

Implementation details of Smart
Contracts in Healthcare and
how it can be used to rank
providers

Blockchain Implementation Details

Whitepaper



Contents

1. Introduction to blockchain
2. Components of blockchain
3. Smart Contracts
4. Providers in healthcare
5. Blockchain Implementation Options
6. What is Hyperledger?
7. AboutHyperledger Fabric
8. Description & Goal of use case
9. Events involved in the use case

1. Introduction to blockchain

The healthcare industry needs a revolution – and it is here now. A blockchain is a distributed tamperproof database which is shared and maintained by multiple parties keeping the records added to it safe. Each record contains a timestamp and secure links to the previous record. Records can only be added to the database, never removed, with each new record cryptographically linked to all previous records in time. When storing healthcare data in a blockchain, cryptography is used for encrypting planned users the contents of a message or transaction to ensure that the content is accessible to only. An encryption process called ‘Public Key Cryptography’ is used. A pair of public and private keys is generated. Public key is spread to all nodes in the network and the private key is only known to its holder. Either key may be used to encrypt a message, but the other key must decrypt the message.

There are multiple potential uses of blockchain technology in healthcare. They have advanced and matured to hold the promise to unite the different processes in the pharmaceutical industry and healthcare ecosystem, reduce costs, improve regulatory compliance, increase data flow, and improve patient experience and outcomes. Features of blockchain are its immutability, Common Consensus, Public/Private key cryptography, Traceability, Decentralized, Disintermediation, Confidentiality and Robustness.

2. Components of blockchain

Blockchain is a design pattern consisting of four main components:

a. Distributed Network

Blockchain is a decentralized peer-to-peer architecture with nodes consisting of network participants.

Each member in the network stores an identical copy of the blockchain and contributes to the collective process of validating and certifying digital transactions for the network.

b. Shared Ledger

This is the logical component. Members in the distributed network record digital transactions into a shared ledger. Transactions are added by members in the network who run algorithms to evaluate and verify the proposed transaction. If a majority of the members in the network agree that the transaction is valid and hence, added to the shared ledger.

c. Digital Transactions

Any type of information or digital asset can be stored in a blockchain, and the network implementing the blockchain defines the type of information contained in the transaction.

Information is encrypted and digitally signed to guarantee authenticity and accuracy.

d. Consensus Algorithm

This is also a logical component. These algorithms enable network participants to agree on the contents of the blockchain in a distributed and trustless manner. This validates information from primary sources and then includes it in the distributed ledger.

3. Smart Contracts

Smart contracts are pre-written computer code and are deployed, replicated, and then executed on a distributed network of computers (blockchain). They allow the performance of credible transactions without third parties. These transactions are trackable and irreversible. Smart contracts execute business logic and reduce costs and improve quality through automation. Attributes dealt with smart contracts are as follows:

- Parties linked with the contract
- Services provided
- Payment terms
- Responsibilities of the parties
- Business processes
- Procedures for claims submission, processing, and payment
- Reimbursement Processes
- Procedures for authorization of services and referrals
- Agreements to exchange information
- Grievance processes
- The term of the contract and the amendment and termination processes

4. Providers in healthcare


















Health care provider is a person or business who provides preventive, curative, promotional or rehabilitative health care services in a systematic way to people, families or communities. Verifying the identity and credentials of licensed providers is core to the healthcare system. A credentialing blockchain would eliminate duplicate requests for information from original sources including physicians, educational institutions, training and credentialing organizations and state licensing boards, reducing the time it takes to assemble information from weeks or months to minutes and enabling real-time updating of records. This will lessen the burden on healthcare providers, minimize risks and cost, enabling providers to focus on improving quality of care. A smart contract can be applied to healthcare, for example, by ensuring claims and reimbursements are executed between payers and providers. This prevents inconsistencies that delay payments and create friction among parties involved, further allowing credentialing and re-credentialing of providers.

5. Blockchain Implementation Options

There are 3 popular key platforms for blockchain development: Ethereum, Hyperledger Fabric, and R3 Corda.

Ethereum is an open-source, public blockchain-based distributed computing platform featuring smart contract (scripting) functionality. Similar to Bitcoins, Ethereum provides a cryptocurrency token that can be transferred between accounts and used to compensate participant nodes for computations performed.

Hyperledger Fabric is a permissioned blockchain infrastructure, originally created by IBM. It provides a modular architecture with a delineation of roles between the nodes in the infrastructure, execution of Smart Contracts (here, chaincode), configurable consensus and membership services. Hyperledger offers other technology stacks, but their Fabric stack is most suited to healthcare industry's current blockchain priorities. If the stack lacks the greatest breadth of features, there's no point to looking deeper. I eliminated those that are specific to an industry and read through the marketing material, white papers and developer documentation to arrive at the answers. Missing features and stacks using proof of work are penalized. Pluggable validation methods (one of them-Byzantine Fault Tolerant) methods are favored. The lack of multi-tenancy is not penalized but support is rewarded in the scoring.

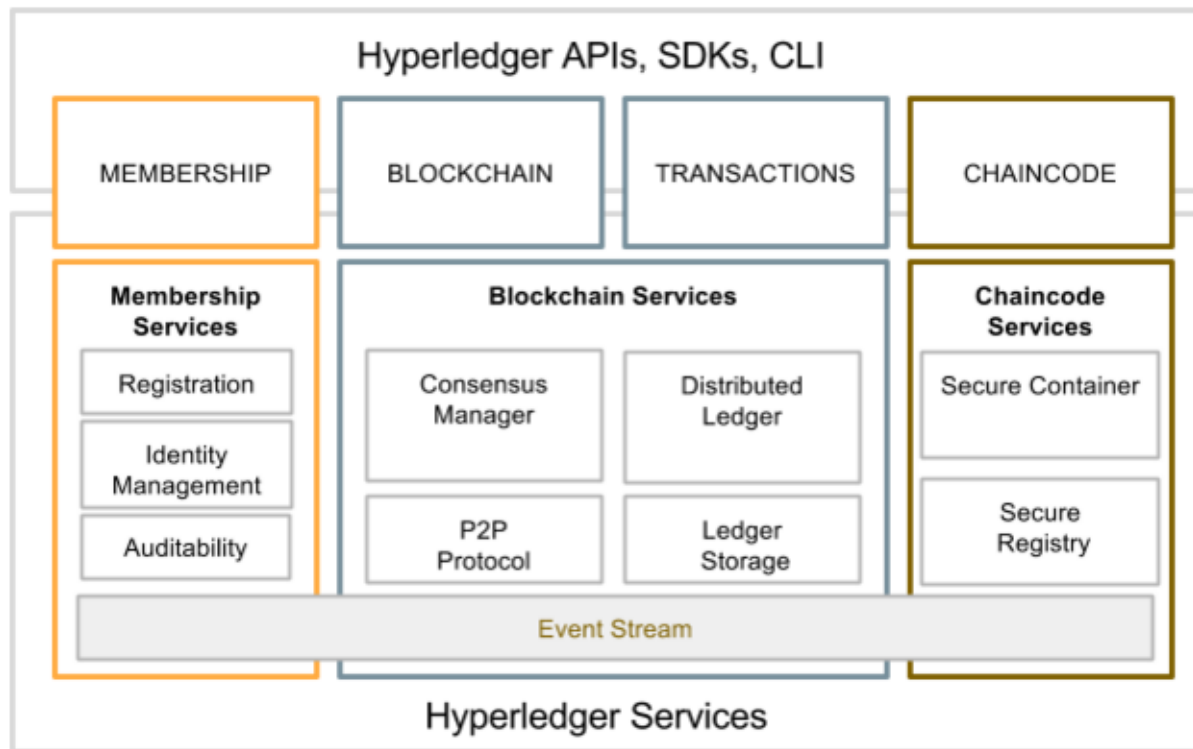
Candidate	Feature Score	Consensus Method	Business Logic?	Independent of Virtual Currency?	Key Management?	Supports Private?	Data Permissions?	User Management?	Multi-tenanted?
Blockstack 	20	Unknown	Unknown	No	Yes	Unknown	Yes	Yes	Unknown
Bloq 	0	Unknown	Unknown	No	Unknown	Yes	Unknown	Unknown	Unknown
Dokchain 	0	Unknown	Unknown	Unknown	Unknown	Unknown	Unknown	Unknown	Unknown
Ethereum 	-70	PoW - Proof of Work	Yes: Smart Contracts	No	No	Yes	No	No	Unknown
Gem 	70	Other	Yes: Smart Contracts	Yes	Yes	Yes	Yes	Yes	Unknown
Hyper Ledger - Fabric 	120	Pluggable	Yes: Smart Contracts	Yes	Yes	Yes	Yes	Yes	Yes
Hyper Ledger - Iroha 	5	BFT - Byzantine Fault Tolerant	No	Yes	No	Yes	No	No	Unknown
Hyper Ledger - Sawtooth Lake 	40	Other	Yes: Smart Contracts	Yes	Unknown	Yes	Unknown	Unknown	Unknown
Kadena 	55	BFT - Byzantine Fault Tolerant	Yes: Smart Contracts	Yes	Unknown	Yes	Unknown	Unknown	Unknown
Lisk 	20	Unknown	Unknown	No	Yes	Unknown	Unknown	Yes	Yes
Monax 	40	Unknown	Yes: Smart Contracts	Yes	Yes	Yes	Unknown	Unknown	Unknown
MultiChain.com 	105	Pluggable	Yes: Other Approach	Yes	Yes	Yes	Yes	Yes	Unknown
NEM 	30	Pol - Proof of Importance	Yes: Smart Contracts	No	Unknown	Yes	Unknown	Yes	Yes
Openchain 	40	Other	Yes: Smart Contracts	Yes	No	Yes	Yes	No	Yes
Quorum 	85	BFT - Byzantine Fault Tolerant	Yes: Smart Contracts	Yes	Yes	Yes	Yes	Unknown	Yes
Stratis 	-30	PoS - Proof of Stake	No	Unknown	No	Yes	No	No	Unknown
STRATO 	-70	PoW - Proof of Work	Yes: Smart Contracts	No	No	Yes	No	No	Unknown

Candidate	Overall Score	Feature Score	Suitability Score
Blockstack	70	20	50
Bloq	25	0	25
Dokchain	-75	0	-75
Ethereum	-20	-70	50
Gem	-5	70	-75
Hyper Ledger - Fabric	245	120	125
Hyper Ledger - Iroha	80	5	75
Hyper Ledger - Sawtooth Lake	115	40	75
Kadena	105	55	50
Lisk	145	20	125
Monax	15	40	-25
MultiChain.com	80	105	-25
NEM	105	30	75
Openchain	115	40	75
Quorum	110	85	25
Stratis	-5	-30	25
STRATO	-45	-70	25

Looking at the aspects that contribute to the risk of using the tech stack, the above table looks at maturity , license for our use and risk of depending on the 'owner' of the tech stack. These factors are combined into a Suitability Score and in combination with the Feature Score, an Overall Score is generated. Hyperledger Fabric appears to be best of breed as of this posting.

6. What is Hyperledger?

Hyperledger is an umbrella project of open source blockchain and related tools, started by the Linux Foundation, to support the collaborative development of blockchain-based distributed ledgers. With it, the Linux Foundation aims to create an environment in which communities of software developer and companies meet and coordinate to build blockchain frameworks. The Linux Foundation founded the platform in December 2015. In February 2016 it announced the first founding members, in March 2016 ten more members joined. Today Hyperledger has an impressive list of more than 100 members.



The Hyperledger Architecture has distinguished the following business blockchain components:

- **Consensus Layer** - Responsible for generating an agreement on the order and confirming the correctness of the set of transactions that constitute a block.
- **Smart Contract (Chaincode) Layer** - Responsible for processing transaction requests and determining if transactions are valid by executing business logic.
- **Communication Layer** - Responsible for peer-to-peer message transport between the nodes that participate in a shared ledger instance.
- **Data Store Abstraction** - Allows different data-stores to be used by other modules.
- **Crypto Abstraction** - Allows different crypto algorithms or modules to be swapped out without affecting other modules.
- **Identity Services** - Enables the establishment of a root of trust during setup of a blockchain instance, the enrollment and registration of identities or system entities during network operation, ensuring authentication and authorization.
- **Policy Services** - Responsible for policy management of various policies specified in the system, such as the endorsement policy, consensus policy, or group management policy. It interfaces and depends on other modules to enforce the various policies.
- **APIs** - Enables clients and applications to interface to blockchain.
- **Interoperation** - Supports the interoperation between different blockchain instances.

The ‘umbrella strategy’ of Hyperledger incubates and promotes a range of business blockchain technologies; framework, libraries, interfaces, and applications. Currently, Hyperledger is the host of the following projects: Hyperledger Sawtooth, Hyperledger Iroha, Hyperledger Burrow, Hyperledger Fabric, etc. Beside these framework projects, Hyperledger has several tool projects

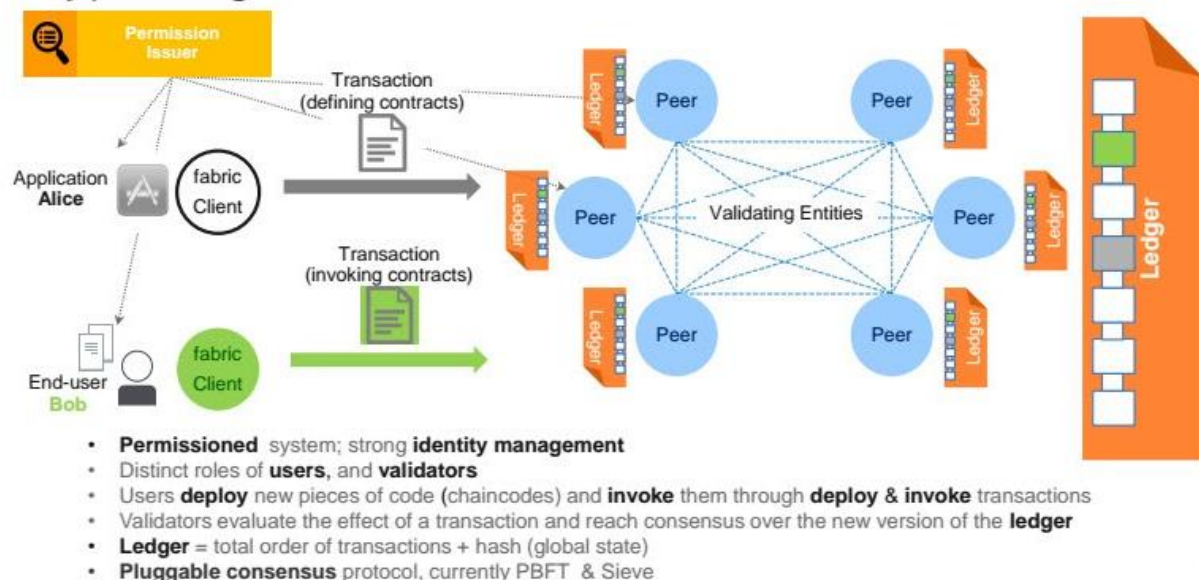
aiming to make the access to and development of blockchain easier and more effective like Cello, Explorer and Indy.

7. About Hyperledger Fabric

With Fabric different components of Blockchain, like consensus and membership services can become plug-and-play. Fabric is designed to provide a framework with which enterprises can put together their own, individual blockchain network that can quickly scale to more than 1,000 transactions per second. Its components are:

- Asset (entity that has some value – patient's medical record)
- Ledger (append-only transaction log maintained by peers, peer ledger and validated ledger)
- Participants (Patients, providers, payers, government, researchers, labs, pharmacies, etc.)
- Chaincode (smart contract, can be implemented in GO or Java)
- Certification Authority (responsible for gatekeeping, performs membership services)
- Channels (One or more channel exists in a blockchain network and is actually a private subnet of communication between two or more specific network members)
- Consensus (general agreement to confirm the correctness of the set of transactions of the block)

Hyperledger-fabric model



Fabric heavily relies on a smart contract system called Chaincode, which every peer of the networks runs in Docker containers.

Consensus in Hyperledger Fabric is broken out into 3 phases: Endorsement, Ordering, and Validation.

- Endorsement is driven by policy (e.g. m out of n signatures) upon which participants endorse a transaction.
- Ordering phase accepts the endorsed transactions and agrees to the order to be committed to the ledger.
- Validation takes a block of ordered transactions and validates the correctness of the results, including checking endorsement policy and double-spending.

This separation confers several advantages: fewer levels of trust and verification are required across node types, and network scalability and performance are optimized. The transaction processing can be precisely put into steps as:

- 1) The transaction proposal is submitted by an application to an endorsing peer.
- 2) The Endorsement policies outline how many and/or what combination of endorsers are required to sign a proposal. The endorser executes the chaincode to simulate the proposal in the network peer, creating a read/write set.
- 3) Then the endorsing peers send back the signed proposal responses (endorsements) to the application.
- 4) The application submits the transactions and signatures to the ordering service, which
- 5) creates a batch, or block, of transactions and delivers them to committing peers.
- 6) When a committing peer receives a batch of transactions, for each transaction it
- 7) validates that the endorsement policy was met or not and checks in the read/write sets to detect conflicting transactions. If both checks pass, the block is committed to the ledger, and the state updates for each transaction are reflected in the state database.

It supports pluggable consensus i.e. we can use different algorithms to confirm our transaction that enables the platform to be more effectively customized to fit particular use cases and trust models. The model file (consisting of assets, participants, transactions, event classes as .cto), transaction logic (.js), permission access control (.acl) and query files (.qry) are all created and packaged into a Business Network Archive (.bna). This is done by Hyperledger Composer and then deployed on Fabric.Chaincode is installed on a peer's file system and then instantiated on a channel. All channel members need to install the chaincode on every peer that will run this chaincode. To use the same chaincode, channel members must give the chaincode the same name and version when they install the chaincode. Then, the instantiation step involves the initialization of key value pairs and the deployment of the chaincode container. Any peer, via which applications will interact with a chaincode, must have the chaincode that is installed on the peer's file system and have a running chaincode container. However, if a peer uses the same chaincode on multiple channels, it needs only a single instance of the chaincode container. The installation and instantiation combination is a powerful feature because it allows for a peer to interact with the same chaincode container across multiple channels. The only prerequisite is for the actual chaincode source files to be installed on the peer's file system. As such, if a piece of common chaincode is being used across dozens of channels, a peer would need only a single chaincode container to perform read/writes on all the channel ledgers. This lightweight approach proves beneficial to computational performance and throughput as networks scale and chaincode become more elaborate.

8. Description & Goal of Use Case

This use case describes interoperability between electronic health records and laboratory systems from the perspective of patients, physicians, pharmacies, hospital, and laboratory results. And then, using the input depending upon factors like number of revisits by consumers and cost charged, providers will be rated and ranked so as to help the consumers choose according to their requirements. Its focus is on widely-available, well-standardized methods that will support the secure access to laboratory results and interpretations in a patient-centric manner for clinical care by authorized parties. The overall goal of this use case is to allow a patient utilize the services provided by physician or pharmacies from one healthcare enterprise and they may also obtain hospital/laboratory data from same/different healthcare enterprise for the purpose of the clinical care of a Patient. This overall goal is achieved by satisfying a number of sub-goals.

9. Events involved in the use case

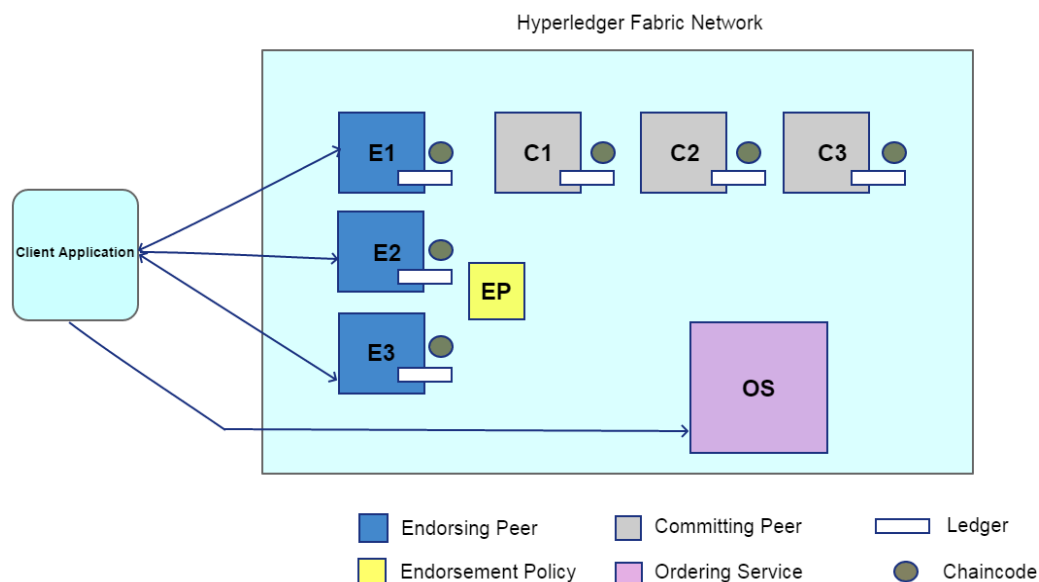
Use Case	Parties involved	Data involved
Patient Visits Physician and initiates Preauthorization	Patient , Physician , Payer	Patient Name, Patient SSN, Provider Name, Provider Tin, Payer details
Consultation happens and lab tests are suggested	Patient , Physician	Patient Name, Patient SSN, Provider Name, Provider Tin
Patient takes Lab tests	Patient , Labs/Hospitals	Patient Name, Patient SSN, Provider Name, Provider Tin
Lab records are taken to Physician	Hospital, Physician	Provider Name, Provider Tin
Physician provides prescription based on lab results	Physician , Patient	Patient Name, Patient SSN, Provider Name, Provider Tin
Patients gets the medicine from Pharmacy	Patient , Pharmacy	Patient Name, Patient SSN, Provider Name, Provider Tin

Case closed until there are related revisits	Patient	Patient Name, Patient SSN
Rank the providers on the basis of no. of revisits and cost	Patient , Physician , Pharmacy , Labs/Hospitals	Patient Name, Patient SSN, Provider Name, Provider Tin

-What these all mean in Hyperledger Fabric terminology: Assets, Participants, Chaincode, Channel, Security and Membership service, Consensus.

-Technology to implement :

- Locally (OS- Ubuntu/iOS) or on Cloud
- Assets, Participants, Transactions put into Model file (.cto)
- Transaction functions put into Script file (.js)
- Access Control Rules put into Access Control file (.acl)
- Query Definitions put into Query file (.qry)
- The package of .cto, .js, .acl and .qry together generate archive of business network definition called business network achieve (.bna)
- Chaincode written in the language Go /Java
- Database- LevelDB/CouchDB
- API- application's API (maybe JavaScript)



- In the above events, Hyperledger Fabric will collect patient's submissions via client application and check if the physician/pharmacy/lab/hospital/patient/pharmacy have access to the each other's record by reading the Asset permission array and verifying with the certificate authority. The Ledger chaincode will execute the correct function, emit logs and return the requested values through the application's API (like JavaScript).
- If the permission check result is affirmative, record will be requested from the database using the record's unique global identifier (SSN), decrypt it with the encryption key received from the Hyperledger and send it to the requestor for viewing.
- Giving access to a record is achieved by giving access to the encryption key for that record.
- If the event needs to be kept private to only few participants, we can use channels to ensure that the chaincode revolves among those selected nodes and the endorsing and validation can only be done among those selected participants.

Client A initiates a transaction via an application	Fabric-client module provides APIs for the applications to interact with the core components of a Hyperledger Fabric-based blockchain network, namely the peers, orderers and event streams. These client applications which might be behind a corporate firewall, may allow access of an initiation to only a few peers w.r.t channel restriction. Identity-based network security makes it secure enough for companies to use in exponentially more applications.
The transaction proposal is constructed and submitted to endorsing peers	The client application leveraging a supported SDK (Nodejs, Java, Python) utilizes one of the available API's which generates a transaction proposal Here, a security protocol: Secure Sockets Layer(SSL) is embedded on the peer's browsers which engage automatically for exchanging sensitive information.
Endorsing peers verify signatures & send back the signed proposal responses/transaction results to the application	The endorsing peers will capture the set of Read and Written data, called RW Sets. Chaincodes are executed on these RW sets are then signed by the endorsing peer via Membership Service and then sent back to the client using fabric-client module and SSL layer enabled.
Application assembles endorsements into a transaction and broadcasts the transaction message to ordering service	For transaction with a valid endorsement, we now start using the ordering service. The submitting client invokes ordering service by broadcasting the endorsements via SSL. If the client does not have capability of

	invoking ordering service directly, it may proxy its broadcast through some peer of its choice. Such a peer must be trusted by the client not to remove any message from the endorsement or otherwise the transaction may be deemed invalid.
Ordering service creates the block of transaction for each channel and sends them to all peer	The ordering service, which is made up of a cluster of orderers accepts the endorsed transactions and specifies the order in which those transactions will be committed to the ledger, followed by specific implementation of 'ordering' by choose the ordering mechanism that best suits that network.
Transaction is validated and committed to blockchain	The committing peer validates the transaction by checking to make sure that the RW sets still match the current world state and the blocks of transactions are added to the shared ledger and state database is updated by the committing peers.
Ledger updated	Lastly, the committing peers asynchronously notify the client application of the success or failure of the transaction. Applications will be notified by each committing peer using the API's (provided by fabric-client module)

Provider Ranking-

To rank the providers on the scale of 1 to 5 on the basis of various parameters, we need to rate the providers and hence, we need to know that on what basis the rating will be done. Several quality measures (out of numerous) will be selected, like number of revisits, cost, overall success rate, waiting time for the consumer, already available CMS ratings, etc. A provider summary score can be calculated by taking the average weighted consideration of these scores calculated via standard deviation. These can be calculated without the consumer efforts. Like, in case of number of revisits, it can be calculated by analyzing each consumer's record for all providers separately, by counting the transactions which relate to services offered to that consumer (patient).

We here have physicians, pharmacies, labs and hospitals as providers. But we consider the providers separately for ranking like first, if we focus on the Physicians and the number of revisits by a patient, we can easily calculate the standard deviation. This standard deviation for each Physician can be given an integral value by fixing values from 1 to 5 to a range of standard deviation values. Similarly, all these Physicians will be rated by using details of other patients as well. And hence, an average weighted score can be measured for each Physician with respect to all patients. Comparison becomes very suitable as what we are left with is only a single value for each hospital under this measure category (parameters). Same is followed for other Providers as well and hence, they can be ranked comfortably.