Universidad Católica del Norte.
Facultad de Ingeniería y Ciencias Geológicas.
Departamento de Ingeniería de Sistemas y Computación.

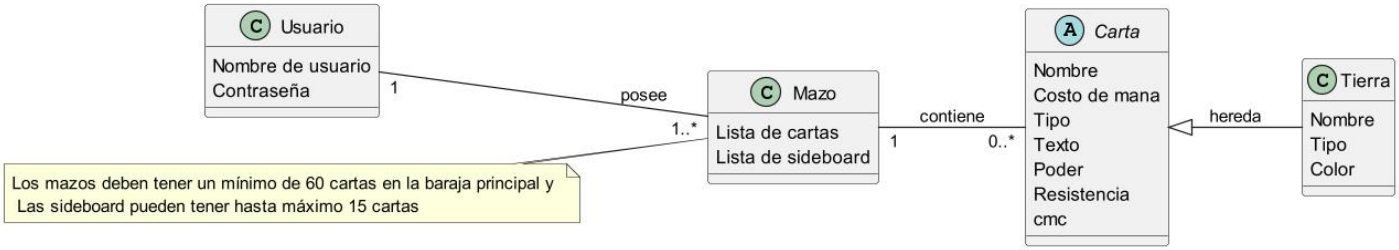
CONTRATOS E INFORME TALLER 2

Programación Avanzada

Franco Olivos Macaya
Franco.olivos@alumnos.ucn.cl
21.645.360-3
Paralelo C2

Profesor: Tomás Reimann
Ayudante: Edgardo Ortiz

Modelo de Dominio



Explicación dominio:

Usuario:
- Representa al usuario al que se le pedirá que inicie sesión con su nombre y contraseña.

- Un Usuario puede poseer uno o varios Mazos, pero cada Mazo pertenece a un único Usuario.

Mazo:
- Representa a las cartas que hay en el sistema, las cuales están contenidas en la "Lista de cartas" que son las cartas que forman parte del mazo principal y las "Lista de sideboard" que son las cartas que tendrá disponible para modificar el mazo principal. También contiene sus listas para luego implementar los métodos correspondientes al sistema.

- Un Mazo puede contener varias Cartas, pero una Carta puede estar en varios Mazos.

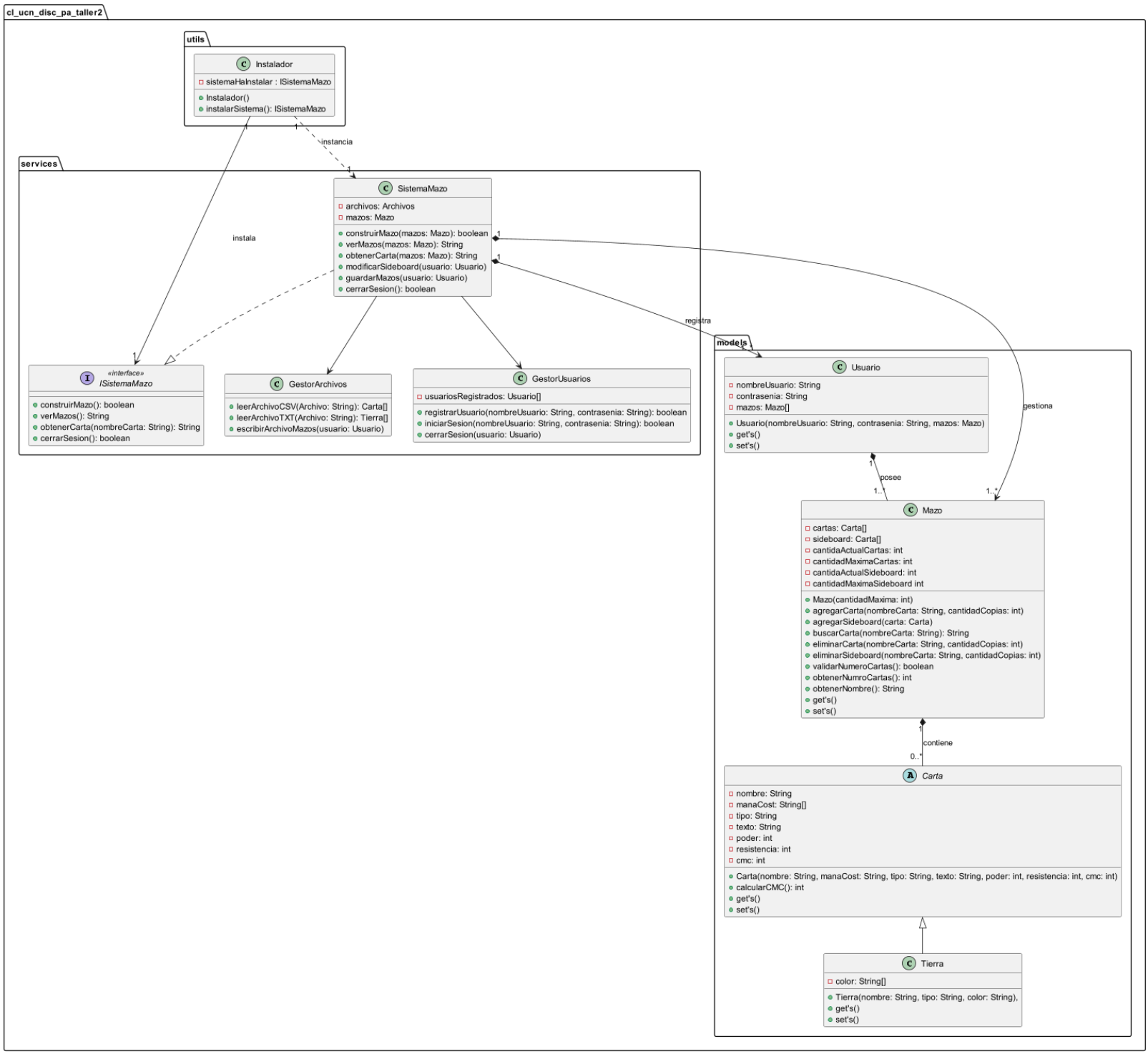
Cartas:
- Es la clase base para las cartas del juego, de esta se puede desprender la clase de "Tierra" que hace referencia al mana o recurso para poder invocar las cartas.

- Las Cartas pueden ser de diferentes tipos, y se representa la relación de herencia con la clase abstracta Carta.

Tierra:
- Hereda de la clase "Carta" y agrega atributos como color o el tipo de carta.

- Las Tierras son un tipo específico de Carta, por lo que heredan los atributos y comportamientos de la clase Carta.

Diagrama de clases



Explicación Diagrama de Clases:

1. **Carta**: Es la clase abstracta que funciona como clase padre o superclase para las cartas en el juego. Tiene atributos como nombre, coste de mana, tipo, texto, poder, resistencia y cmc. También tiene métodos para calcular el CMC y getters y setters para sus atributos.

2. **Tierra**: Es la subclase de la clase Carta. Tiene su atributo el cual es color, y sus propios métodos getter y setter.

3. **Mazo**: Esta clase representa un mazo de cartas. Tiene una lista de Cartas y una lista de Sideboard, que también son Cartas. También tiene métodos para agregar y eliminar cartas del mazo y del sideboard, validar el número de cartas y obtener el número y el nombre de las cartas.

4. **Usuario**: Esta clase representa a un usuario del sistema. Tiene atributos para el nombre de usuario, la contraseña y una lista de Mazos. También tiene métodos getter y setter para sus atributos.

5. **ISistemaMazo**: Es la interfaz que define los métodos que debe implementar el sistema de mazos.

6. **SistemaMazo**: Esta clase implementa la interfaz ISistemaMazo. Tiene atributos para los archivos y los mazos, y métodos para construir mazos, ver mazos, buscar cartas, modificar el sideboard, eliminar cartas, guardar mazos y cerrar sesión.

7. **GestorArchivos**: Esta clase se encarga de gestionar la lectura de archivos CSV y TXT los cuales contienen las cartas que se utilizaran en el sistema, además de escribir el archivo de mazos.

8. **GestorUsuarios**: Esta clase tiene una lista de usuarios registrados y métodos para registrar usuarios, iniciar sesión y cerrar sesión, los cuales se usarán en el sistema.

9. **Instalador**: Se encarga de instalar el sistema y permitir que el programa interactúe con la clase principal Main.

Relaciones entre clases:

Carta y Tierra:

La clase Tierra hereda de la clase abstracta Carta. Esto significa que una carta tierra es un tipo específico de carta y comparte características comunes.

Mazo y Carta:

La relación entre Mazo y Carta indica que un Mazo contiene una colección de cartas, tanto en su lista principal como en su sideboard. Esto significa que un Mazo está compuesto por una o más instancias de la clase Carta.

Usuario y Mazo:

La relación entre Usuario y Mazo indica que un Usuario puede poseer uno o más Mazos. Cada Mazo pertenece a un único Usuario.

SistemaMazo e ISistemaMazo:

SistemaMazo implementa la interfaz ISistemaMazo, lo que significa que proporciona una implementación concreta de los métodos definidos en la interfaz ISistemaMazo.

SistemaMazo, GestorArchivos y GestorUsuarios:

SistemaMazo tiene una relación de dependencia con GestorArchivos y GestorUsuarios. Esto indica que SistemaMazo utiliza los servicios proporcionados por GestorArchivos y GestorUsuarios, pero no está directamente compuesto por ellos.

SistemaMazo, Mazo y Usuario:

SistemaMazo tiene una relación de composición con Mazo y Usuario. Esto indica que SistemaMazo está compuesto por instancias de Mazo y Usuario. Por lo tanto, SistemaMazo contiene instancias de Mazo y Usuario, y los gestiona.

Instalador, SistemaMazo e ISistemaMazo:

Instalador tiene una relación de dependencia con SistemaMazo y ISistemaMazo. Esto significa que Instalador utiliza los servicios proporcionados por SistemaMazo a través de la interfaz ISistemaMazo.

Contratos

OPERACIONES	construirMazo(Mazo mazos)
DESCRIPCION	Construye un mazo y lo agrega a la lista de mazos del usuario.
PRECONDICIONES	El mazo a construir no debe estar vacío y debe contener al menos 60 cartas en la baraja principal y no puede tener ams de 4 cartas iguales, a excepción de las “Tierra”.
POSTCONDICIONES	Se agrega el mazo construido a la lista de mazos del usuario y se devuelve true si se construyó correctamente, false en caso contrario.

OPERACIÓN	verMazos(Mazo mazos)
DESCRIPCION	Muestra los mazos del usuario.
PRECONDICIONES	Debe haber al menos un usuario registrado con un mazo y que este no esté vacío.
POSTCONDICIONES	Se muestran los mazos correspondientes al usuario.

OPERACIÓN	buscarCarta(String nombreCarta)
DESCRIPCION	Busca una carta por su nombre y muestra sus detalles.
PRECONDICIONES	La carta debe existir en el sistema, la lista donde están almacenadas las cartas no puede ser vacía ni negativa.
POSTCONDICIONES	Se muestra la carta que está buscando con sus respectivos detalles e información.

OPERACIÓN	modificarSideboard(usuario: Usuario)
DESCRIPCION	Permite al usuario modificar el sideboard de su mazo.
PRECONDICIONES	El usuario debe estar registrado y tener al menos un mazo con sideboard.
POSTCONDICIONES	Se realizan las modificaciones en el sideboard del mazo del usuario.

OPERACIÓN	eliminarCarta(String nombre, int cantidadCopias)
DESCRIPCION	Permite eliminar una carta del mazo del usuario.
PRECONDICIONES	El usuario debe estar registrado, tener al menos un mazo con cartas y debe existir en el mazo la carta que se desea eliminar.
POSTCONDICIONES	Se elimina la carta seleccionada del mazo del usuario.

OPERACIÓN	guardarMazos(usuario: Usuario)
DESCRIPCION	Guarda el mazo que agrego el usuario al sistema.
PRECONDICIONES	Las cartas deben de existir en el archivo del cual se quieren extraer y el usuario debe estar registrado.
POSTCONDICIONES	El mazo se guarda en el sistema para que siga existiendo, aunque haya cerrado sesión.

OPERACIÓN	añadirCarta(String nombreCarta, int cantidadCopias)
DESCRIPCION	Añade una carta al mazo del usuario con la cantidad de copias especificada.
PRECONDICIONES	El nombre de la carta a añadir debe ser válido y no nulo. La cantidad de copias debe ser mayor que 0 y no negativo.
POSTCONDICIONES	Se agrega la carta al mazo del usuario con la cantidad de copias especificada.

Tiempo dedicado (Aproximado)

- Creación modelo de dominio: 00:45 horas/minutos
- Creación Diagrama de clases: 04:30 horas/minutos
- Creación Informe y contratos, primera entrega: 02:00 horas/minutos