FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

75.74 Sistemas Distribuidos I

TP1 - Concurrencia y Comunicaciones

Centralized Blockchain

1º Cuatrimestre 2021
11/05/2021

Integrante:

NOMBRE	PADRÓN	CARRERA
Olivia Fernandez	99732	Ingeniería Informática

Introducción

A continuación se muestran diagramas para documentar la solución del trabajo práctico con el fin de facilitar el entendimiento de la implementación.

Diagramas

Diagrama de Clases

A continuación se detalla el diagrama de las clases que se utilizaron para resolver el trabajo práctico:

- **Request handler**: La interfaz del sistema, tiene varias conexiones abiertas escuchando requests de los clientes y en base al tipo de request la manda por las distintas colas a las otras entidades
- **Query Handler:** Entidad que se encarga de recibir las consultas y resolverlas de manera acorde. En caso de ser una request de tipo **GET_STATS** las tiene el mismo almacenadas, sino deberá conectarse con la entidad Reader Manager de la Blockchain para poder hacer la consulta. Una vez obtenida la respuesta de la consulta se la debe comunicar al Request handler para que le responda al cliente que la realizó.
- **Miners Handler**: Entidad que recibe los chunks a procesas de los clientes y se encarga de armar los bloques y enviarselos a los miners para que realicen el cómputo para minarlo. También se encarga del orquestado de los mineros para que actúen sincronizadamente y de la forma más óptima posible, para eso tiene tres colas de comunicación, una para enviar los bloques, otra para recibir el resultado del minado y finalmente una para avisar a los miners si deben o no dejar de minar el bloque que están intentando minar y pasar al siguiente.
- **Miner**: Entidad que almacena la condición de minado y tiene un loop para encontrar el hash que se resuelva el desafío criptográfico y se lo envía al Writer Manager de la Blockchain para guardarlo, el cual le puede responder que tuvo éxito en el minado o que no lo tuvo.
- **Writer Manager**: Entidad de la blockchain que se encarga de recibir los bloques a guardar y verificar que sean válidos, si son válidos se los envía a través de una cola a la entidad Writer, y finalmente le responde a cada minero como le fue.
- **Writer**: Entidad que recibe un bloque y lo escribe en su archivo correspondiente y actualiza el índice por hora.
- **Reader Manager**: Entidad que recibe las requests a realizar y las pone en la cola de requests para que los distintos readers las vayan tomando y realizando, de esa forma se logra un mayor paralelismo. También tiene la cola de responses para escuchar las respuestas de los readers.
- **Reader**: Entidad que escucha la cola de requests y las realiza accediendo al o a los archivos devolviendo los bloques (o el bloque) por la cola de responses.

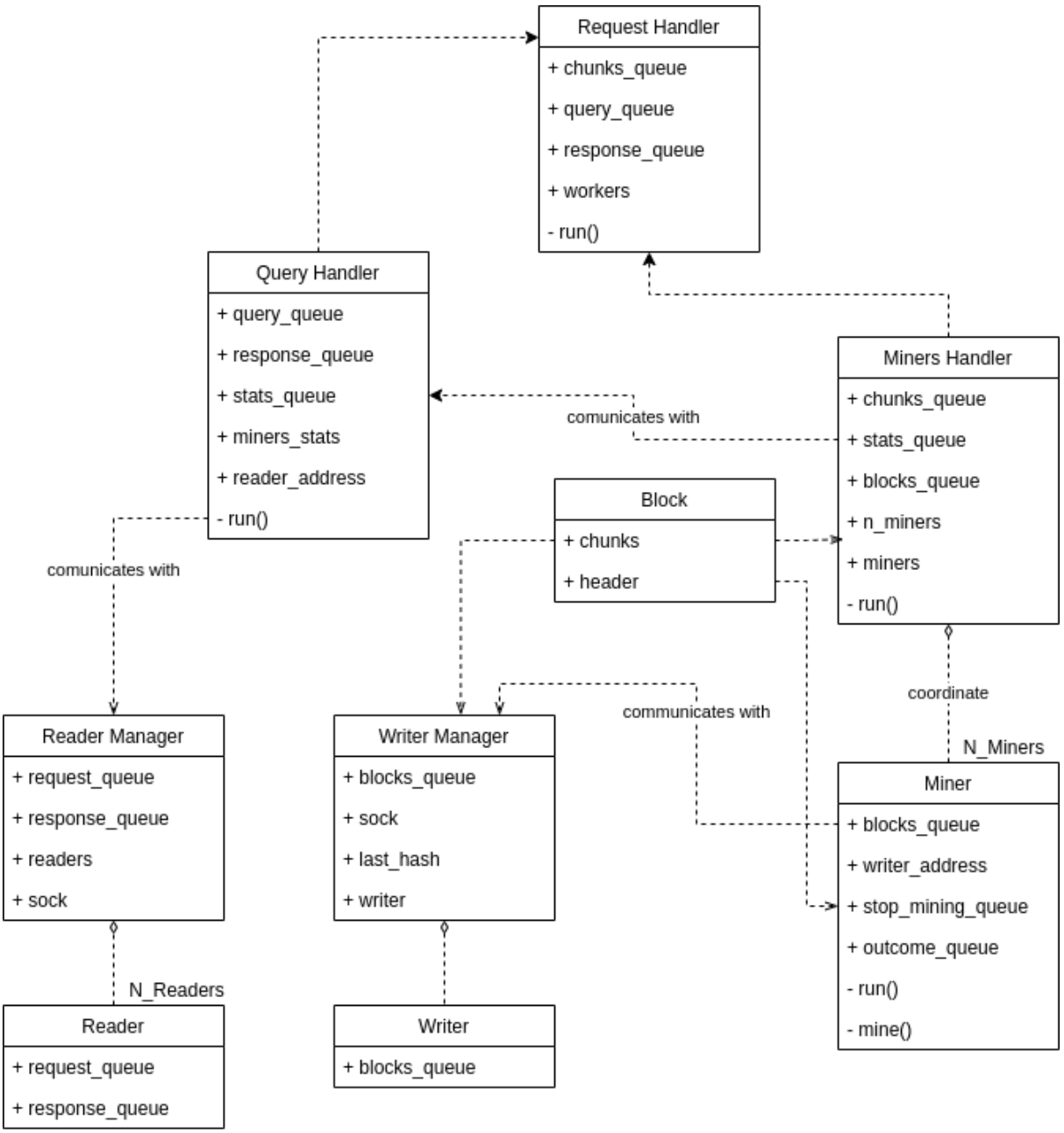


Diagrama de robustez

En el diagrama podemos ver las entidades y cómo se comunican entre ellas.
Además se puede ver cual seria la entrada del Usuario del sistema y cual seria el Nodo de la blockchain aislado para su protección.
En particular a la hora de implementarlo se eligió utilizar en su mayoría threads para utilizar colas de comunicación de manera más simple y solo dejar los sockets para las comunicaciones estrictamente necesarias como eran la comunicación del cliente con el sistema y de algunas entidades con la blockchain.

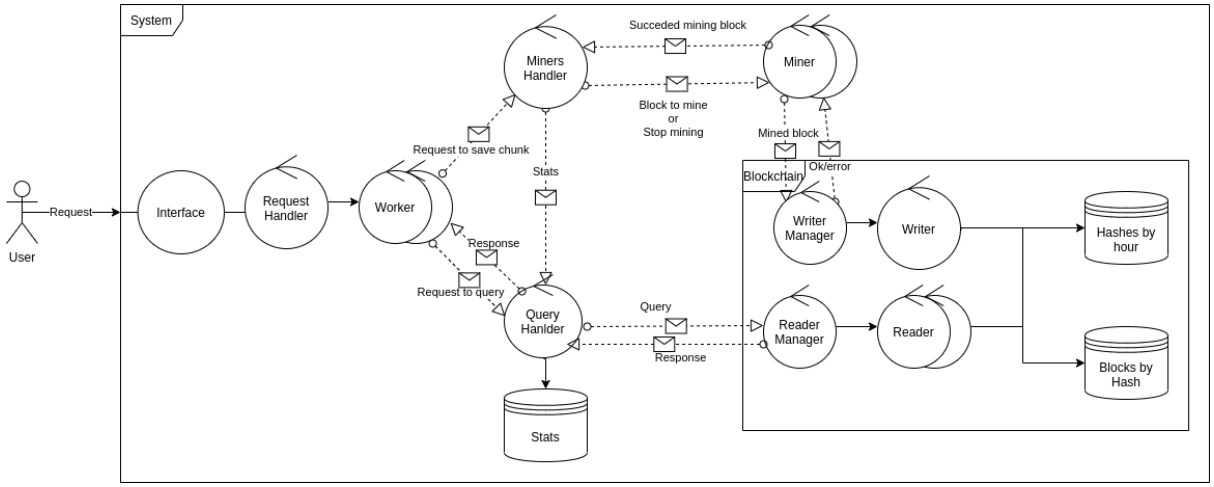


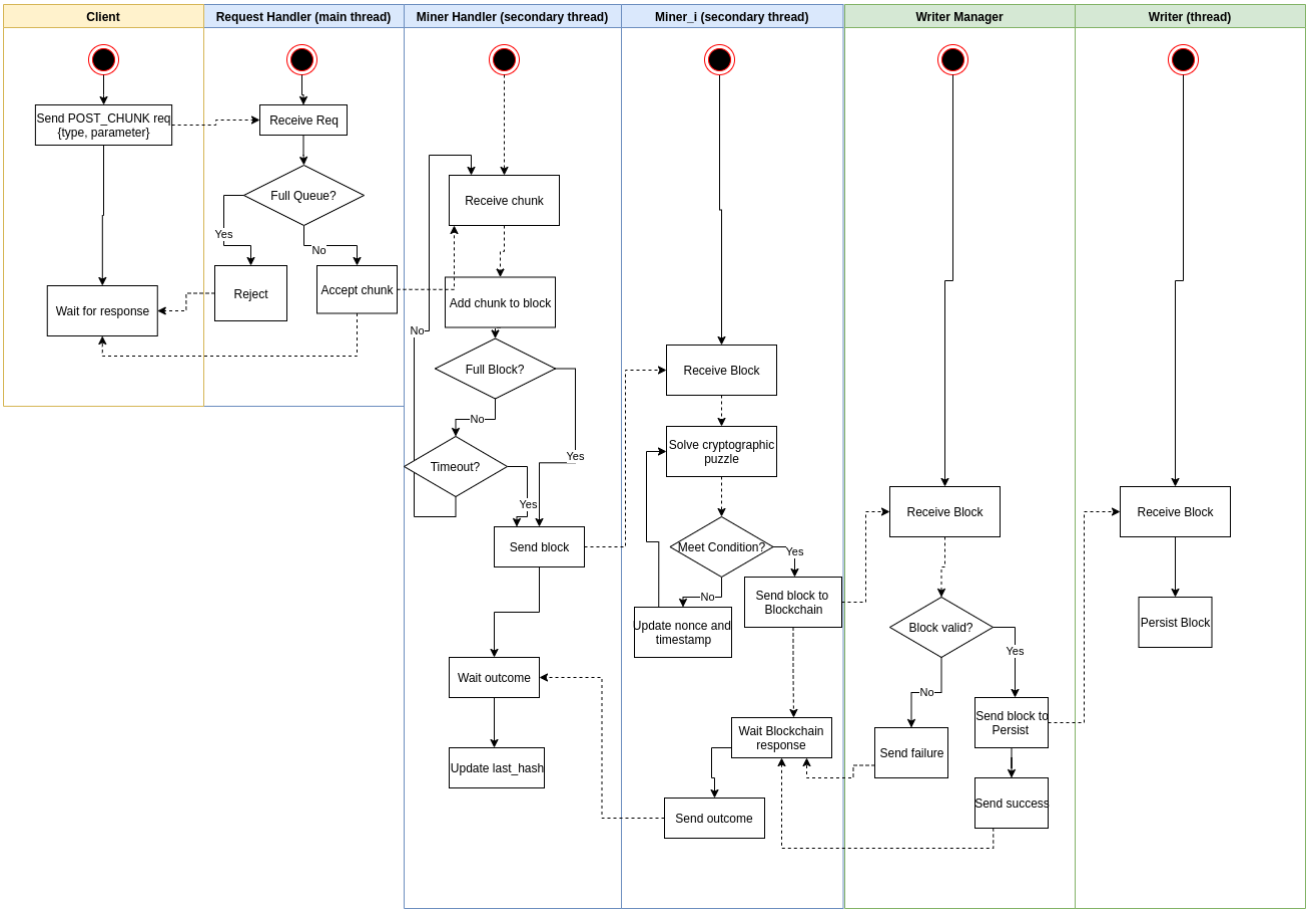
Diagrama de Actividades

Para el diagrama de actividades se eligió mostrar el flujo de **POST_CHUNK** es decir cuando el usuario quiere almacenar un chunk de datos en la blockchain.
Para poder ver los flujos de requests se debería realizar otro diagrama de actividades con ese flujo en particular ya que involucra otras entidades del sistema.

Cabe aclarar que:

- color amarillo se usa para el Nodo Cliente
- color azul Nodo API
- color verde Nodo Blockchain

Por lo tanto la comunicación entre nodos se realiza mediante sockets y la comunicación dentro de un mismo nodo es entre threads y por colas.



También me parece importante aclarar que no hay un fin del ciclo porque todas las entidades del sistema están escuchando indefinidamente por nuevos requests/bloques.