Escuela Superior de Informática

Universidad de Castilla – La Mancha

Práctica 2

Tres en Raya Definitivo Web

Autores
Marchán Loro, Javier
Peralta López, Ángel
Pérez Pascual, Rubén
Ruedas García, Antonio

Asignatura: Ingeniería del Software II
Titulación: Ingeniería Informática
Fecha: 2 de junio de 2014

Índice

1. Introducción

Este documento describe el análisis, desarrollo e implementación llevados a cabo para obtener el juego **Tres en Raya Definitivo Web** [1].

Se trata de una aplicación cliente-servidor web. Es requisito indispensable la existencia y ejecución del servidor de escritorio RMI ya que el servidor web depende de este. El lenguaje utilizado para la implementación ha sido Java y el framework *Google Web Toolkit* (GWT).

Este documento pretende que el lector obtenga una idea clara de la composición del sistema, para ello se muestra la arquitectura del sistema que ayudarán a formarse una imagen mental, a alto nivel, del sistema desarrollado. Tras esto, se explicarán las pruebas realizadas.

[1] Este documento no explica las reglas del juego

2. Arquitectura del sistema

En la Figura 1 se muestra una visión de la arquitectura del sistema. Se trata de una arquitectura cliente-servidor a través de servicios web. Su implementación se ha llevado a cabo usando el framework Google Web Toolkit(GWT). El servidor proporciona e implementa una serie de operaciones a través de un interfaz que extiende de "Remote Service". Este servidor web es un mero proxy para los clientes web, ya que no contiene la funcionalidad del servidor como tal, que la sigue mantiendo el servidor de escritorio.

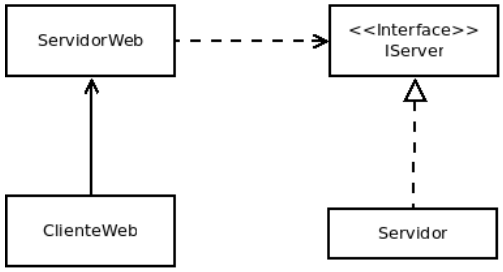

Figura 1: Interacción entre cliente web, servidor web y servidor de escritorio

Los clientes solicitan los servicios que el servidor web ofrece usando la interfaz proporcionada y este, se comunica a través de RMI con el servidor de escritorio. En esta comuncación el servidor web actúa en rol de cliente del servidor de escritorio, que es el componente que contiene toda la lógica propia del servidor como tal. Esta comunicación se realiza mediante RMI. Por lo tanto los flujos de información que se producen, se resumen en:

- Clientes web a servidor web.
- Servidor web con rol de cliente de escritorio a servidor de escritorio.

En la Figura 1 se puede observar la arquitectura descrita. Al igual que en la anterior práctica, la lógica del juego reside en la parte del cliente web.

En la siguiente figura, se puede apreciar la arquitectura de la aplicación web, conformada tanto por el servidor web y el cliente web.

2.1. Arquitectura del Servidor web

Como se puede apreciar en la Figura 2, el servidor web está basado únicamente en una capa de comunicación ya que no posee dominio o persistencia alguna.

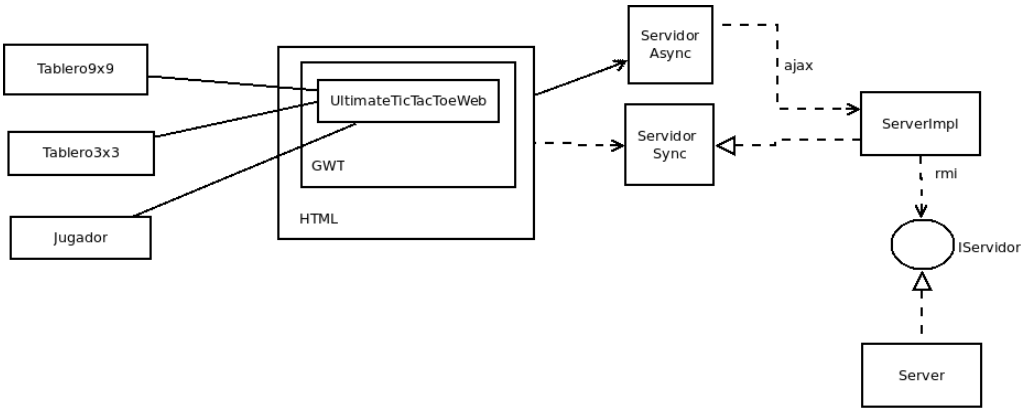

Figura 2: Arquitectura del sistema

- *Comunicación:* La clase encargada de esta función es *ServerImpl*. Implementa la interfaz del Servidor Síncrono en la parte web y usa la interfaz del Servidor RMI, permitiendo la comunicación con otros clientes, tanto web como de escritorio.

2.2. Arquitectura del Cliente web

En la Figura 2 se muestra la arquitectura multicapa del servidor:

- *Comunicación:* La clase encargada de estas tareas es *UltimateTicTacToeWeb*. Simplemente realiza llamadas asíncronas al servidor web y obtiene su respuesta.
- *Interfaz:* Contiene las funciones que representan y modifican la GUI del cliente en el navegador. Se encarga la clase *UltimateTicTacToeWeb* de esta función.
- *Dominio:* Contiene las clases de dominio como las encargadas de representar a cualquier juego. Estas clases son Tablero3x3, Tablero9x9 y Jugador.

3. Pruebas de los casos de uso

3.1. Pruebas para la aplicación web

Los test de la aplicación web han sido llevado a cabo mediante el testbed *Selenium*. Las pruebas que han podido ser cubiertas y reproducibles han sido:

- Pruebas relacionadas con el login, posibles causas de error y de éxito.
- Pruebas relacionadas con el registro, posibles causas de error y de éxito.
- Pruebas relacionadas con el cierre de sesión, posibles causas de error y de éxito.

Para poder realizar estas pruebas ha sido necesario introducir una clase *Broker* para la comunicación con la base de datos y una clase auxiliar que contiene las operaciones de borrado de esta base de datos.

Las pruebas relacionadas con la lógica de dominio por parte del cliente quedan cubiertas con las pruebas que se realizaron para el cliente web, ya que el código se ha cogido tal cual de esa implementación.

Se ha intentado realizar una prueba respecto de la jugabilidad, pero al no ser reproducible, se ha descartado. Estas pruebas han sido llevadas a cabo de forma manual con mucha dedicación y esfuerzo, probando las múltiples posibilidades para poder llevar la cobertura al máximo posible.

Las pruebas se encuentran en el paquete *test.terd.web*.