Instituto Tecnológico de Costa Rica
Centro Académico de Alajuela
IC-2101 Programación Orientada a Objetos

II Semestre 2020
I Proyecto Programado
MSc. Samanta Ramijan Carmiol

II Proyecto Programado

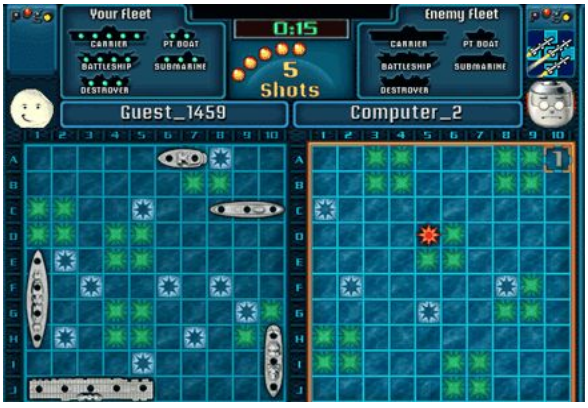
1. Motivación

Como actividad para la puesta en práctica de los conceptos estudiados en los contenidos de Modelado con UML y programación en Java, se propone la realización de una aplicación de complejidad media con ayuda del motor de interfaces gráficas JavaFX.

2. Objetivos Formativos.

La presente asignación pretende servir como herramienta para que las y los estudiantes logren el objetivo general del curso “Entender las actividades de construcción de software vía su aplicación práctica en el desarrollo de programas medianos.”

3. Metodología.



Las y los estudiantes , agrupados en grupos de trabajo de máximo tres estudiantes, deberán modelar y programar el juego de batalla naval. Los detalles generales del juego y de este proyecto en particular se abordan en detalle en la siguiente sección, más adelante también se tratan los aspectos de calificación y organización temporal del proyecto, así como algunos aspectos generales.

4. Especificación.

A continuación se presenta una breve descripción del juego, así como la descripción de los distintos entregables esperados.

Batalla Naval es un juego tradicional de adivinación que involucra a **dos participantes**. Los jugadores manejan dos tableros divididos en casillas, cada tablero representa una zona diferente del mar abierto: la propia y la contraria. En uno de los tableros, el jugador coloca sus barcos y registra los «tiros» del oponente; en el otro, se registran los tiros propios, al tiempo que se deduce la posición de los barcos del contrincante (objetivo del juego).

Objetivo del juego: Hundir los 9 barcos del oponente antes que él o ella acabe con los propios.

Cada jugador tiene dos tableros compuestos por 10 filas y 10 columnas.

1. **Tablero de posición:** Es la zona propia, en él se distribuye la propia flota antes de comenzar la partida, durante la partida se podrá observar la posición de los barcos propios, así como los disparos del oponente, pero no se podrán realizar cambios ni disparos en él.
2. **Tablero principal:** la zona del enemigo, donde tiene desplegada su flota. Será aquí donde se desarrollen los movimientos (disparos) para tratar de hundir los barcos enemigos. Aparecerá en la pantalla del jugador una vez comience la partida y en él quedarán registrados todos los movimientos, reflejando tanto los **disparos al agua** como los disparos acertados (golpes a barcos enemigos).

   Cada jugador tiene una flota de 9 barcos de diferente tamaño, por lo que cada uno ocupará un número determinado de casillas en el tablero: Barcos de la flota
   a. Un Portaaviones: ocupa 4 casillas
   b. Tres Submarinos/ Acorazados: ocupan 3 casillas.
   c. Tres Destructores: ocupan 2 casillas.
   d. Dos Fragatas: ocupan 1 casilla.

Terminología y movimientos: Para disparar es necesario indicar la casilla, indicando fila y columna y son posibles tres resultados tras un disparo:

1. **Agua:** cuando se dispara sobre una casilla donde no está colocado ningún barco enemigo, se dispara al agua. En el tablero principal aparecerá una X. Pasa al turno del oponente.
2. **Tocado:** cuando se dispara en una casilla en la que está ubicado un barco enemigo que ocupa 2 o más casillas y se destruye sólo una parte del barco, es decir que el barco ha sido tocado. En el tablero principal aparece esa parte del barco con fuego.

3. **Hundido:** si se dispara en una casilla en la que está ubicada una fragata (1 casilla) u otro barco con el resto de casillas tocadas, el barco se hunde, es decir, se elimina ese barco del juego. Esto deberá ser visible en el tablero.

El juego continúa hasta que alguna de las dos flotas haya sido hundida por completo.

A partir de la información anterior cada grupo de trabajo deberá modelar e implementar una solución orientada a objetos utilizando el lenguaje de programación Java. Las y los estudiantes que componen el grupo deberán modelar la información y la interacción del juego. Es fundamental que se preste atención y se dedique tiempo, no solamente a la programación, sino especialmente al modelado del sistema.

Con este proyecto se pretende que los grupos de trabajo manejen almacenamiento persistente, de modo que sea posible, en cualquier momento de la ejecución del programa guardar el estado del juego. Además al iniciar el juego por primera vez, será posible iniciar un juego nuevo o cargar uno almacenado. Estas funcionalidades deben implementarse utilizando archivos en formato Json, por lo que se deberán aplicar los conocimientos previamente adquiridos para la lectura de estos archivos y se deberá investigar para efectos de este proyecto, también la escritura.

Se espera que la implementación de la interfaz gráfica se realice utilizando el motor JavaFX[1], las actividades programadas por la profesora (tanto sincrónica como asincrónica) para la catorceava semana de clases, servirán a los grupos de trabajo como insumo para el uso de esta herramienta.

Cada grupo deberá presentar los siguientes entregables, según las fechas establecidas.

1. **Documentación:** Cada grupo de trabajo presentará un documento que deberá completarse según el calendario presente en la siguiente sección.
    a. Portada: Nombre de la institución, la sede, nombre del curso y la profesora, semestre actual, integrantes del grupo (con sus respectivos nombres y carnets).
    b. Diagrama de Casos de uso: Incluye los actores y casos de uso identificados.
    c. Descripciones Detalladas: Se presenta una descripción detallada para cada caso de uso identificado. Se incluye (i) actor primario, (ii) actor

[1] https://openjfx.io/openjfx-docs/ Guía de inicio para trabajar con JavaFX.

secundario, (iii) pre-condiciones, (iv) post-condiciones, (v) flujo principal y (vi) flujos alternos.

d. Diagrama de Clases: Incluya para cada clase (i) nombre de clase, (ii) atributos y (iii) métodos (incluya los tipos de datos), (iv) relaciones con etiqueta y cardinalidad explícita.

e. Diagramas de Secuencia: Muestra la secuencia de intercambio de mensajes en un grupo de objetos al realizar algunas tareas del sistema. Respete la simbología estudiada en clase para diagramar las siguientes acciones:

   i. Atacar. Un jugador realiza un ataque al tablero principal (del enemigo).

   ii. Cargar Juego almacenado: Se toma un archivo en formato JSON almacenado en la máquina local donde se ejecuta el juego y se cargan los valores para continuar con el juego.

   iii. Guardar Juego: toma los valores generados en tiempo de ejecución para almacenarlos de forma permanente en un archivo en formato JSON.

f. Diseño de la interfaz gráfica: Muestra una ilustración de la interfaz gráfica esperada, con los elementos gráficos esperados.

2. **Repositorio remoto:** Se deberá configurar un proyecto grupal en GitHub[2], de modo que todos los integrantes del grupo puedan clonar el proyecto y hacer sus aportes desde su máquina local. Se deberá incluir además a la profesora bajo el nombre de usuaria `tec-ramijan`. A partir de los diagramas realizados, el grupo deberá implementar el código de la solución utilizando ***Java SE versión 11 o superior***, dicho código deberá permitir realizar todas las acciones descritas en esta especificación. La interacción con el sistema deberá realizarse a través de una interfaz gráfica generada con ***JavaFX 15.***

3. **Video explicativo:** Una vez finalizados sus diagramas en formato UML y toda la programación necesaria para que el sistema funcione. Realice un video, capturando la pantalla de su computadora. Primero explique el diagrama de clases y después haga un recorrido por el código explicándolo también. Finalmente muestre una ejecución exitosa del código. El video deberá ser de entre 10 y 15 minutos. Cada estudiante deberá realizar su propio video, y publicarlo a través de YouTube.

[2] https://coursework.vschool.io/starting-a-group-project-on-github/ Starting a group project in GitHub.

5. Calendario.

Entregable	Fecha límite de Entrega
Acceso al repositorio remoto para les estudiantes que participan y la profesora.	Martes 1 de diciembre (11:59 pm)
Diagrama de Casos de Uso + Descripciones Detalladas	Jueves 3 de diciembre (3:00 pm)
Diagrama de Clases + Diseño de la Interfaz Gráfica	Martes 8 de diciembre (3:00 pm)
Diagramas de Secuencia	Jueves 10 de diciembre (3:00 pm)
Código completo en el repositorio	Martes 15 de diciembre (3:00 pm)
Video Explicativo	Jueves 17 de diciembre (3:00 pm)

6. Rúbrica.

Criterio	Indicadores de rendimiento		
	Muy bien	**Regular**	**Deficiente**
Diagrama de Casos de Uso	El diagrama respeta en su totalidad la nomenclatura UML. Se incluyen todas las funcionalidades descritas en esta especificación. Incluye además todos los actores y casos del sistema. 5	El diagrama respeta en su totalidad la nomenclatura UML, pero no se incluyen todos los actores o casos del sistema. 3	El diagrama irrespeta la nomenclatura UML. Hay errores graves o faltan representaciones de elementos importantes. 1
Descripciones Detalladas	La descripción describe un flujo válido del caso de uso. Se consideran	No se consideran posibles flujos alternos. No se respeta el formato	La descripción es escueta o incompleta.

	posibles flujos alternos. Se respeta el formato visto en clase.	visto en clase.	
	5	3	1
Diagrama de Clases	El diagrama respeta en su totalidad la nomenclatura UML. Se incluyen todas las entidades descritas en esta especificación. Hay una relación clara entre el diagrama de clases y el código. Las asociaciones son correctas, así como los modificadores de visibilidad, métodos y atributos.	El diagrama respeta en su totalidad la nomenclatura UML. No se incluyen algunas las entidades descritas en esta especificación.Hay faltas leves en cuanto a las asociaciones,los modificadores de visibilidad, métodos y atributos.	El diagrama no respeta en su totalidad la nomenclatura UML. Hay faltas graves en cuanto a las asociaciones,los modificadores de visibilidad, métodos y atributos.
	20	12	8
Diagramas de Secuencia	Se presentan los tres diagramas solicitados y estos respetan en su totalidad la nomenclatura UML. Se incluyen todos los objetos que interactúan en cada una de las acciones.	Se presentan los tres diagramas solicitados y estos respetan en su totalidad la nomenclatura UML. Se presentan faltas leves en los diagramas.	No se presenta alguno de los diagramas solicitados.
	10	6	4
Uso del repositorio	Se configura de manera correcta el repositorio grupal. Se evidencia la participación de todos los miembros del grupo de trabajo. Se incluyen todos los entregables descritos en esta especificación.	No se evidencia la participación equitativa de todos los miembros del grupo de trabajo. No se incluye alguno de los entregables descritos en esta especificación.	No hay contribuciones de parte de al menos uno de los miembros del grupo.
	10	6	4
Archivos JSON	Es posible cargar un archivo JSON de la máquina donde se ejecuta el juego (10).	Solo se puede cargar el archivo, pero no guardarlo, o ocurre lo contrario, es decir es	Hay código que evidencia un intento fallido por cargar y/o almacenar la

	Es posible guardar en un archivo JSON el estado actual del juego (10) 20	posible guardar pero no cargar. 10	información contenida en un archivo JSON. 5
Interacción del Juego con JavaFX	Se utiliza JavaFX para la implementación de los componentes gráficos (10). Es posible realizar ataques (5) Tras cada ataque se actualiza el estado de la flotilla atacada y se verifica si el juego llegó al final (5). 20	Se descuentan 10 puntos por utilizar java Swing o AWT para implementar la interfaz gráfica. Se descuentan 5 puntos si no es posible realizar ataques. Se descuentan 5 puntos si tras cada ataque no se actualiza el estado de la flotilla atacada o no se verifica si el juego llegó al final 10 - 15	
Video Explicativo (calificación individual)	El video cumple con todos los lineamientos solicitados en este documento. 10	El video no cumple con algunos de los lineamientos solicitados en este documento. 6	El video es poco informativo, no incluye diagramas, código y/o ejecución o evidencia falta de claridad del proyecto. 4
Puntaje total	**100 puntos**	**Valor porcentual**	**25%**

7. Aspectos Generales.

En el repositorio remoto deberá incluirse los siguientes entregables, antes de las fechas límite establecidas en la sección de calendario.

1. Cada una de las versiones de la documentación.
2. El código fuente de todas las clases necesarias para ejecutar el código.
3. Un archivo `README.md` en el que se incluya una breve descripción del proyecto y los enlaces a los videos de todos los integrantes del grupo.