BLOCKCHAIN HYPERLEDGER FABRIC

- Hyperledger: Hyperledger Projects. Hyperledger as a protocol.
- Hyperledger Fabric : Hyperledger Fabric, Fabric architecture, Components of the Fabric. (Text book -1)
- NFT and DLT: Non-Fungible Tokens ,Four Types of NFTs ,DLT: The Beating Heart of Blockchain. (Text book -3)

Hyperledger

- Hyperledger is not a blockchain, but it is a project that was initiated by Linux foundation in December 2015 to advance blockchain technology. This project is a collaborative effort by its members to build an open source distributed ledger framework that can be used to develop and implement cross industry blockchain applications and systems.
- The key focus is to build and run platforms that support global business transactions.
- The project also focuses on improving the reliability and performance of blockchain systems.
- Projects under Hyperledger undergo various stages of development, starting from proposal to incubation and graduating to an active state. Projects can also be deprecated or in End of Life state where they are no longer actively developed. In order for a project to be able to move into incubation stage, it must have a fully working code base along with an active community of developers

Projects

- Currently, there are six projects under the Hyperledger umbrella: Fabric, Iroha, Sawtooth lake, blockchain explorer, Fabric chain tool, and Fabric SDK Py. Corda is the most recent addition that is expected to be added to the Hyperledger project.
- The Hyperledger project currently has 100 members and is very active with more than 120 contributors, with regular meet-ups and talks being organized around the globe.
- A brief introduction of all these projects follows, after which we will provide more details around the design, architecture, and implementation of Fabric and Sawtooth lake

Fabric

- Fabric is a blockchain project that was proposed by IBM and DAH (Digital Asset Holdings).
- This intended to provide a foundation for the development of blockchain solutions and is based on pluggable architecture where various components, such as consensus algorithm, can be plugged into the system as required.
- It is available at https://github.com/hyperledger/fabric

Sawtooth lake

- Sawtooth lake is a blockchain project proposed by Intel in April 2016 with some key innovations focusing on decoupling of ledgers from transactions, flexible usage across multiple business areas using transaction families, and pluggable consensus.
- Decoupling can be explained more precisely by saying that the transactions are decoupled from the consensus layer by making use of a new concept called Transaction families. Instead of transactions being individually coupled with the ledger, transaction families are used, which allows for more flexibility, rich semantics and unrestricted design of business logic.
- Transactions follow the patterns and structures defined in the transaction families. Intel has also introduced a novel consensus algorithm abbreviated as PoET, proof of elapsed time, which makes use of Intel Software Guard Extensions (Intel's SGX) architecture's trusted execution environment (TEE) in order to provide a safe and random leader election process. It also supports permissioned and permission-less setups. This project is available at https://github.com/hyperledger/sawtooth-core

Iroha

- Iroha was proposed by Soramitsu, Hitachi, NTT Data, and Colu in September 2016.
- Iroha is aiming to build a library of reusable components that users can choose to run on their Hyperledger-based distributed ledgers.
- Iroha's main goal is to complement other Hyperledger projects by providing reusable components written in C++ with an emphasis on mobile development.
- This project has also proposed a novel consensus algorithm called Sumeragi, which is a chain based Byzantine fault tolerant consensus algorithm.
- Iroha is available at https://github.com/hyperledger/iroha
- Various libraries have been proposed and are being worked on by Iroha, including but not limited to a digital signature library (ed25519), an SHA-3 hashing library, a transaction serialization library, a P2P library, an API server library, an iOS library, an Android library, and a JavaScript.

Blockchain explorer

- This project aims to build a blockchain explorer for Hyperledger that can be used to view and query the transactions, blocks, and associated data from the blockchain.
- It also provides network information and the ability to interact with chaincode (Smart contract).
- https://hyperledger-fabric.readthedocs.io/en/release-1.4/chaincode4ade.html
- Currently there are two other projects that are in incubation: Fabric chaintool, and Fabric SDK Py. These projects are aimed at supporting Hyperledger Fabric.

Fabric chaintool

- Hyperledger chaincode compiler is being developed to support Fabric chaincode development.
- The aim is to build a tool that reads in a high-level Google protocol buffer structure and produces a chaincode.
- Additionally, it packages the chaincode so that it can be deployed directly.
- This tool will help developers in various stages of development, such as compiling, testing, packaging, and deployment.
- It is available at https://github.com/hyperledger/fabric-chaintool.

Fabric SDK Py

- The aim of this project is to build a python based SDK library that can be used to interact with the blockchain (Fabric).
- It is available at https://github.com/hyperledger/fabric-sdk-py.

Corda

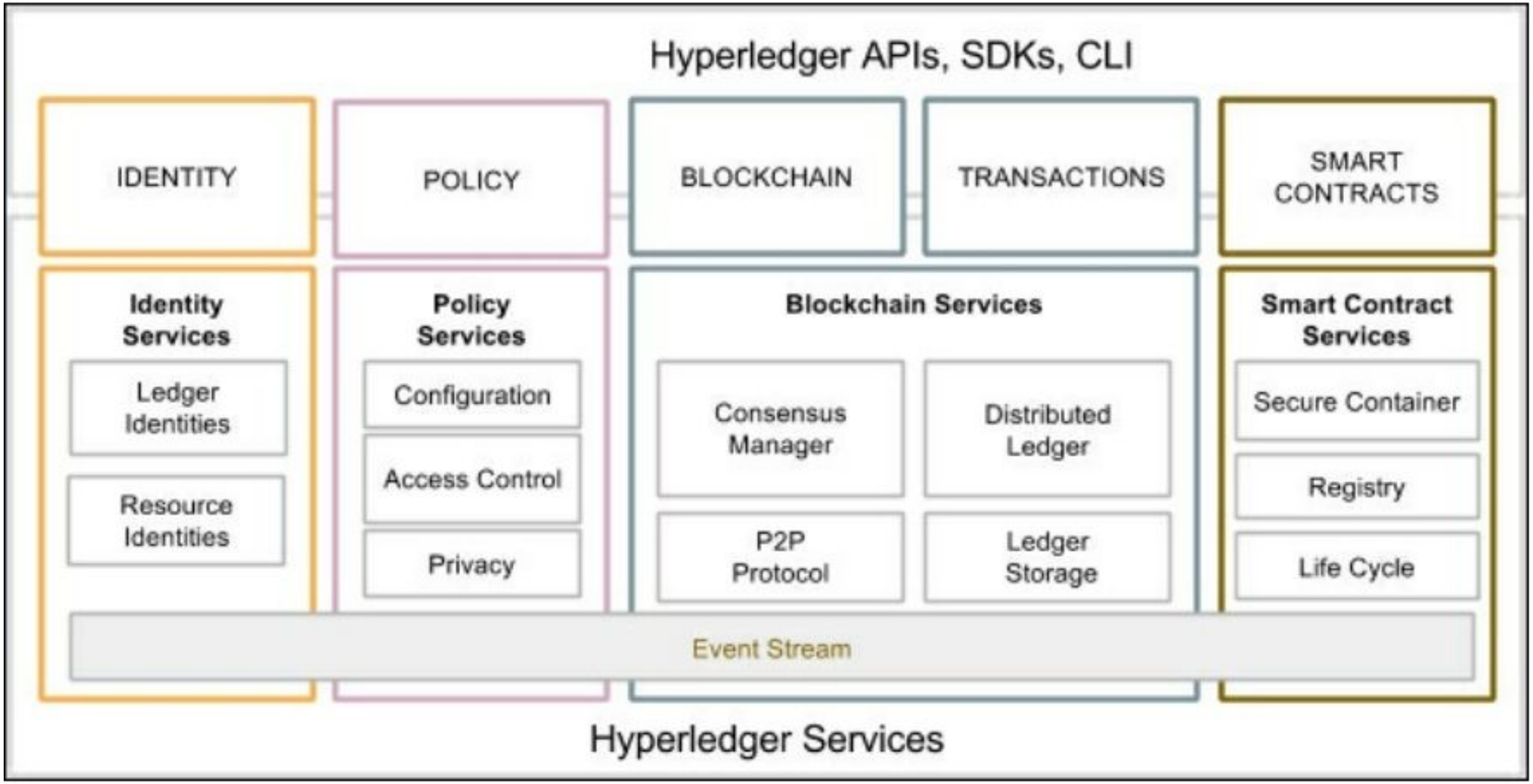
- Corda is the latest project that has been contributed by R3 to the Hyperledger project. It was open sourced on November 30, 2016.
- Corda is heavily oriented towards the financial services industry and has been developed in collaboration with major banks and organizations in the financial industry.
- At the time of writing it is not yet in incubation under the Hyperledger project. Technically, Corda is not a blockchain but has key features similar to those of a blockchain, such as consensus, validity, uniqueness, immutability, and authentication.

Hyperledger as a protocol

- Hyperledger is aiming to build a new blockchain platform that is driven by industry use cases.
- As there have been number of contributions made to the Hyperledger project by the community, Hyperledger blockchain platform is evolving into a protocol for business transactions.
- Hyperledger is also evolving into a specification that can be used as a reference to build blockchain platforms as compared to earlier blockchain solutions that address only a specific type of industry or requirement.
- In the following section, a reference architecture is presented that has been published by the Hyperledger project. As this work is under continuous and rigorous development some changes are expected in this, but core services are expected to remain unchanged

Reference architecture

- Hyperledger has published a white paper with reference architecture that can serve as a guideline to build permissioned distributed ledgers.
- The reference architecture consists of two main components: Hyperledger services and Hyperledger APIs, SDKs, and CLI(Command Line Interface).
- Hyperledger services provide various services such as **identity services, policy services, blockchain services, and smart contract services**.
- On the other hand, Hyperledger APIs, SDKs, and CLIs provide an interface into blockchain services via appropriate application programming interfaces, software development kits, or command line interfaces.
- Moreover, an event stream, which is basically a gRPC channel, runs across all services.
- It can receive and send events. Events are either pre-defined or custom.
- Validating peers or chaincode can emit events to which external application can respond or listen to.


Hyperledger APIs, SDKs, CLI
IDENTITY
POLICY
BLOCKCHAIN
TRANSACTIONS
SMART CONTRACTS
Identity Services
Ledger Identities
Resource Identities
Policy Services
Configuration
Access Control
Privacy
Blockchain Services
Consensus Manager
Distributed Ledger
P2P Protocol
Ledger Storage
Smart Contract Services
Secure Container
Registry
Life Cycle
Event Stream
Hyperledger Services

Requirements

There are certain requirements of a blockchain service. The reference architecture is driven by the needs and requirements raised by the participants of the Hyperledger project and after studying the industry use cases. There are several categories of requirements that have been deduced from the study of industrial use cases and are discussed in the following sections:-

- Modular approach
- Privacy and confidentiality
- Identity
- Auditability
- Interoperability
- Portability

Modular approach

- The main requirement of Hyperledger is a modular structure. It is expected that, as a cross-industry fabric (blockchain), it will be used in many business scenarios. As such, functions related to storage, policy, chaincode, access control, consensus and many other blockchain services should be pluggable.
- The modules should be plug and play and users should be able to easily remove and add a different module that meets the requirements of the business.
- For example, if a business blockchain needs to be run only between already trusted parties and performs very basic business operations, then perhaps there is no need to have advanced cryptographic support for confidentiality and privacy, and therefore users should be able to remove that functionality (module) or replace that with a more appropriate module that suits their needs.
- Similarly, if users need to run a cross-industry blockchain, then confidentiality and privacy can be of paramount importance.
- In this case, users should be able to plug an advanced cryptographic and access control mechanism (module) into the blockchain (fabric).

Privacy and confidentiality

- Privacy and confidentiality of transactions and contracts is of utmost importance in a business blockchain.
- Hyperledger's vision is to provide a wide range of cryptographic protocols and algorithms and it is expected that users will be able to choose appropriate modules according to their business requirements.
- The fabric should be able to handle complex cryptographic algorithms without compromising performance.

Identity and Auditability

- In order to provide privacy and confidentiality services, a flexible PKI model that can be used to handle the access control functionality is also required.
- The strength and type of cryptographic mechanisms is also expected to vary according to the needs and requirements of the users.
- In certain scenarios it might be required for a user to hide their identity, and as such the Hyperledger is expected to provide this functionality.
- Auditability is another requirement of a Hyperledger Fabric. It is expected that an immutable audit trail of all identities, related operations and any changes is kept.

Interoperability

- Currently there are many blockchain solutions available, but they cannot communicate with each other and this can be a limiting factor in the growth of a blockchain based global business ecosystem.
- It is envisaged that many blockchain networks will operate in the business world for specific needs, but it is important that they are able to communicate with each other.
- There should be a common set of standards that all blockchains can follow in order to allow communication between different ledgers.
- It is expected that a protocol will be developed that will allow the exchange of information between many Fabrics.

Portability

- The portability requirement is concerned with the ability to run across multiple platforms and environments without the need to change anything at code level.
- Hyperledger is envisaged to be portable, not only at infrastructure level but also at code, libraries, and API levels so that it can support uniform development across various implementations of Hyperledger.

Fabric

- Fabric can be defined as a collection of components providing a foundation layer that can be used to deliver a blockchain network. There are various types and capabilities of a fabric network, but all fabrics share common attributes such as immutability and are consensus driven.
- Some fabrics can provide modular approach towards building blockchain networks. In this case the blockchain network can have multiple pluggable modules to perform various function on the network.
- For example, consensus algorithms can be a pluggable module in a blockchain network where, depending on the requirements of the network, an appropriate consensus algorithm can be chosen and plugged into the network. The modules can be based on some particular specification of the fabric and can include APIs, access control, and various other components.

- Fabrics can also be designed either to be private or public and can allow the creation of multiple business networks. As an example, bitcoin is an application that runs on top of its fabric (blockchain network).
- blockchain can either be permissioned or permission-less and the same is **true** for fabric in Hyperledger terminology.
- Fabric is also the name given to the code contribution made by IBM to the Hyperledger foundation and is formally called Hyperledger Fabric. IBM also offers blockchain as a service (IBM Blockchain) via its Bluemix cloud service.

Hyperledger Fabric

- Fabric is the contribution originally made by IBM to the Hyperledger project. The aim of this contribution is to enable a modular, open and flexible approach towards building blockchain networks.
- Various functions in the fabric are pluggable, and it also allows use of any language to develop smart contracts. This is possible because it is based on container technology which can host any language.
- Chaincode (smart contract) is sandboxed into a secure container which includes a secure operating system, chaincode language, runtime environment and SDKs for Go, Java, and Node.js. Other languages can be supported too if required.
- Smart contracts are called chaincode in the Fabric. This is a very powerful feature compared to domain specific languages in Ethereum, or the very limited scripted language in bitcoin.

- It is a permissioned network that aims to address issues such as scalability, privacy, and confidentiality.
- The key idea behind this is modular technology, which would allow for flexibility in design and implementation.
- Transactions in fabric are private, confidential and anonymous for general users, but they can still be traced and linked to the users by authorized auditors.
- As a permissioned network, all participants are required to be registered with the membership services in order to access the blockchain network.
- This ledger also provided auditability functionality in order to meet the regulatory and compliance needs.

Fabric architecture

- The Fabric is logically organized into **three main categories** based on the type of service provided. These include **membership services, blockchain services, and chaincode services**. In the following section, all these categories and associated components are discussed in detail.
- The current stable version of Hyperledger Fabric is v2.5,
- **Membership services :**These services are used to provide access control capability for the users of the fabric network. The following list shows the functions that membership services perform:

1. User identity validation.
2. User registration.
3. Assign appropriate permissions to the users depending on their roles.

Membership services makes use of **Public Key Infrastructure (PKI)** in order to support identity management and authorization operations. Membership services are made up of various components:

- Registration authority (RA): A service that **authenticates the users and assesses the identity** of the fabric participants for issuance of certificates.
- Enrolment certificate authority: **Enrolment certificates (Ecerts) are long term certificates issued by ECA to registered participants** in order to provide identification to the entities participating on the network.
- Transaction certificate authority:In order ***to send transactions on the networks***, participants are required to hold a transaction certificate. TCA is responsible for issuing transaction certificates to holders of Enrolment certificates and is derived from Ecerts.
- TLS certificate authority:In order **to secure the network level communication between nodes** on the Fabric, TLS certificates are used. TLS certificate authority issues TLS certificates in order to ensure security of the messages being passed between various systems on the blockchain network.

Blockchain services

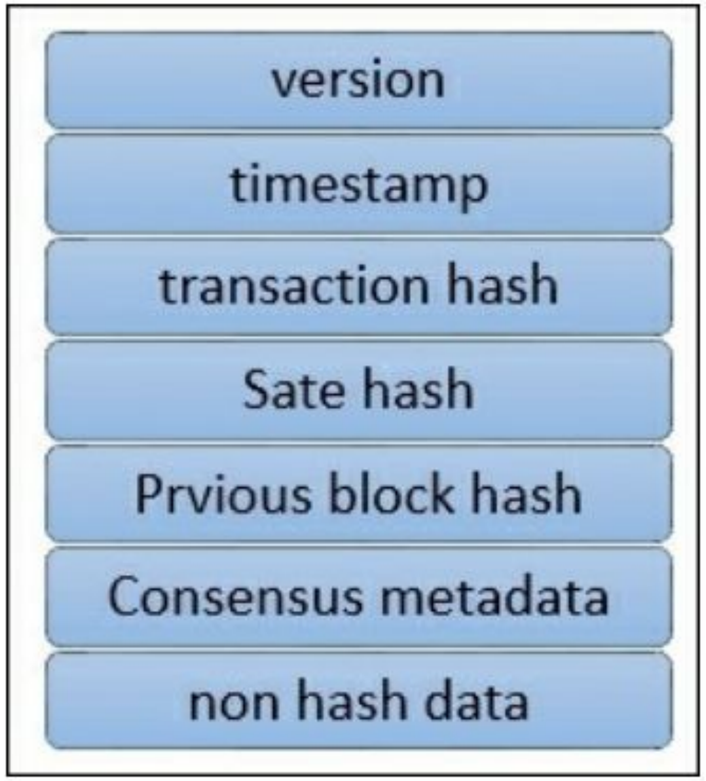
- Blockchain services are at the core of the Hyperledger Fabric. Components within this category are as follows.
- Consensus manager Consensus manager is responsible **for providing the interface to the consensus algorithm**.
- This serves as an adapter that **receives the transaction from other Hyperledger entities and executes them under criteria according to the type of algorithm chosen.**
- Consensus is pluggable and currently there are three types of consensus algorithm in Fabric,
- batch PBFT protocol

  SIEVE algorithm

  NOOPS.

Distributed ledger

- Blockchain and world state are two main elements of the distributed ledger.
- Blockchain is simply a linked list of blocks and world state is a key-value database.
- This database is used by smart contracts to store relevant states during execution by the transactions.
- The blockchain consists of blocks that contain transactions.
- These transactions contain chaincode, which runs transactions that can result in updating the world state.
- Each node saves the world state on disk in RocksDB. The diagram shows a typical block in the Hyperledger Fabric with the relevant fields



Block structure

The fields shown in the preceding diagram are as follows:

- **Version:** Used for keeping track of changes in the protocol.
- **Timestamp**: Timestamp in UTC epoch time, updated by block proposer.
- **Transaction hash:** This field contains the Merkle root hash of the transactions in the block.
- **State hash:** This is the Merkle root hash of the world state.
- **Previous hash:** This is the previous block's hash, which is calculated by applying the SHA3 SHAKE256 algorithm.
- **Consensus metadata:** This is an optional field that can be used by the consensus protocol to provide some relevant information about the consensus.
- **Non-Hash data:** This is some metadata that is stored with the block but is not hashed. This feature makes it possible to have different data on different peers. It also provides the ability to discard data without any impact on the blockchain.

Peer to Peer protocol

- P2P protocol in the Hyperledger Fabric is built using **google RPC** (gRPC).
- It **uses protocol buffers** to define the structure of the messages.
- Messages are passed between nodes in order to perform various functions.
- There are four main types of messages in Hyperledger Fabric:
- Discovery, transaction, synchronization and consensus.
- Discovery messages are exchanged between nodes when starting up in order to discover other peers on the network.

Transaction messages can be divided into two types:

- Deployment transactions
- Invocation transactions.

- Deployment transactions used to deploy new chaincode to the ledger,
- Invocation transactions is used to call functions from the smart contract.
- Transactions can be **public, confidential, and confidential chaincode transactions**.
- **Public** transactions are open and available to all participants.
- **Confidential transactions** are allowed to be queried only by transaction owners and participants.
- **Confidential chaincode transactions** have encrypted chaincode and can only be decrypted by validating nodes.
- **Validating nodes** run consensus, validate the transactions and maintain the blockchain.
- **Non-validating nodes** provide transaction verification, stream server, and REST services. They also act as a proxy between the transactors and the validating nodes.
- Synchronization messages are used by peers to keep the blockchain updated and in synch with other nodes.
- Consensus messages are used in consensus management and broadcasting payloads to validating peers. These are generated internally by the consensus framework.

Chaincode services

- In order to save the state of the ledger, RocksDB is used, and it is stored at each peer.
- RocksDB is a high performance database available at http://rocksdb.org/.

These services allow the creation of secure containers that are used to execute the chaincode. Components in this category are as follows:

- **Secure container**: Chaincode is deployed in Docker containers that provide sandboxed environment for smart contract execution. Currently Golang is supported as the main smart contract language, but any other main stream language can be added and enabled if required.
- **Secure registry**: This provides a record of all images containing smart contracts.

Events and APIs and CLIs

- Events on the blockchain can be triggered by **validator nodes** and **smart contracts.**
- External applications can listen to these events and react to them if required via event adapters.
- API provides an interface into the fabric by exposing various REST APIs.
- command line interfaces that provide a subset of REST APIs and allow for quick testing and limited interaction with the blockchain are also available.

Components of the Fabric

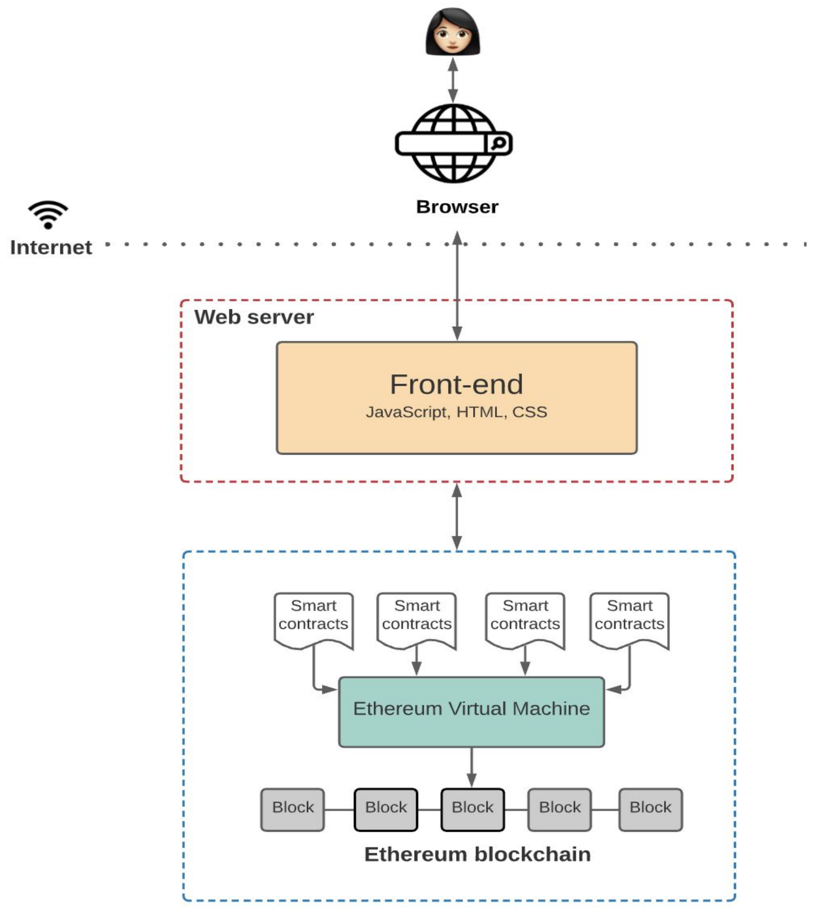
There are various components that can be part of the blockchain. These components include but are not limited to **the ledger, chaincode, consensus mechanism, access control, events, system monitoring and management, wallets and system integration components**.

Peers or nodes :

There are ***two main types of peers*** that can be run on a fabric network: **Validating and non-validating.**

- A validating node runs consensus, creates and validates a transaction, and contributes towards updating the ledger and maintaining the chaincode.
- A non-validating peer does not execute transactions and only constructs transactions that are then forwarded to validating nodes.
- Both nodes manage and maintain user certificates that have been issued by membership services.

Applications on blockchain



- A typical application on Fabric is simply composed of a user interface, usually written in JavaScript/HTML, that interacts with the backend chaincode (smart contract) stored on the ledger via an API layer.

Chain-code implementation

Chain-code is usually written in Golang or Java. Chaincode can be public, confidential or access controlled. These codes serve as a smart contract that users can interact with via APIs.

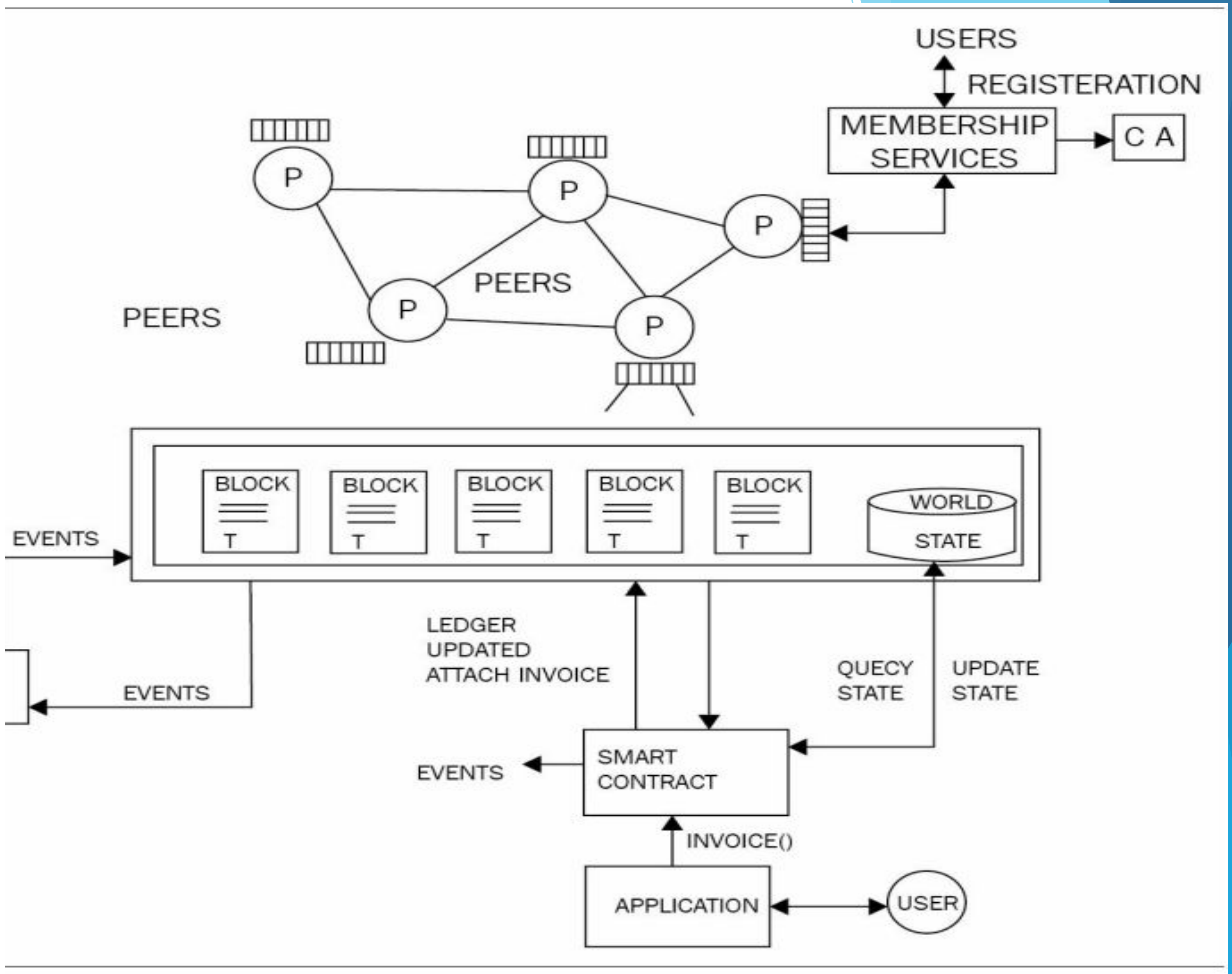
Users can call functions in the chaincode that result in a state change, and consequently updates the ledger.

There are also functions that are only used to query the ledger and do not result in any state change.

The following four functions are required in order to implement the chaincode:

- Init(): This function is invoked when chaincode is deployed onto the ledger. This initializes the chaincode and results in making a state change, which accordingly updates the ledger.
- Invoke(): This function is used when contracts are executed. It takes a function name as parameters along with an array of arguments. This function results in a state change and writes to the ledger.
- Query(): This function is used to query the current state of a deployed chaincode. This function does not make any changes to the ledger.
- Main(): This function is executed when a peer deploys its own copy of the chaincode. The chaincode is registered with the peer using this function.

Overview of Hyperledger Fabric



Application model

Any blockchain application for Hyperledger Fabric follows MVC-B architecture. This is based on the popular MVC design pattern. Components in this model are Model, View, Control, and Blockchain:

- View logic: This is concerned with the user interface. It can be a desktop, web application or mobile frontend.
- Control logic: This is the orchestrator between user interface, data model, and APIs.
- Data model: This model is used to manage the off-chain data.
- Blockchain logic: This is used to manage the blockchain via the controller and the data model via transactions.

Non-Fungible Tokens (NFT)

- Non-Fungible Tokens (NFTs) are unique digital assets stored and managed on a blockchain, a decentralized digital ledger technology.
- Each NFT is distinct and cannot be replicated, making it one-of-a-kind.
- Unlike cryptocurrencies such as Bitcoin or Ethereum, which are fungible and can be exchanged one-to-one, NFTs represent ownership of specific digital content or assets, such as artwork, collectibles, virtual real estate, music, videos, or any other form of digital media.
- Each NFT contains metadata that verifies its authenticity, ownership history, and other relevant information, providing a transparent and immutable record of ownership.

How NFTs Differ From Cryptocurrencies Like Bitcoin

- **Fungibility**
- Cryptocurrencies like Bitcoin are fungible, meaning that each unit is identical and interchangeable with any other unit of the same denomination. For example, one Bitcoin is equivalent in value to any other Bitcoin.
- In contrast, NFTs are non-fungible, meaning each token is unique and cannot be exchanged like-for-like. Each NFT represents ownership of a specific asset, and no two NFTs are identical.
- **Use case**
- Cryptocurrencies like Bitcoin are primarily used as a medium of exchange or store of value, functioning as digital currencies that can be traded or spent.
- NFTs, on the other hand, represent ownership of digital content or assets. They can be bought, sold, and traded like any other asset, but their value is derived from their uniqueness and scarcity rather than their utility as a medium of exchange.

Fungibility vs. Non-fungibility

- Fungibility refers to the property of an asset whereby each unit is interchangeable and indistinguishable from any other unit of the same type. For example, a dollar bill is fungible because one dollar bill is equivalent to any other dollar bill.
- Non-fungibility refers to the property of an asset whereby each unit is unique and cannot be exchanged on a like-for-like basis. Non-fungible assets have distinct characteristics or properties that differentiate them from one another. For example, a piece of artwork is non-fungible because each artwork is unique and cannot be replaced by any other artwork of the same type.

How NFTs Work

- The decentralized nature of blockchain technology ensures that NFT transactions are transparent and resistant to censorship or tampering. When an NFT is created, its ownership information, metadata, and transaction history are recorded on the blockchain, providing a permanent and verifiable record of ownership.

Types of NFTs

- Digital Art and Collectibles
- Digital art NFTs encompass many artistic creations that are tokenized and sold as NFTs, including illustrations, paintings, animations, and 3D models.
- Artists can upload their digital artwork to NFT marketplaces, where collectors can purchase them using cryptocurrency.
- Digital art NFTs often come with metadata that includes information about the artwork, such as the artist's name, creation date, edition number (if applicable), and any additional details or descriptions.
- Collectibles are digital items that hold value due to their rarity, uniqueness, or sentimental value. Examples include virtual trading cards, virtual pets, and virtual fashion items.

- Virtual Real Estate and Gaming Assets
- NFTs are increasingly being used to represent ownership of virtual real estate within virtual worlds, metaverses, and decentralized gaming environments.
- Virtual real estate NFTs allow users to buy, sell, and trade virtual land parcels, buildings, and other assets within virtual environments.
- Gaming assets NFTs include in-game items, characters, skins, weapons, and other virtual goods that players can own, customize, and trade within video games and online gaming platforms.
- NFTs enable players to own their in-game assets and transfer them between games, platforms, and marketplaces.

- Music and Other Digital Media
- NFTs are revolutionizing the music industry by providing artists and musicians new ways to monetize their music, engage with fans, and retain ownership and control over their creative works.
- Music NFTs can represent ownership of digital albums, songs, concert tickets, exclusive access to live performances, and other music-related assets.
- Other digital media NFTs include videos, podcasts, e-books, articles, and other forms of digital content that can be tokenized and sold as NFTs.
- NFTs allow creators and content creators to monetize their content directly without relying on traditional intermediaries like record labels, publishers, or streaming platforms.

- Intellectual Property Rights and Ownership
- In a decentralized and transparent manner, NFTs can be used to tokenize and manage intellectual property rights, such as patents, trademarks, copyrights, and licenses.
- Creators and rights holders can use NFTs to prove ownership, establish authenticity, and manage the licensing and distribution of their intellectual property.
- NFTs provide a secure and immutable record of ownership and transaction history, reducing the risk of copyright infringement, piracy, and unauthorized use of intellectual property.

DLT: The Beating Heart of Blockchain.

"Distributed Ledger Technology (DLT): The Beating Heart of Blockchain" emphasizes the critical role that DLT plays within blockchain networks. In this context:-

- DLT refers to the foundational technology that enables the decentralized, secure, and synchronized recording of transactions across a network of nodes or participants.-
- The Beating Heart metaphorically highlights that DLT is at the core of every blockchain. It symbolizes the essential function of DLT in maintaining the life and integrity of the blockchain ecosystem.-
- Blockchain is a specific application of DLT, organizing data into blocks and linking them together in a chain. DLT provides the underlying structure and consensus mechanisms that make blockchain secure and trustworthy.

This concept underscores the idea that without DLT, blockchain as we know it would not exist. DLT is responsible for **creating the distributed and tamper-resistant ledger that forms** the basis of blockchain systems, enabling trustless transactions, smart contracts, and a wide range of applications beyond cryptocurrencies.