Documentación del tercer obligatorio

Repositorio

https://github.com/federicocarbonell/ProgDeRedes

Autores

- Federico Carbonell - 224359
- Juan Pablo Sobral - 192247

Manual de usuario

1. Iniciar aplicación StateServer
    - Ejecutar ServerEstado/StateServer.exe
2. Iniciar resto de las aplicaciones
    - Ejecutar AdminAPI/AdminServer.exe
    - Ejecutar ServerCliente/Server.exe
    - Ejecutar Cliente/Client.exe
3. Iniciar aplicación LogAPI
    - Esta aplicación se la puede ejecutar desde el principio, luego de haber operado, etc dado que los mensajes existen mas allá de el consumidor.
    - Ejecutar LogsAPI/LogAPI.exe

Endpoints APIs REST

Pruebas en ruta /pruebasPostman

USERS

- POST /users crea usuario (userDTO en el body)
- GET /users ver todos los usuarios
- DELETE /users/{id} borra (logico) usuario con id dado
- PUT /users/{id} actualiza usuario (userDTO en el body)
- POST /users/{id}/games?gameId={gameId} asocia al juego con el id $gameId al usuario con el id $id

```
//userDTO
{
    "username": "prueba",
    "password": "prueba"
}
```

GAMES

- POST /games crea juego (gameDTO en el body)
- GET /games ver todos los juegos

- DELETE /games/{id} borra (logico) juego con id dado
- PUT /games/{id} actualiza juego (gameDTO en el body)
- POST /games/{id}/owners?buyerName={buyerName} asocia al juego con el id $gameId al usuario con el username(secondary key) $buyerName

```
//gameDTO
{
    "name": "cyberpunk 2077",
    "genre": "rpg",
    "description": "inspired by the cyberpunk tabletop game"
}
```

LOGS

- GET /logs obtiene los logs generados por el sistema
- GET /logs?usuario={usuario}&juego={juego}&fecha={fecha} soporta filtrado de la siguiente forma.

Alcance proyecto

Alcance LogServer

- API REST
- Permite ver todos los logs o filtrar en funcion de parametros como fecha, usuario, juego.
- Ejemplos de request en la coleccion de postman.

Alcance AdminServer

- API REST
- ABM de juegos, usuarios y posibilidad de asociarlos
- Ejemplos de request en la coleccion de postman.

Alcance StateServer

- API gRPC
- Fuente autoritaria de datos.

Alcance Server

- Servidor para protocolo propietario.
- Utiliza StateServer como fuente de datos.

Alcance Cliente

- Cliente para protocolo propietario
- Conexión y desconexión al servidor.

  El cliente se conecta al servidor de manera automática al iniciarse la aplicación. Se puede desconectar con la opción 0 del menú principal, cerrando así también la aplicación.

- Publicación de juego.

  Opción 1 del menú del cliente. Luego de dado de alta al juego se notifica al cliente con un mensaje del lado del servidor. Se puede chequear la adición haciendo uso de la opción 6 del menú (ver todos).

  PARAMS PUBLICACION

Nombre	Genero	Descripcion	RutaCaratula
string	string	string	string

  Por esta iteración, no tenemos muchos chequeos sobre los datos recibidos del lado del servidor, lo cual puede llegar a llevar a problemas con las carátulas dado que se asume que los nombres de los juegos van a ser únicos, y por lo tanto guardamos las carátulas del lado del servidor como nombreJuego.jpg .

- Baja y modificación de juego.

  Opciones 2 y 3 del menú del cliente. Si el juego está en el sistema, modifica sus datos con los recibidos o lo marca como borrado.

  PARAMS BORRADO

Id
int

  PARAMS ACTUALIZACION

Id	Nombre	Genero	Descripcion	RutaCaratula
int	string	string	string	string

- Búsqueda de juegos.

  Opción 7 del menú del cliente. Se puede buscar por nombre del juego, el cual retorna matches parciales. Se puede buscar por categoría, que retorna solo matches absolutos. Se puede buscar por calificación, se retornan juegos con promedio de calificaciones >= al parámetro de búsqueda.

  Importante a la hora de probar búsqueda por calificación, tener en cuenta que los juegos que aún no han sido calificados tienen una calificación nula, es decir, no van a ser tenidos en cuenta a la hora de evaluar los juegos que cumplan con la condición.

  PARAMS BUSQUEDA

Modo	Nombre(opcional)	Genero (opcional)	Rating minimo (opcional)
int	string	string	int

- Calificación de un juego.

  Opción 4 del menú del cliente. Se permite calificar juegos, luego podemos verificar que la calificación quedó registrada de manera exitosa en el detalle del juego calificado.

PARAMS CALIFICACION

Id	Rating	Comentario
int	int	string

- Detalle de un juego.

  Opción 5 del menú del cliente. Se busca por id del juego, el cual se puede obtener utilizando la opción de listar todos. Nos trae toda la información del juego disponible en el servidor, incluida la lista de las calificaciones obtenidas con sus respectivos comentarios.

  La funcionalidad de la descarga de la carátula no se implementa dado que se recibió instrucción de dejarlo para próxima iteración, pero sería relativamente sencillo, replicando de manera inversa el envío realizado del cliente al servidor.

  PARAMS DETALLE

Id
int

- Compra de juego.

  Opción 9 del menú del cliente. Se le pregunta al cliente como que usuario quiere comprar el juego, y el id del juego a adquirir. Se puede chequear la compra viendo la lista de juegos del usuario.

  PARAMS COMPRA

Id	Username
int	string

- Ver juegos del usuario.

  Opción 8 del menú del cliente. Se le pregunta al cliente la lista de juegos de que usuario desea ver, y se la devuelve.

  PARAMS VER

Id
int

Servidor

- Aceptar pedidos de conexión de un cliente.

  El servidor acepta varias conexiones en paralelo, y se maneja perfectamente respecto al acceso a datos.

- Ver catálogo de juegos.

El servidor permite ver el catálogo de juegos desde cualquier cliente, más allá de que se hayan realizado las adiciones en uno y la lectura en otro.

- Adquirir un juego.

  Este requerimiento queda para próxima iteración dado que se decidió dejar directamente el manejo de usuarios para otra iteración.

- Publicar un juego.

  Una vez publicado un juego, se puede verificar la creación del mismo desde otro cliente sin problemas.

- Publicar una calificación de un juego.

  Al igual que con los juegos, una vez publicada de manera exitosa la calificación se puede verificar desde otro cliente sin problemas.

- Buscar juegos.

  Al igual que con el resto de las funcionalidades, dado que podemos ver el catálogo también podemos filtrar el mismo.

- Ver detalle de un juego.

  Funciona de manera correcta igual que la anterior. No se permite la descarga de carátula aún pero la adición de esta funcionalidad no debería ser compleja.

Manejo de paralelismo y concurrencia

Este fue el principal desafío encontrado en este obligatorio. Nos decantamos por la utilización de una clase ServerState que almacenase el estado del servidor durante su ejecución en memoria. Esta clase es estática y aplica el patrón Singleton, al cual le agregamos el uso de una serie de locks para asegurar la integridad de las operaciones sobre la misma. Las operaciones de lectura sobre las listas de entidades de dominio son de libre acceso, mientras que las de escritura sobre las mismas tienen un lock individual (ej, si el cliente A está escribiendo a la lista de reviews, el cliente B puede al mismo tiempo escribir a la lista de usuarios).

Este fue el manejo previo utilizado para los obligatorios 1 y 2.

En esta instancia se creo un servicio separado que se encargeu de mantener el estado, atrás de una API gRPC. Utiliza de igual forma singletons a la hora de manejar los datos por el mismo motivo. Se utiliza un unico locker a diferencia de la vez pasada que se usaba un locker para lectura y otro para escritura.

Protocolo propietario

El protocolo que utilizamos es muy similar al descrito en la letra del obligatorio:

- Es orientado a caracteres.
- Esta implementado sobre TCP/IP.
- Valores alineados a la izquierda, bytes de relleno tienen valor 0.
- Campos HEADER, CMD, LARGO van con largo fijo, DATOS tiene largo variable segun valor indicado en largo.
- Formato general de la trama

Para la serializacion y deserializacion de los datos, utilizamos la siguiente metodologia:

- Calculamos largo del dato a insertar
- Pasamos a bytes este valor
- Pasamos a bytes el dato a insertar
- Insertamos en el array de datos los bytes del valor seguidos por los bytes del dato

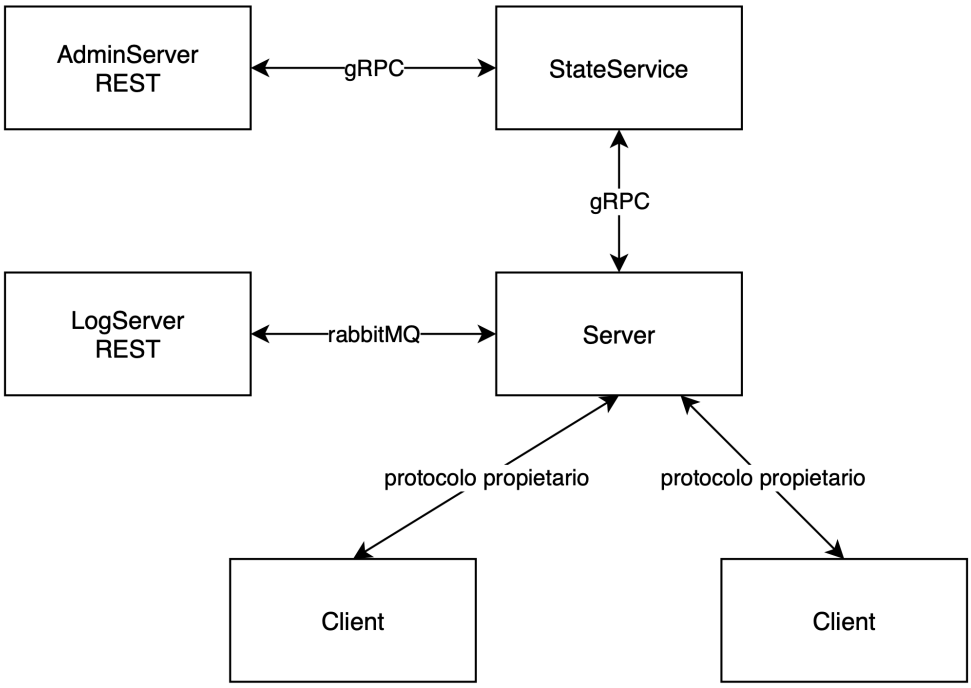
Esto nos permite luego hacer una lectura precisa del lado del servidor. Supongamos que quisieramos enviar un dato que vale 1567.

En ese caso, nuestra trama de datos tendria el siguiente formato => |4|0|0|0|1|5|6|7|

Al querer deserializar los datos el servidor, comienza a leer sabiendo que los primeros 4 bytes se corresponden al largo del dato. Luego, asigna al dato el valor correspondiente a la deserializacion de los siguientes cuatro bytes.

Comunicación entre aplicaciones

Para la comunicacion entre aplicaciones utilizamos las tecnologias gRPC, Rabbit MQ, Rest Api y un protocolo propietario. En el diagrama que se encuentra debajo se encuentra especificado como se utilizaron.

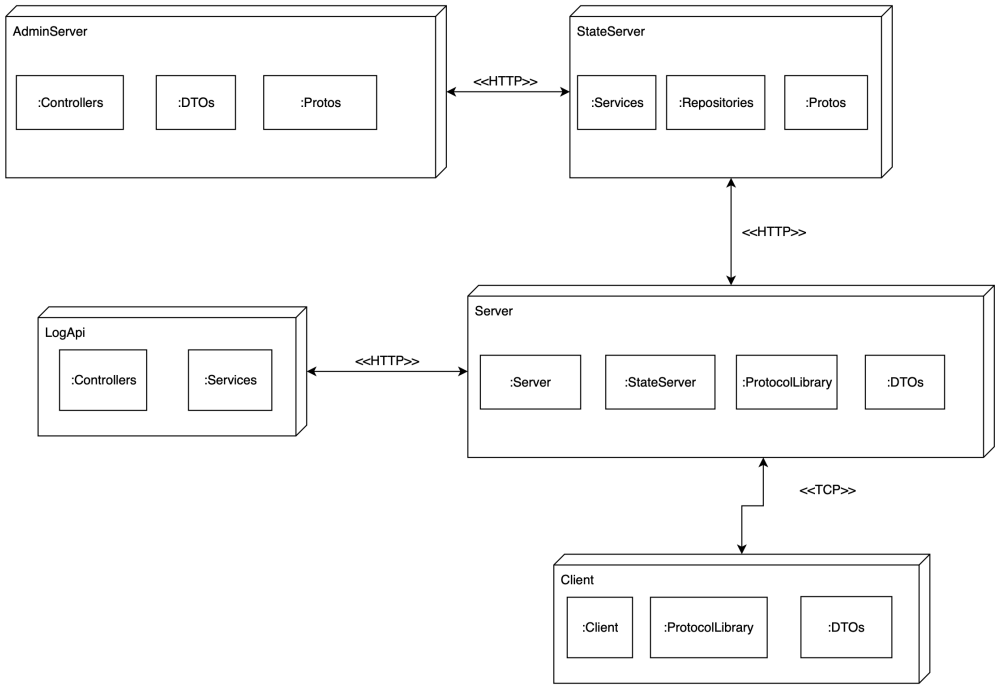


Arquitectura

La arquitectura del sistema esta compuesta por un servidor y un cliente, ellos se comunicaran mediante el protocolo TCP. Como se menciono previamente, se pueden conectar diversos clientes al mismo servidor. Los ejecutables mencionados se pueden deployar en un mismo host, no es necesario tenerlo en separados hosts.

- El servidor se compone por:
  - Server.dll
  - StateServices.dll
  - ProtocolLibrary.dll
- El cliente se compone por:

- Client.dll
- ProtocolLibrary.dll



Responsabilidad de los paquetes

- Paquete Server

  Es el encargado del servidor, recibe las consultas, las procesa y retorna la informació.

- Paquete StateServer

  Es el encargado de manejar la consistencia de datos entre conexiones.

- Paquete ProtocolLibrary

  Es el encargado de almacenar la información necesaria para serializar-deserializar datos y enviar y recibir mensajes.

- Paquete Client

  Es el encargado del servicio de cliente que va a comunicarse con el Servidor. Procesa el input del usuario para luego enviarlo al servidor y tambien recibir y mostrar mensajes del mismo.

- Paquete AdminServer

  Es la api rest encargada de la administracion de usuarios y juegos.

- Paquete LogApi

  Es la api rest encargada de obtener y filtrar logs..

Comandos

Desde el lado del cliente existen los siguientes comandos:

0. Desconectar cliente.
1. Agregar juego.
2. Eliminar juego.
3. Modificar juego.
4. Calificar juego.
5. Ver detalles de un juego.
6. Ver todos los juegos.
7. Buscar juegos.
8. Ver juegos comprados.
9. Comprar juego.

Desde el lado del servidor existe un unico comando:

0. Desconectar servidor.