Ingeniería en computación

Universidad de La Serena

II Semestre, 2020

Versión número 1

Diseño

**AIPA prototipo I**

*Miercoles 13 de octubre*

**Integrantes:**Yair Gallardo   
Norton Irarrázabal   
Sebastian Rojas

**Docente**   
Guillermo Leyton.

# Propósito

Este documento tiene como finalidad especificar el diseño sobre el cual se desarrollará el software Aipa, facilitando su construcción y siendo la base para su implementación.

Está dirigido a los desarrolladores del software y a las partes involucradas en la asignatura software II:

* SQA.
* Docente.

# Descripción del problema

El uso de inteligencia artificial en el ámbito de los jugadores virtuales, es una **herramienta de uso investigativo y publicitario** que cuenta con larga data dentro de los últimos 30 años. En el área investigativa, se pretende encontrar el alcance de las capacidades de simulación racional con las que cuentan las máquinas; el logro de altos niveles por estos jugadores virtuales suele generar gran interés en el público.

Otra perspectiva a valorar es el amplio mercado de los videojuegos, donde la dificultad, la búsqueda de retos y experiencias satisfactorias suelen ser una de las áreas de interés para los jugadores más exigentes.

Intereses que pueden ser abordados por estos jugadores virtuales, siendo un reto y por otra una “ayuda” para jugadores menos experimentados que busquen encontrar o descubrir nuevas jugadas que faciliten su éxito mejorando su experiencia y satisfacción. Por consiguiente, es de interés para empresas desarrolladoras que buscan satisfacer a este tipo de clientes.

Si bien el software a desarrollar se desenvolverá en un ámbito concreto, juego de estrategia **ajedrez**; el enfoque es el mismo, siendo un buen prototipo aplicable en otros entornos.

# Diseño

Para la implementación del software se utilizará diseño por contrato, por lo tanto, se especificarán los métodos pertenecientes a las clases con su respectiva precondición (condiciones de entrada, que se espera) y postcondición (condiciones de salida, que se garantiza).

El diseño del software se especificará mediante:

* Los paquetes.
* Las clases.
* Las clases y sus métodos.
* Métodos por contrato.
* Muestra visual.

## Especificación de paquetes

|  |  |  |
| --- | --- | --- |
| **ID** | **Nombre** | **Descripción** |
| P01 | Controlador | Aquí se encontrarán todas las clases necesarias para dar soporte de comunicación entre el modelo y las vistas. |
| P02 | Modelo | En este paquete se encontrarán los objetos que conforman el software. |
| P03 | Vista | Se encontrarán todas las vistas del software. |

## Especificación de clases

|  |  |  |
| --- | --- | --- |
| **ID** | **Paquete** | **Clase** |
| C01 | Controlador | DrawHandler : IDisposable |
| C02 | Historial\_acciones |
| C03 | Historial\_movimiento |
| C04 | Sprite |
| C05 | Tiempo |
| C06 | Modelo | Alfil |
| C07 | Caballo |
| C08 | Celda\_tablero |
| C09 | Jugador |
| C11 | enum Tipo\_de\_jugador |
| C12 | Movimiento |
| C13 | enum Tipo\_de\_movimiento |
| C14 | Peon |
| C15 | Pieza |
| C16 | enum UnColor |
| C17 | Reina |
| C18 | Rey |
| C19 | Tablero |
| C20 | Torre |
| C21 | Vista | Recursos |
| C22 | Ventana\_bienvenida |
| C23 | Ventana\_coronacion |
| C24 | Ventana\_juego |

## Especificación de clases y sus métodos

|  |  |  |
| --- | --- | --- |
| **ID** | **Clase** | **Métodos** |
| M01 | DrawHandler : IDisposable | DrawHandler() |
| M02 | Dispose() |
| M03 | Draw(precondición) |
| M04 | Historial\_acciones | Historial\_acciones() |
| M05 | Historial\_movimiento | ***propiedades*** |
| M06 | Sprite | Sprite(precondición) |
| M07 | **(Virtual)** Draw(DrawHandler drawHandler) |
| M08 | Tiempo | Tiempo() |
| M09 | Alfil : Pieza | Alfil(precondición) : base(image, color) |
| M10 | Caballo : Pieza | Caballo(precondición) : base(image, color) |
| M11 | Celda\_tablero | ***propiedades*** |
| M12 | Jugador | Jugador(precondición) |
| M13 | enum Tipo\_de\_jugador | ***Clase enumerada*** |
| M14 | Movimiento | Movimiento(precondición) |
| M15 | enum Tipo\_de\_movimiento | ***Clase enumerada*** |
| M16 | Peon : Pieza | Peon(precondición) : base(image, color) |
| M17 | abstract Pieza : Sprite | Pieza(precondición) : base(image, new Point()) |
| M18 | **(Override)** Draw(precondición) |
| M19 | enum UnColor | ***Clase enumerada*** |
| M20 | Reina : Pieza | Reina(precondición) : base(image, color) |
| M21 | Rey : Pieza | Rey(precondición) : base(image, color) |
| M22 | Tablero : Sprite | Tablero(precondición) : base(boardImage, new Point()) |
| M23 | Desmarcar\_celdas() |
| M24 | **(Override)** void Draw(precondición) |
| M25 | Torre : Pieza | Torre(precondición) : base(image, color) |
| M26 | Recursos | ***propiedades*** |
| M27 | partial Ventana\_bienvenida : Form | Ventana\_bienvenida() |
| M28 | Timer1\_Tick(precondición) |
| M29 | Timer2\_Tick precondición |
| M30 | Ventana\_bienvenida\_Load\_1(precondición) |
| M31 | partial Ventana\_coronacion : Form | Ventana\_coronacion(precondición) |
| M32 | Reina\_click(precondición) |
| M33 | Alfil\_click(precondición) |
| M34 | Caballo\_click(precondición) |
| M35 | Torre\_click(precondición) |
| M36 | partial Ventana\_juego : Form | Ventana\_juego() |
| M37 | Boton\_cerrar\_Click(precondición) |
| M38 | Boton\_minimizar\_Click\_1(precondición) |
| M39 | Barratitulo\_MouseMove(precondición) |
| M40 | Boton\_maximizar\_Click(precondición) |
| M41 | Boton\_restaurar\_Click(precondición) |
| M42 | Boton\_activar\_consejos(precondición) |
| M43 | Boton\_manual\_usuario\_click  (precondición) |
| M44 | btnStart\_Click(precondición) |
| M45 | Ventana\_juego\_Canvas\_MouseUp  (precondición) |
| M46 | Initialize() |
| M47 | Cargar\_imagen(precondición) |
| M48 | Comenzar\_juego() |
| M49 | Siguiente\_turno(precondición) |
| M50 | Obtener\_ubicacion\_pieza(precondición) |
| M51 | Añadir\_pieza(precondición) |
| M52 | Remover\_pieza(precondición) |
| M53 | Set\_pieza\_seleccionada(precondición) |
| M54 | Set\_movimientos\_posibles() |
| M55 | Obtener\_movimientos\_posibles  (precondición) |
| M56 | Validar\_movimiento(precondición) |
| M57 | Validar\_movimiento\_especial  (precondición) |
| M58 | Mover\_pieza(precondición) |
| M59 | Movimiento\_especial\_pieza(precondición) |
| M60 | Update(precondición) |
| M61 | Draw(precondición) |
| M62 | enum Estado | ***Clase enumerada*** |

## Especificación de métodos por contrato

|  |  |  |
| --- | --- | --- |
| **Precondición** | **Método** | **Postcondición** |
| Controlador | | |
| Int widht  Int height | DrawHandler() | Asignación de propiedades:  Image Imagen\_base,  Graphics graphics |
| Graphics graphics | Dispose() | Libera los recursos utilizados por Graphics graphics |
| Image image  Point position  Graphics graphics | Draw(precondicion) | Dibujar imagen en pantalla |
|  | Historial\_acciones() | Asignación de propiedades: List<Historial\_movimiento>, List<Pieza> Pieza\_removida, Pieza Pieza\_añadida |
| Image image  Point Ubicacion | Sprite(precondición) | Asignación de propiedades  Image image,  Point Posicion |
| DrawHandler drawHandler | (virtual) Draw(DrawHandler drawHandler) | Dibuja todos los sprites en pantalla |
|  | Tiempo() | Asignación de propiedades:  DateTime Fecha\_inicial,  DateTime Frame\_fecha\_actual |
| Modelo | | |
| Image image,  UnColor color | Alfil(precondición)  : base(image, color) | Asignación de propiedades:  Image image,  UnColor color,  Movimiento Movimientos |
| Image image,  UnColor color | Caballo(precondición)  : base(image, color) | Asignación de propiedades:  Image image,  UnColor color,  Movimiento Movimientos |
| UnColor color, Tipo\_de\_jugador tipo\_jugador,  int numero | Jugador(precondición) | Asignación de propiedades:  UnColor Color,  Tipo\_de\_jugador Tipo\_jugador,  int Numero |
| int x,  int y,  bool isLinear = true, Tipo\_de\_movimiento tipo\_de\_movimiento = Tipo\_de\_movimiento.normal | Movimiento() | Asignación de propiedades:  Point Direccion,  Tipo\_de\_movimiento Tipo\_de\_mov,  bool IsLinear |
| Image image,  UnColor color | Peon(precondición)  : base(image, color) | Asignación de propiedades:  Image image,  UnColor color,  Movimiento Movimientos |
| Image image,  UnColor color | Pieza(precondición)  : base(image, new Point()) | Asignación de propiedades:  Image image,  UnColor color |
| DrawHandler drawHandler | override void Draw(precondición) | Dibuja todos los sprites en pantalla |
| Image image,  UnColor color | Reina(precondición)  : base(image, color) | Asignación de propiedades:  Image image,  UnColor color,  Movimiento Movimientos |
| Image image,  UnColor color | Rey(precondición)  : base(image, color) | Asignación de propiedades:  Image image,  UnColor color,  Movimiento Movimientos |
| Image boardImage,  Image mover\_imagen | Tablero(precondición) : base(boardImage, new Point()) | Asignación de propiedades:  Image boardImage,  Image Mover\_imagen.  Generar instancia con la información del tablero de juego, particularmente ubicación de cada celda. |
|  | Desmarcar\_celdas() | Desmarcar celdas que permiten movimiento |
| DrawHandler drawHandler | override void Draw(precondición) | Dibuja todos los sprites en pantalla |
| Image image,  UnColor color | Torre(precondición)  : base(image, color) | Asignación de propiedades:  Image image,  UnColor color,  Movimiento Movimientos |
| Vista | | |
| showDialog() | Ventana\_bienvenida() | Inicializa los componentes |
| object sender,  EventArgs e | Timer1\_Tick(precondición) | Hace visible el form gradualmente según un timer |
| object sender,  EventArgs e | Timer2\_Tick precondición | Hace invisible el form gradualmente según un timer luego cierra el form |
| object sender,  EventArgs e | Ventana\_bienvenida\_Load\_1  (precondición) | Controla el evento de carga del formulario. |
| Recursos recursos, UnColor color | Ventana\_coronacion(precondición) | Inicializa los componentes y propiedades |
| object sender,  EventArgs e | Reina\_click(precondición) | Guarda la selección del usuario de la coronación y cierra el form |
| object sender,  EventArgs e | Alfil\_click(precondición) | Guarda la selección del usuario de la coronación y cierra el form |
| object sender,  EventArgs e | Caballo\_click(precondición) | Guarda la selección del usuario de la coronación y cierra el form |
| object sender,  EventArgs e | Torre\_click(precondición) | Guarda la selección del usuario de la coronación y cierra el form |
| Application.Run | Ventana\_juego() | Inicializa los componentes |
| object sender,  EventArgs e | Boton\_cerrar\_Click(precondición) | Cierra el form |
| object sender,  EventArgs e | Boton\_minimizar\_Click\_1(precondición) | Minimiza el form |
| object sender, MouseEventArgs e | Barratitulo\_MouseMove(precondición) | Mover el form |
| object sender,  EventArgs e | Boton\_maximizar\_Click(precondición) | Maximizar el form |
| object sender, EventArgs e | Boton\_restaurar\_Click(precondición) | Restaura el form |
| object sender, EventArgs e | Boton\_activar\_consejos(precondición) | Permite activar los consejos del agente y modifica el color del botón |
| object sender, EventArgs e | Boton\_manual\_usuario\_click  (precondición) | Permite apertura del manual y modifica el color del botón |
| object sender, EventArgs e | btnStart\_Click(precondición) | Permite comenzar la partida desde 0. |
| object sender, MouseEventArgs e | Ventana\_juego\_Canvas\_MouseUp  (precondición) | Muestra los movimientos disponibles para la pieza seleccionada en el tablero |
|  | Initialize() | Permite cargar los recursos necesarios para mostrar el tablero particularmente imágenes. |
| string path | Cargar\_imagen(precondición) | Devuelve una variable de tipo Image |
|  | Comenzar\_juego() | Asignación de propiedades:  List<Historial\_acciones>ActionLog  List<Pieza> Piezas  Tablero (Image,Image) Board  Estado GameState,  Añade todas las piezas al tablero y llama al siguiente turno. |
| bool firstTurn | Siguiente\_turno(precondición) | Finaliza el turno actual y recalcula los movimientos para cada pieza del tablero,  Recalcula los movimientos disponibles,  Revisa si el rey se encuentra en jaque,  Asigna el estado al match, |
| Point location | Obtener\_ubicacion\_pieza(precondición) | Devuelve coordenadas (modificadas) de la pieza acordes al tablero (Point). |
| Pieza piece, Historial\_acciones log = null | Añadir\_pieza(precondición) | Permite añadir una pieza al listado de piezas |
| Pieza piece, Historial\_acciones log | Remover\_pieza(precondición) | Elimina una pieza del listado de piezas |
| Point cell\_Location | Set\_pieza\_seleccionada(precondición) | Desmarca las celdas del tablero,  Deselecciona todas las piezas,  Colorea las celdas donde tiene movimiento la pieza seleccionada. |
| List<Pieza> Piezas | Set\_movimientos\_posibles() | Obtener los movimientos habilitados para cada pieza. |
| Pieza piece, List<Pieza> boardPiezas | Obtener\_movimientos\_posibles  (precondición) | Envia un arreglo con las coordenadas de los movimientos a la cual puede moverse una pieza |
| Pieza piece, Point newLocation | Validar\_movimiento(precondición) | Devuelve un booleano true si el movimiento es valido false en caso contrario, esto es para evitar que quede en jaque al mover una pieza. |
| Pieza piece, Movimiento move,  Pieza rivalPiece | Validar\_movimiento\_especial  (precondición) | Devuelve un booleano que revisa si puede desplazarse la pieza seleccionada utilizando un movimiento especial, true si puede false en caso contrario |
| Point cell\_Location | Mover\_pieza(precondición) | Devuelve un booleano que determina si es posible mover la pieza a la ubicación especificada, además pinta en el tablero la pieza desplazada en su nueva ubicación y llama al siguiente turno. |
| Pieza piece, Point targetLocation, Historial\_acciones log | Movimiento\_especial\_pieza(precondición) | Ejecuta un movimiento especial, por ejemplo, coronación, comer al paso, enroque corto y largo. |
| Tiempo gameTime | Update(precondición) | Actualiza la posición de todas las piezas en el tablero |
| DrawHandler drawHandler, | Draw(precondición) | Dibuja todos los sprites en pantalla |

## Muestra

**Ventana de bienvenida**



**Ventana del juego**



**Ventana cuando ocurre selección de pieza**



**Ventana cuando se requiere coronación de peón.**



*Nota Prototipo 1: Se encuentra desarrollado el aspecto visual del software y también el aspecto lógico de todas las reglas del juego incluyendo movimientos especiales:*

* *Al paso.*
* *Enroque.*
* *Coronación.*