tercer número. Este número simboliza una etapa en la que está el software y al aumentarlo se está representando una modificación significativa de la aplicación.

**Segundo número**: Se incrementa cuando se añade una nueva funcionalidad compatible con la versión anterior, si algún método se marca como obsoleto debe aumentarse este número.

**Tercer número**: Aumenta solo cuando se corrigen errores que no modifican ninguno de los métodos públicos, es decir, no realizan cambios en el comportamiento.

**4.3 Plan de esquemas de ramas a usar**

*Rama de integración:* se define como la rama de integración principal donde se llevarán a cabo todas las implementaciones de funcionalidades. Esta rama se crea de forma predeterminada por el repositorio de control de versiones cuando se agrega un archivo.

*Ramas de desarrollo:* son aquellas ramas donde se codifica el desarrollo de nuevas características / funcionalidades del programa.

*Ramas de lanzamiento:* este tipo de ramas debe ser creado por el CM y se crean para indicar aquí las versiones de código de una versión específica y se bifurcarán desde la rama de integración. El nombre de la rama Release será branch- <versión> -rel

**4.4 Políticas de fusión de archivos**

En la rama de integración o rama principal se mantiene el código principal, pero en el caso de alguna funcionalidad extra que tenga que ser desarrollada en simultáneo por otros desarrolladores, creamos una rama para su posterior Fusión.

Para fusionar esta rama con el código principal nuevamente, se debe hacer un pedido de fusión con un mensaje descriptivo de lo que se desarrolló en la rama y que conflictos se tuvieron que resolver.

En caso de que la fusión se realice de forma correcta, se procede a un etiquetado de acuerdo a la política de versionados.

**4.5 Forma de entrega de los “releases”**

Los releases y las actualizaciones del producto se entregarán por medio de Google Play Store para que los consumidores tengan la posibilidad de tener un acceso más simple a la aplicación y además tener la posibilidad de tener acceso a las críticas y opiniones de los usuarios y a la posibilidad de informar un error en caso de fallo de la aplicación o de los componentes de la aplicación.

# **Introducción**

## **Propuesta y alcance**

Este es el Documento de requerimientos (SRS) para el proyecto “TBreaker”.

Se explicaran los requerimientos funcionales y no funcionales. Aquí se incluirá un diagrama de casos de uso y un diagrama de actividades para mejorar el detalle de los requerimientos y además sirve para una mejor interpretación de los mismos, ya que existe una variedad de lectores.

El documento servirá de guía para desarrollar el software y para el control en futuras modificaciones.

## **Descripción del proyecto**

Para el proyecto se utilizara el lenguaje de programación java. El software es una aplicación de entretenimiento para dispositivos móviles llamada “TBreaker”. Utilizaremos un conjunto de herramientas para gestionar y producir el software.

La aplicación contiene dos juegos. Uno similar al tetris .En el otro hay que destruir bloques, si se destruyen todos los bloques se pasa a otro nivel donde aumenta la dificultad y se pierde cuando no se le pega a la bola.

# **Especificación de Requerimientos**

## **Casos de uso y Requerimientos**

### **Definición**

La especificación de los requerimientos y los casos de uso engloban todos los servicios que provee el sistema y como debe comportarse ante determinadas situaciones .Los requerimientos se redactaron completos y consistentes para evitar ambigüedades y de forma comprensible para los usuarios que no cuentan con un conocimiento técnico.

### **Modelo del Sistema**

Actores:

* Usuario

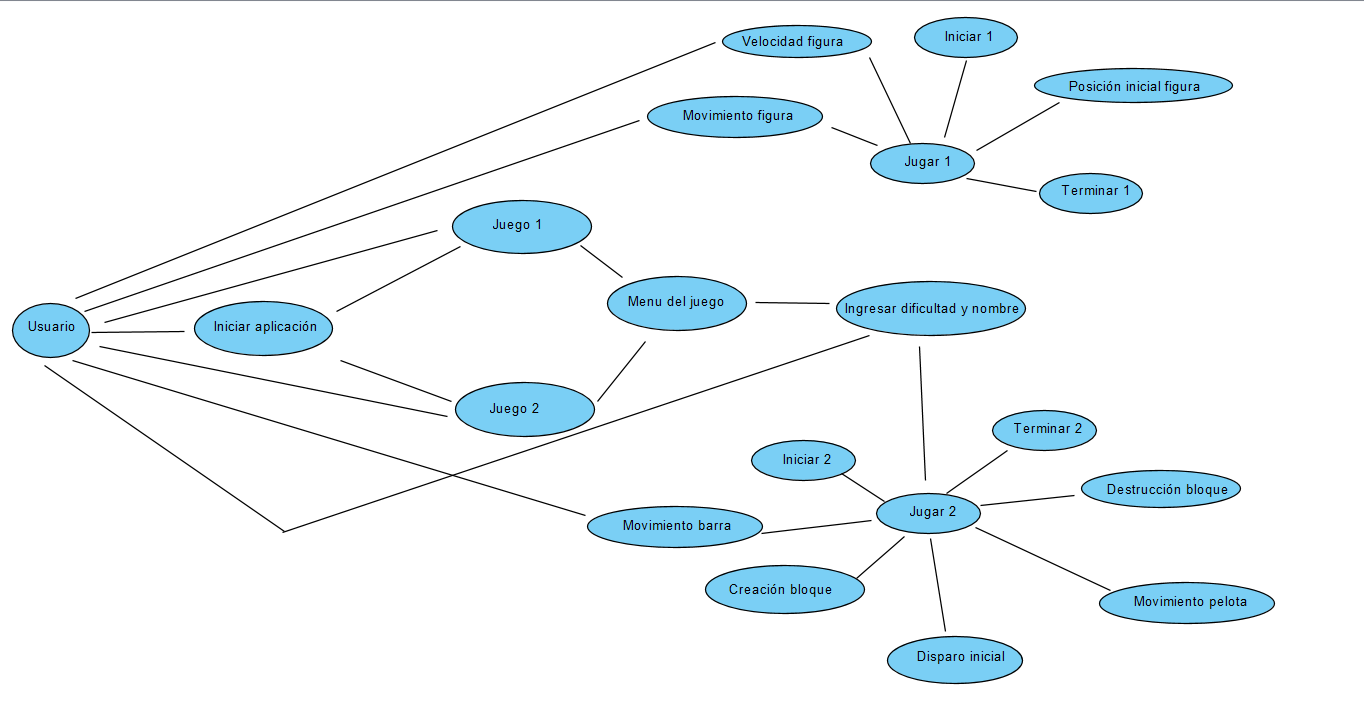
Objetivo general:

* Ofrecer a los jugadores la posibilidad de jugar a los dos juegos que contiene la aplicación.

Eventos que alteran el normal funcionamiento del sistema:

* En ambos juegos no hay actividad cuando el usuario gana.

A continuación, se busca representar gráficamente relaciones entre componentes del sistema, sistemas externos y ambiente.



### Casos de uso

1. Iniciar aplicación
2. Juego 1
3. Juego 2
4. Menú del juego
5. Ingresar nombre y elegir dificultad
6. Jugar 1
7. Iniciar 1
8. Movimiento de la figura
9. Velocidad de la figura
10. Posición inicial de la figura
11. Terminar 1
12. Jugar 2
13. Iniciar 2
14. Creación de bloque
15. Disparo inicial
16. Movimiento de la pelota
17. Movimiento de la barra
18. Destrucción de bloque
19. Terminar 2

A. Iniciar aplicación

1. Precondiciones: ninguna
2. Función: Se muestra una pantalla donde hay dos botones para elegir el juego 1 o el juego 2
3. Postcondiciones:

(P1): Si selecciona el juego 1 se abre la pantalla de menú para completar los datos para jugar el primer juego.

(P2): Si selecciona el juego 2 se abre la pantalla de menú para completar los datos para jugar el segundo juego.

B. Juego 1

1. Precondiciones: Haber apretado el botón de juego 1 (P1).
2. Función: Se prepara el juego que selecciono el usuario.
3. Postcondiciones:

(P1): Se abre la ventana de menú y los datos que se ingresan modifican al juego que se seleccionó.

C. Juego 2

1. Precondiciones: Haber pulsado el botón de juego 2(P2).
2. Función: Se prepara el juego que selecciono el usuario
3. Postcondiciones:

(P1): Se abre la ventana de menú y los datos que se ingresan modifican al juego que se seleccionó.

D. Menú del juego

1. Precondiciones: Haber pulsado el botón de juego 1 (P1) o el botón juego 2(P2).
2. Función: Aquí se encuentran dos campos para ingresar el nombre y la dificultad del juego. También está el botón jugar para iniciar el juego. Existe la posibilidad de usar un menú donde se puede salir del juego o entrar a help para conocer los controles del juego.
3. Postcondiciones:

(P1): En el caso de apretar el botón jugar, si antes se seleccionó el juego 1 se inicia el juego 1 con la dificultad que se elija.

(P2): En el caso de apretar el botón jugar, si antes se seleccionó el juego 2 se inicia el juego 2 con la dificultad que se elija.

(P3): En el caso de entrar al menú y apretar help, se abre una ventana como una pequeña explicación de los controles para jugar.

(P4): En el caso de entrar al menú y apretar el botón salir, se cierra el juego.

E. Ingresar nombre y elegir dificultad

1. Precondiciones: El usuario ha seleccionado el juego 1 o juego 2.
2. Función: Permite al usuario elegir el nivel de dificultad. Este puede ser fácil, normal o difícil. La dificultad solo acepta un rango de números 1 a 3 que corresponden a los niveles de dificultad y el nombre solo debe contener letras. Si se ingresan mal los datos el juego avisa y se deben reingresar de nuevo los datos.
3. Postcondiciones: Se presiona el botón jugar y se inicia la partida.

F. Jugar 1

1. Precondiciones: El usuario ha elegido la dificultad para el juego 1.
2. Función: Se crea la figura y se habilitan los botones para el usuario.
3. Postcondiciones: El usuario puede usar los controles para mover la figura.

G. Iniciar 1

1. Precondiciones: Haber elegido la dificultad.
2. Función: Comienza el juego y el usuario puede desplazarla hacia los lados o aumentar la velocidad de la misma para colocarla en el espacio que decida.
3. Postcondiciones: Se puede mover las figuras que aparecen.

H. Movimiento de la figura

1. Precondiciones: Haber pulsado uno de los botones para realizar un desplazamiento horizontal.
2. Función: La figura se mueve hacia la izquierda o derecha.
3. Postcondiciones: La figura es situada en la posición deseada.

I. Velocidad de la figura

1. Precondiciones: Haber pulsado el botón para cambiar de velocidad.
2. Función: La figura aumenta su velocidad.
3. Postcondiciones: Cae a la posición con mayor velocidad.

J. Posición inicial de la figura

1. Precondiciones: Haber iniciado el juego.
2. Función: La figura se crear en una determinada posición de la pantalla del juego.
3. Postcondiciones: La figura se desplaza en un sentido vertical.

K. Terminar 1

1. Precondiciones: Haber acumulado piezas hasta llegar a la máxima altura.
2. Función: El juego finaliza.
3. Postcondiciones: Se cierra el juego.

L. Jugar 2

1. Precondiciones: El usuario ha elegido la dificultad para el juego 2.
2. Función: Se prepara el juego y se crea la barra, el bloque y la pelota.
3. Postcondiciones: Se habilitan los controles para el usuario.

M. Iniciar 2

1. Precondiciones: Haber elegido la dificultad.
2. Función El usuario puede mover la barra para que pueda impactar la pelota.
3. Postcondiciones: La pelota comienza a moverse.

N. Creación de bloque

1) Precondiciones: Haber iniciado el juego 2 o haber destruido un bloque.

2) Función: Se arma el bloque, este puede crearse en posiciones al azar.

3) Postcondiciones: el usuario sigue jugando.

O. Disparo inicial

1. Precondiciones: Haber iniciado el juego.
2. Función: Dispara la pelota.
3. Postcondiciones: Comienza a moverse la pelota.

P. Movimiento de la pelota

1. Precondiciones: Haber disparado la pelota.
2. Función: Se encarga de mover la pelota.
3. Postcondiciones: La pelota sigue un movimiento determinado por el choque con algún objeto ya sean bloques, la misma barra o los extremos de la pantalla.

Q. Desplazamiento de la barra

1. Precondiciones: Haber pulsado los botones para desplazamiento.
2. Función: La barra se desplaza hacia la derecha o hacia la izquierda, dependiendo del botón presionado.
3. Postcondiciones: La barra es colocada en la posición deseada.

R. Destrucción bloque

1. Precondiciones: La pelota impacta con el bloque.
2. Función: Se mueve de posición al bloque.
3. Postcondiciones: El usuario sigue jugando.

S. Terminar 2

1. Precondiciones: La bola no pudo ser impactada por la barra.
2. Función: El juego finaliza.
3. Postcondiciones: Se abre una ventana con el puntaje del usuario y se cierra el juego.

**Requerimientos No Funcionales**

### **Definición**

Los requerimientos redactados están relacionados con las propiedades de fiabilidad, tiempo de respuesta y uso del almacenamiento. Además definen restricciones sobre la implementación del sistema. Estos restringen características del sistema como un todo. Por esta razón son muy importantes para que el sistema opere correctamente.

### **Listado**

#### **De Producto**

De eficiencia:

No debe utilizar más de 100 MB de RAM

No debe ocupar más de 50 MB de almacenamiento

Cualquier actividad no debe tardar más de un segundo

De usabilidad:

Los usuarios tienen acceso a un botón de tutorial en el caso de tener dudas en la jugabilidad.

De confiabilidad:

La aplicación no requiere de permisos al iniciar el juego

De seguridad:

No se requieren datos personales del usuario

#### **Organizacionales**

Operacional:

Se debe ingresar un nombre de usuario para registrar el puntaje obtenido en el juego.

Desarrollo:

El lenguaje de programación es java.

Ambiental:

Se utilizara NetBeans para desarrollar la aplicación.

#### **Externos**

* El sistema será gratuito, código abierto en el que cualquiera podrá cambiar el software, sin patentes.
* El sistema no revelara a los desarrolladores datos de los usuarios

## **Evolución del sistema**

Modificaciones que se podrían hacer en el futuro:

* Mejorar la calidad de los gráficos
* Añadir ítems nuevos
* Efectos de sonido
* Compartir el puntaje con otros jugadores

# **Casos de Prueba**

## **Requerimientos Funcionales**

|  |  |
| --- | --- |
| Código |  |
| Descripción | Abrir aplicación |
| Pasos | 1. Abrir aplicación 2. Mostrar pantalla |
| Resultado Esperado | Que se visualicen los botones de la pantalla |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Iniciar juego 1 |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 1  3. Ingresar la dificultad  4. Ingresar el nombre de usuario  5. Pulsar en el botón jugar |
| Resultado Esperado | Que comience el juego 1 y que caigan las piezas. |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Iniciar juego 2 |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 1  3. Ingresar la dificultad  4. Ingresar el nombre de usuario  5. Pulsar en el botón jugar |
| Resultado Esperado | Que comience el juego 2. Que se armen los bloques y aparezca la barra y la bola. |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Menú del juego |
| Pasos | 1. Abrir aplicación 2. Seleccionar juego 1 o juego 2 |
| Resultado Esperado | Debe mostrar una pantalla nueva con campos para ingresar el nombre y la dificultad, el botón jugar y menú. |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Realizar tutorial de los juegos |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 1 o juego 2  3. Ingresar al menú y seleccionar el la opción help |
| Resultado Esperado | Se abrirá una ventana con la explicación de ambos juegos |
|  |  |
| Código |  |
| Descripción | Salir del juego |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 1 o juego 2  3. Ingresar al menú y seleccionar el la opción salir |
| Resultado Esperado | El juego se debería cerrar |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Ingresar nombre de usuario |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 1 o juego 2  3 Completar el campo donde se ingresa el nombre |
| Resultado Esperado | Se almacena el nombre del jugador |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Ingresar nombre de usuario y/o dificultad (incorrectos) |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 1 o juego 2  3 Completar el campo sin ingresar nada o colocando datos inválidos |
| Resultado Esperado | Se pedirá el reingreso del nombre de usuario y la dificultad |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Seleccionar dificultad juego 1 |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 1  3. Ingresar un numero de 1 a 3 para elegir la dificultad del juego |
| Resultado Esperado | Se almacena la configuración de dificultad |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Seleccionar dificultad juego 2 |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 2  3. Ingresar un numero de 1 a 3 para elegir la dificultad del juego |
| Resultado Esperado | Se almacena la configuración de dificultad. |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Armado de bloque |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 2  3. Ingresar la dificultad  4. Ingresar el nombre de usuario  5. Pulsar en el botón jugar |
| Resultado Esperado | Que se arme un bloque al iniciar el juego y cada vez que se destruya un bloque |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Movimiento de la pelota |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 2  3. Ingresar la dificultad  4. Ingresar el nombre de usuario  5. Pulsar en el botón jugar  6. Disparo inicial de la pelota |
| Resultado Esperado | Que la bola tenga un movimiento deseado y que pueda rebotar con los extremos laterales de la pantalla, con la barra y con los bloques. |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Desplazamiento de barra |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 2  3. Ingresar la dificultad  4. Ingresar el nombre de usuario  5. Pulsar en el botón jugar  6. Mover la barra al presionar los botones de desplazamiento |
| Resultado Esperado | Que el movimiento hacia la izquierda o hacia la derecha sea preciso. |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Terminar partida de juego 2 |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 2  3. Ingresar la dificultad  4. Ingresar el nombre de usuario  5. Pulsar en el botón jugar  6. Jugar  7.Perder todas las vidas |
| Resultado Esperado | Que muestre el puntaje acumulado y que se cierre el juego |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Terminar partida juego 1 |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 1  3. Ingresar la dificultad  4. Ingresar el nombre de usuario  5. Pulsar en el botón jugar  6. Jugar  7.Perder todas las vidas |
| Resultado Esperado | Que muestre el puntaje acumulado y que se cierre el juego |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Ganar juego 2 |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 2  3. Ingresar la dificultad  4. Ingresar el nombre de usuario  5. Pulsar en el botón jugar  6. Jugar  7.Romper todos los bloques |
| Resultado Esperado | Que se muestren los puntos acumulados y se cierre el juego |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Posición inicial de la figura |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 1  3. Ingresar la dificultad  4. Ingresar el nombre de usuario  5. Pulsar en el botón jugar |
| Resultado Esperado | Que se cree la figura |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Eliminar bloque |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 2  3. Ingresar la dificultad  4. Ingresar el nombre de usuario  5. Pulsar en el botón jugar  6. Jugar  7.La pelota debe impactar al bloque |
| Resultado Esperado | Que se elimine el bloque y se cree otro en otra posición. |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Velocidad de la figura |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 1  3. Ingresar la dificultad  4. Ingresar el nombre de usuario  5. Pulsar en el botón jugar  6. Apretar el botón para aumentar la velocidad |
| Resultado Esperado | Que al presionar el botón la pieza descienda con mayor velocidad. |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Desplazamiento de la figura |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 1  3. Ingresar la dificultad  4. Ingresar el nombre de usuario  5. Pulsar en el botón jugar  6. Apretar los botones de desplazamiento |
| Resultado Esperado | Que al presionar los botones la pieza se desplace hacia la izquierda o hacia la derecha según se desee. |

## **Requerimientos No Funcionales**

|  |  |
| --- | --- |
| Código |  |
| Descripción | Velocidad de carga |
| Pasos | 1. Abrir aplicación  2. Pulsar juego 1 o juego 2  3. Ingresar la dificultad  4. Ingresar el nombre de usuario  5. Pulsar en el botón jugar  6. Jugar |
| Resultado Esperado | Que el tiempo de carga entre los menús y los niveles sea menor a 1 segundo. |

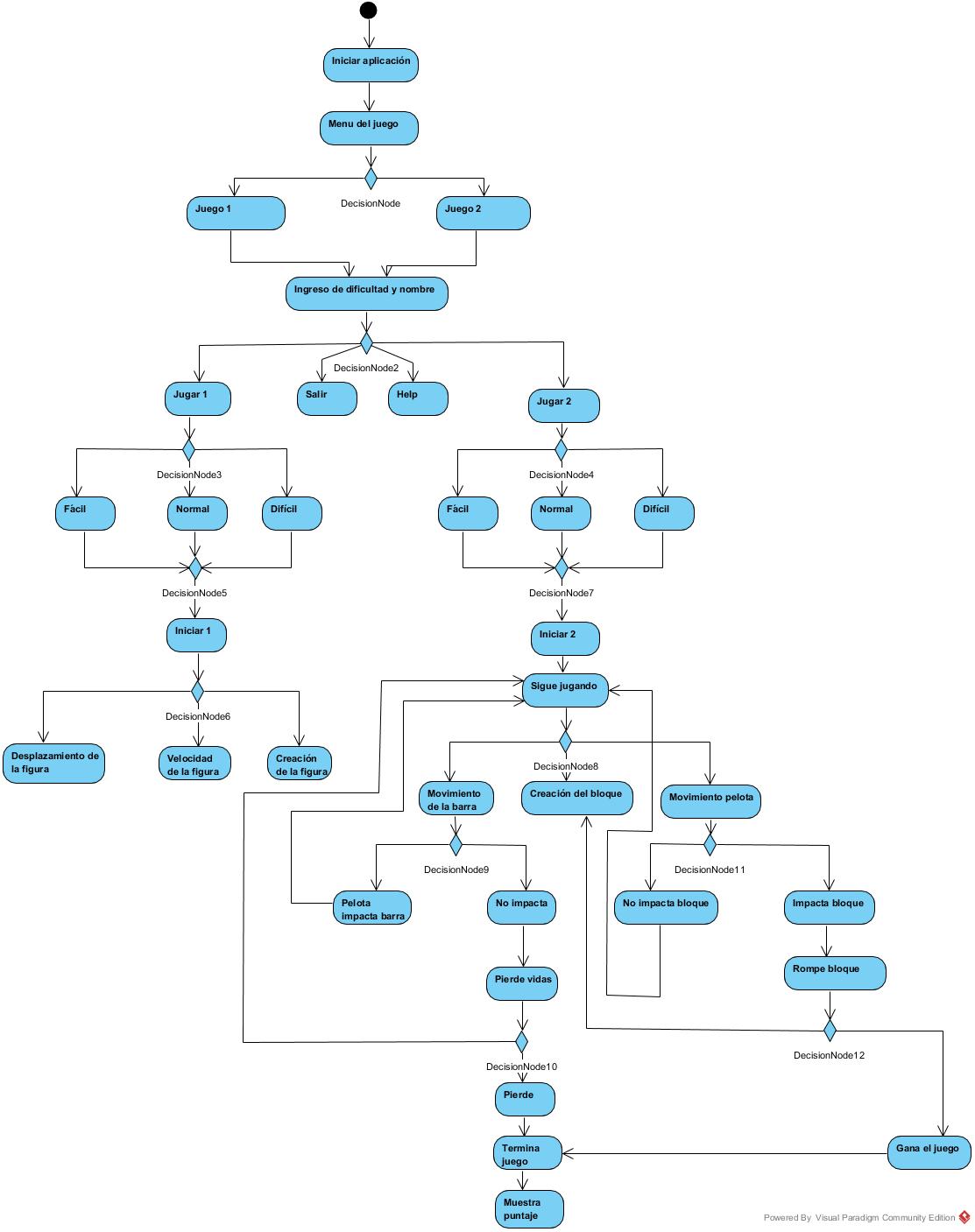
|  |  |
| --- | --- |
| Código |  |
| Descripción | Consumo de RAM |
| Pasos | 1. Abrir aplicación 2. Observar consumo de RAM con otro software |
| Resultado Esperado | No deberá superar la utilización de 100 MB de memoria RAM |

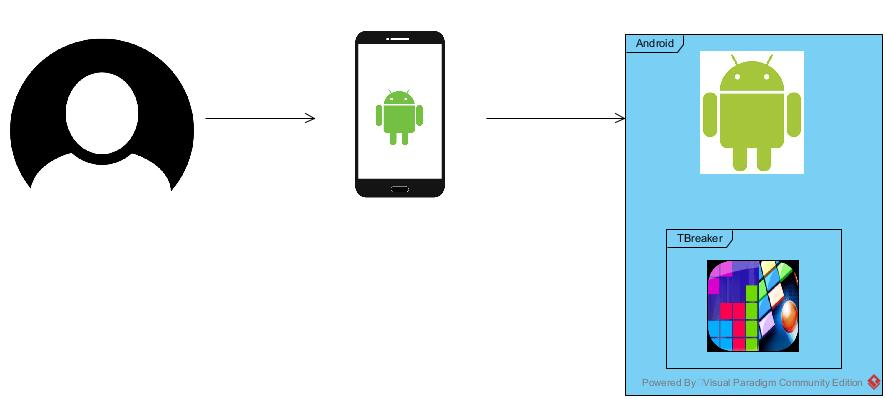
|  |  |
| --- | --- |
| Código |  |
| Descripción | Consumo de almacenamiento |
| Pasos | 1. Instalar aplicación 2. Observar consumo de almacenamiento con otro software |
| Resultado Esperado | No deberá superar la utilización de 50 MB de almacenamiento |

## **3.3 Matriz de Trazabilidad**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Casos de uso**  **Requerimientos del sistema** | Iniciar aplicación | Juego 1 | Juego 2 | Menú del juego | Dificultad y nombre | Jugar 1 | Iniciar 1 | Movimiento de figura | Velocidad figura | Posición inicial figura | Terminar 1 | Jugar 2 | Iniciar 2 | Armado de bloque | Disparo inicial | Movimiento de la pelota | Movimiento de la barra | Destrucción del bloque | Terminar 2 |
| Botón para seleccionar juego 1 | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Botón para seleccionar juego 2 | x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Ingresar dificultad |  | x | x | x | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Ingresar nombre de usuario |  | x | x | x | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| El nombre no puede ser vacío o contener más de 8 caracteres y la dificultad deben ser números entre 1 y 3 |  |  |  | x | x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Ver tutorial |  |  |  | x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Salir del juego |  |  |  | x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Que comience el juego en la dificultad elegida por el usuario |  |  |  |  |  | X |  |  |  |  |  | X |  |  |  |  |  |  |  |
| Cae una pieza aleatoria para comenzar el juego |  |  |  |  |  |  | X |  |  | x |  |  |  |  |  |  |  |  |  |
| Se crea el bloque y se muestra la barra y la bola para comenzar el juego |  |  |  |  |  |  |  |  |  |  |  |  | X | X |  |  |  |  |  |
| Si se rompen todos los bloques se gana el juego |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | x | x |
| Si se destruye un bloque se crea uno nuevo |  |  |  |  |  |  |  |  |  |  |  |  | X | X |  |  |  | X |  |
| Si el usuario pierde, se guardan los datos, se muestra el puntaje obtenido en la partida y se cierra el juego |  |  |  |  |  |  |  |  |  |  | x |  |  |  |  |  |  |  | x |
| Movimiento de la figura hacia la izquierda o derecha |  |  |  |  |  |  | x | X |  |  |  |  |  |  |  |  |  |  |  |
| mover la barra hacia la izquierda o derecha |  |  |  |  |  |  |  |  |  |  |  |  | X |  |  |  | X |  |  |
| Al presionar un botón la figura descender más rápidamente |  |  |  |  |  |  | x | X | x |  |  |  |  |  |  |  |  |  |  |
| La bola sale en diagonal a la barra |  |  |  |  |  |  |  |  |  |  |  |  | X |  | X |  |  |  |  |
| Movimiento de la figura en sentido vertical durante el juego |  |  |  |  |  |  | x | x |  |  |  |  |  |  |  |  |  |  |  |
| Que la bola rebote con los extremos laterales de la pantalla, con la barra y con los bloques. |  |  |  |  |  |  |  |  |  |  |  |  | x |  |  | x |  |  |  |

# **Diagrama de actividades**



**Arquitectura General**

## **Gráfico de arquitectura**

## **Patrón de arquitectura**

MVC (Model View Controler): Es un patrón de arquitectura que separa la presentación y la interacción de los datos del sistema. Tiene una estructura compuesta por tres componentes lógicos relacionados entre sí. Los componentes son los siguientes:

*Modelo:* se encarga de manejar los datos del sistema y los métodos y operaciones asociados a esos datos. Es la representación de la información con la cual el sistema opera, por lo tanto gestiona todos los accesos a dicha información, tanto consultas como actualizaciones, implementando también los privilegios de acceso que se hayan descrito en las especificaciones de la aplicación. Envía a la “vista” aquella parte de la información que en cada momento se le solicita para que sea mostrada. Las peticiones de acceso o manipulación de información llegan al “modelo” a través del “controlador”.

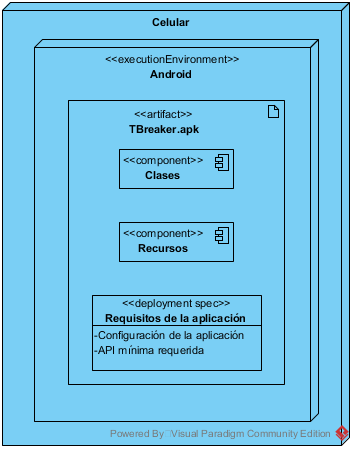
*Vista:* Define la presentación de los datos hacia el usuario. Ni el modelo ni el controlador se preocupan de cómo se verán los datos, esa responsabilidad es únicamente de la vista.

*Controlador:* Dirige la interacción del usuario y pasa estas interacciones a vista y modelo. Las interacciones pueden ser por ejemplo: cuando se presiona una tecla. También puede enviar comandos a su “vista” asociada si se solicita un cambio en la forma en que se presenta el “modelo”.

Se va a utilizar MVC ya que permite facilitar el mantenimiento de la aplicación. Por ejemplo: si lo que queremos es modificar la apariencia de la aplicación sólo modificaremos el componente Vista; si queremos cambiar de sistema de almacenamiento de datos sólo tendremos que modificar el componente Modelo; y si queremos modificar el código para optimizar rendimiento lo que haremos será modificar la capa del Controlador.

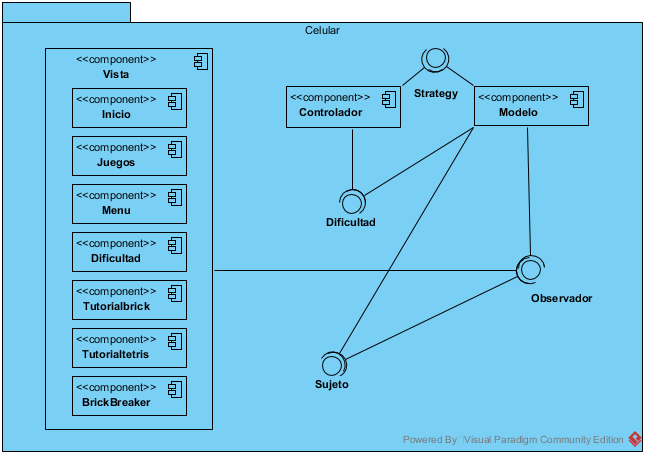
# **Diagramas**

## **Diagrama de despliegue**



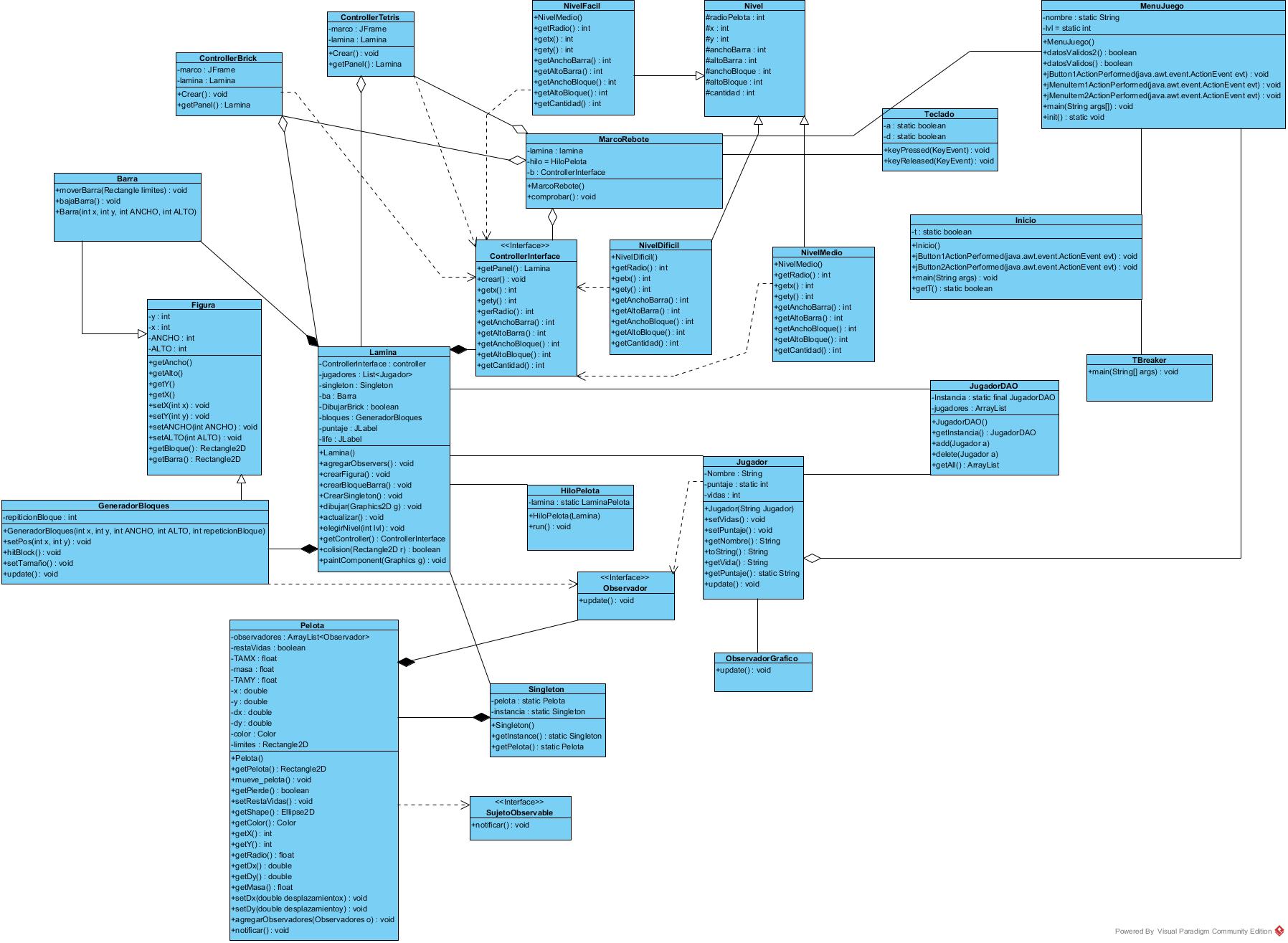
El diagrama de despliegue de UML sirve para modelar la disposición física de los artefactos del software y su relación con los elementos hardware. En el caso presente se tiene un celular, el mismo tiene un entorno de ejecución en Android en donde correrá la aplicación (TBreaker.apk). Esta última tiene distintos componentes como las clases que la integran y las especificaciones de despliegue.

## **Diagrama de componentes**

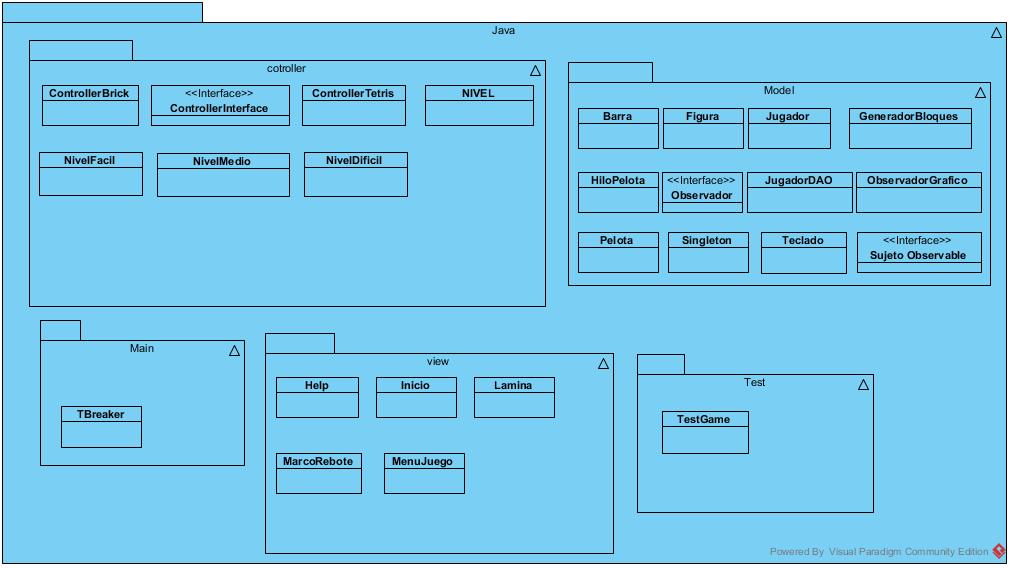


El diagrama de componentes muestra cómo un sistema de software es dividido en componentes y muestra las dependencias entre los mismos. Esta expone la interacción entre el usuario, la vista, el controlador y el modelo. El primero provee entradas a las distintas pantallas dispuestas en la vista, el controlador toma estos cambios detectados en la misma y realiza operaciones con el modelo (utilizando la lógica programada en sí), finalmente el modelo les avisa a las vistas que deben actualizarse con los nuevos datos.

## **Diagrama de Clases**



## **Diagrama de Paquetes**



# **Patrones de Diseño**

## Stratregy

El patrón de diseño strategy fue implementado en la clase MarcoRebote y en la clase Lamina. De esta forma se facilita los cambios en los juegos según en nivel de dificultad que seleccione el usuario y además para preparar el tipo de juego que se eligió. La interfaz controller define las operaciones para hacer los cambios en la barra, figura y bloques según las distintas clases de niveles de dificultad y las clases Tetris y Brick se encargan de preparar los objetos que se mostraran en pantalla usando también los métodos de la interfaz controller.

## Singleton

Este patrón se utilizó para crear una única instancia de la clase pelota y además para crear una única instancia de una colección de jugadores para poder guardar los puntajes de cada usuario que ingrese al juego sin crear otra lista nueva.

## Observer

Este patrón se lo implemento en la clase pelota, en clase jugador y en GenerarBloque. Su funcionamiento consiste en avisar cada vez que la pelota toque un bloque, entonces cuando lo toca, se destruye un bloque y se crear otro nuevo en otra posición. En la clase jugador cuando se recibe la notificación del impacto se incrementa el puntaje .También se utilizó este patrón para avisar cuando la pelota toque el extremo inferior para restar vidas del jugador y además para desplegar un cartel que indica el fin del juego si se perdieron todas las vidas o si se ganó el juego.

# Pruebas Unitarias

Las pruebas unitarias fueron realizadas utilizando la herramienta JUnit aplicando Assert para verificar el funcionamiento del software.

Se utilizó una sola clase test, esta verifica que se crea el juego según los niveles de dificultad que se seleccionan. También se verifica el funcionamiento de la lista de los jugadores y además se verifica el comportamiento del juego cuando implementa el patrón observer para incrementar el puntaje del jugador cuando la pelota toca un bloque.

## **Release Note**

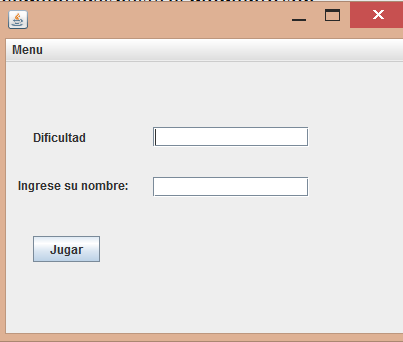
El software a la fecha (04/02/2019) aún no es entregable a los usuario finales debido a que se requiere modificar en el código fuente del software porque aún queda pendiente hacer terminaciones en el juego tetris, ya que este no está completamente terminado.

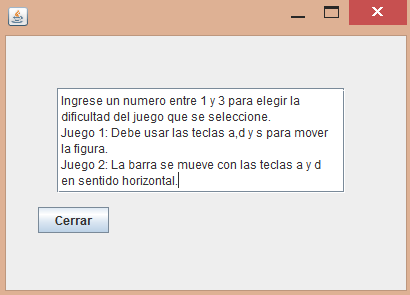
Las funcionalidades implementadas hasta el momento son:

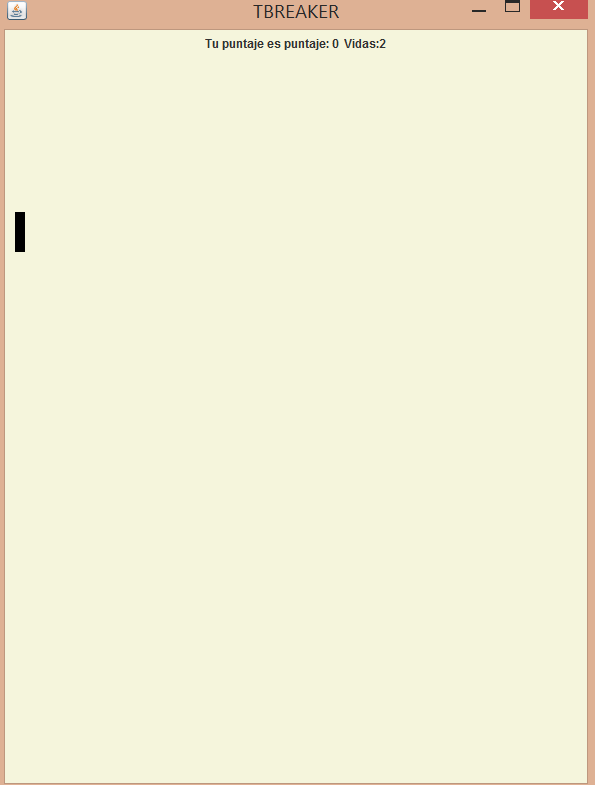
* El usuario puede ingresar al programa y jugar al “BrickBreaker” y Tetris.
* El juego le muestra al usuario en pantalla el puntaje que está realizando.
* El usuario tiene 2 vidas y cuando pierde las mismas el juego termina.
* El usuario puede ver un tutorial antes de comenzar a jugar.
* El usuario puede ver tu puntaje final al ganar o perder.

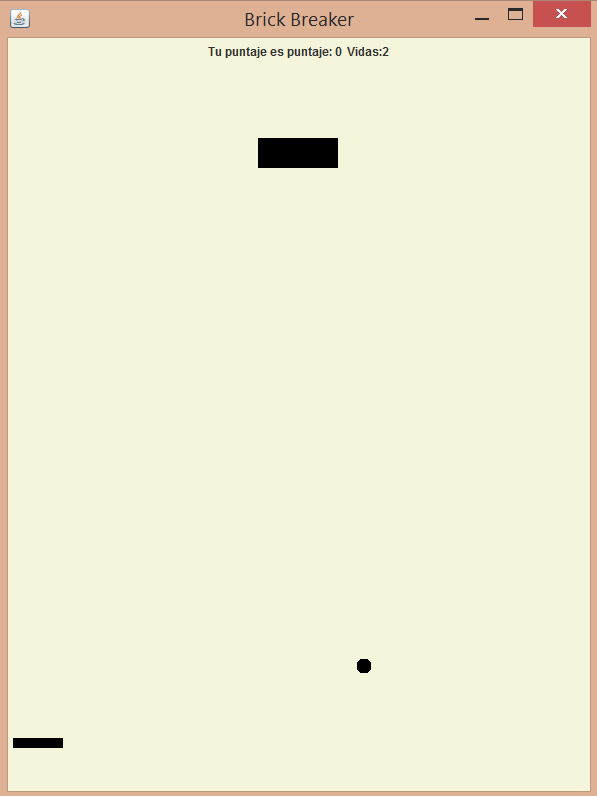
El aspecto visual del software es el siguiente:





****

****

****

**Casos de prueba de sistema**

**Funcionales:**

|  |  |
| --- | --- |
| Código |  |
| Descripción | Abrir aplicación |
| Pasos | 1. Abrir aplicación |
| Resultado Esperado | Que se visualicen los botones de la pantalla de inicio. |
| Pass/Fail | Pass |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Seleccionar juego |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 1 o juego 2 |
| Resultado Esperado | Que ingrese al menú del juego seleccionado. |
| Pass/Fail | Pass |

|  |  |
| --- | --- |
| Código |  |
| Descripción | Ingresar nombre y nivel dificultad |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 1  3.Colocar nombre de usuario y nivel de dificultad  4.Pulsar el botón jugar |
| Resultado Esperado | Que comience el juego 1 y aparezca la figura |
| Pass/Fail | Pass |
| Código |  |
| Descripción | Iniciar juego 2 |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 2  3.Colocar nombre de usuario y nivel de dificultad  4.Pulsar el botón jugar |
| Resultado Esperado | Que comience el juego 2. Que se armen los bloques y aparezca la barra y la pelota. |
| Pass/Fail | Pass |
|  |  |
| Código |  |
| Descripción | Realizar tutorial de los juegos |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 1 o juego 2  3. Pulsar el botón menú y pulsar en help |
| Resultado Esperado | Se abrirá una ventana con la explicación de ambos juegos |
| Pass/Fail | Pass |
| Código |  |
| Descripción | Ingresar nombre de usuario y dificultad |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 1  3. Colocar nombre de usuario y nivel de dificultad |
| Resultado Esperado | Se almacena el nombre y la dificultad modifica el juego |
| Pass/Fail | Fail |
|  |  |
| Código |  |
| Descripción | Ingresar nombre de usuario o dificultad (nombre incorrecto y/o dificultad) |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 1 o juego 2  3. Colocar nombre de usuario y nivel de dificultad |
| Resultado Esperado | Se pedirá el reingreso del nombre de usuario y/o dificultad |
| Pass/Fail | Fail |
|  |  |
| Código |  |
| Descripción | Seleccionar dificultad juego 2 |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 2  3. Colocar nombre de usuario y nivel de dificultad |
| Resultado Esperado | Se almacena la configuración de dificultad y el nombre |
| Pass/Fail | Pass |
| Código |  |
| Descripción | Armado de bloques |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 2  3. Colocar nombre de usuario y nivel de dificultad  4.Pulsar jugar |
| Resultado Esperado | Que se armen los bloques para que el jugador pueda jugar. |
| Pass/Fail | Pass |
| Código |  |
| Descripción | Movimiento de la bola |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 2  3. Colocar nombre de usuario y nivel de dificultad  4.Pulsar jugar |
| Resultado Esperado | Que la bola tenga un movimiento deseado y que pueda rebotar con los extremos laterales de la pantalla, con la barra y con los bloques. |
| Pass/Fail | Pass |
| Código |  |
| Descripción | Desplazamiento de barra |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 2  3. Colocar nombre de usuario y nivel de dificultad  4.Pulsar jugar |
| Resultado Esperado | Que el movimiento hacia la izquierda o hacia la derecha sea preciso. |
| Pass/Fail | Pass |
| Código |  |
| Descripción | Terminar partida juego 2 |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 2  3. Colocar nombre de usuario y nivel de dificultad  4.Pulsar jugar  5.Perder o ganar |
| Resultado Esperado | Que muestre el puntaje. |
| Pass/Fail | Pass |
| Código |  |
| Descripción | Guardar puntaje |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 2  3. Colocar nombre de usuario y nivel de dificultad  4.Pulsar jugar  5.Perder o ganar |
| Resultado Esperado | Que guarde el puntaje en el dispositivo del usuario. |
| Pass/Fail | Fail |
| Código |  |
| Descripción | Desplazamiento de la figura |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 1  3. Colocar nombre de usuario y nivel de dificultad  4.Pulsar jugar  5. Presionar algunas de las teclas de desplazamiento horizontal |
| Resultado Esperado | Que la figura se mueva hacia la izquierda o derecha. |
| Pass/Fail | Pass |

**No funcionales**

|  |  |
| --- | --- |
| Código |  |
| Descripción | Velocidad de carga |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 1  3. Colocar nombre de usuario y nivel de dificultad  4.Pulsar jugar |
| Resultado Esperado | Que el tiempo de carga entre los menús y los niveles sea menor a 1 segundo. |
| Pass/Fail | Pass |
| Código |  |
| Descripción | Consumo de RAM |
| Pasos | 1. Abrir aplicación  2. Seleccionar el juego 1  3. Colocar nombre de usuario y nivel de dificultad  4.Pulsar jugar  5.Jugar  6. Observar el consumo de la ram |
| Resultado Esperado | No deberá superar la utilización de 100 MB de memoria RAM |
| Pass/Fail | Pass |
| Código |  |
| Descripción | Consumo de almacenamiento |
| Pasos | 1. Instalar aplicación 2. Observar consumo de almacenamiento con otro software |
| Resultado Esperado | No deberá superar la utilización de 50 MB de almacenamiento |
| Pass/Fail | Pass |

**Defectos identificados y corregidos**

* En el juego BrickBreaker la barra cuando se desplazaba se perdía de vista porque no detectaba los límites en los márgenes de la pantalla. Se logró corregir este error para que no pase los márgenes laterales de la pantalla.
* En un principio no se veía la barra al comenzar a jugar, luego pudimos corregirlo ajustando la posición de la misma de acuerdo al tamaño de la pantalla.
* No se podía crear una colección de bloques entonces se decidió crear un bloque a la vez cada vez que se lo destruya.
* La bola se movía muy rápido por la pantalla y se arregló variando la velocidad hasta que se obtuvo una razonable.
* La barra se movía muy lento y se arregló variando la velocidad de la misma hasta obtener una velocidad adecuada respecto a la velocidad de la bola.

**Defectos conocidos (no resueltos)**

* El puntaje de cada jugador no se guarda cuando se cierra el juego porque no se implementó una base de datos para guardar la información.
* Al seleccionar la dificultad en el juego tetris no hay cambios debido a que no hubo tiempo para terminar dicho juego.
* La figura en el juego tetris se desplaza más allá del límite de la pantalla.
* No existe un fin para el juego tetris.
* La visualización de las vidas y puntaje en el juego tetris no se ven afectadas en el juego.
* La clase hiloPelota no funciona correctamente.

**Lugar/link del entregable y de las instrucciones de instalación**

La aplicación se subirá a GitHub con un enlace público. Este se encuentra disponible en la siguiente Link: <https://github.com/locxha96/UNC-ing-software-fuplace>

## **Informe sobre el trabajo realizado**

Detalle de dedicación de esfuerzo para realizar el trabajo (en Personas Horas), distinguiendo la contribución personal de cada miembro del grupo y el esfuerzo invertido por cada tarea realizada. Esto incluye todas las tareas para poder construir la aplicación, documentarla y elaborar el informe y la presentación de esta materia.

|  |  |  |  |
| --- | --- | --- | --- |
| Integrantes | Desarrollo de aplicación | Documentación de software | Preparación del informe y la presentación |
| Velazquez, Cristian | 70 hs | 30 hs | 20 hs |

**Lecciones aprendidas durante la elaboración del práctico y errores cometidos**

Hemos aprendido la importancia del trabajo en equipo para la elaboración de un producto de software. Y hemos visto que la coordinación de tareas a realizar por cada integrante es fundamental para que el trabajo realizado por cada uno fuese útil, eficiente y eficaz.

La ingeniería de software nos brinda éstos lineamientos fundamentales, de los cuales algunos de ellos no hemos podido ponerlos en práctica de la mejor manera, en la mayoría de los casos por inexperiencia previa.

Algunas de éstas deficiencias que hemos tenido desarrollando el software, fue la falta de común acuerdo de las interfaces de los componentes del programa que necesitaban ser implementados, lo cual nos dificultó enormemente dividirnos las tareas de programación y más aún integrarlos luego.

Se aprendió a crear ventanas en java implementado el uso de botones para pasar de una ventana a otra. Y además se aprendió a generar la interacción de los botones.

En este proyecto primero se escribió el código y luego se observó que al implementar los patrones de diseño se necesitaron menos líneas de código para poder realizar lo mismo. También ayudo para mejorar las técnicas de programación.

Tuve dificultades al momento de realizar los test, ya que no conocía el funcionamiento de los botones en las ventanas para la interfaz por lo que no puede hacer tests para comprobar el funcionamiento de los métodos que se encargan de verificar el ingreso del nombre o si el nivel esta comprendido en el rango permitido y además que sea un numero entero.

Más allá de los inconvenientes se ha logrado un mayor entendimiento de los conceptos dados en la materia y de su importancia para la elaboración de Software profesional. Si bien el software que hemos elaborado para éste trabajo final ha sido pequeño, hemos visto claramente que si desarrolláramos un proyecto mucho más grande y con un equipo más grande, sin aplicar principios de Ingeniería de Software sería prácticamente imposible concretarlo correctamente.