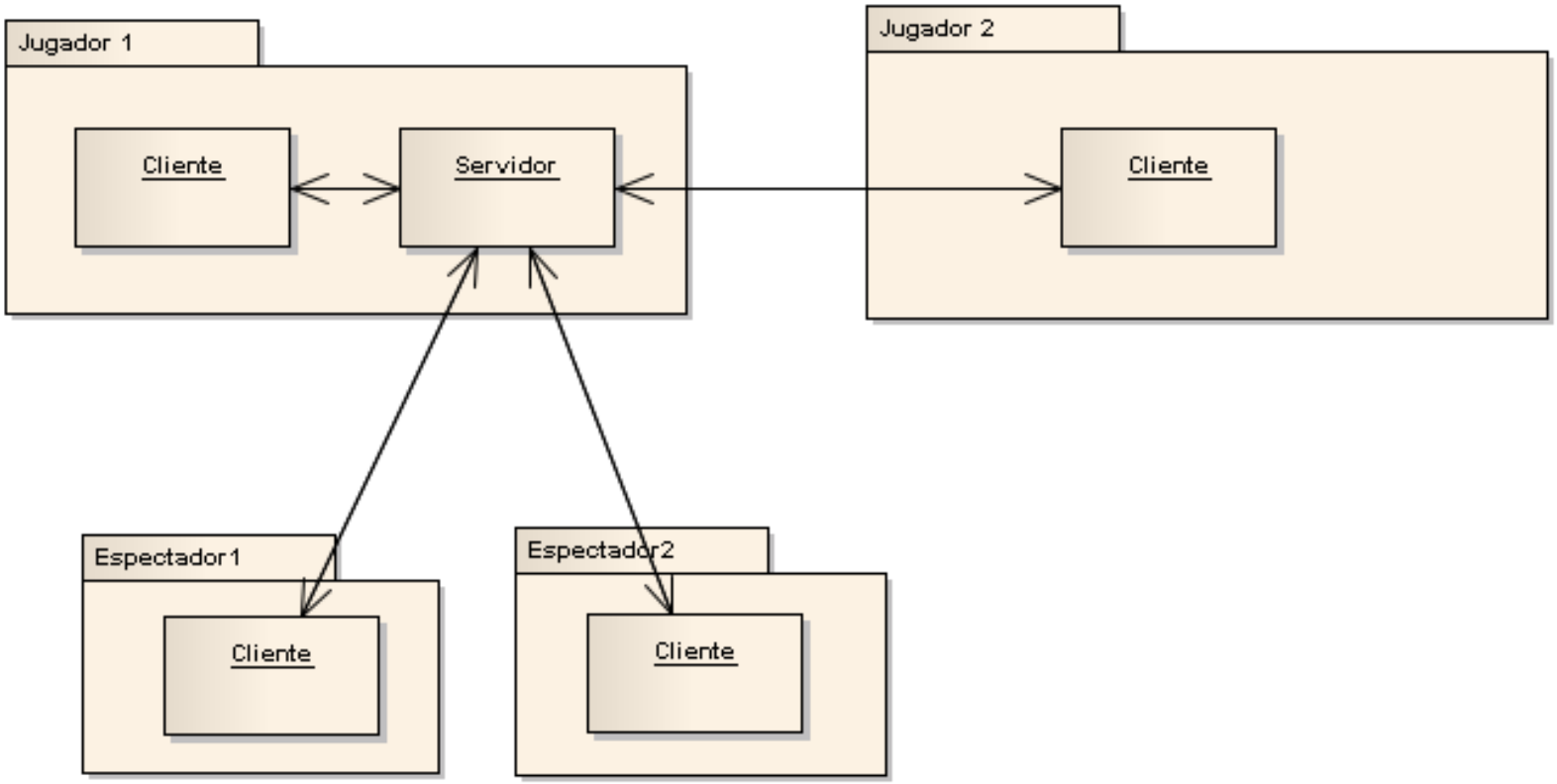
Arquitectura de Batalla Naval

Alumnos:
- Calderón Rodrigo
- Rebella Santiago
- Repetti Nicolás

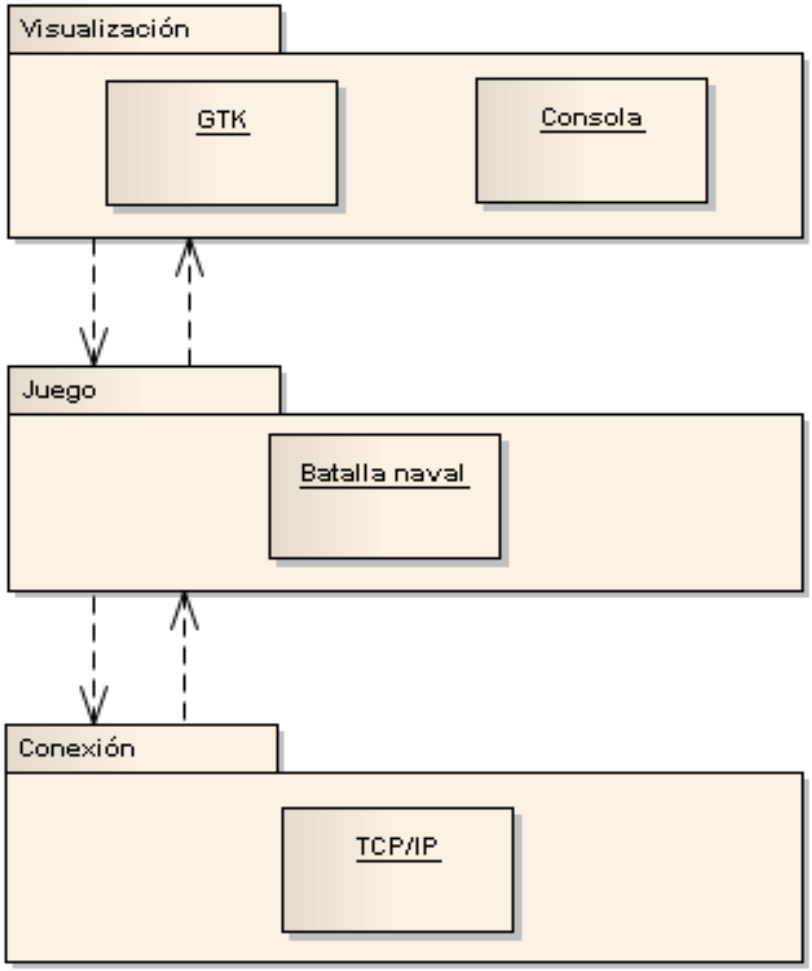
Profesor: Morixe Martín

Fecha: 26 de Agosto de 2011

Modelo Cliente/Servidor



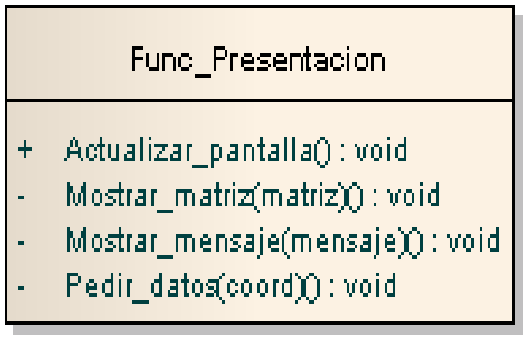
Arquitectura en Capas

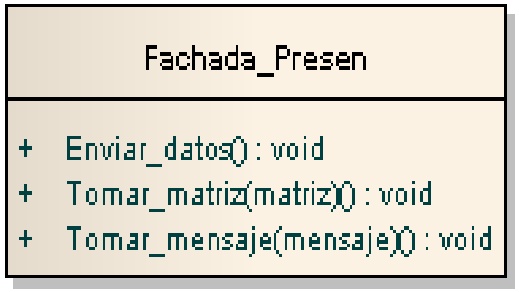


- Se desarrollan 3 capas que se conectan a través de interfaces definidas
- Cada clase contiene clases con atributos y métodos que veremos a continuación

Clases - Presentación

- Se encarga de refrescar la consola para que los nuevos datos sean visibles para jugadores y espectadores, y de proveer datos de los jugadores a la capa lógica
- En futuras versiones también se incluirá un modo gráfico con GTK





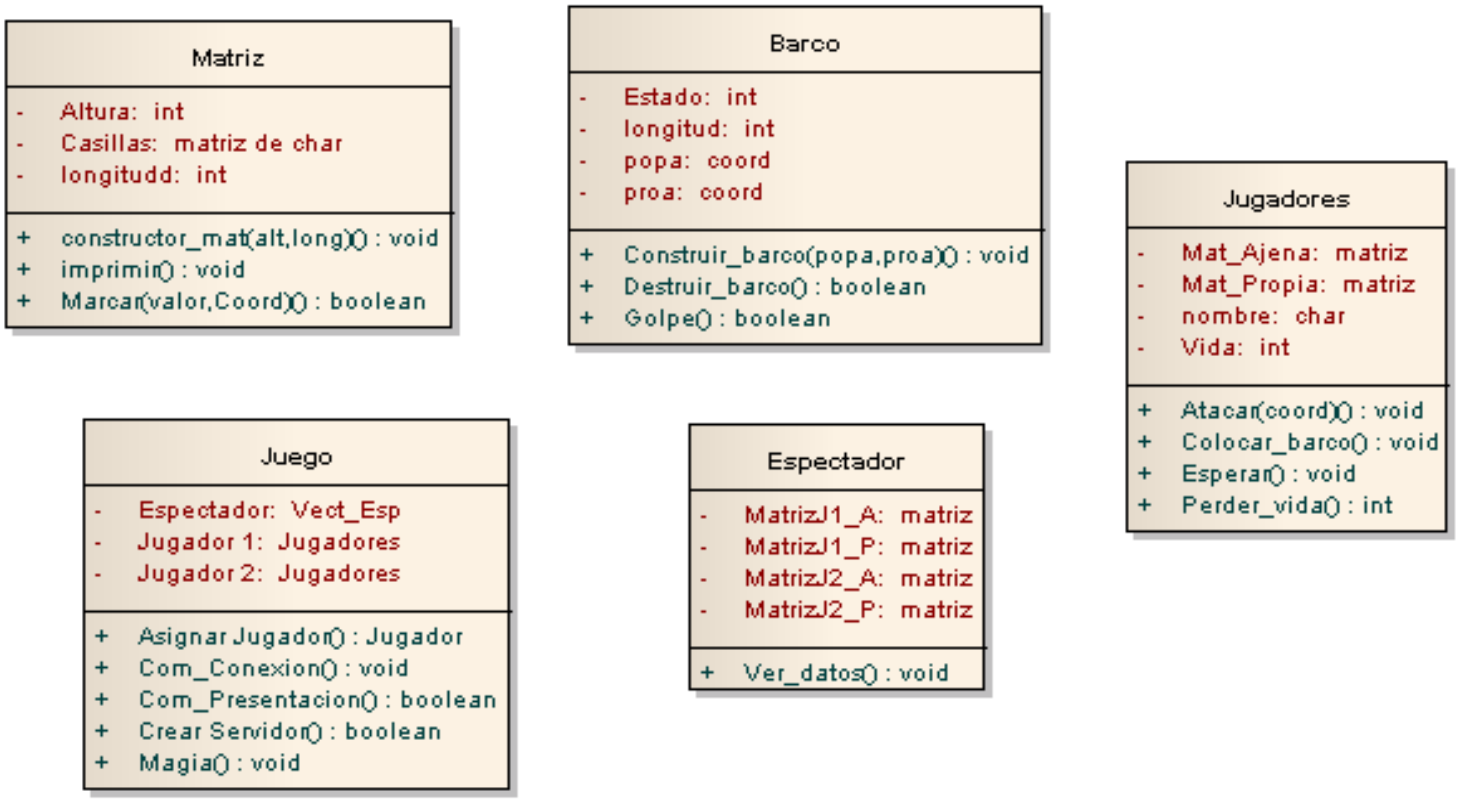
Clases – Presentación
Func-Presentación

- Actualizar_pantalla: imprime las matrices y mensajes correspondientes en la consola del jugador (usando las funciones privadas abajo nombradas)
- Mostrar_matriz: imprime la matriz indicada en la consola del jugador
- Mostrar_mensaje: imprime el mensaje indicado en la consola del jugador
- Pedir_datos: pide datos al jugador para enviarlos a la capa de lógica

Clases – Presentación Fachada-Presen

- Enviar_datos: manda datos hacia la capa de lógica (coordenadas)
- Tomar_matriz: toma los datos recibidos desde la lógica (dos matrices) y los devuelve a la capa presentación
- Tomar_mensaje: toma los mensajes recibidos desde la lógica (dos matrices) y los devuelve a la capa presentación

Clases - Lógica



- Se encarga de la lógica, los turnos, los movimientos posibles, la definición del partido y de conectar la capa de presentación con la de conexión

Clases – Lógica
Matriz

- Altura: cantidad de filas
- Longitud: cantidad de columnas
- Casillas: matriz de (Altura x Longitud) caracteres
- Construir_mat: crea una instancia de matriz
- Imprimir: manda mensaje con todos los contenidos de sus celdas
- Marcar: cambia el valor de una casilla de la matriz

Clases – Lógica
Barco

- Estado: indica si el barco está hundido o no
- Longitud: casillas que ocupa el barco
- Proa: casilla inicial del barco
- Popa: casilla final del barco
- Construir_barco: crea una nueva instancia de barco llenando sus atributos
- Destruir_barco: decrece el numero de barcos de jugador y borra el barco de la matriz
- Golpe: marca un golpe en el barco

Clases – Lógica Jugadores

- Mat_ajena: instancia de matriz que contiene los misiles enviados por el jugador y el resultado proporcionado por el contrincante
- Mat_propia: instancia de matriz que indica sus barcos propios y los misiles lanzados por el contrincante
- Nombre: nick del jugador
- Vida: barcos restantes del jugador
- Atacar: envía un misil a las coordenadas especificadas
- Colocar_barco: crea un barco en la posición indicada
- Esperar: impide acciones al jugador hasta la respuesta del servidor
- Perder_vida: decrece la cantidad de barcos del jugador

Clases – Lógica
Juego

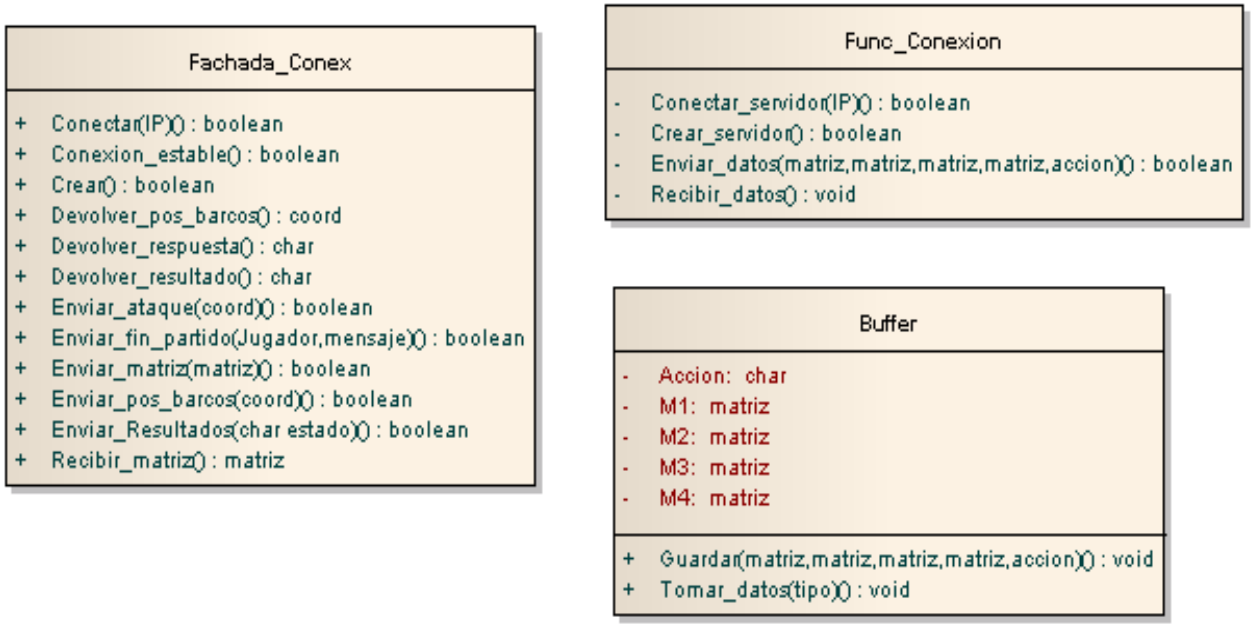
- Espectador: vector que contiene la información de todos los espectadores
- Jugador1: instancia de la clase Jugador
- Jugador2: instancia de la clase Jugador
- Asignar_jugador: relaciona a una persona real (física) con una instancia dentro del juego
- Com_conexion: interfaz con la capa “Conexión”
- Com_presentacion: interfaz con la capa “Presentación”
- Crear_servidor: crea el servidos en la máquina en la que se ejecuta el programa
- Magia: arbitra toda la lógica del juego

Clases – Lógica Espectador

- MatrizJ1_A: contiene la matriz ajena del Jugador1
- MatrizJ1_P: contiene la matriz propia del Jugador1
- MatrizJ2_A: contiene la matriz ajena del Jugador2
- MatrizJ2_P: contiene la matriz propia del Jugador2
- Ver_datos: muestra todas las matrices indicadas arriba

Clases - Conexión

- Manejar el envío y la recepción de los datos necesarios, y provee la conexión con el servidor



Clases – Conexión Fachada_Conex

- Conectar: conecta un jugador al servidor
- Conexion_estable: chequea que la conexión con los jugadores se mantenga
- Crear: crea el servidor
- Devolver_pos_barcos: envía al servidor la posición de los barcos puestos por el cliente
- Devolver_respuesta: devuelve a la capa lógica el último ataque realizado por algún jugador
- Devolver_resultado: devuelve a la capa lógica la consecuencia del último ataque realizado
- Enviar_ataque: envía desde la capa lógica los ataques a la capa de conexión

Clases – Conexión Fachada_Conex

- Enviar_fin_partido: envía a los jugadores el mensaje de fin de partido con sus respectivos mensajes (gana o pierde)
- Enviar_matriz: envía las matrices resultantes a los objetivos deseados (jugadores y/o espectadores)
- Enviar_pos_barco: envía la información de la posición de los nuevos barcos al servidor
- Enviar_resultados: envía la consecuencia del ataque de un jugador al mismo desde el servidor
- Recibir_matriz: pasa la matriz enviada por el servidor a la capa lógica del cliente

Clases – Conexión
Func_Conexión

- Conectar_servidor: conecta al cliente al servidor ubicado en la IP otorgada
- Crear_servidor: crea un servidor en la máquina local
- Enviar_datos: envía las 4 matrices y opciones de acción a cierto objetivo (jugador, servidor o espectador)
- Recibir_datos: toma los datos recibidos desde el buffer y los devuelve a la capa lógica

Clases – Conexión Buffer

- Accion: indica la acción realizada en la última conexión (ataque, nuevos barcos, etc)
- M1: contiene la matriz ajena del Jugador1
- M2: contiene la matriz propia del Jugador1
- M3: contiene la matriz ajena del Jugador2
- M4: contiene la matriz propia del Jugador2
- Guardar: guarda los datos nuevos en el buffer
- Tomar_datos: devuelve los datos especificados

Caso de Uso 1

Nombre: Crear Conexión	
Actores: Jugador 1, Jugador 2, Espectadores (0 a n)	
Precondiciones: Jugador 1 tiene corriendo el server.	
Acciones del usuario	Respuesta del sistema
1) Una persona intenta conectarse al servidor de “Jugador 1” 4) Otra persona intena conectarse al mismo servidor	2) El servidor le permite la conexión y lo identifica como “Jugador 2” 3) Arranca la partida entre Jugador 1 y Jugador 2 5) El servidor lo identifica como “Espectador”
Notas:	
El paso 4 y 5 pueden repetirse varias veces	

Caso de Uso 2

Nombre: Colocar Barcos	
Actores: Jugador 1, Jugador 2	
Precondiciones: Jugador 1 y Jugador 2 son reconocidos por el servidor	
Acciones del usuario	Respuesta del sistema
1) Jugador 1 coloca su barcos. 3) Jugador 1 finaliza 4) Jugador 2 coloca sus barcos 5) Jugador 2 finaliza 7) Jugador 1 comienza la partida.	2) En la consola de Jugador 2 se ve “esperando” 4) En la consola de Jugador 1 se ve “esperando” 6) Se muestran dos matrices en cada terminal (la propia con los barcos y la ajena toda en 0)
Notas:	

Caso de Uso 3

Nombre: Ataca Jugador1	
Actores: Jugador 1, Jugador 2	
Precondiciones: Jugador 1 y Jugador 2 iniciaron el juego	
Acciones del usuario	Respuesta del sistema
1) Jugador 1 selecciona una coordenada.	2) Si no fue seleccionada anteriormente, se envia el ataque. Caso contrario se vuelve al paso 1. 3) Se envia un mensaje con los datos (“ataque”,c1,c2) 4) Se recibe el ack del mensaje. 5) Se procesan los datos. Se devuelve (Resultado, vidas_restantes) 6) Se hace un refresh en las matrices de ambos jugadores y la última acción realizada
Notas:	
Resultado: toma los valores Agua, Golpe, Hundido. Si vidas_restantes es 0. Se impimr el mensaje “has ganado” y se da por terminado el juego	

Caso de Uso 4

Nombre: Actualizar Pantalla (Jugador)	
Actores: Jugador 1, Jugador 2	
Precondiciones: Jugador 1 y Jugador 2 son reconocidos por el servidor	
Acciones del usuario	Respuesta del sistema
1) El jugador 1 ingresa su jugada. 4) La capa de presentación recibirá los datos de ese jugador (através Refresh()), y los muestra en pantalla.	2) Se efectúan los cambios producidos por la jugada (CU anterior). 3) Se envía los datos de las matrices al jugador 1, a través de la función Imprimir().
Notas:	

Caso de Uso 5

Nombre: Actualizar Pantalla (Espectadores 0 a n)	
Actores: Jugador 1, Jugador 2	
Precondiciones: Jugador 1 y Jugador 2 son reconocidos por el servidor	
Acciones del usuario	Respuesta del sistema
1)No realiza ninguna acción. 3) La capa de presentación recibirá los datos de los jugador (através Refresh()), y muestra en su pantalla las cuatro matrices(2 de cada jugador). Tambien se muestra la jugada realizada por cada jugador.	2) Envia periódicamente datos al viewer, luego de cada envío de ataque de los jugadores 1 y 2.
Notas:	