Proyecto Bitcoin

Integrantes:

- Lucas Aldazabal 107705
- Bautista Bosselli 107490
- Juan Francisco Gulden 107985

Comportamiento de Nodo

Node

Nuestra representación de nuestro nodo en la red P2P de bitcoin

El nodo se encarga de actualizar nuestro estado, almacenar los peers a los que está conectado y enviarles las acciones que tienen que ejecutar cada uno de estos

Peers

Nuestra representación de cualquier peer de la red P2P al que estemos conectado

Se encarga del handshake con el peer de la red. Almacena el stream, escucha y manda los mensajes adecuados en cada situación

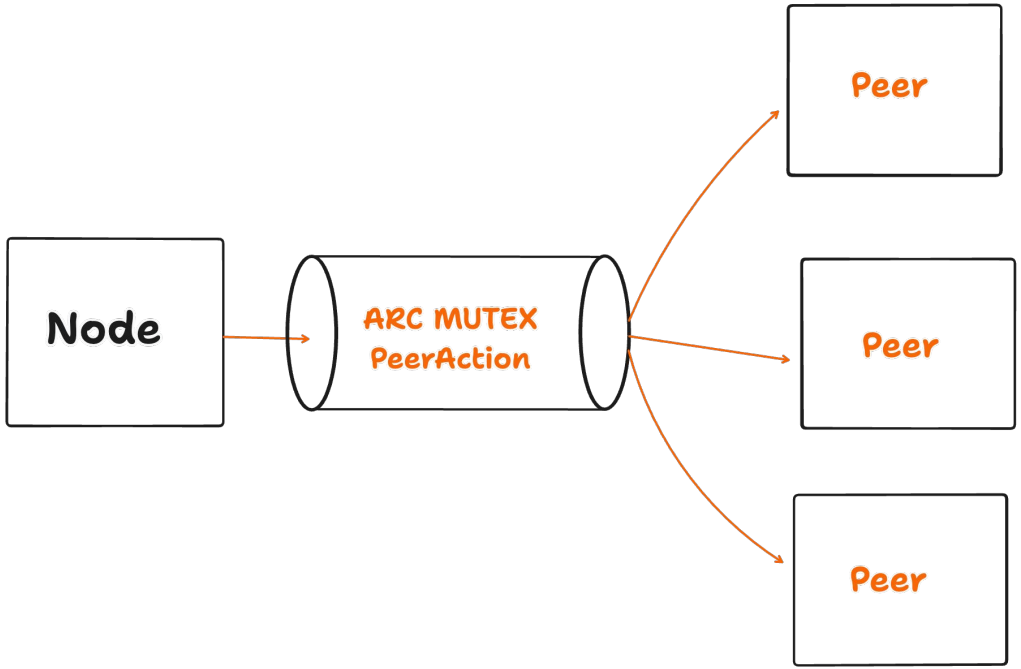
Comunicación entre entes

PeerActions

Son las acciones que se mandan a los Peers que indican que mensajes tienen que enviar por el stream.

Son:

- Obtener headers,
- Obtener datos
- Enviar transacciones

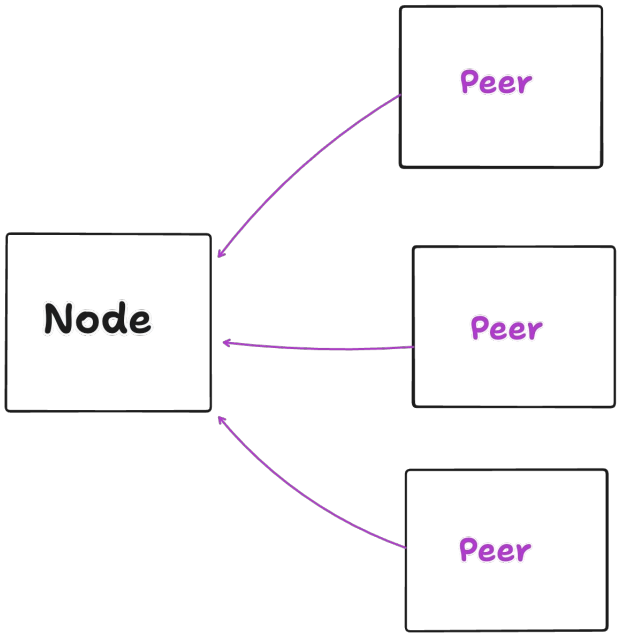


NodeActions

Son los mensajes que se mandan al Node con las acciones que tiene que ejecutar

Algunas son:

- Guardar Headers
- Guardar Blocks
- Hacer Transacciones
- Informacion de errores con Peers
- Etc

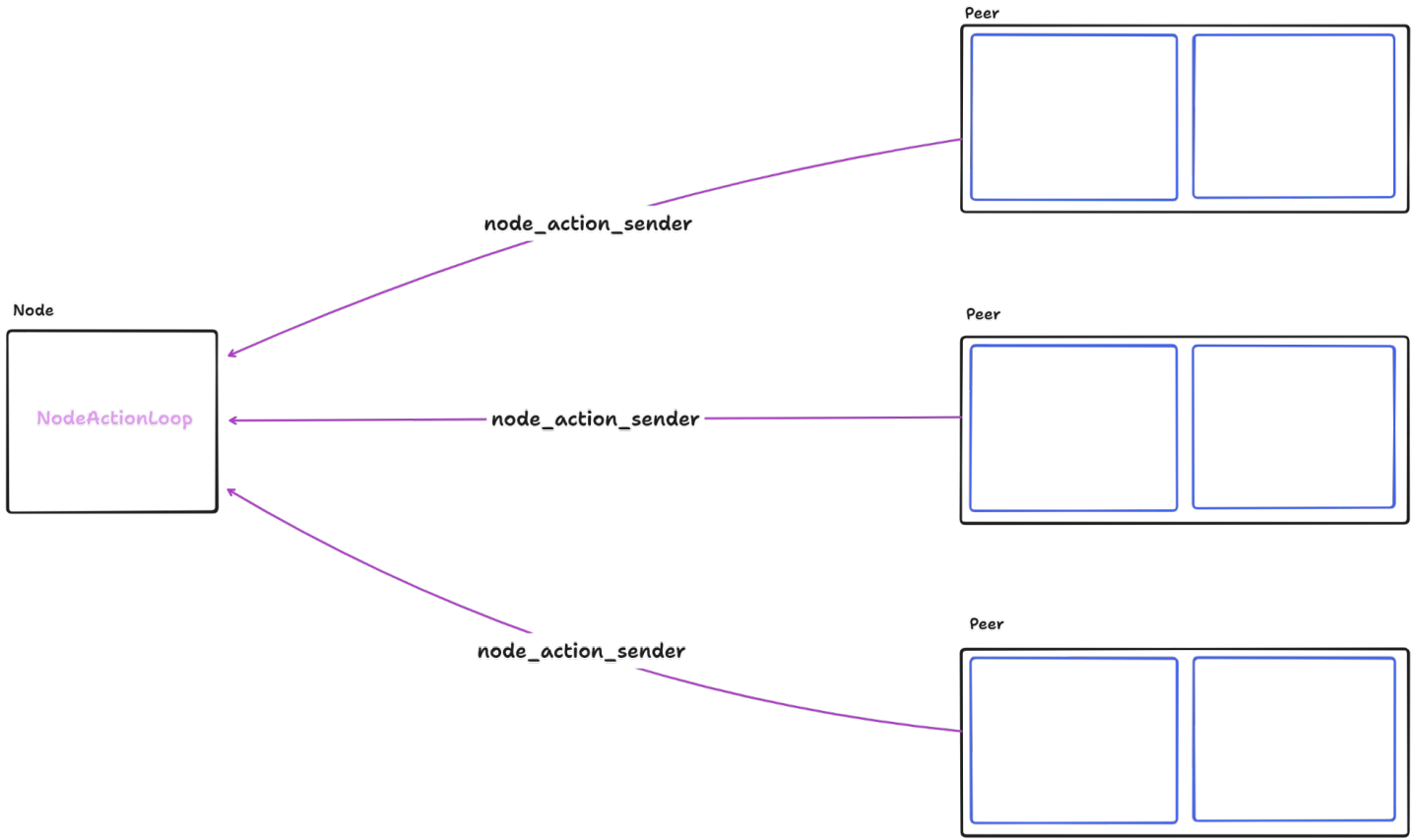


Loops

NodeActionLoop

Genera un loop infinito leyendo los NodeActions que recibe el Node y hace las acciones correspondientes

NodeAction


Peer
node_action_sender
Node
NodeActionLoop
Peer
node_action_sender
Peer
node_action_sender

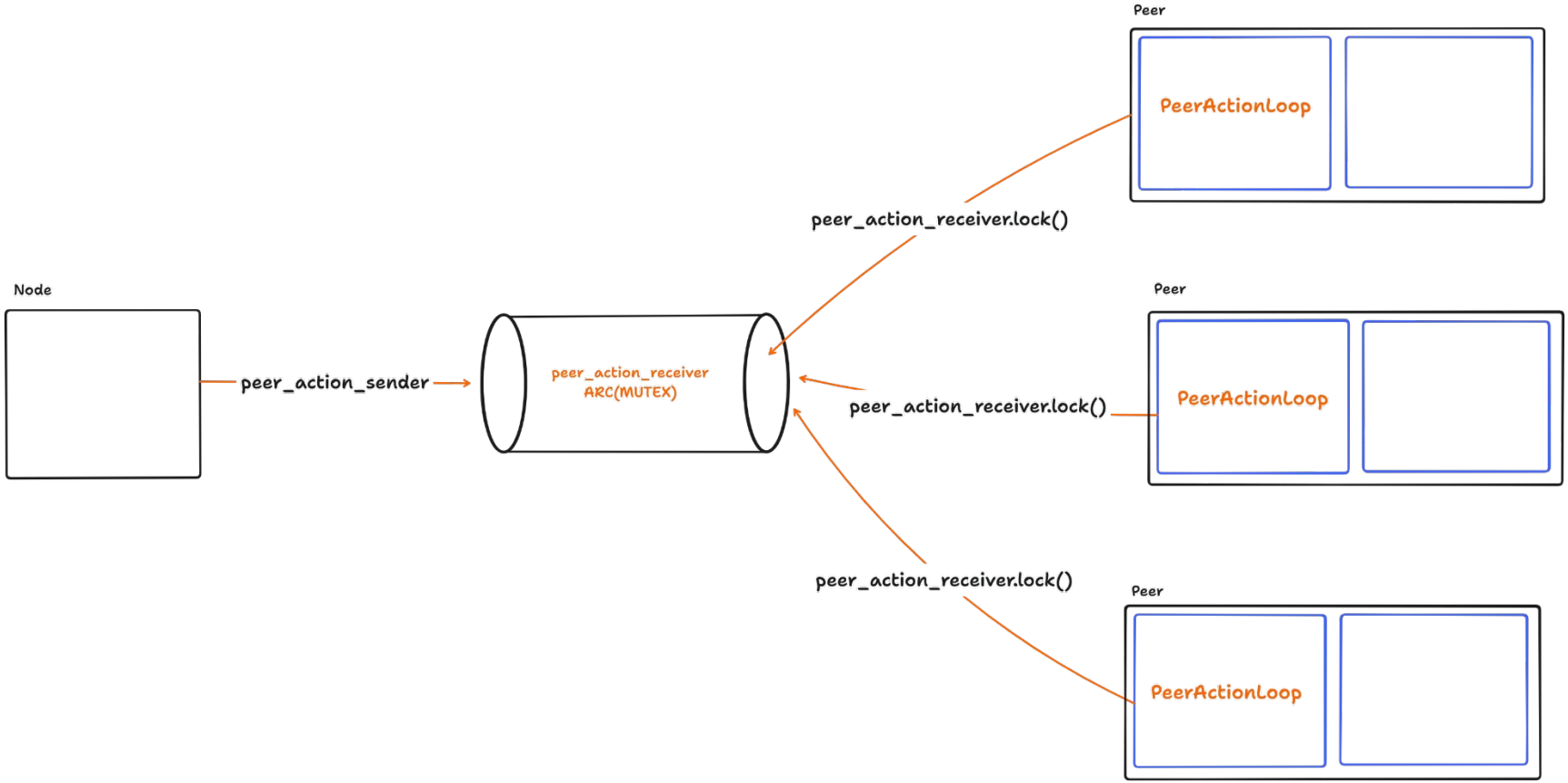
PeerActionLoop

Genera un loop infinito leyendo las PeerActions que reciben los Peers y hace las acciones correspondientes

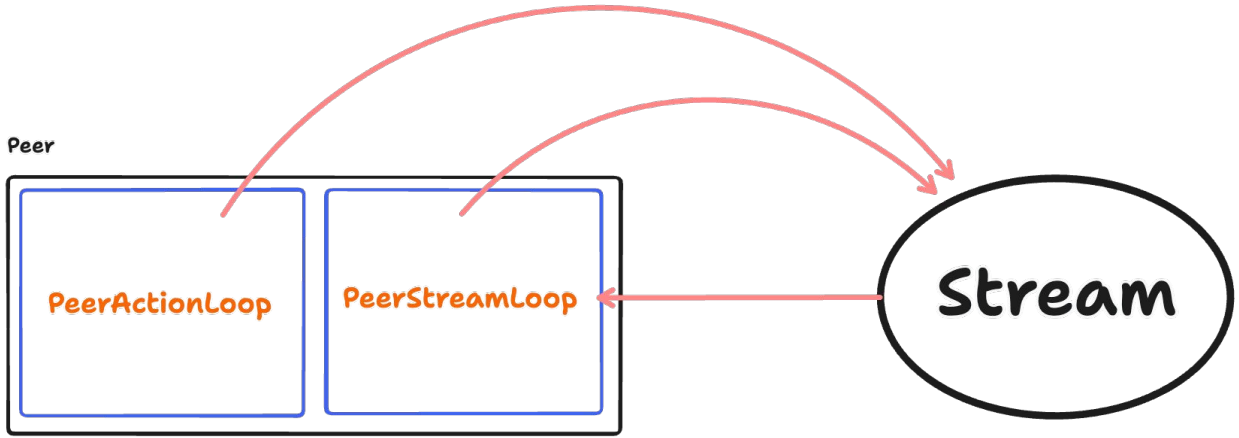
PeerStreamLoop

Genera un loop infinito leyendo el stream con la conexion en la red P2P

PeerAction


Peer
PeerActionLoop
peer_action_receiver.lock()
Node
peer_action_sender
peer_action_receiver
ARC(MUTEX)
Peer
peer_action_receiver.lock()
PeerActionLoop
peer_action_receiver.lock()
Peer
PeerActionLoop

PeerStream



Envio de mensajes

Trait Message

Requiere implementar

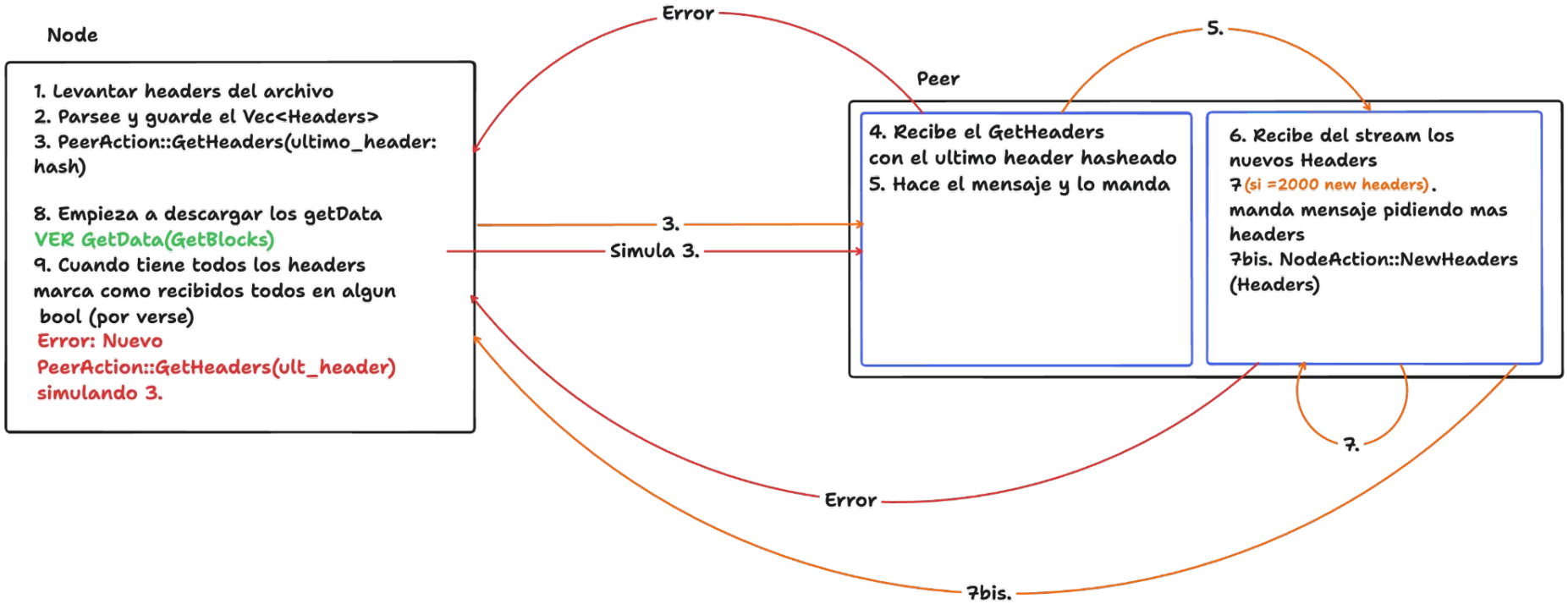
- GetCommand
- Parse
- Serialize

Implementados

- Send
- Read

Cuando se hace send se genera automáticamente un **MessageHeader** con los datos del trait

Ejemplo de una comunicación completa

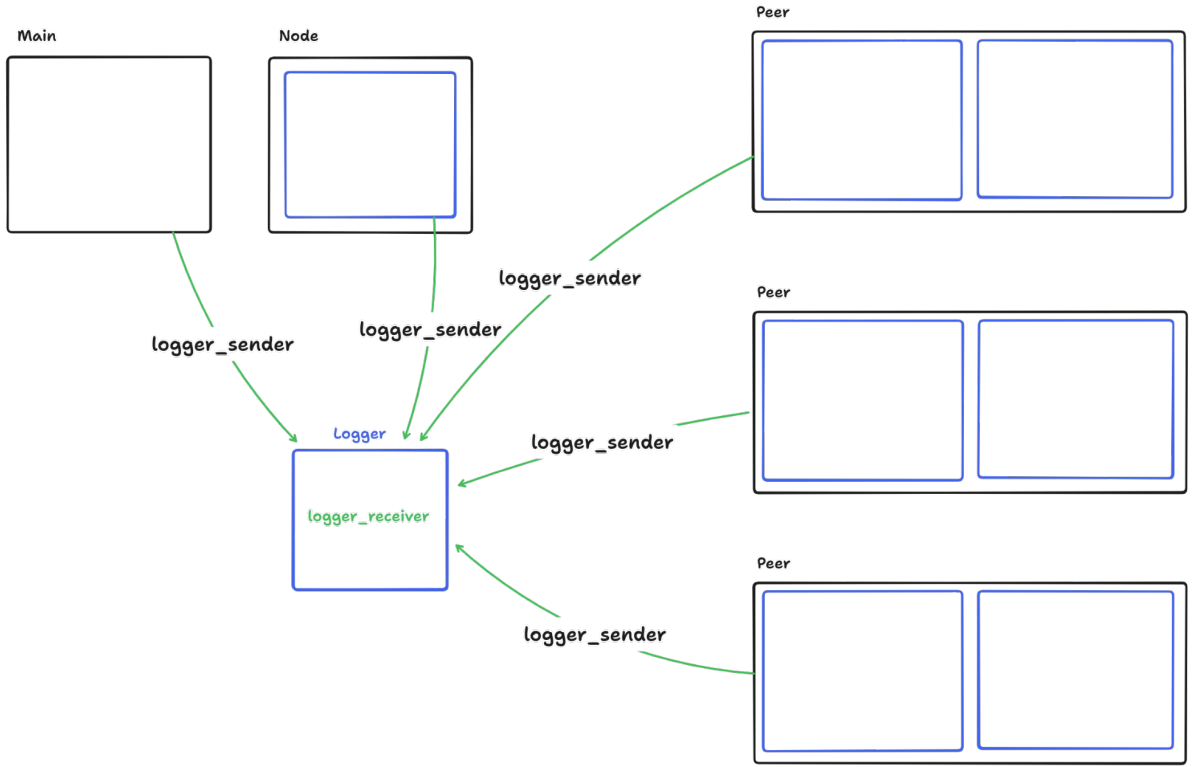


Logger

Logger

Tenemos un enum Log, donde categorizamos a los mensajes entre:

- Message(String)
- Error(CustomError)



NodeState

Sincronización

El Node state está sincronizado con la red cuando termina:

- sincronización de los headers
- sincronización de los bloques
- sincronización del UTXO

Información del NodeState

- Headers
- UTXO
- Pending Blocks
- PendingTxs
- Wallets

UTXO

Una vez sincronizados los Blocks y los Headers se genera el UTXO

Consiste en un HashMap que guarda la información necesaria de las utxo

Una vez generado se guarda un archivo como **“checkpoint”** del hashmap en base al timestamp del último bloque procesado

La siguiente vez se usa ese checkpoint para no tener que procesar todos los bloques nuevamente

Wallets

De las wallets se almacena lo siguiente

- Nombre
- Public y Private Key
- Historial de **Movements**

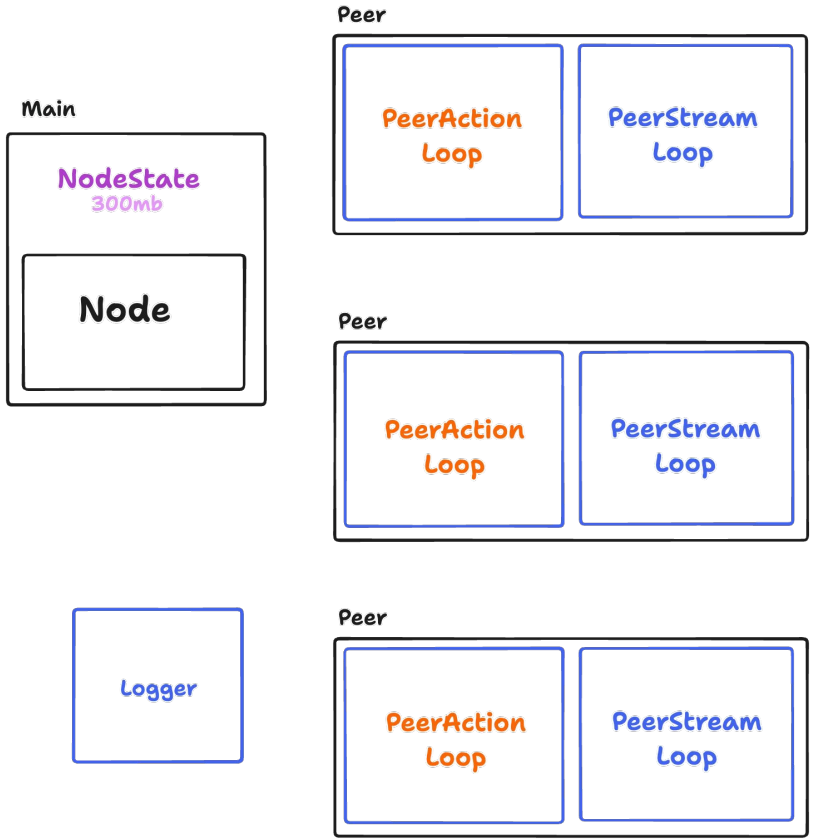
Pending Blocks

Llevamos un registro de los bloques que solicitamos y todavía no recibimos

Hay un **thread** que revisa los que tardaron demasiado y los solicita nuevamente a otro Peer

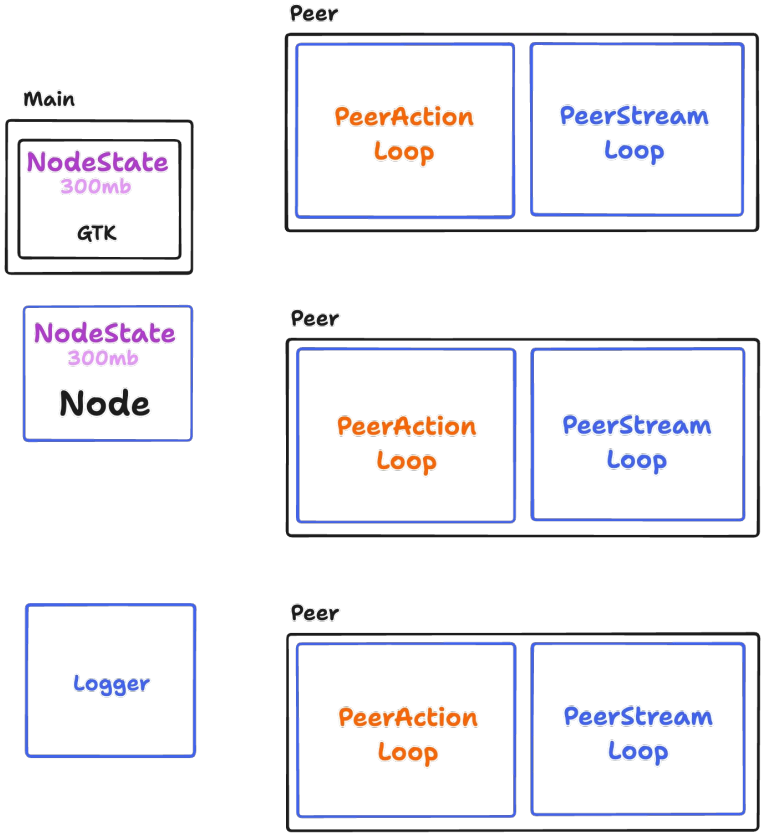
Hasta ahora...

Paso siguiente, agregar el GTK



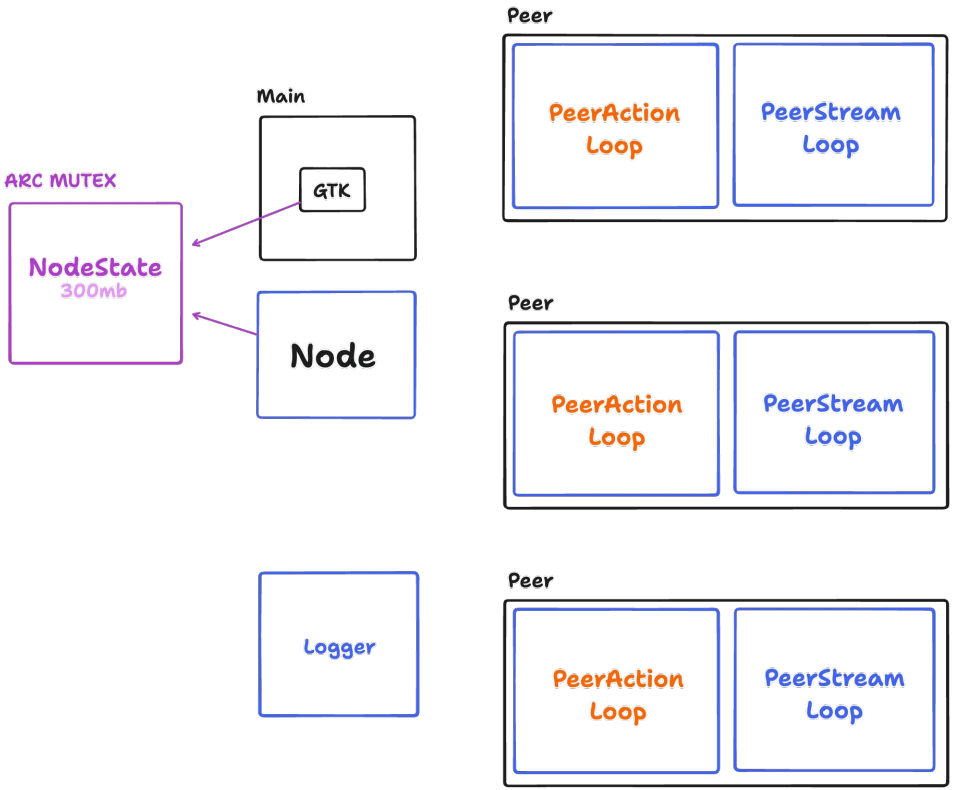
Una opción...

Consume demasiados recursos y rompe con la única fuente de verdad



Nuestra Solución

Un Arc mutex para el NodeState



Interfaz

La interfaz hace lo siguiente:

- Agregar y cambiar de wallets
- Ver balance
- ver tx pendientes
- Ver Utxo
- ver historial
- hacer transacciones
- ver todos los bloques
- Mostrar Merkle Path en Utxo e historial

Partes de la interfaz

Para mostrar los datos y para hacer la interfaz reactiva creamos varias abstracciones que hacen las siguientes tareas

- Manejo de Eventos (GUI Events)
- Manejo de Interactividad

Por ejemplo

- GUI Wallet
- GUI Balance
- GUI Transfer
- etc

GUI Events

Los GUI Events son las acciones que el node state y el node mandan a la GUI vía un **MainContext** para informar que ocurrieron algunas condiciones que hacen que estas se tengan que actualizar

Ej:

- LLegan headers nuevos > GUIEvent::NewHeaders > Blocks se actualiza
- LLega pending Tx de nuestras wallets > GUIEvent::WalletsUpdated > Balance se actualiza

¡MUCHAS GRACIAS!