NEAR Data Flow

Data flow In this video we give a quick overview on the main concept of how the data flows across the NEAR Protocol blockchain. NEAR Protocol blockchain data flow might be a bit tricky at a glance. But it is pretty straightforward and follows well-defined rules. In this article, we are going to have a closer look at how the data flows in NEAR Protocol blockchain.

In fact, any blockchain flow can be represented as an infinity timeline that has a start but has no end.

And a block appears on that timeline with some interval. Each of these blocks holds the information about the previous one, thus, creating achain of blocks .

NEAR Protocol has a sharded nature, meaning that more than one parallel network, called aShard , can be live at any moment. And each Shard produces a chunk of a block at a given interval. A Block in NEAR Blockchain is a collection of the block chunks from all shards. Block chunk is shortened toChunk in NEAR Protocol documentation.

Returning to the data flow itself. The best we can do is to imagine tracks like we could see in audio/video editing apps. Each Shard has its own set of tracks. The top track is for Chunks. They appear no matter what at a given interval, for NEAR Blockchain the interval is about one second. Chunks are being produced even if nothing is happening on the blockchain.

But what do we mean by saying something is happening? We mean that something has triggered some changes in the blockchain. And the most well-known method to trigger any changes is to send aTransaction to the blockchain with instructions of what we want to change and who has requested those changes.

A transaction needs to be constructed, signed and sent to the blockchain. Once it is executed we expect to have a result -ExecutionOutcome . Oh, that is simple. But it’s not really accurate for NEAR Blockchain.

In the beginning there is aTransaction , it contains instructions we want to be executed on the blockchain. Transaction is sent to NEAR Blockchain.

And yes, it is executed there immediately BUT the immediate result of the transaction execution is always justan acknowledgement that it will be executed on the chain; this internal execution request is known asReceipt . You can think of theReceipt as an internal transaction that exists to pass information across shards.

Let’s get back to the tracks to look at the example.

Assuming we have two accounts living on differentShards alice.near andbob.near .alice.near creates aTransaction to send a few tokens tobob.near . TheTransaction is immediately executed and theExecutionOutcome for theTransaction is always aReceipt .

But thisReceipt cannot be executed here, becausebob.near doesn’t live on the same Shard asalice.near , sothe Receipt must be executed on the receiver’s Shard . So the Receipt is moving to the Shard wherebob.near belongs to.

On the destination Shard the Receipt is executed and the process is considered as done.

The explanation here is simplified Please refer to theToken transfer flow article So the final scheme will look like:

Summary

We have learnt the main principles of how data is flowing in NEAR Protocol. We have found out thatTransactions execute and theExecutionOutcome of the Transaction is always aReceipt .

Now we know thatReceipt is a main internal asset for NEAR Protocol blockchain and it has a power of traveling betweenShards . We learnt the NEAR Data flow on a simple example. Of course in real life with more complex transactions, that involve cross-contract calls, there will be more Receipts and ExecutionOutcomes.

We hope this article is useful and you will be able to build your dApps and indexers easily with the knowledge of how data if flowing in NEAR Protocol. Edit this page Last updatedonMar 29, 2024 byVadim Volodin Was this page helpful? Yes No

Previous Lifecycle of a Transaction Next Token Transfer