Algoritmos y Programación III

Trabajo Práctico N°2 - AlgoBlocks

Arrua Rocío Ayelén; Lozano Ramiro; Sayos Alberto Daniel

Introducción

El presente informe reúne la documentación de la solución del segundo trabajo práctico de la materia Algoritmos y Programación III que consiste en desarrollar una aplicación de ejecución de una secuencia de bloques de movimientos que permiten crear un dibujo, utilizando los conceptos del paradigma de la orientación a objetos vistos hasta ahora en el curso.

Supuestos

A continuación se enumeran los supuestos que se realizaron en base al análisis de la aplicación, y a lo largo del desarrollo.

1. El sector dibujo se encuentra dividido en posiciones x e y, del estilo (0,0), (1,0), etc. Las posiciones del tablero son: de 0 a N para abajo (filas) y de 0 a N para la derecha (columnas).

2. Posición del personaje al inicio de la aplicación: (5,5).

3. El personaje se crea con el lápiz sujetado de forma que no se apoye sobre el dibujo (o sea que al moverse, por ahora, no se dibujará nada en el sector dibujo).

4. Para los bloques de Repetir e Invertir Comportamiento, suponemos que los tipos de bloques que aceptan alterar son solo los de Movimiento (Arriba, Abajo, Izquierda, Derecha) y Escritura (Subir Lapiz, Bajar Lapiz).

Diagramas de clase

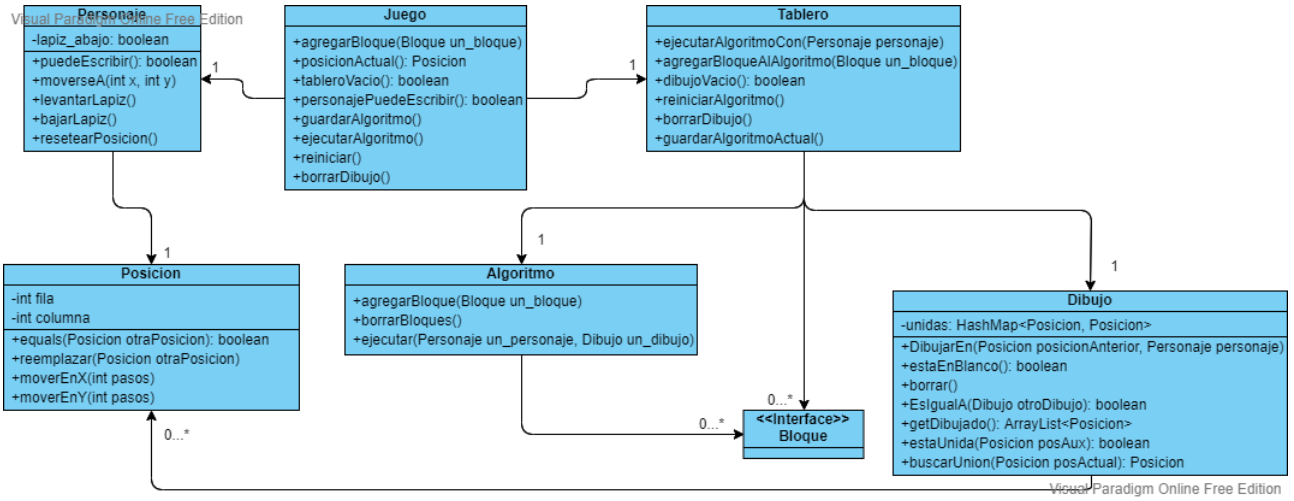

Figura: diagrama de clases para el objeto principal Juego

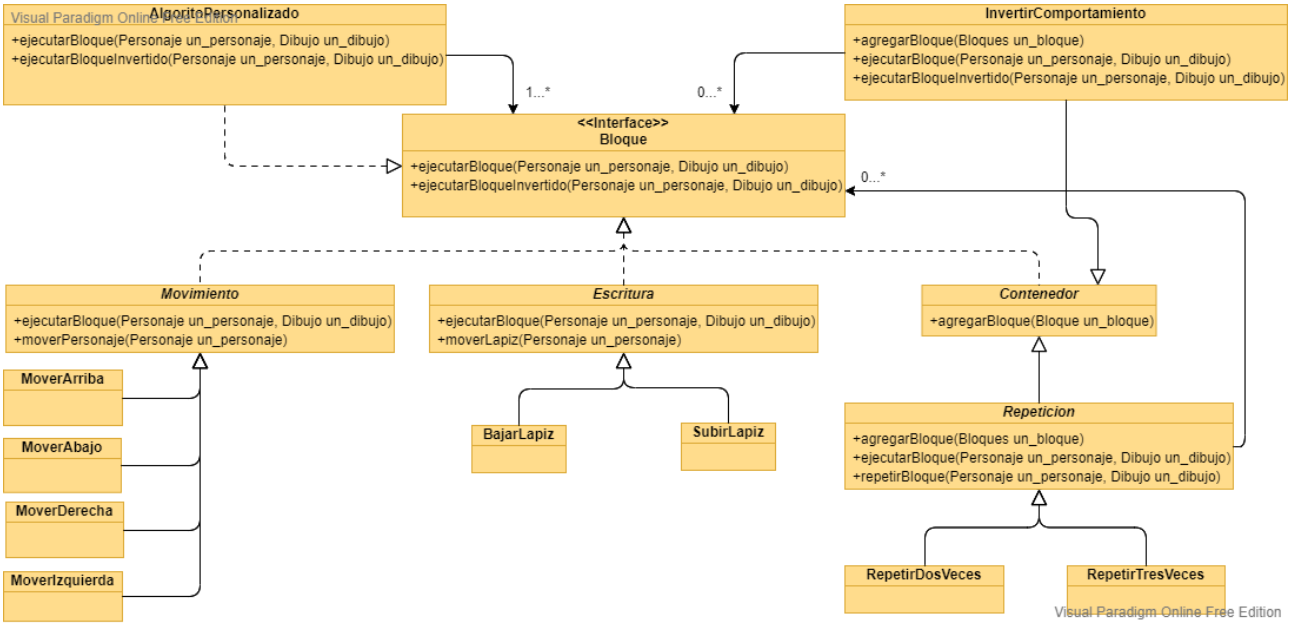

Figura: diagrama de clases de relaciones de herencia de Bloque

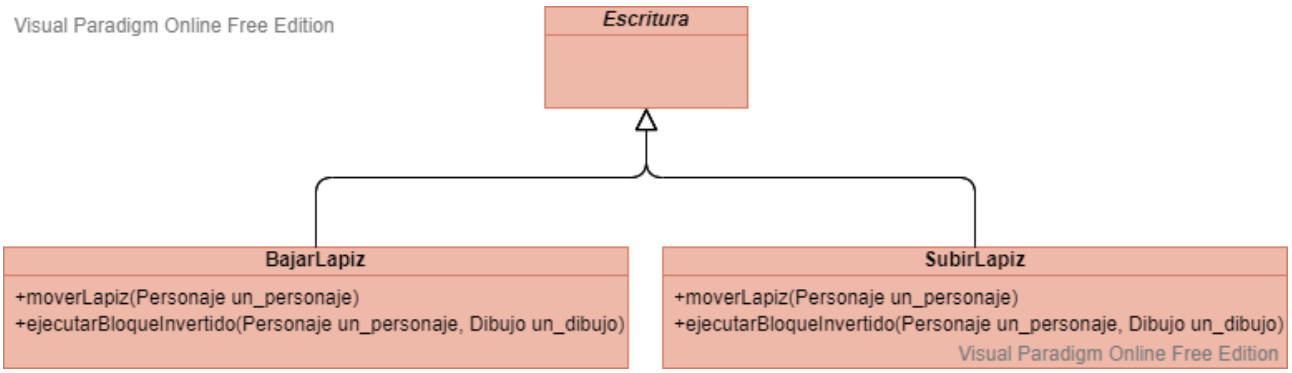

Figura: diagrama de clase abstracta Escritura

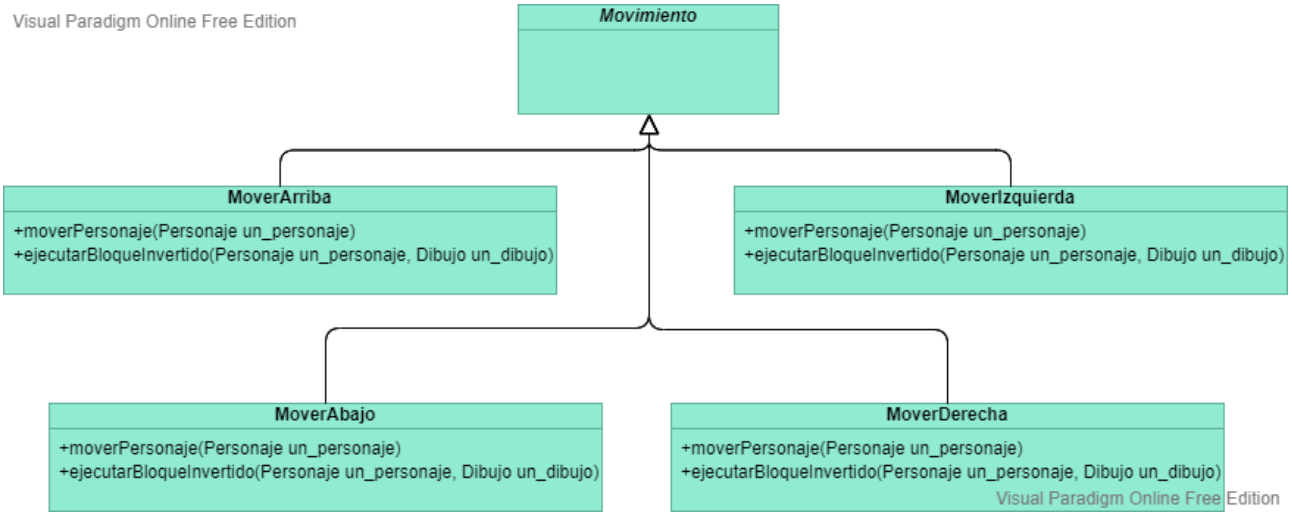

Figura: diagrama de clase abstracta Movimiento

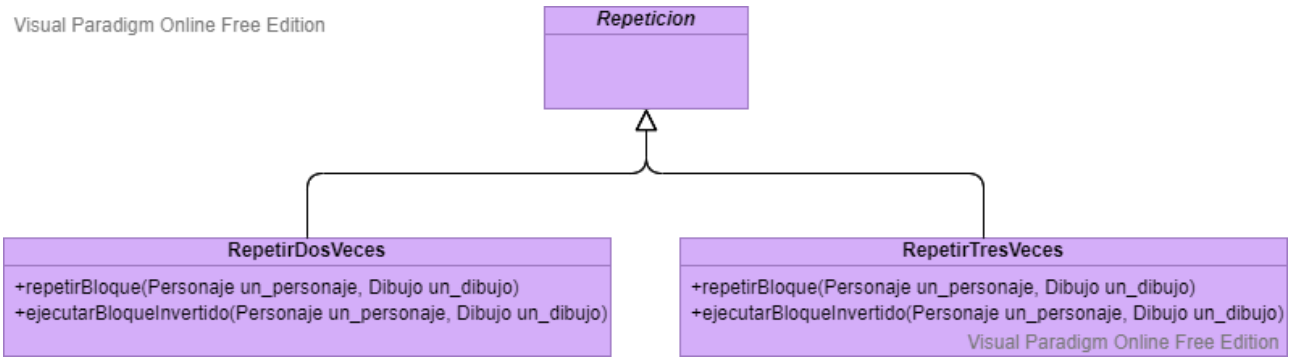

Figura: diagrama de clase abstracta Repeticion

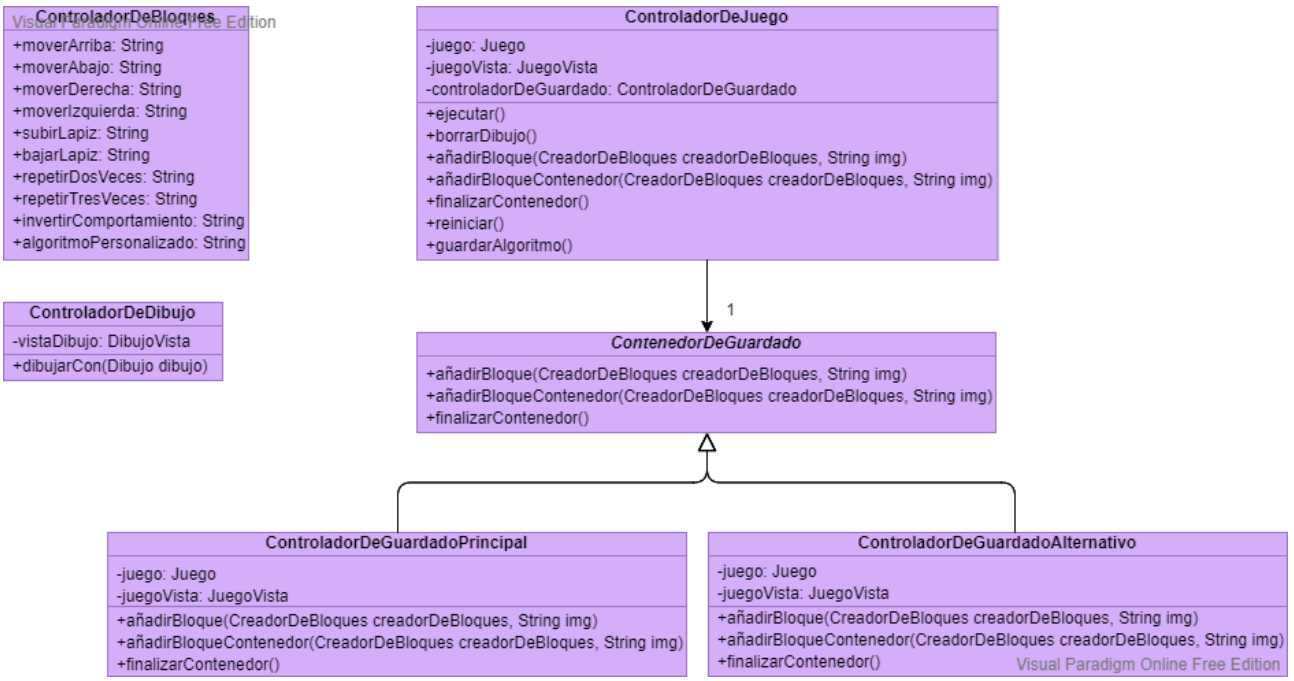

Figura: diagrama de clases para el paquete “control”

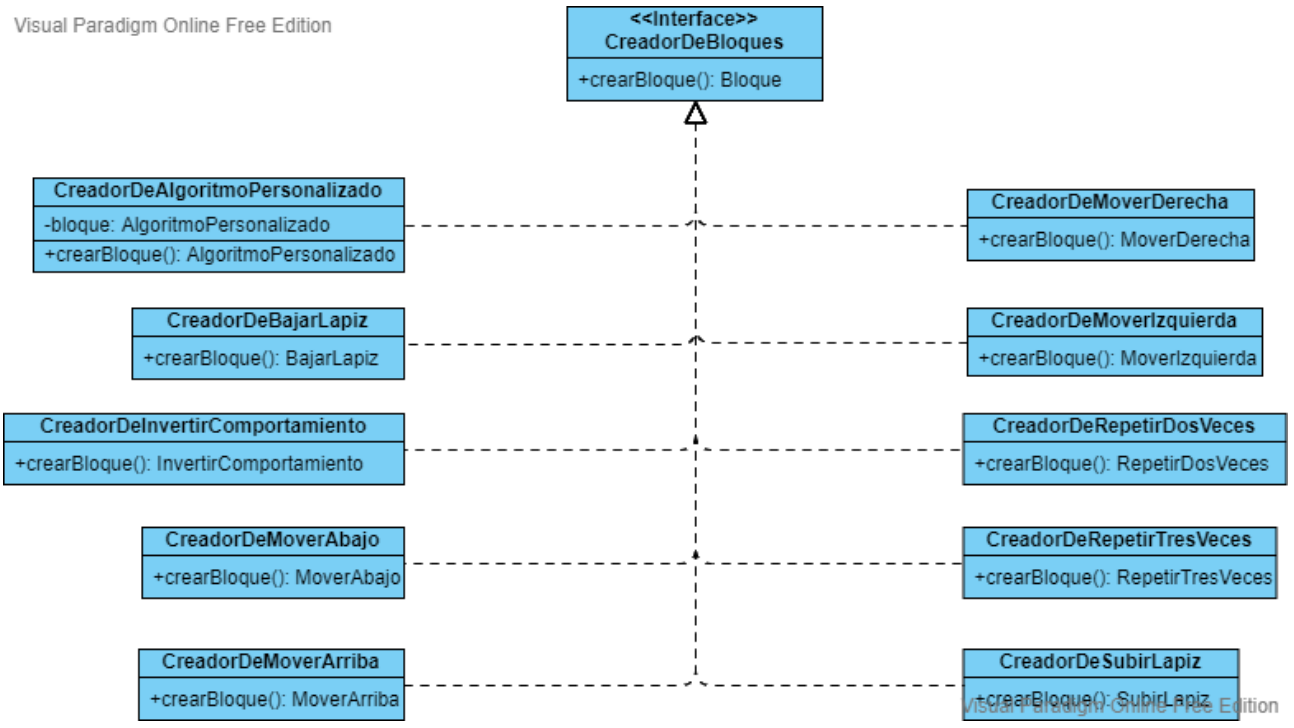

Figura: diagrama de clases para el paquete “creadores”

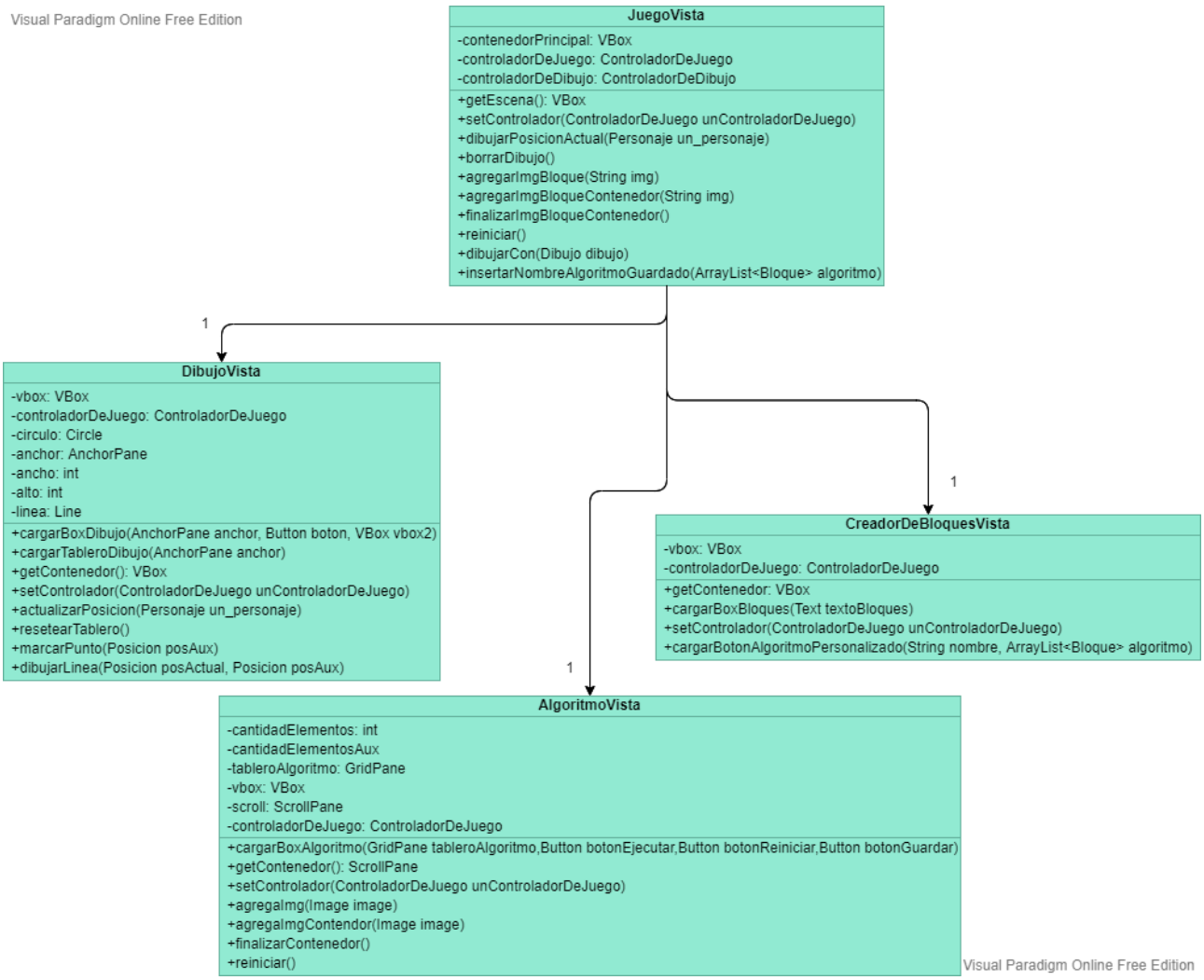

Figura: diagrama de clases para el paquete “visual”

Diagramas de secuencia

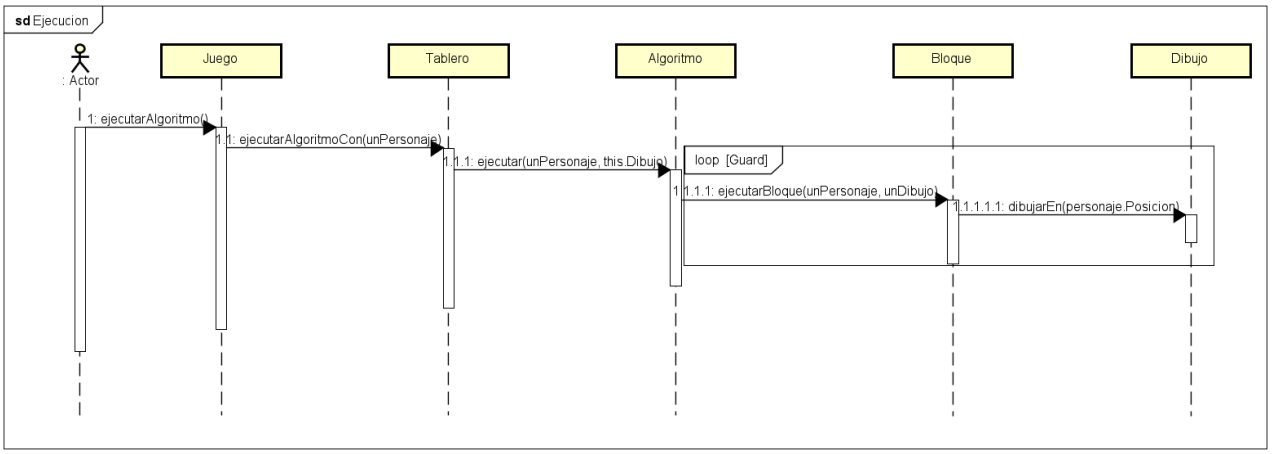

Figura: secuencia de ejecución de un algoritmo

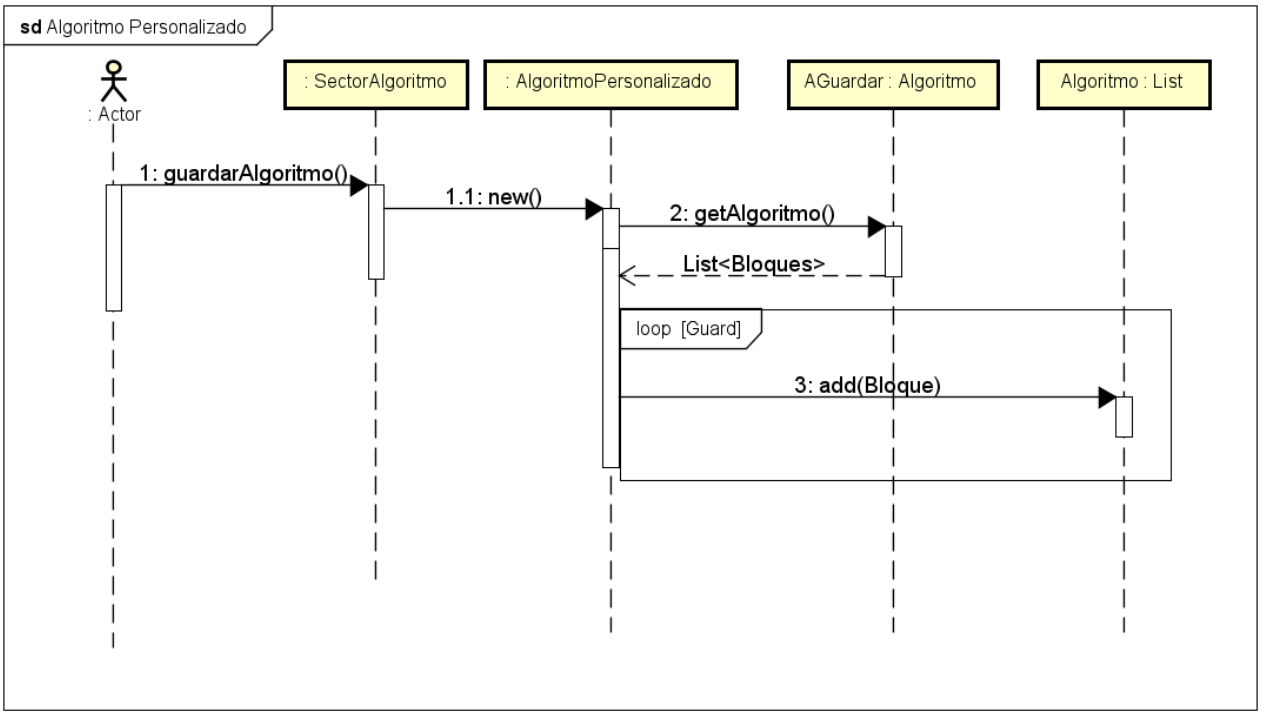

Figura: secuencia de creación de un algoritmo personalizado

Diagramas de paquetes

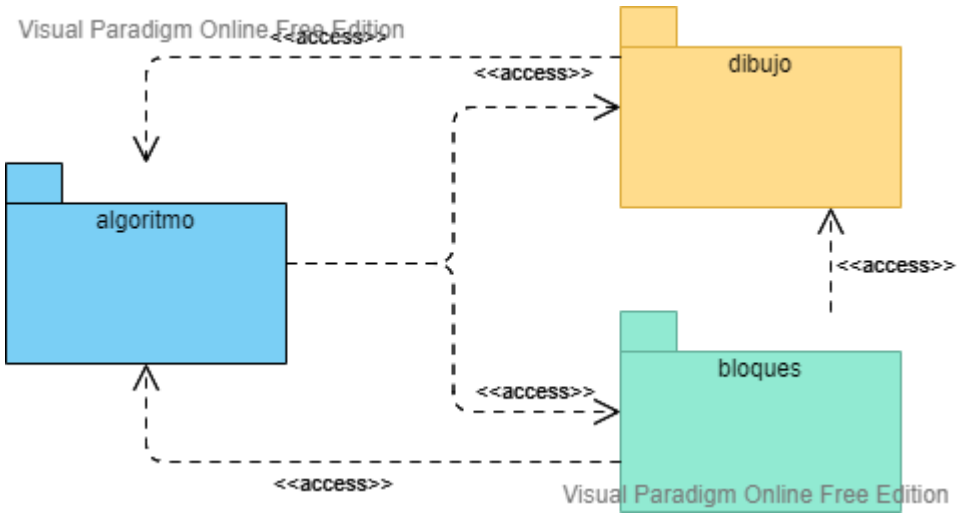

Figura: relación entre paquetes del modelo

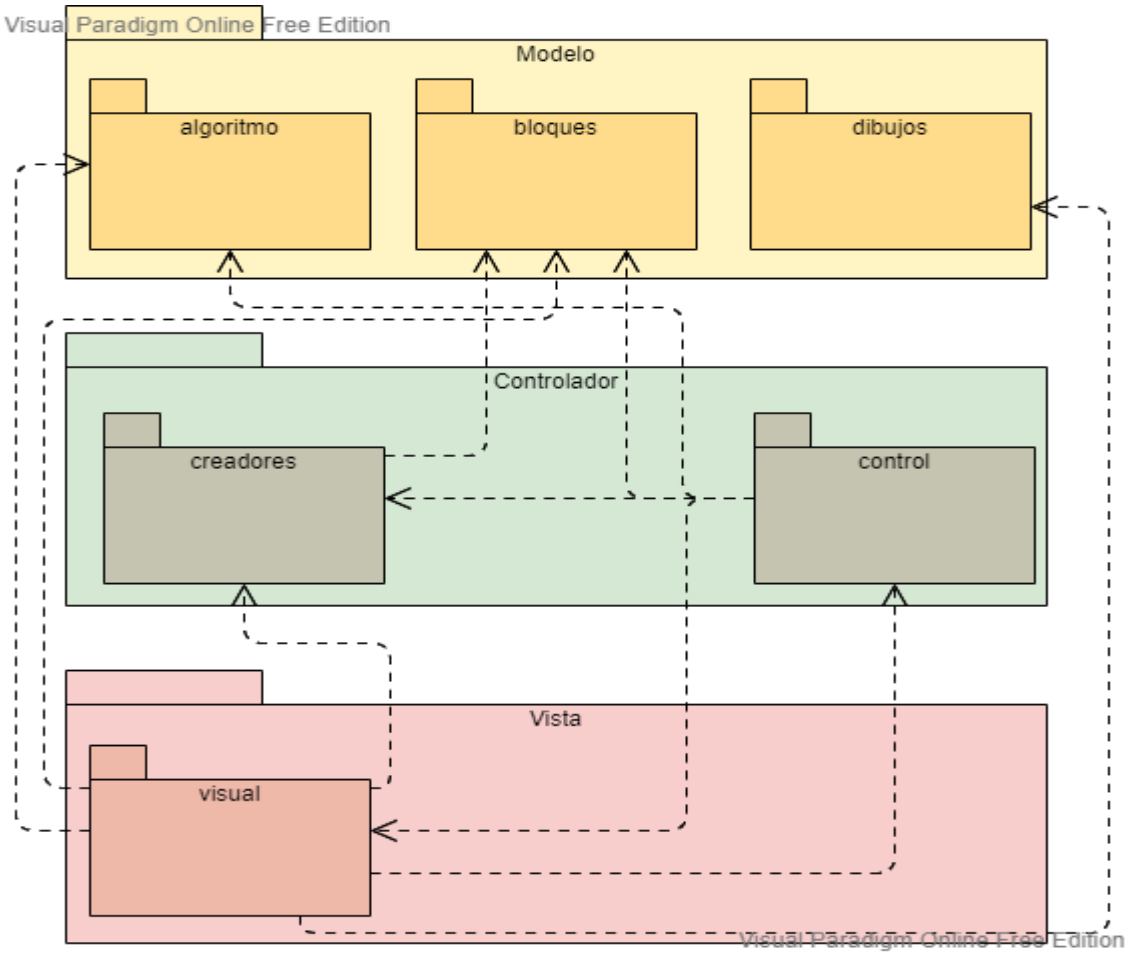

Figura: esquema MVC del programa

Diagramas de estado

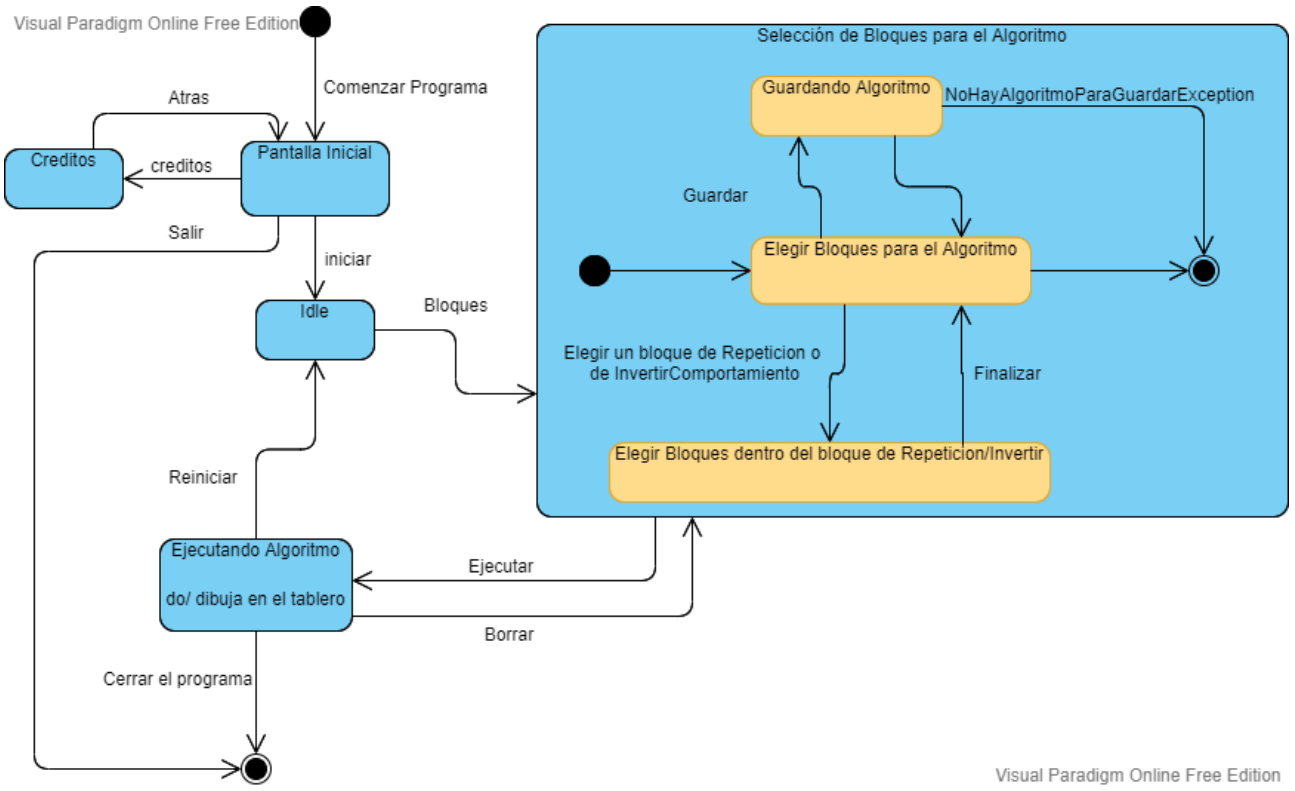

Figura: variación de estados al ejecutar el programa

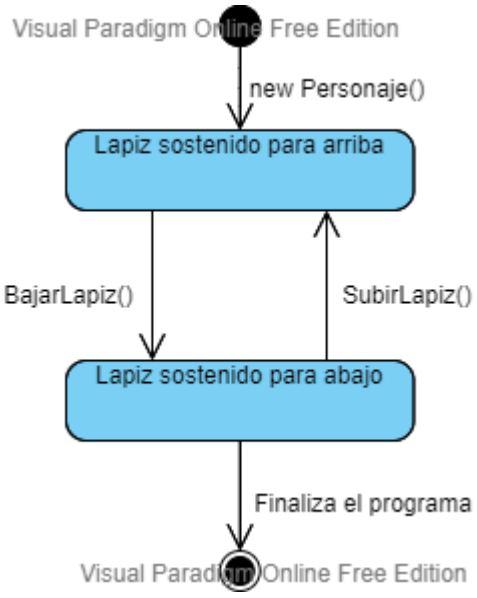

Figura: estados pertenecientes al movimiento de lápiz del Personaje

Detalles de implementación

A la hora de implementar la entidad Bloques, notamos que es un agente que puede tomar distintos estados, cada uno con funciones y usos diferentes. Como consecuencia, se optó por implementar dicho objeto como una interfaz de la cual hereden 3 objetos abstractos que definen el tipo de carácter que describiría cada bloque: Escritura (BajarLapiz, SubirLapiz), Movimiento (MoverArriba, MoverAbajo, MoverIzquierda, MoverDerecha) y Repeticion (RepetirDosVeces, RepetirTresVeces). Así, logramos mantener la abstracción del modelo utilizando polimorfismo y ayudando a hacer más flexible la aplicación a la hora de agregar o corregir diferentes tipos de bloques.

Patrón Creador: se implementó este patrón, para ceder la responsabilidad de crear bloques a cada entidad creadora de los distintos bloques

Patrón MVC (modelo, vista, controlador): se implementó este patrón para diferenciar las responsabilidades y disminuir el acoplamiento del modelo (objetos), la vista (interfaz) y los controladores de la aplicación.

Excepciones

- **NoHayAlgoritmoGuardadoException**: a la hora de pedir guardar un algoritmo, en el caso de que no se haya insertado ningún tipo de bloque en el algortimo antes de realizar esta acción, salta este error.