Universidad de los Andes

Ingeniería de Sistemas y Computación

ISIS 1205 - Algorítmica y Programación 2

Taller teórico nivel 12

Cupi2

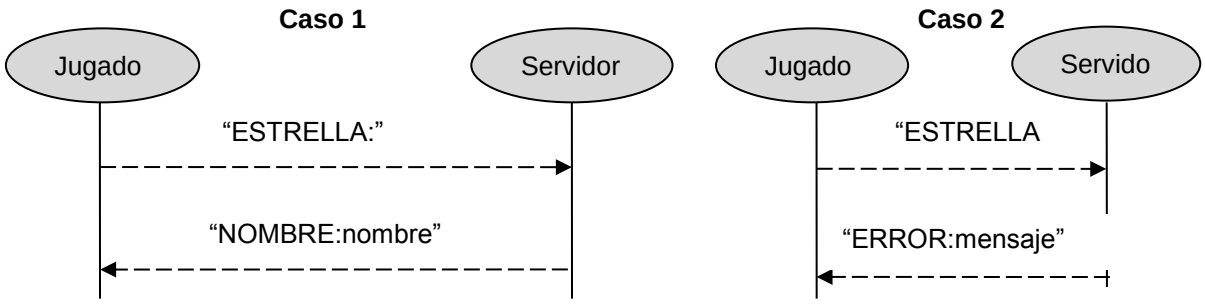
Se desea extender la aplicación del juego de cartas Continental para que: 1) los jugadores puedan saber el nombre del jugador "Estrella". El jugador "Estrella" es el que más partidos ha ganado durante la historia del juego y 2) saber cuántos partidos ha jugado un participante.

Suponga que la base de datos contiene la siguiente tabla:

```
Tabla: jugadores

Campo: nombre varchar(32)

Campo: ganados int

Campo: perdidos int

Campo: jugados int

PRIMARY KEY: nombre
```

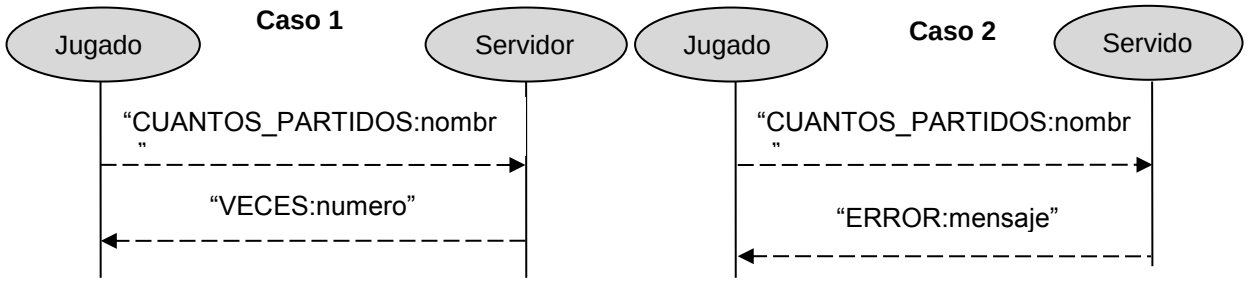
El protocolo de comunicación entre cliente y servidor para saber el nombre del jugador "Estrella" es el siguiente:

En esta interacción se pueden presentar dos casos:



- Caso 1: el servidor puede obtener el nombre del jugador, en cuyo caso retorna el nombre.
- Caso 2: el servidor obtiene algún error haciendo la consulta y retorna ERROR con el mensaje de error adecuado.

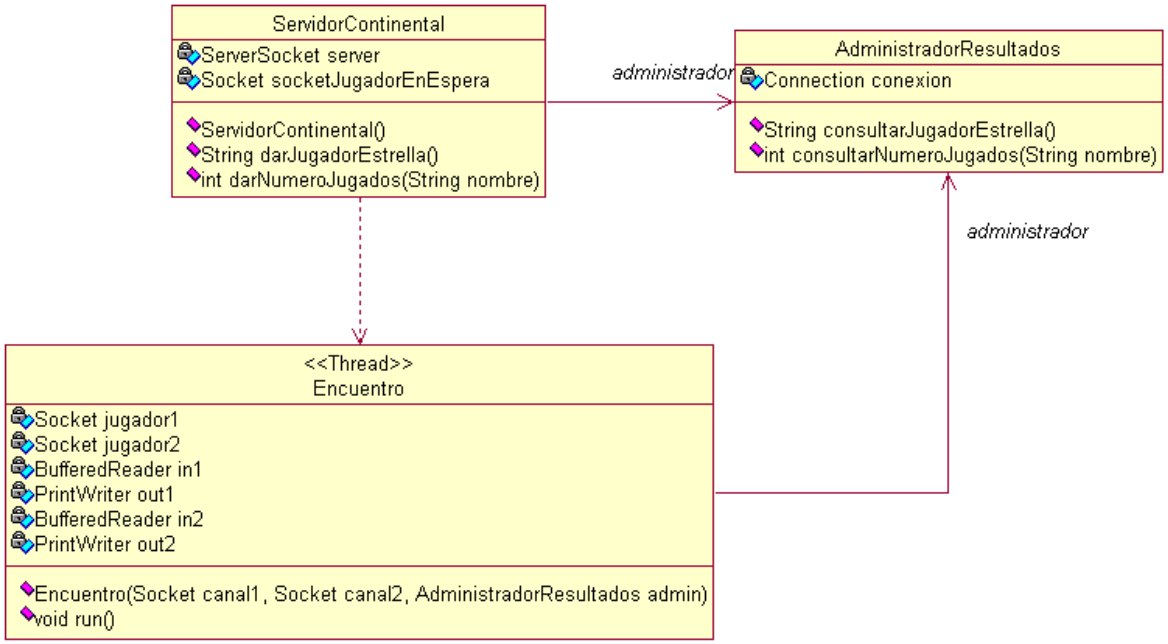
El protocolo de comunicación entre cliente y servidor para saber cuántos partidos ha jugado un participante es el siguiente:



En esta segunda interacción se pueden presentar dos casos:

- Caso 1: el servidor puede obtener el número de veces jugadas por un participante, en cuyo caso retorna el número.
- Caso 2: el servidor obtiene algún error haciendo la consulta y retorna ERROR con el mensaje de error adecuado.

El modelo siguiente muestra los aspectos más relevantes para satisfacer las extensiones del juego.



Complete los siguientes métodos del servidor y del cliente para que interactúen de la forma descrita en los protocolos.

1. (10%) Servidor: Clase ServidorContinental – Método recibirConexiones

Este método se encarga de recibir todas las conexiones de los jugadores. Debe abrir el socket y para cada pareja de conexiones de jugadores entrantes construir un Encuentro.

```
public void recibirConexiones( )

{

       int puerto = 7777;

//COMPLETAR: Abre un puerto de escucha para recibir solicitudes de conexión

              server =

             while( true )

             {

//COMPLETAR: Espera una nueva solicitud de conexión

                 Socket socketNuevoCliente =

                 // Intentar iniciar un encuentro con el nuevo cliente

                 if( socketJugadorEnEspera == null )

                 {

                 //No hay un oponente aún, así que hay que dejarlo en espera

                     socketJugadorEnEspera = socketNuevoCliente;

                 }

                 else // Ya se tiene un oponente

                 {

                   Socket jug1 =

                   Socket jug2 =

                   socketJugadorEnEspera = null;

//COMPLETAR: Crea un nuevo objeto de la clase Encuentro y lanza el nuevo hilo de ejecución









                }
```

```
            }//cierra while

           server.close( );

}
```

2. (10%) Servidor: Clase Encuentro – Método constructor

```
public Encuentro( Socket canal1, Socket canal2, AdministradorResultados
admin ) throws IOException

    {

        administrador = admin;

//COMPLETAR: Inicializa el canal de comunicación con el jugador1, crea
el flujo de escritura y lectura asociado al canal del jugador1









//COMPLETAR: Inicializa el canal de comunicación con el jugador2, crea
el flujo de escritura y lectura asociado al canal del jugador2









    }
```

3. (30%) Servidor: Clase Encuentro – Método run

Este método se encarga de recibir los mensajes enviados por el cliente al servidor. Usted debe escribir **UNICAMENTE** el código correspondiente a:

- La recepción del mensaje enviado por el cliente mediante el cual éste pregunta al servidor el nombre del jugador “Estrella”.
- La recepción del mensaje enviado por el cliente mediante el cual éste pregunta al servidor el número de veces que un jugador ha jugado.
- El procesamiento del mismo según los protocolos descritos anteriormente

```
public void run( )

{









}
```

4. (20%) Cliente – Método consultarJugadorEstrella

Este método, de la clase principal del mundo del cliente, se encarga de preguntar al servidor el nombre del jugador estrella según el protocolo descrito anteriormente. En caso de que el servidor no haya podido responder satisfactoriamente a la solicitud, este método debe retornar una excepción. Este método recibe como parámetro, el canal de comunicación con el servidor.

```
public String darJugadorEstrella( Socket server )throws JugadorException

{


   //COMPLETAR: Obtiene el flujo de entrada y de salida del canal de comunicación (socket)









   //COMPLETAR: Se comunica con el servidor

















   // Retorna la respuesta









}
```

5. (10%) Escriba la instrucción SQL para saber cuántos partidos ha jugado el participante "Juan Tamariz".

6. (20%) Complete el siguiente método de la clase que se encarga de manejar la base de datos, que retorna un vector con los nombres de todos los jugadores cuyos partidos ganados superen el número que se recibe como parámetro.

```
public ArrayList darJugadoresSuperiores( int num ) throws SQLException

{

   String sql = "SELECT nombre FROM jugadores WHERE ganados > '" + num + "'";

   Statement st = conexion.createStatement( );

   ResultSet resultado = st.executeQuery( sql );

   //COMPLETAR: Procesa los registros









   resultado.close( );

   st.close( );

   //COMPLETAR: Retorna el vector



}
```