[**http://hyperledger-fabric.readthedocs.io/en/release-1.0/txflow.html**](http://hyperledger-fabric.readthedocs.io/en/release-1.0/txflow.html)

[**https://www.blogsaays.com/transaction-flow-hyper-ledger-fabric/**](https://www.blogsaays.com/transaction-flow-hyper-ledger-fabric/)

[**How Transaction Flow Works in Hyper-Ledger Fabric**](https://www.blogsaays.com/transaction-flow-hyper-ledger-fabric/)

What is Transaction Flow?

A transaction flow basically outlines the mechanics that take place during a standard asset exchange between usually two or more clients.

The scenario includes two clients, A and B, who are buying and selling mobile. They each have a peer on the network through which they send their transactions and interact with the ledger.

Example blockchain1

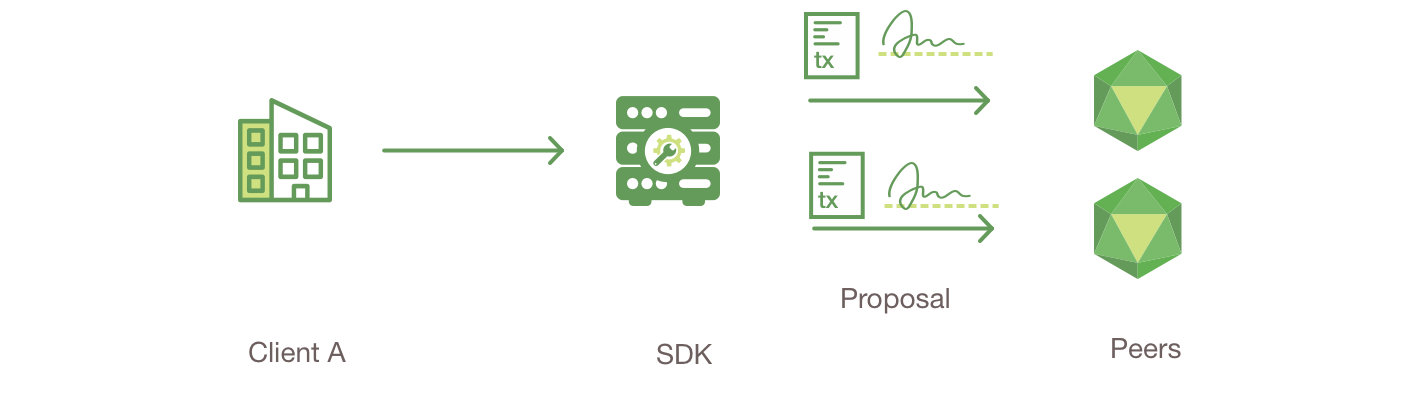
Basic assumptions

There are 4 basic or minor assumptions that are kept in mind.

1. The first assumption of the flow is that a channel is set up and running.
2. The application user has registered and enrolled with the (CA) organization’s certificate authority and received back necessary cryptographic material, which is used to authenticate to the network.
3. The chaincode is installed on the peers and instantiated on the channel.

A chaincode simply contains a set of key value pairs representing the initial state of the market (Mobile market in this case). These contain logic defining a set of transaction instructions and the agreed upon price for a mobile (ie; Product/Service).

1. An endorsement policy has also been set for this chaincode, stating that both peerA and peerB must endorse any transaction.



Client A initiates a transaction

The initiation of the trade begins with Client A sending a request to purchase [mobiles](https://www.blogsaays.com/top-reasons-buy-asus-zenphone-2-laser/).

peerA and peerB, who are respectively representative of Client A and Client B are the request targets.

The request goes to peerA and peerB since the endorsement policy states that both peers must endorse any transaction.

Transaction proposal is constructed

the proposal is a request to invoke a **chaincode function**.

The transaction proposal is then packaged into the properly architected format by this SDK which then takes the user’s [cryptographic](https://en.wikipedia.org/wiki/Cryptography) credentials to produce a unique signature for this transaction proposal.

Endorsing peers verify signature & execute the transaction

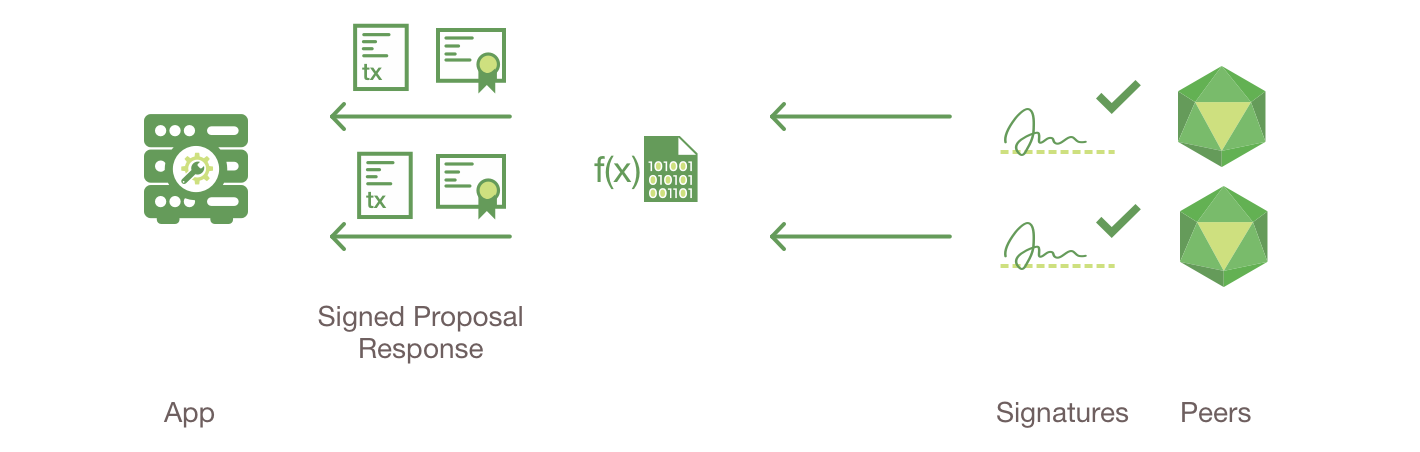
the endorsing peers verify that

* The transaction proposal is well formed.
* It has not been submitted already in the past, to satisfy the replay-attack protection.
* The signature is valid (using MSP). Membership service provider
* The submitter (Client A) is properly authorized to perform the proposed operation on that channel.

The transaction proposal inputs are taken as arguments to the invoked chaincode’s function by the endorsing peers. The chaincode is then executed to produce transaction results against the current state database. These typically include a response value, read set and write set.

It is further made sure that no updates are made to the ledger at this point.

The set of these values, along with the endorsing peer’s signature is passed back as a “proposal response” to the SDK which parses the payload for the application to consume.



NOTE -:

The MSP is a peer component that allows them to verify transaction requests arriving from clients and to sign transaction results(endorsements).

The Writing policy is defined at channel creation time, and determines which user is entitled to submit a transaction to that channel.

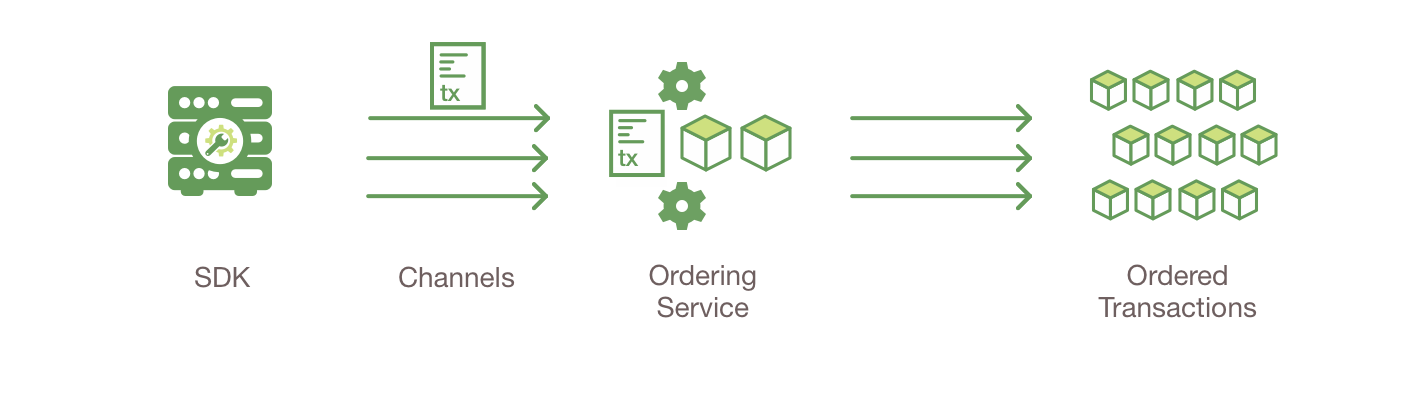
Proposal responses are inspected

The application verifies the endorsing peer signatures and compares the proposal responses to determine if the proposal responses are the same.

If the chaincode only queried the ledger, the application would inspect the query response and would typically not submit the transaction to Ordering Service.

If the client application intends to submit the transaction to Ordering Service to update the ledger, the application determines if the specified endorsement policy has been fulfilled before submitting (i.e. did peerA and peerB both endorse).

The architecture is such that even if an application chooses not to inspect responses or otherwise forwards an unendorsed transaction, the endorsement policy will still be enforced by peers and upheld at the commit validation phase.



Client assembles endorsements into a transaction

**Wipro-interview-2**

1. Managing the fund, Is it a real money or token
2. Existing network you used or created your own network
3. Are you using Ether or your own cryptocurrency?
4. If you create your own coin how the linking happens, what kind of value assigned to that coin?
5. If new person wants to participate in a campaign how do you provision it? How do I know which contract I have to call?
6. What is the consensus you have used?
7. Is it like all new comer has to do mining or only specific node?
8. I have lot of money and want to contribute, when my node become part of the network, so where do I get my coins now?
9. What your contract does and what it is all about?
10. What kind of token contract that you have used?
11. How do you move the coins?
12. What features you used in Hyperledger 1.0
13. How did you achieve privacy in Hyperledger?
14. Were there any private transactions happening?
15. Did you use channel in your project?
16. How do you compile and deploy solidity code in your network?
17. If all these tools are not available, how will you deploy the contract?
18. I have written a smart contract and I want to deploy that contract inside my network. How will you do that.

Have you heard of ABI

1. Tomorrow if you move to your own coins. How this transfer will take place? How do you achieve this using smart contract?
2. How do you create your own coin?

We don’t have coins right now, basically we are private network ether and transferring it there.