[AA3] Práctica - Bloque 1: TCP

El objetivo es implementar un cluedo online que funcione sobre la topología cliente/servidor y sobre la topología P2P.

Para la topología cliente/servidor debe incluir una capa de lobby que permita la creación de partidas.

Ítem 1: Funcionamiento del Enti-Cluedo

Las reglas del juego son conocidas, pero vamos a especificarlas/relajarlas para esta práctica.

Elementos del juego

- 1 tablero de juego
- 6 fichas de personaje
- 6 fichas de arma
- 2 dados
- 6 cartas de personajes
- 6 cartas de armas
- 9 cartas de sala
- 21 cartas de pista. Para nosotros, habrá tres cartas de pista: de personaje, de arma y de sala. Si nos sale una carta de personaje, podemos indicar qué personaje queremos que se revele y el jugador que lo tiene, lo muestra. Lo mismo con armas y salas.
- Fichas para apuntar pistas para cada jugador

Preparación de la partida

- Cada jugador escoge un personaje (como mínimo habrá 3 jugadores y como máximo 6)
- El personaje escogido se coloca en una posición aleatoria de entre las que no pertenecen a ninguna sala
- Una mano inocente escoge una carta de personaje, una carta de arma y una carta de sala y las reserva. Ese es el asesinato a resolver.
- El resto de cartas se reparten entre los jugadores.
- Empieza el turno el primer jugador que se conectó a la partida.

Qué ocurre en un turno

- Se lanzan los dados (hay dos dados).
    - Si sale un 1, se nos ofrece una carta de pista.
- Movemos indicando la casilla destino o bien desplazándonos con las teclas de dirección hasta que se acaba la puntuación de los dados.
    - Simplificamos que las salas no tienen puertas, podemos entrar desde cualquier punto.
    - Tampoco tenemos pasadizo secreto entre las salas de los extremos. Pero impleméntalos si lo consideras.
- Si hemos conseguido llegar a una sala, podemos hacer una deducción. Ejemplo:
    - Orquídea ha matado con el Puñal en la Sala de Billar.
- En sentido inverso al de turno de tirada, el resto de jugadores tienen que intentar desmentir esa deducción. Ejemplos:
    - Si el jugador de la izquierda tuviera a Orquídea, mostraría esa carta sólo al jugador que ha hecho la deducción.
    - Si el jugador de la izquierda tuviera a Orquídea y el Puñal, podría escoger qué carta mostrar.
    - Si el jugador de la izquierda no pudiera desmentir, lo intentaría hacer el de más a la izquierda.
    - Si ningún jugador pudiera desmentir, se da la opción al jugador inicial de hacer una acusación formal o de lo hacerla.
        - Si el jugador acierta en su acusación, gana el juego.
        - Si el jugador no acierta, queda eliminado.
- En el momento en que un jugador hace una acusación formal, el juego se para y en el resto de turnos, cada jugador debe hacer una acusación formal.
    - Gana el primero que acierta.
    - Podría darse el caso de que no ganara nadie.

Cuándo acaba la partida

- Cuando un jugador hace una acusación formal y acierta. Gana ese jugador.
- Cuando ningún jugador ha sido capaz de acertar en su acusación formal. No gana ningún jugador.

Ítem 2: Protocolos de comunicación

La topología client/server funciona de una forma concreta y hace que el protocolo de comunicación necesite intercambiar mensajes entre clientes y servidor. Esto genera un protocolo de comunicación ad-hoc a esta topología.

En la topología peer-to-peer sabemos que no hay servidor de juego y todos los nodos tienen su propia copia del estado del juego. Eso puede hacer que el protocolo de comunicación para conseguir la funcionalidad final cambie respecto a client/server.

El objetivo de este apartado es que estructures un protocolo de comunicación para cada caso.

Protocolo de comunicación para Client/Server

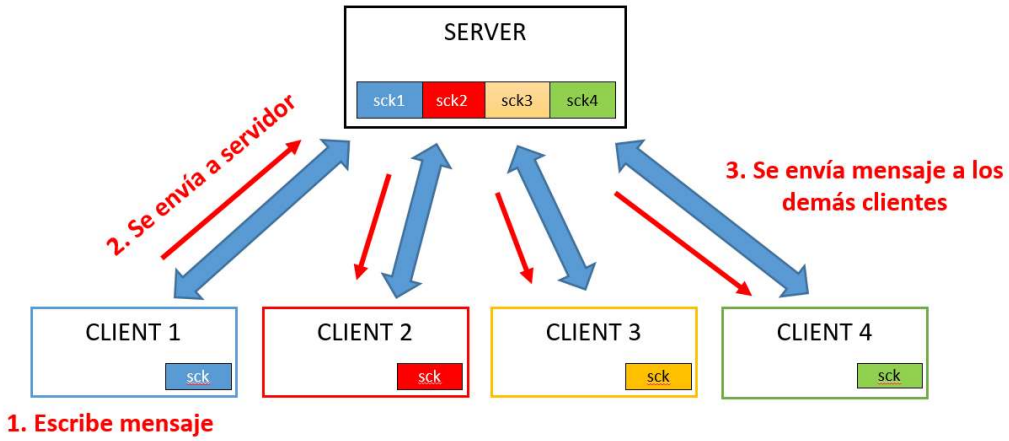
Estructura mensaje	Sentido	Acción asociada

Protocolo de comunicación para Peer-to-Peer

Estructura mensaje	Acción asociada

Cuando vayas implementando, estas listas irán cambiando. Procura tenerlas actualizadas. Te ayudarán.

Ítem 3: Implementación en topología cliente/servidor

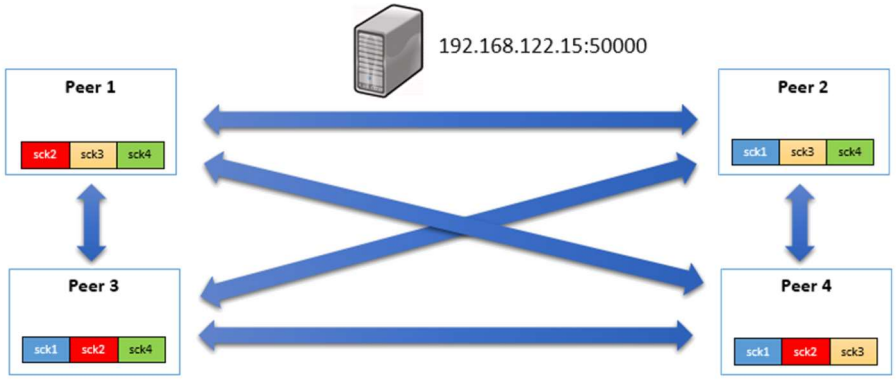


Recuerda:

1. El servidor no tiene parte gráfica. Pero es el que mantiene el estado del juego. Así que debe guardar la información de todos los jugadores. Recuerda que para esto se utilizan los ClientProxy.
2. El cliente tiene sólo la información del juego necesaria para la representación gráfica. El jugador envía los comandos a servidor y no puede ejecutarlos hasta que el servidor le

"da permiso". Sólo si servidor lo considera válido, se ejecuta en este cliente y en todos los demás (ya que todos deben ver la misma versión del juego).
3. El servidor indica cuándo empieza la partida y asigna turnos (si los hay).
4. El servidor indica cuándo finaliza la partida y quién es el ganador.

Ítem 4: Implementación en topología peer-to-peer



Recuerda:

1. En esta topología tenemos un Bootstrap Server que nos ayuda sólo al inicio para que todos los nodos establezcan conexión.
2. En el momento que se dan cuenta de que ya han llegado los N jugadores necesarios, ya puede empezar la partida.
3. A partir de ahí, todos los nodos están al mismo nivel. Todos tienen una copia "verdadera" del mundo.
4. Todos los nodos pueden comunicarse con todos los demás.
5. Cada nodo valida sus propias acciones y da por buenas las acciones de los demás. (Podría añadirse alguna verificación extra, pero no es necesario para esta práctica).

Ítem 5: Juego en modo cliente/servidor con varias salas de juego

La implementación cliente/servidor debe permitir que el servidor mantenga varias partidas a la vez.

Concretamente, serán los propios jugadores lo que creen sus propias salas de juego o se unan a salas ya existentes.

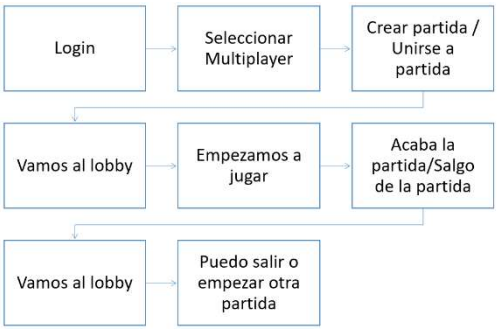
El jugador que quiera crear una sala de juego debe:

1. Darle un nombre que no exista para otra sala de juego.
2. Tener la posibilidad de darle una contraseña de entrada a la partida.
3. Poder indicar el número de jugadores. Cuando ese número de jugadores esté en la sala de juego, se puede iniciar la partida.

El jugador que quiere unirse a una sala puede:

1. Ver un listado de las salas disponibles en el que aparecerá, al menos, el nombre de la sala.
2. Hacer búsquedas.
    a. Por salas con/sin contraseña
    b. Por salas completas/incompletas.

Los jugadores deben poder comunicarse a través de un chat. Podrás elegir cuál es el lugar más adecuado para hacerlo.



Ten claro cuál será el flujo de navegación del jugador a través de las diferentes fases de creación / unión a partidas.

Recuerda:

1. Cuando seleccionamos la sala de juego, no entramos directamente en la partida. Debemos ir a parar a un lobby en el que, como mínimo, nos podremos ver con el resto de jugadores
2. Cuando la partida acabe, el cliente no se acaba, se le debe dar la oportunidad de jugar otra partida.

Item 6: Próximamente

…..

Importante

A la hora de evaluar se considerará **muy importante**:

1. Que los protocolos de comunicación diseñados cubran todas las situaciones del juego
2. Que los protocolos de comunicación estén correctamente implementados
3. La implementación de la comunicación por red: Entre jugadores y servidor / entre nodos
4. Que estén controlados los errores de comunicación: Sobre todo a la hora de conectarse y desconectarse (acabar partida)

Entregable de la práctica → Protocolos

1. Un documento Word/Excel o similar en el que se detalle el protocolo utilizado para la topología cliente/servidor. El protocolo debe contemplar la creación/unión de/a partidas.
2. Otro documento Word/Excel o similar en el que se detalle el protocolo utilizado para la topología P2P. El protocolo debe contemplar la creación/unión de/a partidas.

Entregable de la práctica → Código

1. Nos aseguramos de que la solución compila.
2. Nos aseguramos de que el proyecto no utiliza paths absolutos para linkar las librerías.
3. Eliminamos la carpeta oculta .vs
4. Hacemos un zip o 7zip de la solución

Lo subimos todo al Classlife en este formato:

AA3_PracticaTCP_Nombre1Apellido1Nombre2Apellido2.zip

Rúbrica → En breve…