Protocolo de comunicación del equipo 1 con merge equipo 2

Equipo 1:
- Nathalie Alfaro Quesada, B90221
- Jesús Alonso Porras Arguedas C26007
- Jordan Esteban Barquero Araya C30965
- Rodrigo Mendoza Quesada C04813

Propósito

- Objetivo: Estandarizar el diálogo Cliente ↔ Tenedor ↔ Servidor de Figuras para listar, agregar y consultar figuras de tipo ASCII.
- Inspiración: Sintaxis HTTP (líneas y encabezados), pero con nombres propios y reglas simplificadas.

Transporte (IPC) y codificación

- Codificación: UTF-8 en todo el mensaje.
- Conexión lógica: Persistente hasta que se cierre la conexión. Los Tenedores envían un aviso mediante UDP a los demás dispositivos de las otras islas, para que se comuniquen. Dicho aviso debe indicar la dirección IP en qué puerto se encuentra activo. De esta forma, los demás equipos pueden conocer qué nodos están disponibles y listos para establecer conexiones a través de TCP.

Conexión

Antes de iniciar la comunicación, los actores que participan en esta deben estar conscientes de la presencia de los demás. Para lograr esto, se propone que los componentes establezcan una conexión al ser encendidos por medio de un acuerdo.

Hay 2 casos para este modelo de comunicación. Caso uno, el componente se enciende, realiza un broadcast con dirección IP, puerto y tipo de componente (Tenedor, Servidor o Cliente); como es el único, no recibe respuesta y queda a la espera. Caso dos, cuando ya hay otro componente, en este caso cuando el segundo componente se enciende, recibe respuesta y puede establecer de una vez la conexión. Cada uno debe recibir el reconocimiento del otro que recibió la información de la conexión, de lo contrario debe esperar a que otro componente inicie la conexión.

El flujo entre el Tenedor y el Servidor en el descubrimiento se vería así:

> [Servidor: espera] espera un mensaje desde el broadcast para solicitar conexión (no recibe nada, queda en espera)
> [Tenedor: espera] broadcast para solicitar conexión a las islas-> [Servidor]
> [Servidor: iniciado] envía al tenedor la respuesta con su información -> [Tenedor]
> [Tenedor: conectado] establece una conexión formal -> [Servidor: conectado]

Formato del mensaje

Los mensajes tendrán una estructura sencilla de dónde comienza y termina el mensaje para saber que solo hay que tomar lo que está entre las etiquetas creadas, estas etiquetas están en mayúscula y el inicio se denota con la palabra BEGIN seguido por un slash y luego continua el cuerpo o la información relevante, luego sigue otro slash con la palabra END que denota que termina ese comunicado, un ejemplo de la estructura principal es:

BEGIN/data/END

Delimitación de encabezados:

- BEGIN/
- /END

Las instrucciones definidas por el momento presentan este tipo de estructura. Cuando se ejecuta el programa, se enciende el Tenedor y el Servidor, por lo cual estos indican su estatus, su dirección IP de la computadora donde están funcionando y el puerto de comunicación:

BEGIN/status/actor/IP/port/END

Servidor:

- BEGIN/ON/SERVIDOR/IP/port/END
- BEGIN/OFF/SERVIDOR/IP/port/END

Tenedor:

- BEGIN/ON/TENEDOR/IP/port/END
- BEGIN/OFF/TENEDOR/IP/port/END

Para estandarizar las peticiones UDP, se define el siguiente puerto para el mensaje de broadcast en Tenedores y Servidores:

- Tenedor: 4321
- Servidor: 1234

Estos puertos se deben usar para recibir mensajes de conexión y en caso de que luego de un ciclo de reconexión no se presente un mensaje de algún servidor, se procederá a descartar y cerrar la conexión principal con aquel servidor.

En lo que respecta a la comunicación TCP, se usará en las conexiones principales entre Clientes-Tenedor y Tenedor-Servidor, gracias a la información de los mensajes de broadcast, y para no interferir con aquella lógica se usará los siguientes puertos:

- Cliente - Tenedor: 8080
- Tenedor - Servidor: 8081

En caso de que el Cliente solicite una figura que no aparece en la lista de figuras, es porque no está contenida en el sistema de archivos, por lo cual se le desplegará un mensaje de error con un código:

- BEGIN/404 - The figure does not exist in the file system/END

En caso de que algunos de los tres protagonistas (Cliente, Tenedor o Servidor) pierda la conexión por diversas razones, estos emitirán un código más un mensaje que indican el incidente:

- BEGIN/100 - Server offline/END
- BEGIN/101 - Fork offline/END
- BEGIN/102 - Client offline/END

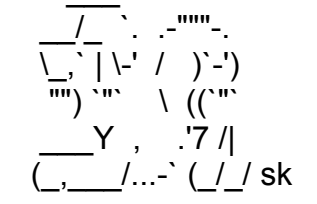
Comandos

Mostrar la lista de figuras con el comando LIST y si esta acción tiene éxito, despliega la lista de figuras junto con el código de 200 OK, el “200 OK” indica que la acción se realizó con éxito.

> **LIST** BEGIN/listFigures/END
>
> Respuesta:
> BEGIN/200 OK/contentLength22/END
> ballena
> pez
> tortuga

Para solicitar una figura en específico se usa el comando GET más el nombre de la figura y si se realiza la acción con éxito, se despliega el código “200 OK” con la figura completa:

> **GET** BEGIN/nameFigure/END
>
> Respuesta:
> 200 OK perro



Si desea agregar una figura que no está contenida en el sistema de archivos, esta acción se podrá realizar con el comando ADD más el nombre del archivo de texto que

contiene la figura, si la figura fue añadida correctamente, seguido se muestra el código de éxito y un mensaje:

ADD BEGIN/newFigure.txt/contentLength<size>/END

Respuesta:
200 OK Figure added: flowers

Aquí se agrega el resto de la figura, el “<size>” corresponde al tamaño de la misma.

Administración de recursos

Para realizar el ruteo, por cada nueva conexión, se haga una solicitud de LIST por parte del Tenedor a Servidor, y se proceda a añadir los nombres de las figuras en un unordered_map o alguna estructura similar con un valor que corresponda a un puntero socket TCP al servidor correspondiente, por otro lado los punteros dinámicos de los sockets se guardarán en un unordered_map con la dirección IP para su fácil eliminación e identificación.

Se considerará que un servidor de figuras se encuentra muerto cuando no responda al llamado del broadcast de cada cinco minutos y se procederá a cerrar los sockets correspondientes y eliminar las figuras del unordered_map.

Existirán cuatro tipos de hilos:

- Hilo de asignación (Tenedor, Servidor): Por cada solicitud que llegue a un Servidor, se encargará de crear un hilo de atención que continúe con la solicitud por su cuenta.

- Hilo de atención (Tenedor, Servidor): Tiene la función de encargarse de la solicitud, redireccionando al servidor pertinente y esperando por la respuesta, para luego enviarla al cliente.

- Hilo de escucha (Servidor): Se encarga de esperar el mensaje con el encabezado "BEGIN/data/END" para completar el mensaje y enviar al tenedor lo siguiente: "BEGIN/status/IP/port/END".

- Hilo de respuesta (Tenedor): Se encarga de enviar cada 5 minutos el mensaje con el encabezado "BEGIN/data/END" y esperar por una respuesta con los atributos correspondientes.

Resumen de cambios

Se cambió la lógica de descubrimiento y muerte, los servidores esperaran el mensaje broadcast en la isla correspondiente escuchando el puerto correspondiente, cada servidor enviará su respuesta al tenedor que haya hecho el broadcast con el mismo

formato que se definió en un inicio, el tenedor usará la información para establecer una conexión, luego pedirá la lista de figuras desde la conexión recién hecha y la procesa, esta llamada se hará cada cinco minutos, en caso de no responder en un solo ciclo se procederá a cerrar la conexión.

Las conexiones TCP/SSL:

- El puerto 8080 será para conexión Cliente-Tenedor.
- El puerto 8081 será para la conexión estable Tenedor-Servidor.

Sé agrego una descripción de cómo se piensa implementar el Servidor-Tenedor.

Se estandarizaron los mensajes.

Se agregaron roles de hilos.