Implementación del juego Sudoku por medio del paradigma cliente/servidor

- Diseño del Software -

José Antonio Martínez López - *joseantonio.martinez7@alu.uclm.es*

Pedro Manuel Gómez-Portillo López - *pedromanuel.gomezportillo@alu.uclm.es*

Índice

1. Resumen

Breve descripción del proyecto

Este proyecto tiene como objetivo el desarrollo de una aplicación y un servidor para jugar al mítico juego japonés Sudoku. La aplicación correrá en un dispositivo Android mientras que el servidor se ejecutará en un ordenador gracias a Apache Tomcat y Struts2 y se comunicarán por medio del intercambio de mensajes JSON.

2. Descripción del alcance.

Qué se va a hacer y qué no. Cómo se juega al juego.

Se ejecutarán el cliente para el juego (que funcionará en el móvil) y el servidor (que correrá en un PC). El segundo debe poder comunicar varios clientes para que puedan jugar entre ellos. Un cliente, previamente registrado, se loguea y si no hay ningún otro jugador esperando entrará a la sala de espera. Si por el contrario ya hay un jugador esperando el servidor creará una partida, los meterá en ella y les enviará el mismo tablero de Sudoku a ambos jugadores, y quien lo resuelva primero ganará.

Cuando el jugador toque una casilla del tablero podrá introducir un número en ella. Si un jugador coloca un número en su tablero, su oponente verá la posición en la que lo ha colocado, pero no qué número fue.

Una vez que un jugador rellene todas las casillas de su tablero el servidor le comunicará si su tablero está resuelto correctamente. Si lo está, anunciará a ambos jugadores que ha ganado, además del tiempo en el que lo ha hecho, y la partida terminará. Pero si por el contrario hay fallos le comunicará a este jugador cuántos fallos hay, pero sin decirle cuáles.

Mientras dure la partida, cualquier jugador podrá abandonarla graciosamente saliendo de ella, lo que le concederá la victoria al oponente.

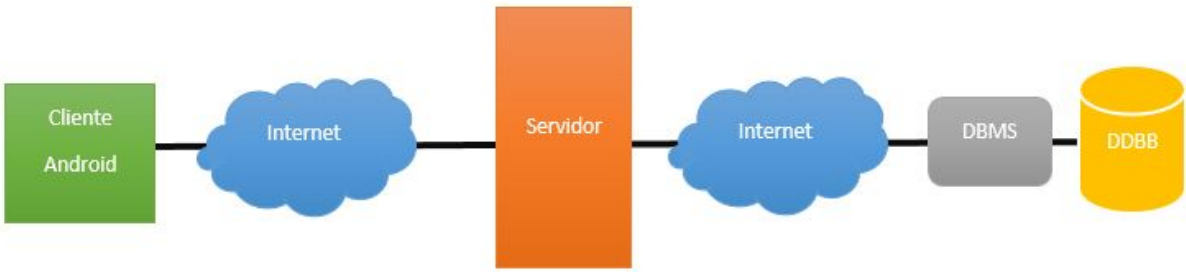
Además, se mantendrá un ranking con todas las victorias de cada jugador, y los clientes podrán acceder a ella por medio de la aplicación Android.

3. Diagramas comunes

Diagramas que formarán parte de ambas partes

3.1. Diseño arquitectónico de la solución

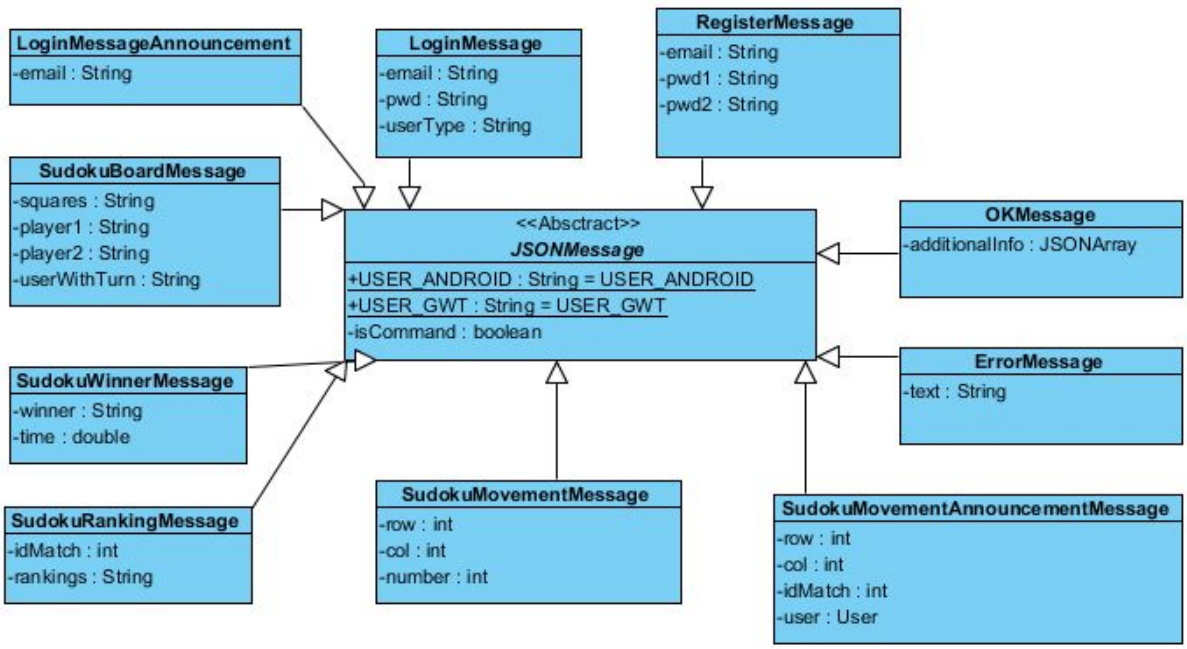
El juego se desarrollará en tres sistemas independientes a la vez; tendremos un cliente Android, quien será el que el que realmente jugará al juego, un servidor encargado de gestionar las partidas y actuar como intermediario entre los jugadores y un sistema gestor de bases de datos, que registrará jugadores y comprobar si estos existen en la base de datos. Estos sistemas se comunicarán a través de internet como se muestra en el siguiente diagrama de diseño arquitectónico.



3.2. Mensajes intercambiados entre C/S

El cliente y el servidor se comunicarán por medio de mensajes JSON. Para ello se ha definido una superclase, JSONMessage, de la que heredarán el resto de mensajes que se intercambiarán. El sistema soporta los siguientes mensajes;

- *RegisterMessage*. Lo enviará un cliente al servidor cuando quiera registrarse.
- *LoginMessage*. Lo enviará un cliente al servidor cuando quiera hacer login.
- *LoginMessageAnnouncement*. Lo enviará el servidor a los clientes conectados cuando un nuevo jugador haga login.
- *SudokuBoardMessage*. Lo enviará el servidor a dos jugadores que estén jugando una partida al inicio de la misma con el tablero del sudoku que van a jugar.
- *SudokuMovementMessage*. Lo enviará un jugador al servidor cuando coloque un número en su tablero durante una partida.
- *SudokuMovementMessageAnnouncement*. Lo enviará el servidor al adversario cuando un jugador coloque un número en su tablero.
- *SudokuWinnerMessage*. Lo enviará el servidor a los dos jugadores de una partida al final de la misma informando quién ha sido el vencedor.
- *SudokuRankingMessage*. Lo enviará el servidor al cliente cuando éste lo pida informándole de los rankings de las partidas jugadas.



3.3. Plan de desarrollo guiado por casos de uso

Para el desarrollo efectivo del proyecto entero se han planeado distintas fechas para las que deberían estar acabadas las distintas iteraciones definidas,

ID	Fecha	Iteración	Estado
1	15/2/2016	Conseguir registrar usuarios mediante el DBMS.	15/2/2016
2	22/2/2016	Verificar que un usuario existe cuando se intente loguear	22/2/2016
3	14/3/2016	Informar al resto de usuarios cuando un jugador se loguee.	12/4/2016
4	21/3/2016	Implementar una sala de espera donde un jugador esperará mientras llega un segundo	12/4/016
5	28/3/2016	Crear una partida con un sudoku al azar de una lista e incluir en ella a dos jugadores que estén esperando.	27/4/2016
6	4/4/2016	Una vez ambos clientes estén jugando se recibirán mensajes con los diferentes números colocados por ambos jugadores y se informará a sus oponentes de la posición en la que se han colocado.	4/5/2016
7	11/4/2016	Cuando todas las casillas del tablero de un jugador estén llenas se comprobará si la solución es correcta, tras lo que se informará a ambos jugadores del ganador. En caso contrario, se informará al jugador del tablero completo cuántos errores tiene.	5/5/2016
8	21/4/2016	Permitir que un jugador abandone una partida e informar al oponente de ello.	6/5/2016
9	29/4/2016	Permitir al usuario preguntar por el ranking de las partidas de Sudokus	6/5/2016
10	6/5/2016	**DEADLINE.** Entrega del servidor al cliente	6/5/2016
11	16/5/2016	Implementar un jugador robot en el servidor	10%

Leyenda

Completado	Fecha completitud
En proceso	Porcentaje completado
Sin empezar	

4. Cliente

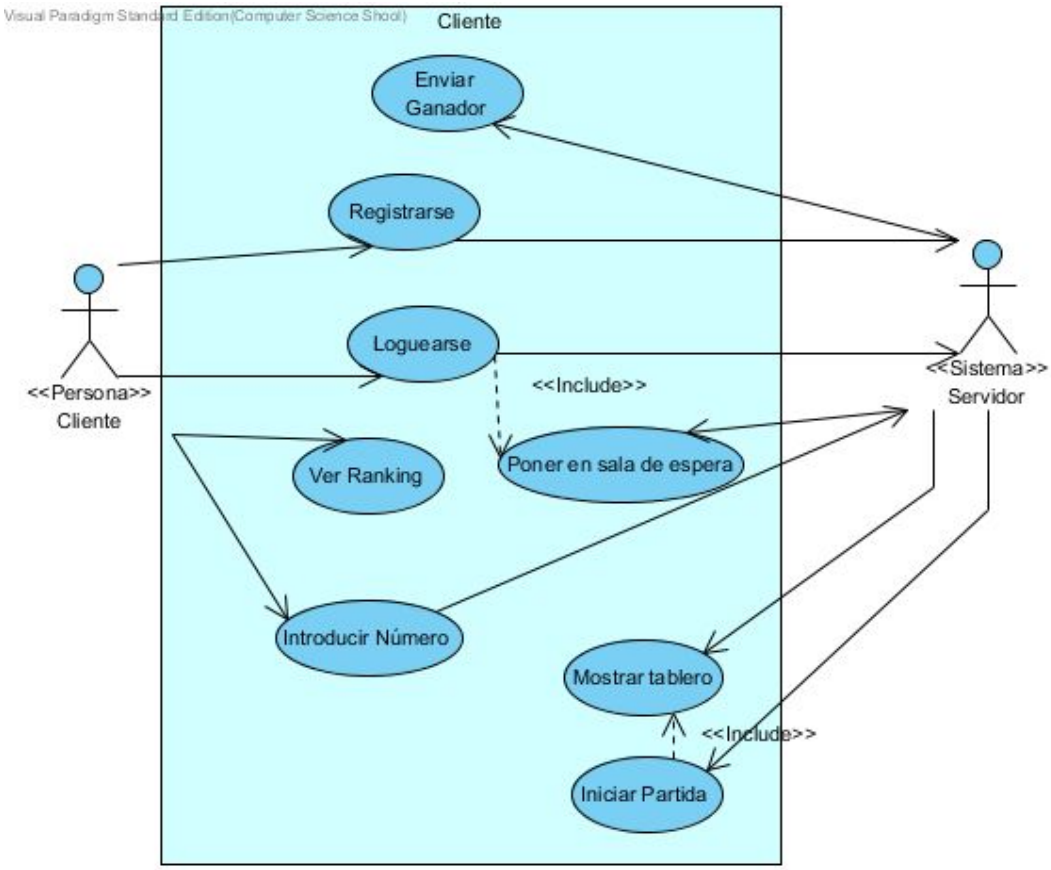
Parte de la documentación corresponde a la implementación del cliente

https://github.com/jamlkht/Sudoku_Client

4.1. Casos de Uso

1. Registrarse
2. Hacer login
3. Entrar a sala de espera
    3.1. Empezar partida
        3.1.1. Mostrar tablero
4. Introducir número
    4.1. Finalizar partida
5. Hacer logout

4.1.1. Diagrama de casos de uso



4.1.2. Descripción textual de los casos de uso más importantes

Nombre: Registrarse
Precondiciones: --
Prioridad/iteración: 1ª iteración
Flujo normal de eventos: el jugador introduce su email y repite dos veces la contraseña para su cuenta, y la aplicación le informa de que se ha registrado con éxito. Postcondiciones: el jugador aparecerá reflejado en la base de datos del sistema
Flujo alternativo 1: el email con el que se intenta registrar el jugador ya está registrado en el sistema Postcondiciones: la aplicación informará al usuario de que ya existe ese email
Flujo alternativo 2: la confirmación de la contraseña es distinta a la contraseña Postcondiciones: La aplicación informa de que son distintas
Descripción: Mediante este CDU el usuario podrá registrarse en la aplicación y así poder jugar

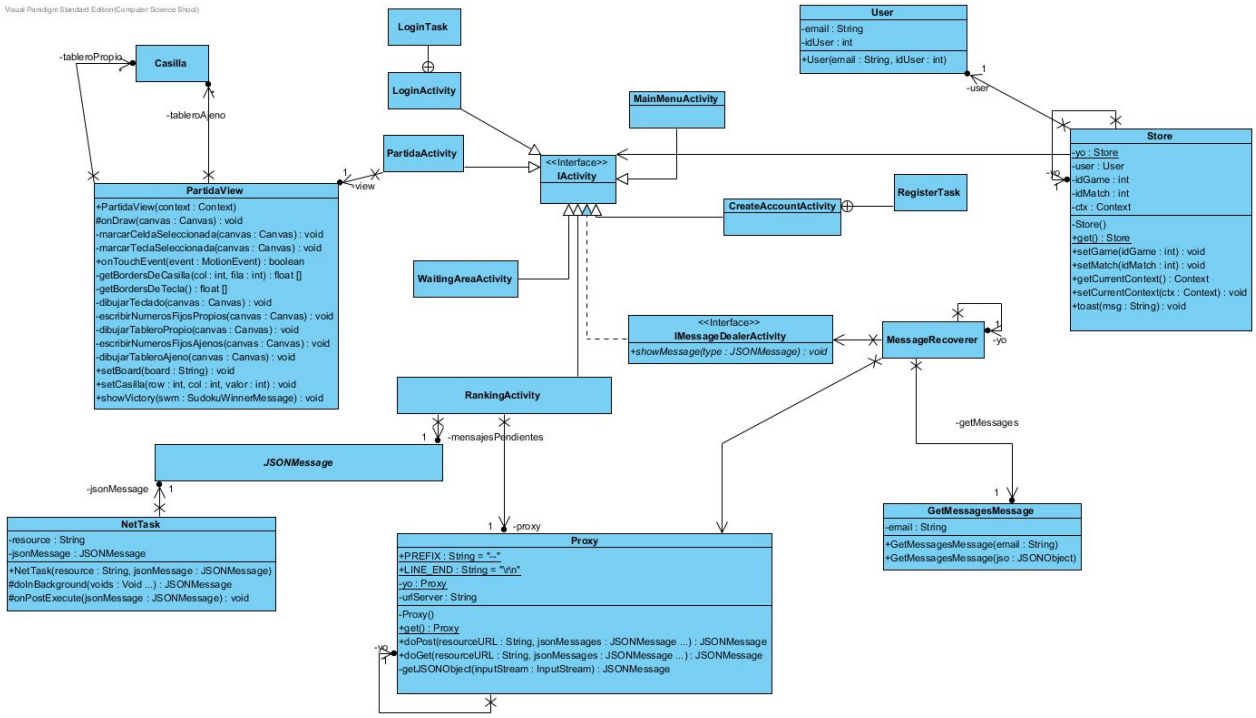
Nombre: Loguearse
Precondiciones: El usuario debe estar registrado y conocer su email y contraseña
Prioridad/iteración: 2º iteración
Flujo normal de eventos: el jugador introduce el correo y la contraseña con los que se registró en la aplicación y la aplicación le introduce en la sala de espera. Postcondiciones: el usuario estará en la sala de espera
Flujo alternativo 1: el email que introduce el usuario no se encuentra registrado en el sistema Postcondiciones: la aplicación informará al usuario de este error
Flujo alternativo 2: la contraseña no coincide con el email Postcondiciones: la aplicación informará al usuario de este error
Flujo alternativo 3: si hay alguien esperando en la sala, el que llega entra directamente a partida Postcondiciones: Comienza el juego
Descripción: mediante este CDU el jugador podrá entrar en la sala de espera de la aplicación y allí esperar a otros jugadores para comenzar la partida

Nombre: introducir número en el tablero en una partida
Precondiciones: El usuario debe estar jugando la partida.
Prioridad/iteración: 3ª iteración
Flujo normal de eventos: El jugador toca un cuadro y aparece un teclado numérico en el que puede introducir el número que crea Postcondiciones: el usuario habrá introducido un número y se le habrá comunicado al adversario
Flujo alternativo 1: El jugador introduce el número que completa el sudoku pero el sudoku no es correcto. Postcondiciones: la aplicación comprueba si el sudoku es correcto y comunica al jugador el número de errores.
Flujo alternativo 2: El jugador introduce el número que completa el sudoku y es correcto.. Postcondiciones: la aplicación informará al usuario de su arrasadora victoria y comunica al perdedor lo miserable que es
Descripción: mediante este CDU el jugador juega la partida y conoce si gana o pierde.

Nombre: Abandonar una partida

Precondiciones: El usuario debe estar jugando una partida
Prioridad/iteración: 4º iteración
Flujo normal de eventos: l jugador clica en “Abandonar la partida” Postcondiciones: la partida acabará y se le concederá la victoria al contrincante
Flujo alternativo 1: -- Postcondiciones: --
Descripción: mediante este CDU el jugador podrá salirse de una partida cuando quiera

4.2. Diagrama de clases de la solución



5. Servidor

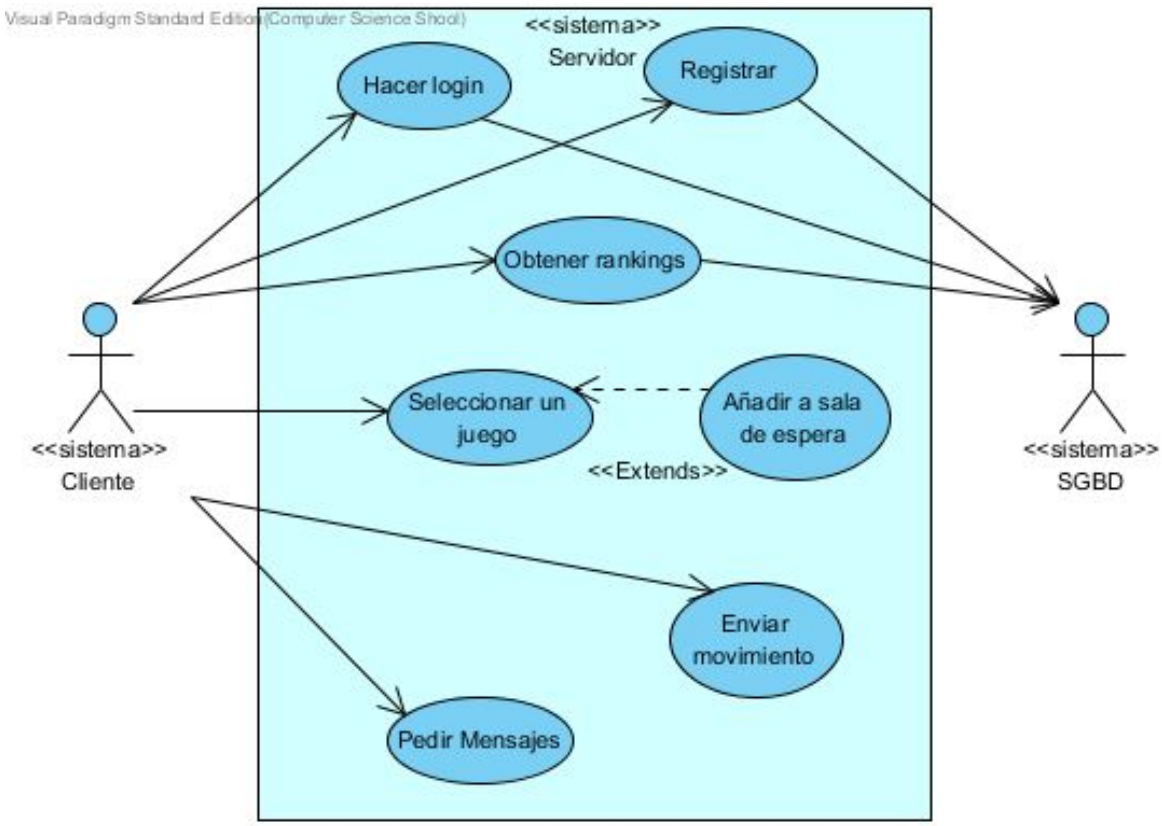
Parte de la documentación correspondiente a la implementación del servidor

https://github.com/gomezportillo/Sudoku_Server

5.1. Casos de uso

1. Registrar
2. Hacer login
3. Obtener rankings
4. Seleccionar un juego
   - 4.1. Añadir a sala de espera
   - 4.2. Notificar usuarios en sala de espera
   - 4.3. Enviar tablero
5. Enviar movimiento

5.1.1. Diagrama de casos de uso



5.1.2. Descripción textual de los casos de uso más importantes

1. Registrar
Precondiciones: El usuario no debe estar registrado
Prioridad/iteración: Alta
Flujo normal de eventos: El servidor recibe un mensaje JSON con los datos del nuevo usuario y se los manda al DBMS, quien lo registrará. Postcondiciones: el usuario quedará registrado en el sistema
Flujo alternativo 1: El campo de la contraseña1 y el de la contraseña2 no coinciden Postcondiciones: el servidor informará al usuario del error
Flujo alternativo 2: El email ya aparece registrado en la BBDD. Postcondiciones: el servidor informará al usuario del error
Descripción: mediante este CDU un usuario puede registrarse en el sistema

2. Hacer login
Precondiciones: El usuario debe estar registrado
Prioridad/iteración: Alta
Flujo normal de eventos: el sevidor recibe un mensaje JSON con la información del usuario y comprueba que existe mediante el DBMS Postcondiciones: el usuario aparecerá registrado en la BBDD y el servidor informará al resto de usuarios del nuevo login
Flujo alternativo 1: el usuario no está registrado en el sistema Postcondiciones: el servidor le informará y le pedirá que se registre
Descripción: mediante este CDU el jugador hace login en el sistema

3. Obtener rankings
Precondiciones: --
Prioridad/iteración: Baja
Flujo normal de eventos: el sevidor recibe un mensaje JSON con el identificador del juego del que se quieren obtener los rankigs Postcondiciones: el cliente recibirá la información que ha pedido
Flujo alternativo 1: el jueog del que se pude información no existe en el sistema Postcondiciones: el jugador recibe un error
Descripción: mediante este CDU el jugador obtiene información de los resultados de las partidas

4. Seleccionar un juego
Precondiciones: El usuario debe haber hecho login
Prioridad/iteración: Alta
Flujo normal de eventos: Cuando haya dos jugadores esperando para jugar, el servidor creará una partida con una tablero de sudoku al azar de una lista y los introducirá en ella Postcondiciones: los jugadores podrán empezar a jugar
Descripción: mediante este CDU el jugador puede jugar una partida

5. Enviar movimiento
Precondiciones: El cliente debe estar jugando una partida
Prioridad/iteración: media
Flujo normal de eventos: El servidor recibe un mensaje JSON de un usuario con la información relativa al movimiento que ha realizado Postcondiciones: el servidor actualiza el tablero de ese jugador e informa a su contrincante del movimiento
Descripción: mediante este CDU el jugador puede realizar movimientos en una partida

5.2. Diagrama de clases de la solución

En esta sección se muestra el diagrama con las clases más importantes del sistema.