Universidad ORT Uruguay

Facultad de Ingeniería

Bernard Wand Polak

Programación de redes

Obligatorio 2

Esteban Guruceaga - 225042

Diego Pizzarello - 211576

Índice.

Arquitectura.

El sistema está compuesto por los siguientes componentes:

- Client (viejo)
- Client Admin
- Server (viejo)
- Server Admin
- Server Log

Client: Inicializa dos conexiones con el servidor utilizando Sockets, una se utiliza para las funcionalidades realizadas por el cliente y la otra para escuchar las notificaciones que recibe del servidor (notas asignadas).
Utiliza la clase System.Threading para crear dos Threads, en uno escuchando al socket de notificaciones y el otro a las peticiones del usuario.

Client Admin: En este cliente están disponibles las nuevas funcionalidades del sistema (mirar logs, crear docente y calificar material). Se comunica mediante el protocolo HTTP con el Server Admin.
Además, tiene intercambio de información con el *Server Log* utilizando la tecnología WFC (Windows Communication Foundation) para solicitar los logs.

Server: Dentro de la solución se encuentra estructurado en carpetas el dominio y la lógica de negocio.
Al iniciar el servidor crea dos Threads, en uno escucha a los clientes y el otro despliega un menú para atender las solicitudes del servidor (crear alumno, crear/eliminar curso, calificar alumno).
Cuenta con una clase *ClientMenuHandler* que hace de nexo entre la lógica de negocio y la respuesta al *Client*.
En base a las nuevas funcionalidades requeridas para la segunda entrega, agregamos dos métodos en la lógica de negocio y una clase (*Teacher*) en el *Domain.*

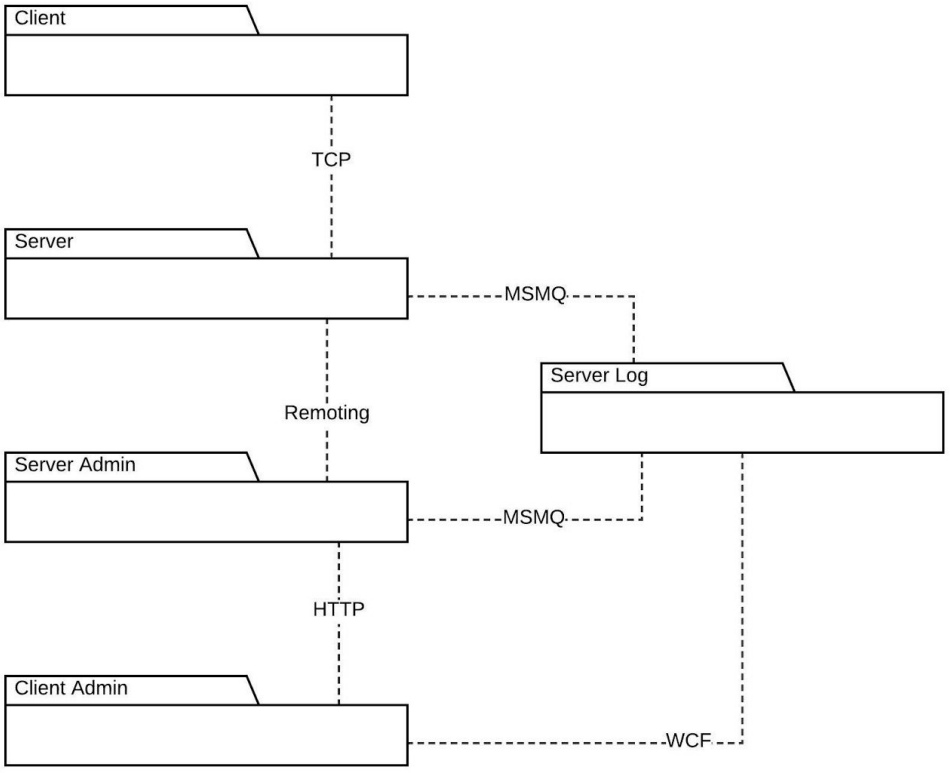
Server Admin: Servidor web que atiende las peticiones del *Client Admin* a través de HTTP y se comunica con el *Server* mediante Remoting.

Server Log: Su funcionalidad es iniciar un host con el servicio *LogService* el cual se encarga de devolver los logs filtrados. Este servicio utiliza la tecnología MSMQ para consumir los logs.

Diagramas

Paquete

El siguiente diagrama de paquetes muestra la arquitectura mencionada anteriormente, dejando en claro cada tecnología utilizada para la comunicación. El objetivo de este diagrama no es entrar en detalle, sino tener un panorama general del sistema.



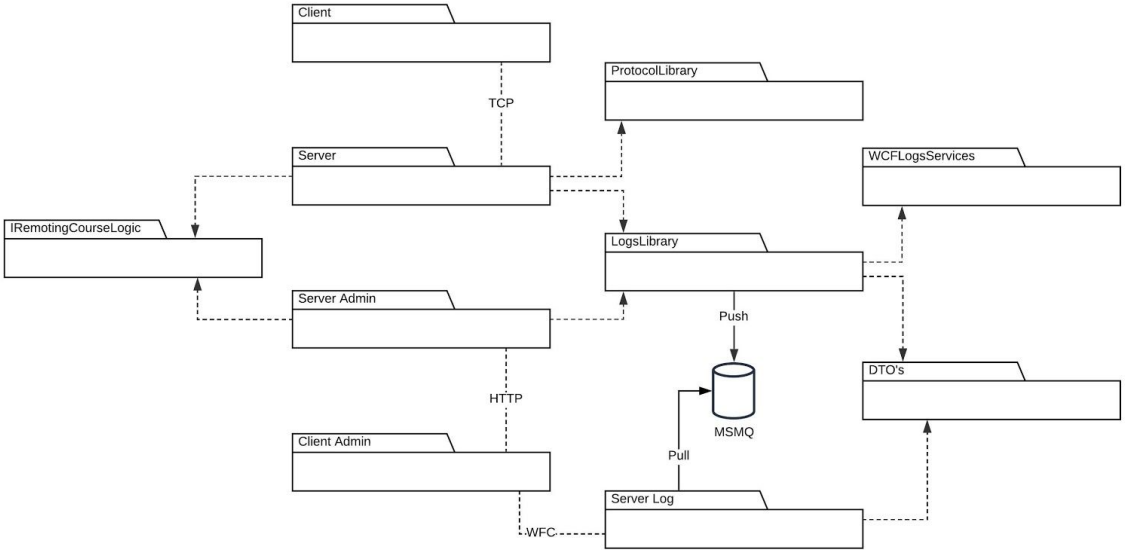
Paquete (en detalle)

Logs Library: Ambos servidores (*Server* y *Server Admin*) utilizan esta clase para enviar logs a través de MSMQ.

Protocol Library: Creamos esta solución para encapsular la lógica del protocolo creado, brindando al cliente métodos para enviar y recibir data. Decidimos colocar constantes los commands utilizados para las diferentes response y requests.

IRemotingCourseLogic: Definimos la interfaz con la cual se va a comunicar el *ServerAdmin* con el *Server*.

Podemos ver estos nuevos componentes en el siguiente diagrama.



Secuencia

En un diagrama de secuencia podemos representar un caso específico de la comunicación cuando se quiere agregar una calificación desde el *ClientAdmin*. En primer lugar se comunica con el *ServerAdmin* a traves de HTTP utilizando el método del HttpClient PostAsJsonAsync, luego se utiliza Remoting para la comunicación entre el nuevo servidor y el viejo. Una vez retorna ok el servidor viejo, el *ServerAdmin* utiliza a *LogsLibrary* para llamar al método de crear un Timestamp. El Timestamp es creado utilizando MSMQ.

DIAGRAMA DE SECUENCIA: AGREGAR CALIFICACION