

Blockchain Honor Degree Sem VII

HBCC 601 : Blockchain Platforms

Module - 1 : Introduction to Blockchain Platforms (4 Hours)

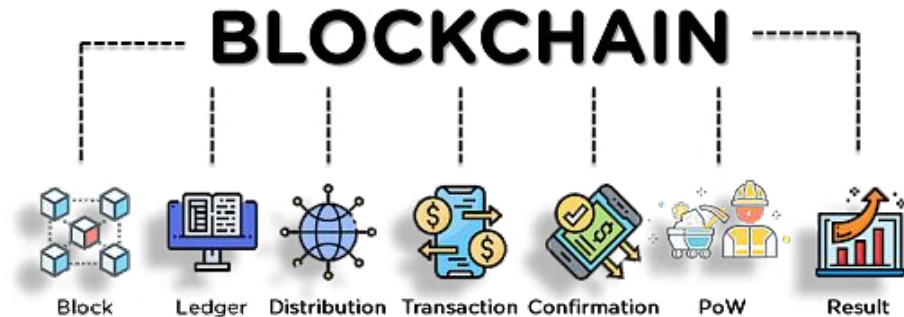
Instructor : Mrs. Lifna C S

Topics to be covered

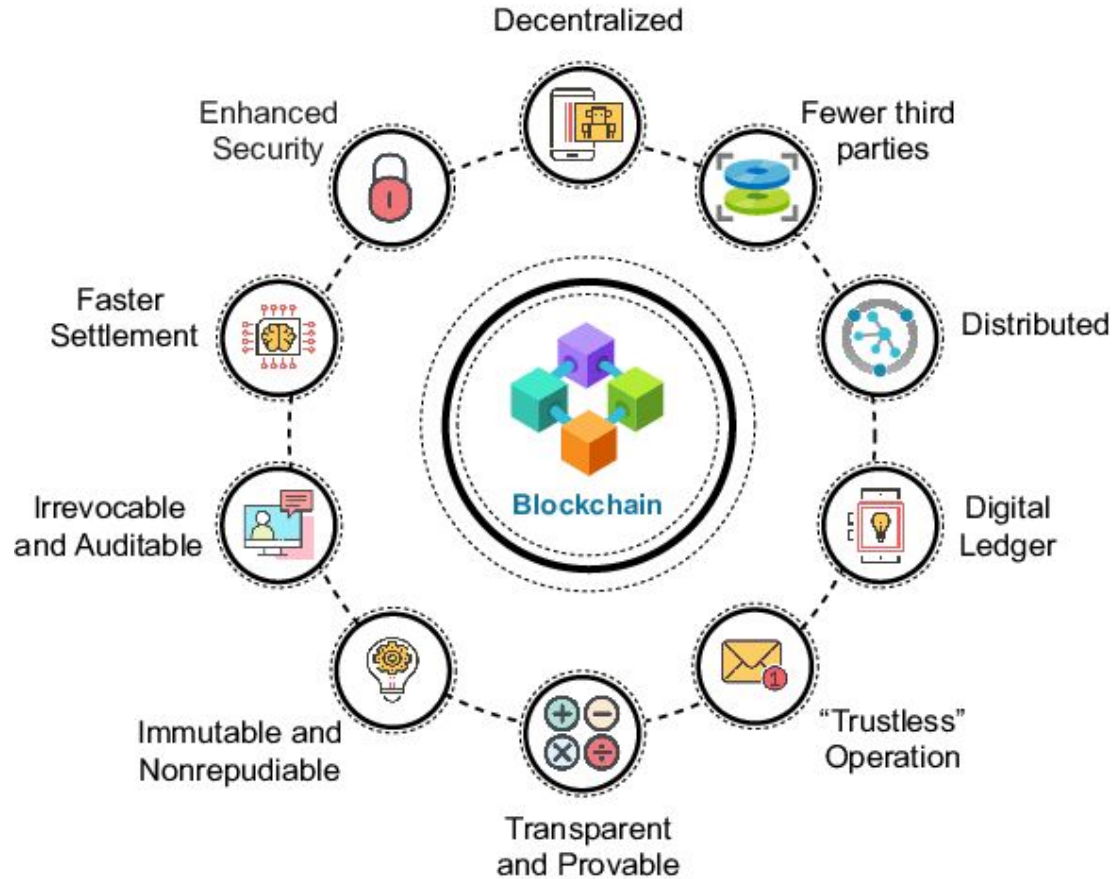
- Why Blockchain Platform:
 - Platform types,
 - Public,
 - Private,
 - technology requirements for implementation.
- Introduction to Ethereum, Hyperledger, and Smart Contracts.
- Case study of blockchain Application.

Self-learning Topics: Study different applications of blockchain.

- Decentralized database that is shared among computer network nodes.
- Transactional data from numerous sources may be readily collected, integrated, and shared using blockchain cloud services.
- Blocks linked together using cryptographic hashes as unique IDs.
- Data integrity is ensured via Blockchain, which uses a single source of truth to eliminate data duplication and increase security.
- Fraud and data tampering is prevented in a blockchain system since data can't be changed without the permission of the nodes of the parties.



Blockchain Characteristics



Why Blockchain Platforms? (3 / 10)

Key aspects that make blockchain platforms appealing:

1. **Decentralization:**

- Traditional systems often rely on a central authority, like a bank or government, to validate and record transactions.
- In a blockchain, the ledger is distributed across a network of nodes, eliminating the need for a central authority.

2. **Security:**

- Blockchain uses cryptographic techniques to secure transactions and control the creation of new units.
- The decentralized and consensus-based nature of blockchain makes it resistant to tampering and fraud.
- Once a block is added to the chain, it is extremely difficult to alter previous blocks, enhancing the overall security of the system.

3. **Transparency:**

- All participants in a blockchain network have access to the same information.
- This transparency helps build trust among users, as they can independently verify the transactions and data stored on the blockchain.

Why Blockchain Platforms? (4 / 10)

4. Immutability:

- Once a block is added to the blockchain, it becomes very difficult to change.
- This immutability ensures that historical transactions are preserved and cannot be altered, providing a reliable and auditable record of events.

5. Smart Contracts:

- Smart contracts are self-executing contracts with the terms of the agreement directly written into code.
- They automatically execute and enforce the terms when predefined conditions are met, eliminating the need for intermediaries.

6. Efficiency and Cost Reduction:

- Blockchain can streamline and automate processes, reducing the need for intermediaries and minimizing the associated costs.
- The decentralized nature of blockchain also eliminates the single point of failure found in centralized systems, improving overall system reliability.

7. Global Accessibility:

- Blockchain operates on a global network, allowing participants from different parts of the world to engage in transactions without the need for intermediaries or traditional banking systems.
- This can be particularly beneficial for cross-border transactions.

Why Blockchain Platforms? (3 /10)

8. Tokenization:

- Blockchain enables the creation of digital tokens, representing various assets such as real estate, art, or even shares in a company.
- This facilitates new forms of ownership, investment, and crowdfunding.

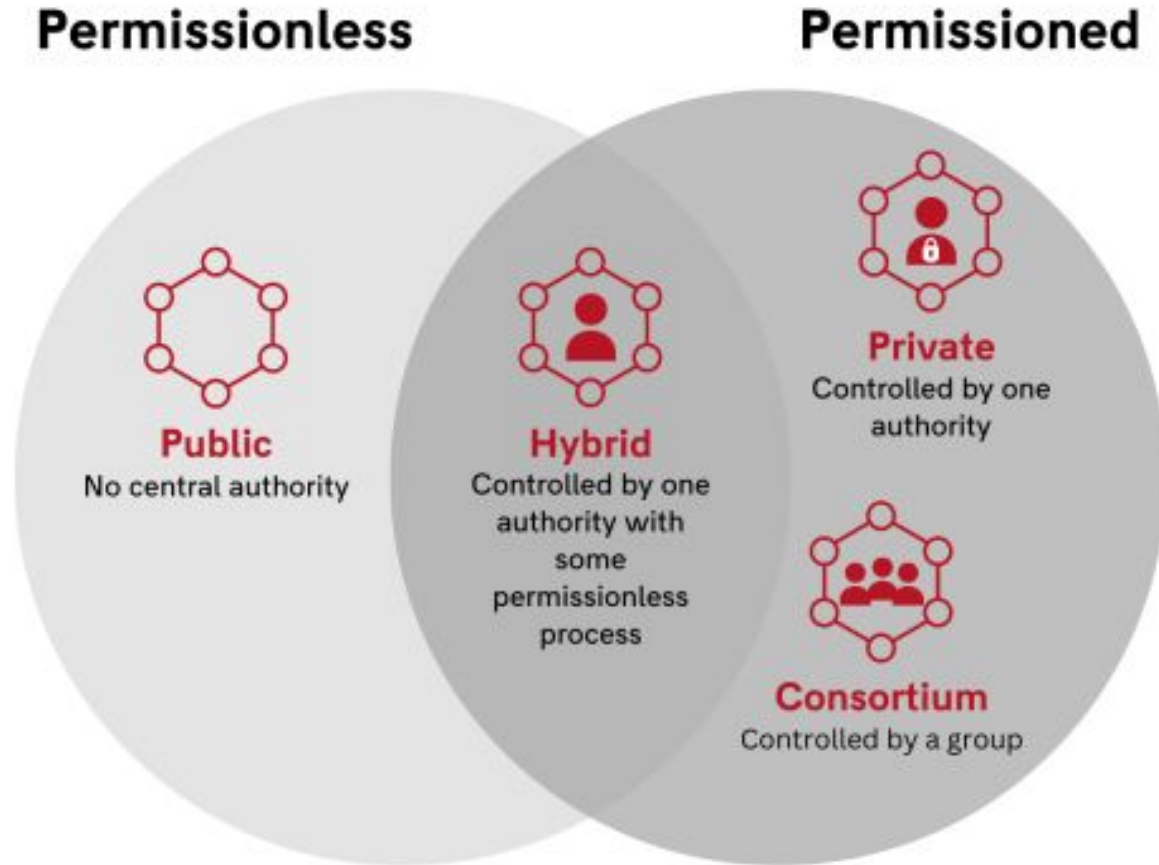
9. Data Integrity:

- Blockchain ensures data integrity by design.
- Each block contains a unique identifier (hash) and references the previous block's hash.
- This interconnection makes it extremely difficult for malicious actors to alter data without detection.

10. Use Cases:

- Blockchain has found applications in various industries, including **finance** (cryptocurrencies like Bitcoin and decentralized finance platforms), **supply chain management, healthcare** (patient data management), **voting systems**, and more

Blockchain Platform Types



Blockchain Platform Types

	Permissioned	Permissionless
Preferred Purpose	Popular among industry-level firms and enterprises	Popular for public use
Decentralization	Limited decentralization	Broad decentralization
Development	Generally proprietary	Generally open source
Transparency	Less transparent	More transparent
Use cases	Manage supply chains	Cryptocurrency blockchains
	Create contracts	
	Verify payment between parties	

Blockchain Platform Types (3 / 11)

- Categorized into several types based on their **design, consensus mechanisms, and use cases**.
- Common types of blockchain platforms:
 1. **Public Blockchains:**
 - Open to anyone.
 - Decentralized and distributed ledger.
 - Examples: Bitcoin, Ethereum.
 2. **Private Blockchains:**
 - Restricted access to a specific group or organization.
 - Often used for internal purposes or within a consortium.
 - Offers more control over the network.
 - Examples: Hyperledger Fabric, Corda.
 3. **Consortium Blockchains:**
 - Semi-decentralized, involving a group of organizations.
 - Shared control among multiple entities.
 - Enhances efficiency through collaboration.
 - Examples: R3 Corda (which can be used both as private and consortium).

Blockchain Platform Types (4 / 11)

4. Permissionless Blockchains:

- Anyone can join and participate.
- Typically associated with public blockchains.
- Examples: Bitcoin, Ethereum.

5. Permissioned Blockchains:

- Participants are known and must have permission to join.
- Offers more privacy and control.
- Common in private and consortium blockchains.
- Examples: Hyperledger Fabric, Quorum.

6. Smart Contract Platforms:

- Allows the execution of smart contracts.
- Automated, self-executing contracts with the terms written into code.
- Examples: Ethereum, Binance Smart Chain.

7. Cryptocurrency Platforms:

- Focused on the creation and management of digital currencies.
- Often includes a native cryptocurrency.
- Examples: Bitcoin, Litecoin.

Blockchain Platform Types (4 / 11)

8. Supply Chain Blockchain:

- Designed for tracking and managing supply chain activities.
- Improves transparency and traceability.
- Examples: IBM Food Trust, VeChain.

9. Identity Management Blockchains:

- Focuses on secure and decentralized identity verification.
- Helps prevent identity theft and fraud.
- Examples: Sovrin, uPort.

10. Cross-Chain Platforms:

- Allows interoperability between different blockchains.
- Enables the transfer of assets and data between blockchains.
- Examples: Polkadot, Cosmos.

11. Tokenization Platforms:

- Used for creating and managing digital tokens.
- Enables the representation of real-world assets on the blockchain.
- Examples: Ethereum (with ERC-20, ERC-721 standards), Binance Smart Chain.

Blockchain Platform Types - Public Blockchain

- completely permissionless.
- Blockchain networks are decentralised, meaning that no single organisation or individual lies at the center of it, controlling it and users can remain anonymous.
- With no permissions, anyone has the liberty to join and participate in the primary activities of a public blockchain.
- That's why public blockchains are known to be self-governed because users have complete freedom to read, write and audit activities on its network.
- **Main features of public blockchain:**
 - Permissionless
 - Largely decentralised
 - Self-governed
 - Maintain's anonymity of users
- **Examples of public blockchain:**
 - [Bitcoin](#)
 - [Ethereum](#)
 - [Litecoin](#)

Blockchain Platform Types - Private Blockchain

- it requires users on its network to be verified.
- Users can only join such a network through invitation when their identity is authenticated.
- They are also given express permission on the level of access they have and the activities they can perform. This means that private blockchains have a single central authority or owner that controls granting access to it and also has the right to override, edit, or delete entries on the blockchain when it deems necessary.
- This results in the network being partially decentralized and smaller than public blockchains because of restricted access and central control.
- However, on the upside, this also means that the networks are fast, efficient and trusted.
- **Main features of private blockchain:**
 - Permissioned
 - Partially decentralised
 - Highly efficient
 - Trusted network
- **Example of private blockchain:**
 - Linux Foundation's [Hyperledger Fabric](#)






Blockchain Platform Types - Hybrid Blockchain

- Amalgamation of a private and public blockchain, combining the best of both networks into one.
- It has the capability of constructing a private, permissioned based system alongside a public, permissionless one.
- This creates a network that is not restricted by one blockchain's limitations
- Allows a single owner or central authority to control who has access to certain data and what is made public.
- **Main features of hybrid blockchain:**
 - Controlled by a central authority
 - A combination of permissioned and permissionless blockchain
 - Information can be accessed via smart contracts
- **Example of hybrid blockchain:**
 - XRP token

Blockchain Platform Types - Consortium Blockchain

- Known as **federated blockchain**,
- Another type of permissioned blockchain with its main point of differentiation being that it is a group of private blockchains owned by multiple individuals or entities.
- While each organisation manages their own blockchain in a consortium blockchain.
- it also allows data from several different sources to be integrated while ensuring a secure and efficient data flow.
- **Main features of consortium blockchain:**
 - Permissioned
 - Partially Decentralized
 - Controlled by a group of entities
 - Privacy and security of data is ensured
- **Examples of consortium blockchain:**
 - R3's [Corda](#)
 - ConsenSys' [Quorum](#)

Blockchain Platform Types : Comparison

	Public	Private	Hybrid	Consortium
 Permissioned/Permissionless	Permissionless	Permissioned	Permissioned & Permissionless	Permissioned
 Control	No control by a central authority	Control by a central authority	Control by a central authority	Control by multiple central authorities
 Main Advantages	<ul style="list-style-type: none"> ✓ Independence ✓ Transparency 	<ul style="list-style-type: none"> ✓ Performance ✓ Scalability 	<ul style="list-style-type: none"> ✓ Performance ✓ Low Cost 	<ul style="list-style-type: none"> ✓ Performance ✓ Security
 Main Disadvantages	<ul style="list-style-type: none"> ✗ Performance ✗ Scalability Issues 	<ul style="list-style-type: none"> ✗ Security ✗ Trust 	<ul style="list-style-type: none"> ✗ Transparency ✗ Upgrading 	<ul style="list-style-type: none"> ✗ Transparency
 Examples	Bitcoin Litecoin	Hyperledger Fabric	XRP token	Corda Quorum

Technology Requirements for Blockchain Implementation (4 /19)

- Several key requirements are required to ensure a successful and secure deployment.
- Requirements may vary based on the type of blockchain (public, private, consortium), its use case, and the desired features.
- Common requirements for implementing blockchain technology:
 1. **Clear Use Case:**
 - Clearly define the purpose and use case for implementing blockchain technology.
 - Identify specific problems or inefficiencies that blockchain can address.
 2. **Consensus Mechanism:**
 - Choose an appropriate consensus algorithm based on the network's goals and requirements (e.g., Proof of Work, Proof of Stake, Practical Byzantine Fault Tolerance).
 3. **Security Measures:**
 - Implement robust security measures to protect against cyber threats and attacks.
 - Use cryptographic techniques for secure transactions and data integrity.
 4. **Decentralization Strategy:**
 - Determine the level of decentralization required for the application.
 - Decide on the number and types of nodes participating in the network.

Technology Requirements for Blockchain Implementation (5 / 19)

5. Node Infrastructure:

- Set up and maintain a network of nodes (computers) that participate in the blockchain network.
- Ensure reliable and secure infrastructure for nodes.

6. Data Privacy and Encryption:

- Incorporate encryption methods to protect sensitive data.
- Define privacy settings and access controls, especially for private and consortium blockchains.

7. Smart Contract Development:

- If applicable, develop and test smart contracts that execute predefined rules autonomously.
- Use programming languages suitable for smart contract development (e.g., Solidity for Ethereum).

8. Interoperability:

- Consider interoperability with other systems or blockchains if required.
- Implement standards that facilitate communication between different blockchain networks.

9. Scalability:

- Address scalability concerns, especially for public blockchains with a large number of transactions.
- Implement solutions like sharding, layer 2 scaling solutions, or sidechains.

Technology Requirements for Blockchain Implementation (5 / 19)

10. Regulatory Compliance:

- Ensure compliance with relevant regulations and legal frameworks.
- Consider issues related to data protection, anti-money laundering (AML), and know your customer (KYC) regulations.

11. User Interface (UI) and User Experience (UX):

- Develop user-friendly interfaces for interacting with the blockchain application.
- Consider the ease of use for both technical and non-technical users.

12. Integration with Existing Systems:

- Plan for seamless integration with existing systems and databases.
- Consider the impact on legacy systems and data migration.

13. Network Monitoring and Management:

- Implement tools for monitoring the health and performance of the blockchain network.
- Have a strategy for managing and maintaining the network.

14. Documentation and Training:

- Create comprehensive documentation for developers, administrators, and end-users.
- Provide training programs to ensure that stakeholders understand how to use and maintain the blockchain system.

Technology Requirements for Blockchain Implementation (5 / 19)

15. Testing and Quality Assurance:

- Conduct thorough testing, including unit testing, integration testing, and security audits.
- Implement a quality assurance process to identify and address potential issues.

16. Governance Model:

- Define a governance model for decision-making within the blockchain network.
- Establish rules for protocol upgrades, dispute resolution, and consensus changes.

17. Backup and Recovery:

- Implement robust backup and recovery mechanisms to prevent data loss in case of failures or disasters.

18. Community and Ecosystem Building:

- Foster a community around the blockchain project to encourage development and adoption.
- Consider building partnerships within the industry or ecosystem.

19. Continuous Improvement:

- Plan for ongoing development and improvements based on feedback and changing requirements.
- Stay informed about advancements in blockchain technology.








Checklist for Choosing a Blockchain Platform



5 Top Blockchain Platforms

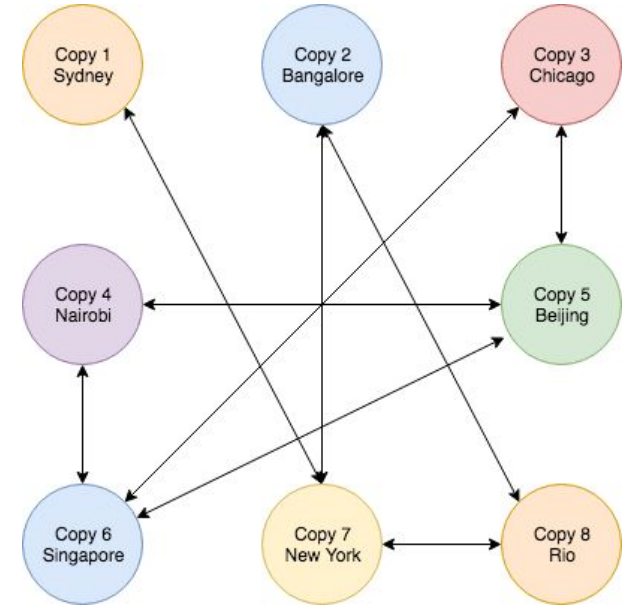


Comparison on the Top Blockchain Platforms

	LEDGER TYPE	TRANSACTION SPEED	CONSENSUS ALGORITHM	SMART CONTRACT	INDUSTRY FOCUS	
ETHEREUM	Permissionless	~20 TPS	Proof of Work Stake	Yes	Cross-Industry	
HYPERLEDGER FABRIC	Permissioned	>2000 TPS	Solo, Kafka, Raft	Yes	Cross-Industry	
HYPERLEDGER SAWTOOTH	Permissioned/Permissionless	>1000 TPS	PBFT, PoET, Raft	Yes	Cross-Industry	
HYPERLEDGER IROHA	Permissioned	≤1000 TPS	YAC Algorithm	Yes, but pre-defined	Cross-Industry	
CORDA	Permissioned	~170 TPS	Pluggable Consensus	Yes	Financial/ Cross-Industry	
RIPPLE	Permissioned	~1500 TPS	Probabilistic Voting	No	Financial Industry	
QUORUM	Permissioned	~100 TPS	Raft, Istanbul BFT	Yes	Cross-Industry	

What is Ethereum?

- a **public, blockchain based distributed computing platform.**
- as **one big computer** made up of small computers around the world.
- Eg: One can write applications and run them on this global computer.
- The platform **guarantees that your application will always run without any downtime, censorship, fraud or third-party interference.**
- Ethereum blockchain can **also transfer money between 2 parties without a central authority**



What is Ethereum?

- A [Blockchain network](#) that introduced a built-in **Turing-complete programming language** that can be **used for creating** various decentralized applications(also called **Dapps**).
- Ethereum network is fueled by its own **cryptocurrency called 'ether'**.
- Allow the implementation of **smart contracts**. Smart contracts can be thought of as 'cryptographic bank lockers' which contain certain values.
- These cryptographic lockers can only be unlocked when certain conditions are met.
- Ethereum is a network that can be applied to various other sectors.
- called **Blockchain 2.0** → it proved that blockchain can be used beyond the financial sector.
- The consensus mechanism used in Ethereum is [Proof of Stakes\(PoS\)](#), which is **more energy efficient** than, [Proof of Work\(PoW\)](#).
- PoS depends on the **amount of stake a node holds**.



History of Ethereum

2013

- Ethereum was **first described** in **Vitalik Buterin's white paper**
- Goal of **developing decentralized applications**.

2014:

- **EVM was specified** in a paper by **Gavin Wood**, and the formal development of the software also began.

2015:

- Ethereum **created its genesis block** marking the **official launch of the platform**.

2018:

- Ethereum **took second place** in Bitcoin in terms of market capitalization.

2021:

- a major network upgrade named **London** included **Ethereum improvement proposal 1559**
- **introduced a mechanism** for **reducing transaction fee volatility**.

2022:

- Ethereum has **shifted from PoW(Proof-of-Work) to PoS(Proof-of-State)**
- Also known as **Ethereum Merge**.
- It has **reduced Ethereum's energy consumption by ~ 99.95%**.



Features of Ethereum

1. Smart contracts:
 - a. Ethereum allows the creation and deployment of smart contracts.
 - b. Smart contracts are created mainly using a programming language called **solidity**.
2. Ethereum Virtual Machine (EVM):
 - a. **a runtime environment for compiling and deploying Ethereum-based smart contracts.**
3. Ether:
 - a. **cryptocurrency of the Ethereum network.**
 - b. **only acceptable form of payment for transaction fees** on the Ethereum network.
4. Decentralized applications (Daaps):
 - a. **has its backend code running on a decentralized peer-to-peer network.**
 - b. Has a frontend and UI to make calls and query data from its backend.
 - c. They **operate on Ethereum** and **perform the same function irrespective of the environment** in which they get executed.
5. Decentralized autonomous organizations (DAOs):
 - a. **works in a democratic and decentralized** fashion.
 - b. **relies on smart contracts for decision-making** within the organization.



Real-World Applications of Ethereum

1.

Voting:

- a. Voting systems are **adopting Ethereum**.
- b. The **results of polls are available publicly, ensuring a transparent fair system thus eliminating voting malpractices**.

2.

Agreements:

- a. With Ethereum smart contracts, agreements and contracts **can be maintained and executed without any alteration**.
- b. Ethereum can be **used for creating smart contracts and for digitally recording transactions based on them**.

3.

Banking systems:

- a. **Due to the decentralized nature** of the Ethereum blockchain it becomes **challenging for hackers to gain unauthorized access to the network**.
- b. **Makes payments on the Ethereum network secure**

4.

Shipping:

- a. **Ethereum provides a tracking framework** that helps with the **tracking of cargo and prevents goods from being misplaced**.

5.

Crowdfunding:

- a. **helps to increase trust and information symmetry**.
- b. **It creates many possibilities for startups by raising funds** to create their own digital cryptocurrency.



- An open-source collaborative effort aimed at advancing cross-industry blockchain technologies.
- Hosted by The Linux Foundation, Hyperledger brings together a diverse community of developers and organizations to collaborate on the development of modular frameworks and tools for enterprise-grade blockchain solutions.
- The project was launched in December 2015 and has since grown to include a variety of blockchain frameworks, libraries, and tools.
- **Key components and features of Hyperledger**
 1. **Collaborative Ecosystem:**
 - Hyperledger fosters a collaborative and vendor-neutral environment, bringing together contributors from various industries, including finance, healthcare, supply chain, and more. It encourages open participation and contributions from individuals and organizations.



HYPERLEDGER

Distributed Ledgers



**HYPERLEDGER
BESU**

Java-based
Ethereum client



**HYPERLEDGER
BURROW**

Permissionable smart
contract machine (EVM)



**HYPERLEDGER
FABRIC**

Enterprise-grade DLT
with privacy support



**HYPERLEDGER
INDY**

Decentralized identity



**HYPERLEDGER
IROHA**

Mobile application focus



**HYPERLEDGER
SAWTOOTH**

Permissioned & permissionless
support; EVM transaction family

Libraries



**HYPERLEDGER
ARIES**



**HYPERLEDGER
QUILT**



**HYPERLEDGER
TRANSACTION**



**HYPERLEDGER
URSA**

Tools



**HYPERLEDGER
AVALON**



**HYPERLEDGER
CACTUS**



**HYPERLEDGER
CALIPER**



**HYPERLEDGER
CELLO**



**HYPERLEDGER
EXPLORER**

Domain-Specific



**HYPERLEDGER
GRID**



**HYPERLEDGER
LABS**

2. Modular Frameworks:

- Hyperledger provides a range of modular blockchain frameworks, each designed to meet specific enterprise use cases. Notable frameworks include:
 - Hyperledger Fabric: A permissioned blockchain framework with a modular architecture, supporting plug-and-play components for consensus algorithms, membership services, and more.
 - Hyperledger Sawtooth: A modular and scalable framework with a unique consensus algorithm called Proof of Elapsed Time (PoET).
 - Hyperledger Besu: An Ethereum-compatible client designed for enterprise use within the Hyperledger consortium.
 - Hyperledger Indy: Focused on decentralized identity, enabling the creation of self-sovereign identities.

3. **Permissioned Blockchains:**

- Hyperledger frameworks are generally designed for permissioned blockchains, where participants are known entities and have defined roles and permissions. This approach is well-suited for enterprise use cases, emphasizing privacy, scalability, and performance.

4. **Smart Contracts and Chaincode:**

- Hyperledger Fabric supports smart contracts, referred to as "chaincode," which are written in languages like Go, JavaScript, or Java. Chaincode allows the execution of business logic on the blockchain.

5. **Consensus Mechanisms:**

- Hyperledger frameworks offer flexibility in choosing consensus mechanisms. Hyperledger Fabric, for example, supports various consensus algorithms, including Practical Byzantine Fault Tolerance (PBFT), Raft, and others.

6. Interoperability and Standards:

- Hyperledger emphasizes interoperability and collaboration between different blockchain networks. It aims to establish common standards to facilitate seamless integration between various Hyperledger frameworks and other blockchain technologies.

7. Identity and Privacy:

- Hyperledger Indy focuses specifically on decentralized identity solutions, providing tools and libraries for creating and managing self-sovereign identities. Privacy and identity protection are critical considerations in Hyperledger projects.

8. Open Source and Apache 2.0 License:

- All Hyperledger projects are released under the open-source Apache 2.0 license, promoting transparency, accessibility, and collaboration within the community.

9. Enterprise Adoption:

- Hyperledger frameworks are designed with enterprise requirements in mind, such as scalability, performance, and compliance with regulatory standards. This focus has led to widespread adoption in industries seeking to leverage blockchain technology for various use cases.

10. Education and Community Support:

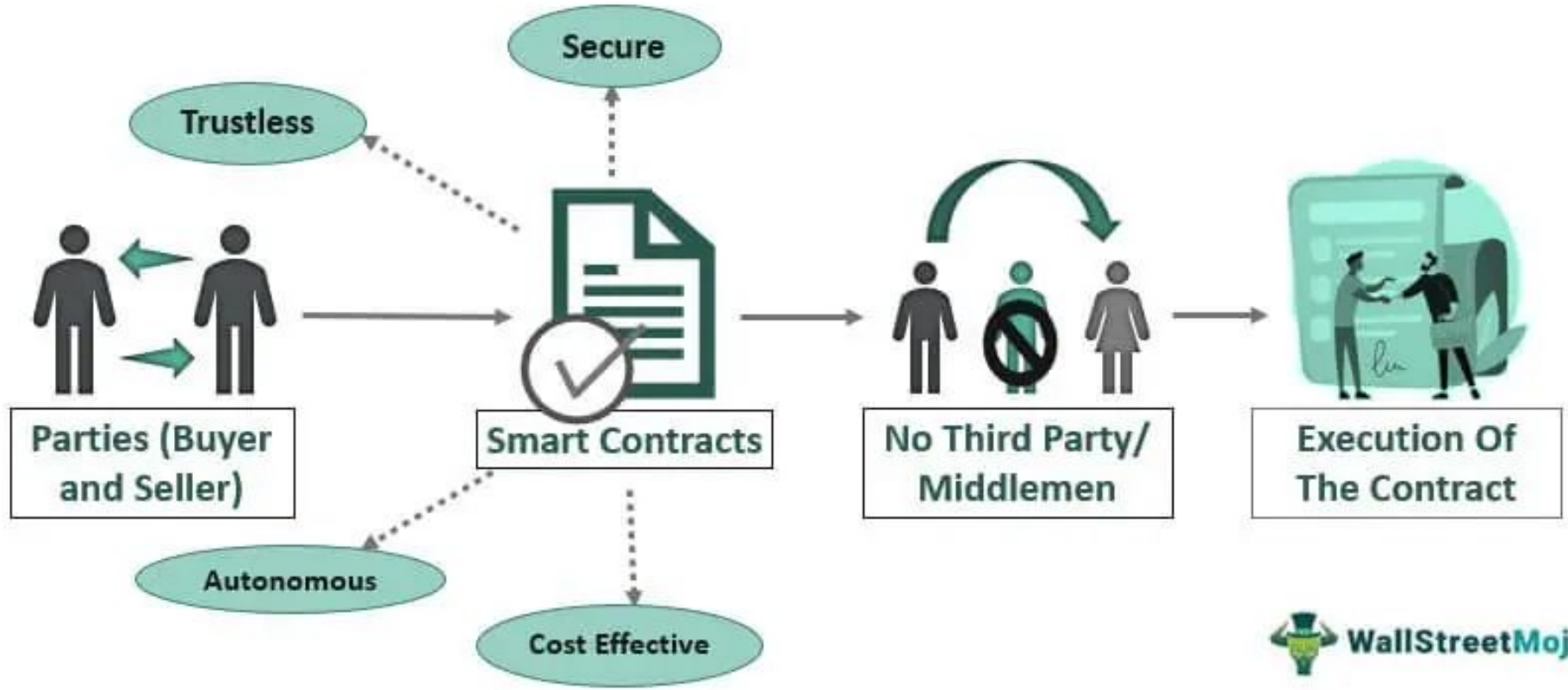
- Hyperledger offers educational resources, including documentation, tutorials, and training programs, to help developers and organizations understand and implement blockchain solutions using Hyperledger frameworks.

What is a Smart Contract ?

- A **self-executing program** that automates the actions required in an agreement or contract. Once completed, the transactions are trackable and irreversible.
- Smart contracts **permit trusted transactions and agreements to be carried out among disparate, anonymous parties** without the need for a central authority, legal system, or external enforcement mechanism.
- **do not contain legal language**, terms, or agreements
- **Only code that executes actions when specified conditions** are met.
- Nick Szabo, an American computer scientist
 - invented a virtual currency called "**Bit Gold**" in 1998,
 - defined smart contracts as computerized transaction protocols that execute the terms of a contract



What is a Smart Contract ?



Features of Smart Contracts

1. **Distributed:** Everyone on the network is guaranteed to have a copy of all the conditions of the smart contract and they cannot be changed by one of the parties. A smart contract is replicated and distributed by all the nodes connected to the network.
2. **Deterministic:** Smart contracts can only perform functions for which they are designed only when the required conditions are met. The final outcome will not vary, no matter who executes the smart contract.
3. **Immutable:** Once deployed smart contract cannot be changed, it can only be removed as long as the functionality is implemented previously.
4. **Autonomy:** There is no third party involved. The contract is made by you and shared between the parties. No intermediaries are involved which minimizes bullying and grants full authority to the dealing parties. Also, the smart contract is maintained and executed by all the nodes on the network, thus removing all the controlling power from any one party's hand.
5. **Customizable:** Smart contracts have the ability for modification or we can say customization before being launched to do what the user wants it to do.
6. **Transparent:** Smart contracts are always stored on a public distributed ledger called blockchain due to which the code is visible to everyone, whether or not they are participants in the smart contract.
7. **Trustless:** These are not required by third parties to verify the integrity of the process or to check whether the required conditions are met.
8. **Self-verifying:** These are self-verifying due to automated possibilities.
9. **Self-enforcing:** These are self-enforcing when the conditions and rules are met at all stages.



Benefits of Smart Contracts



Autonomy



Accuracy



Transparency



High Speed



Data storage



Trustability



Cost Savings



Robust Backup

Types of Smart Contract

1. Smart Legal Contracts

- Legally enforceable
- Require the parties to fulfill their contractual obligations.
- Failure to do so may result in strict legal actions against them.

2. Decentralized Autonomous Organizations (DAO)

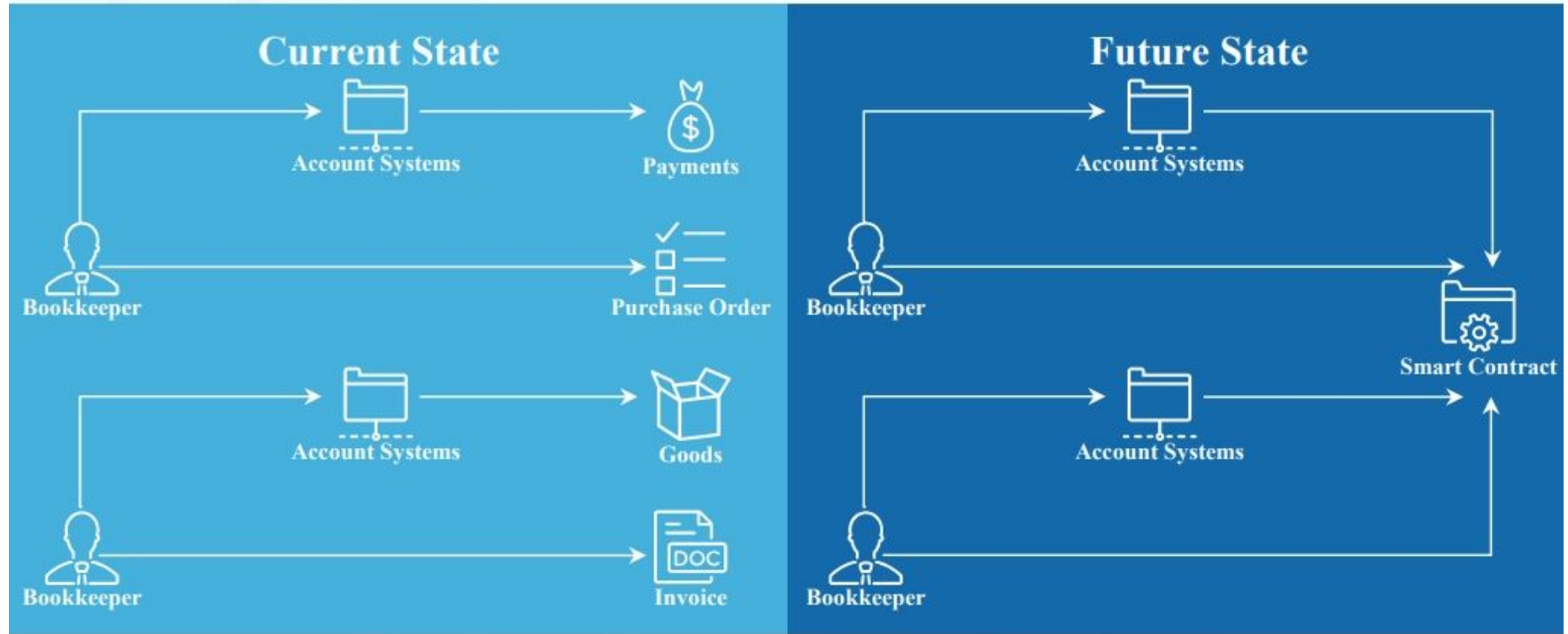
- These are blockchain communities that are bound to specific rules coded into blockchain contracts combined with governance mechanisms.
- Any action taken by the community members gets replaced by a self-enforcing code.

3. Application Logic Contracts (ALC)

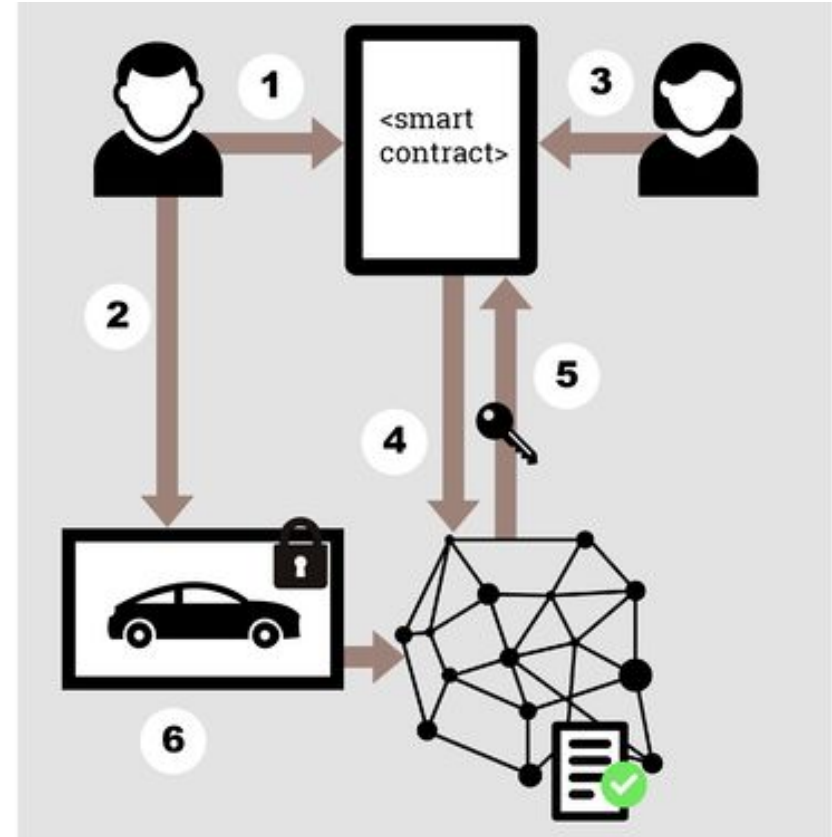
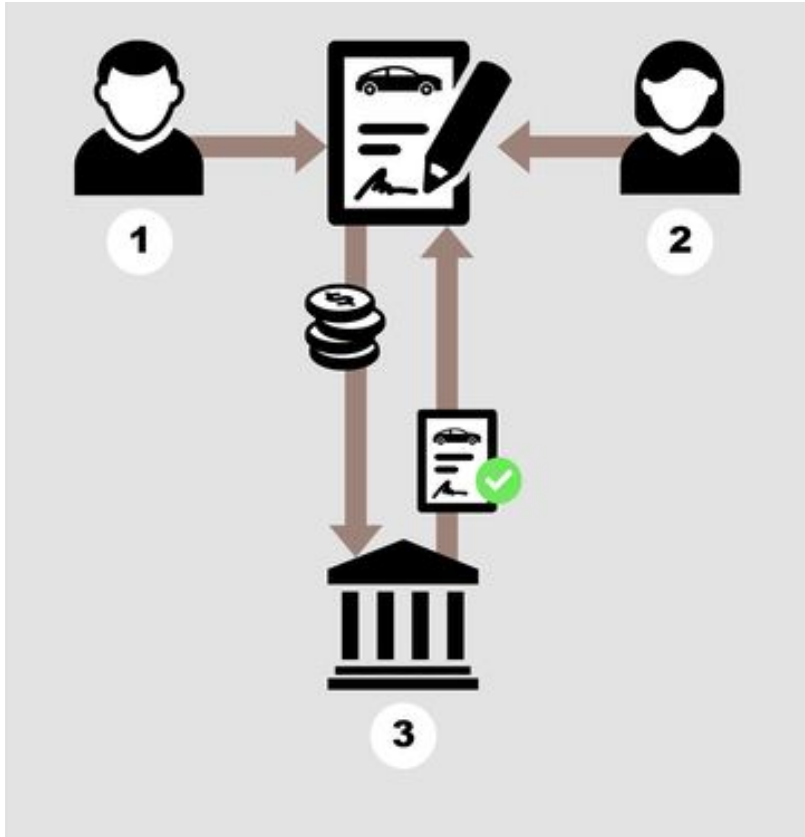
- contain an application-based code that remains in sync with other blockchain contracts.
- It enables communication across different devices, such as the merger of the Internet of Things with blockchain technology.



Future of Financial Reporting



Traditional Contracts Vs Smart Contracts



Traditional Contracts Vs Smart Contracts - Explained

Traditional Contracts

1. Bob would like to sell his car.
2. Alice would like to buy a car.
3. A third party (intermediary) enables the trust that is needed in order to transfer the ownership of the car. Mostly different intermediaries are needed: motor vehicle registration authority, notary, insurance company. All middlemen take fees.

Traditional Contracts Vs Smart Contracts - Explained

Smart Contracts

1. Bob would like to sell his car. He defines in a smart contract the conditions by which he will sell the car and signs the contract with his private key.
2. Bob leaves his car locked with a smart lock in his garage. The car has its own blockchain address and the smart lock is controlled by a smart contract.
3. Alice would like to buy a car. She finds Bob's car on an internet platform and signs the smart Bob's contract with her private key. She adds the agreed amount from her blockchain address to Bob's blockchain address.
4. As soon as the smart contract is executed the whole blockchain network will check if Bob is the real owner of the car and if Alice has enough money to buy the car.
5. If all peers in the blockchain network agree on the same state, it means that all conditions in the smart contract are met. The access code for the smart lock will be transferred to Alice and the blockchain address of the car will be registered to Alice. Bob will get the defined amount of money in his blockchain address.
6. Alice will be able to open the smart lock with her private key.

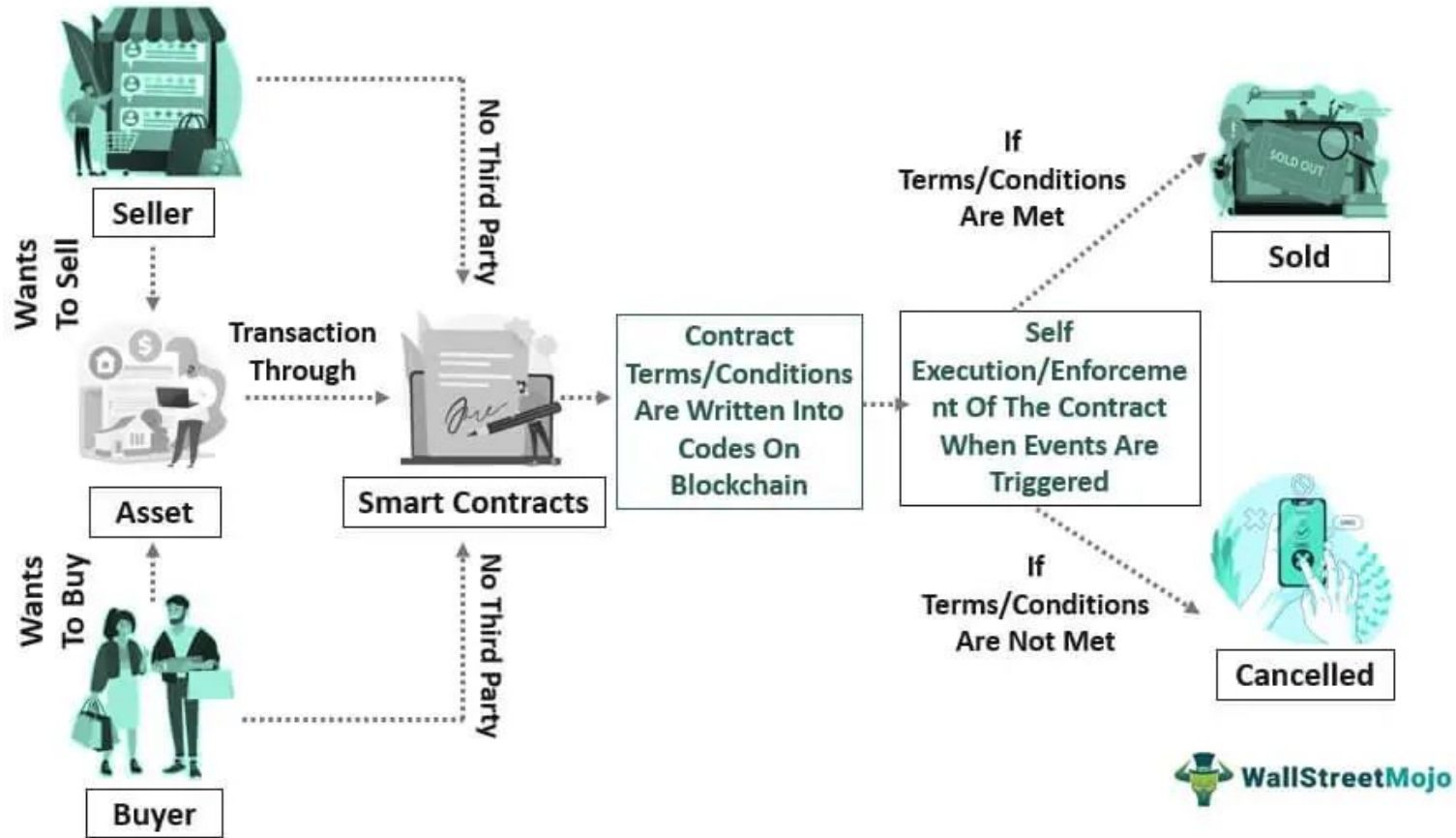


Future of Financial Reporting

	Written contract	Smart contract
Language/Code	Human language	Machine computer code
Automation	All parts of the agreement	Only transactions to be automated
Recording	Conditions can be written on a paper by the concerned parties	Embedded into blockchain or another ledger
Modification of an internal state	Subject to interpretation	Generally immutable



Smart Contract Functioning



How do smart contracts work ?



Identify Agreement

Multiple parties identify the cooperative opportunity and desired outcomes.



Network updates

All the nodes on the network update their ledger.



Set conditions

Smart contracts are executed automatically when certain conditions are met.



Execution and processing

The code is executed and outcomes are memorialized.



Code business logic

A computer program is written



Encryption and blockchain technology

Encryption provides a secure transfer of messages between parties.



How do smart contracts work ?

1. **Identify Agreement:** Multiple parties identify the cooperative opportunity and desired outcomes and agreements could include business processes, asset swaps, etc.
2. **Set conditions:** Smart contracts could be initiated by parties themselves or when certain conditions are met like financial market indices, events like GPS locations, etc.
3. **Code business logic:** A computer program is written that will be executed automatically when the conditional parameters are met.
4. **Encryption and blockchain technology:** Encryption provides secure authentication and transfer of messages between parties relating to smart contracts.
5. **Execution and processing:** In blockchain iteration, whenever consensus is reached between the parties regarding authentication and verification then the code is executed and the outcomes are memorialized for compliance and verification.
6. **Network updates:** After smart contracts are executed, all the nodes on the network update their ledger to reflect the new state. Once the record is posted and verified on the blockchain network, it cannot be modified, it is in append mode only.



Steps involved in the Smart Contracts



Identify parties and establish the terms of the agreement



Define conditions for contract execution



Write smart contract code



Deploy contract to a blockchain platform



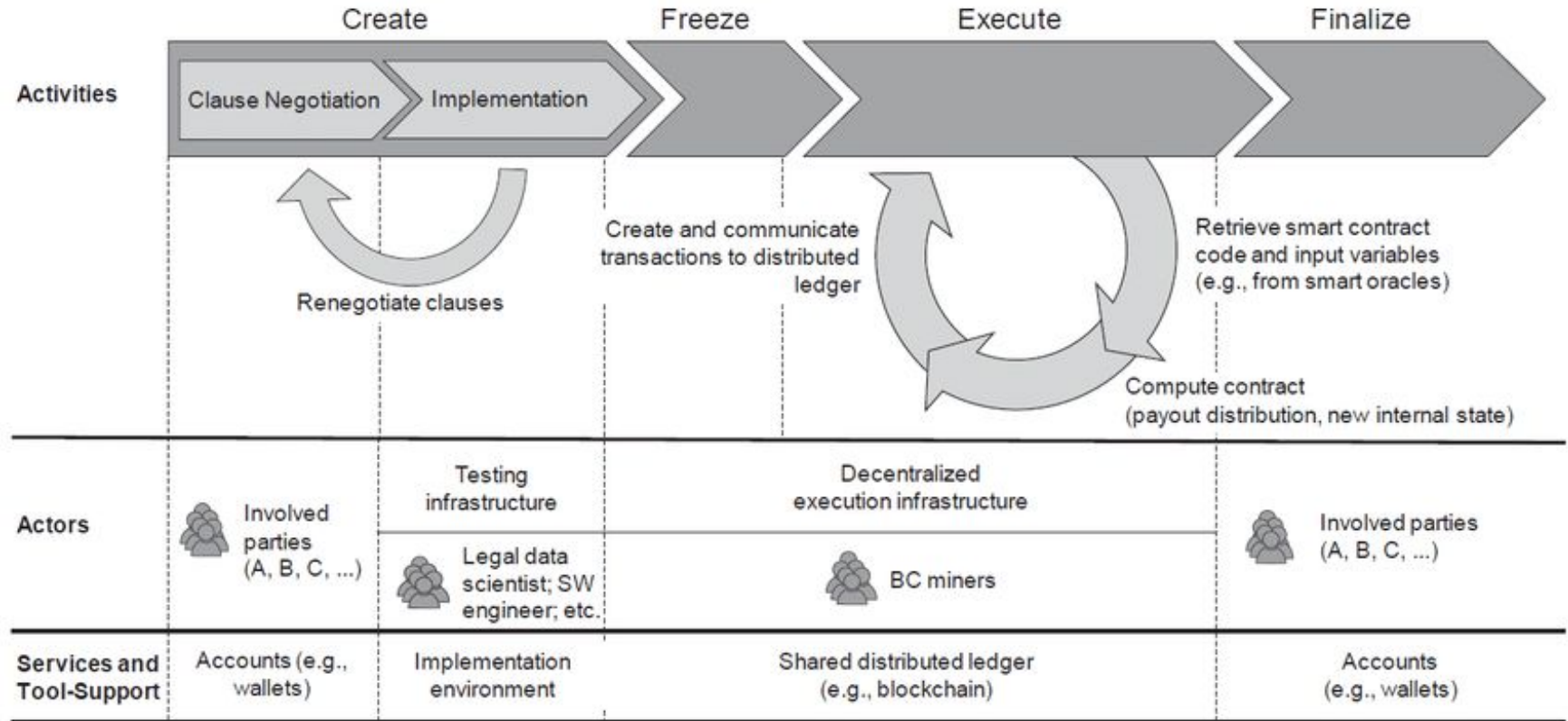
Trigger contract execution automatically



Record contract details on the blockchain ledger



Life Cycle of a Smart Contract



Life Cycle of a Smart Contract

1. Creation Phase:

- consists of iterative contract negotiation and an implementation phase.
- First, the parties must agree on the broad content and goals of the contract.
- Similar to typical contract negotiations and can be conducted online or in person contracts
- During this phase, the following tasks are completed:
 1. Multiple-party bargaining.
 2. Design, implementation, and validation of smart

2. Freeze:

- The validation of transactions on a blockchain is performed by a network of computers known as nodes. The blockchain miners are these nodes.
- To prevent the ecosystem from being swamped with smart contracts, miners must be paid a tiny fee in exchange for this service.



Life Cycle of a Smart Contract

3. Execution:

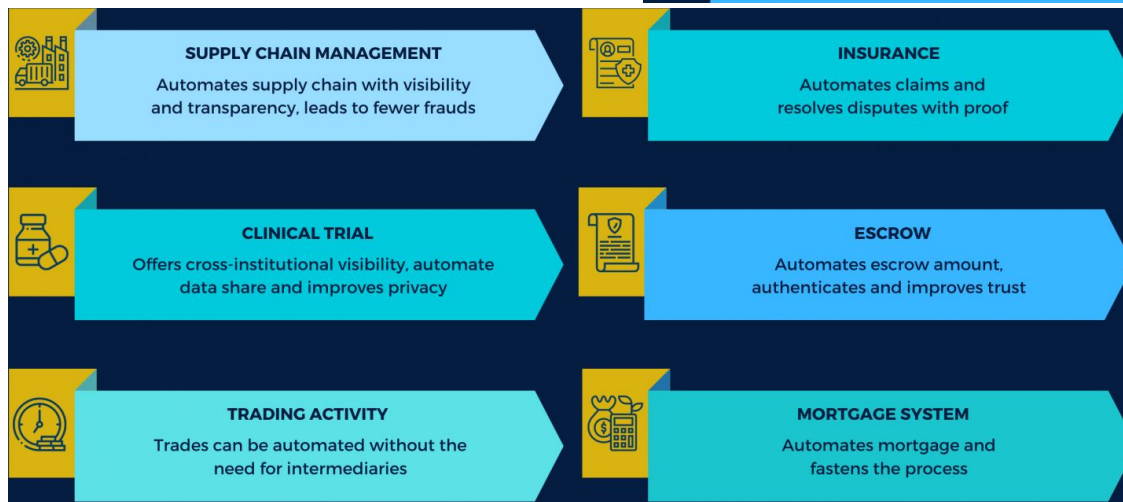
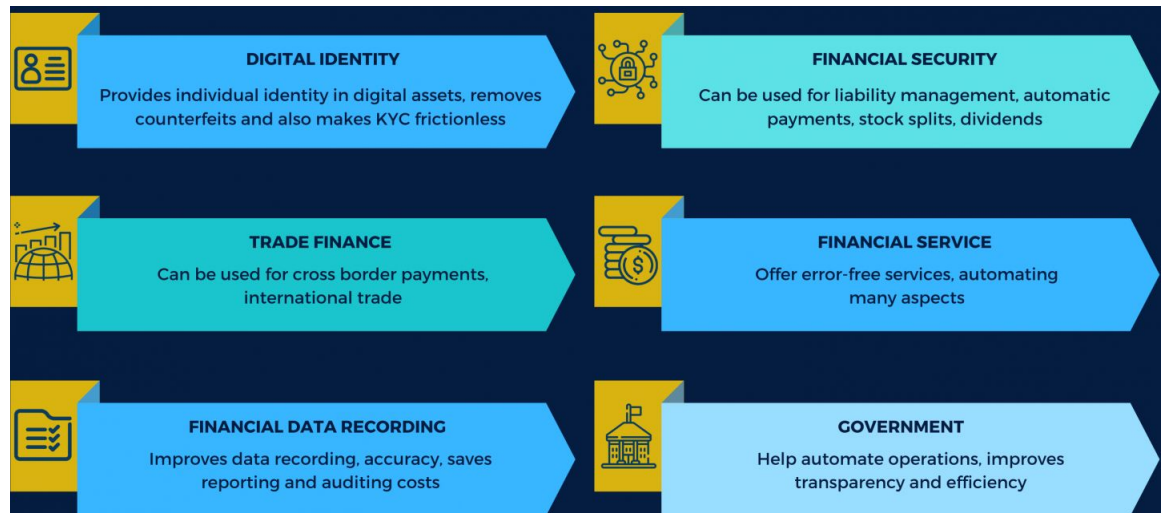
- Contracts placed on the distributed ledger are read by participating nodes.
- The authentication nodes validate the integrity of a smart contract,
- the code is performed by the smart contract interference engine (or by the compiler).
- When one party's inputs for execution are received in the form of coins (commitment to goods via coins), the interference engine **generates a transaction triggered by the met criterion.**
- The execution of the smart contract **results in a new set of transactions and a new state for the smart contract.**
- The **discoveries and new state information** are entered into the distributed ledger and validated using the consensus procedure.

4. Finalize:

- the resulting transactions and updated state information are recorded in the distributed ledger and confirmed via the consensus process.
- The previously pledged digital assets are transferred (assets are unfrozen), and the contract is signed to confirm all transactions.



Use cases of Smart Contracts



Identity Management and Access Control

- Identity Data Protection
- Decentralized Identity Management
- Decentralized Access Control

Real Estate

- Improved Secure Transaction Process
- Eliminated Processing Fees and Commissions

Internet of Things

- Smart Contracts for Scalable Resource Sharing in IoT
- Smart Contracts in Edge Computing
- Smart Contracts for the Enforcement of IoT Security
- UAV
- Smart Cities

Telecommunication

- Smart Contracts for Automated Resource Sharing in IoT
- Tenant Identity Management and Access Control
- Enforcement of Roaming Security

Logistics

- Ensuring Sea/Air Freight Supply Chain Tracability and Compliance
- Special Commodity Supply Chain Tracability
- Agricultural Supply Chain Provenance and Tracability

eGovernment/ Law

- Enforcement of Law by Smart Contracts
- Smart Contracts to Automate Contractual Agreements
- Smart Contracts for Public Services

Healthcare

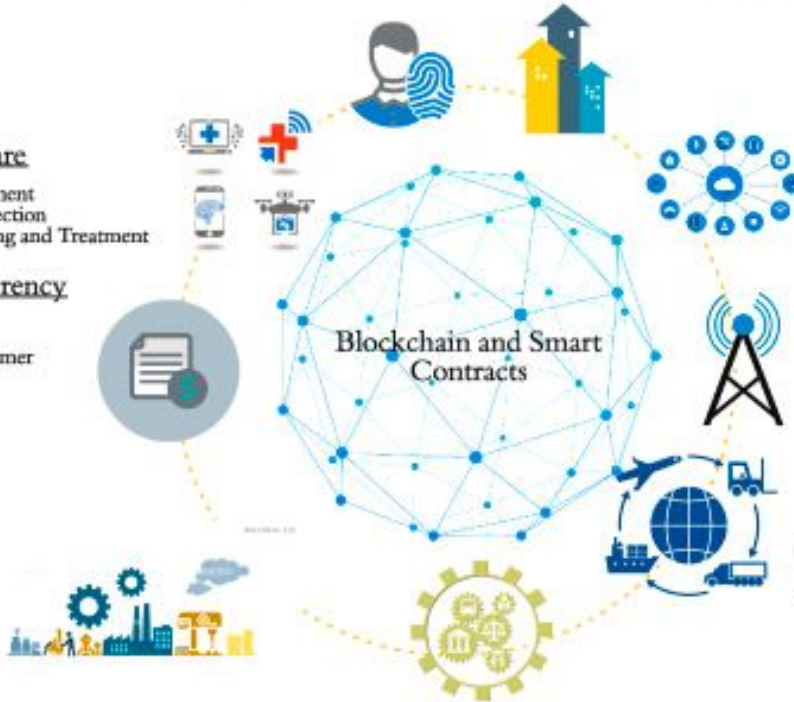
- Health Information Management
- Clinical Research Data Protection
- Automated Patient Monitoring and Treatment

Currency

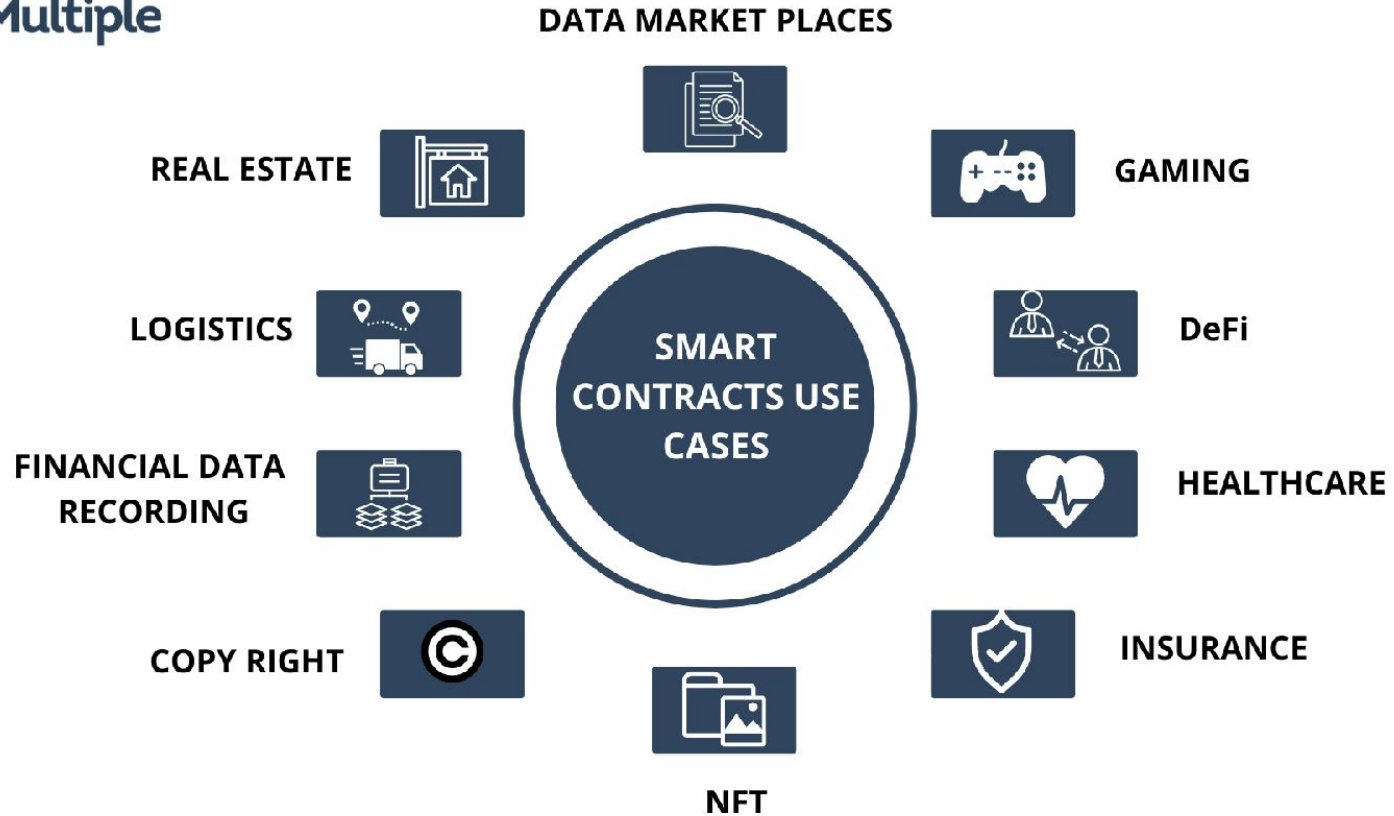
- Currency
- Know Your Customer
- Escrow
- Insurance
- Lending
- Auditing
- Stock Trading

Cross Industry

- Energy Trading
- Waste Management
- Automotive Industry
- Additive Manufacturing



Top 10 Use cases of Smart Contract 2023



Common Use cases of Smart Contract

- **DeFi**

- Cryptocurrencies and smart contracts have allowed decentralized finance platforms to provide financial services without a need for a middleman.
- DeFi had a total value locked of [\\$94](#) billion by end of 2021.
- DeFi has evolved to be more than just peer-to-peer transactions.
- Smart contracts have enabled sophisticated transactions such as lending, borrowing, and derivative transactions on DeFi platforms.
- For example:
 - [AAVE](#) is a DeFi platform that allows borrowing and lending in different cryptocurrencies.
 - [Opyn](#) is a DeFi derivative trading platform that utilizes smart contracts for options trading.

- **NFTs**

- \$17 billion worth of [Non-fungible tokens \(NFTs\)](#) were traded in 2021, making it one of the most impactful smart contract use cases.
- Even though the market has cooled down in the 2nd quarter of 2022, NFTs have real-life [use cases](#) which can lead to long-term use of NFTs.
- Smart contracts have [enabled](#) the creation of non-fungible tokens (NFTs) by allocating ownership and managing the transferability of NFTs. These contracts can also be modified to include additional features such as royalty payments and access rights to a platform or software.



Smart Contract Opportunities



1. **Trust and Transparency:** Smart contracts operate on decentralized blockchain networks, ensuring trust and transparency in transactions. Parties involved can rely on the code's execution rather than trusting a centralized authority.
2. **Reduced Intermediaries:** By automating contract execution, smart contracts eliminate the need for intermediaries like banks, lawyers, or notaries, reducing costs and delays associated with traditional contracts.
3. **Efficiency:** Smart contracts execute automatically once predefined conditions are met. This reduces manual errors, speeds up processes, and lowers administrative costs.
4. **Security:** Blockchain technology provides a high level of security through encryption and consensus mechanisms. Once data is on the blockchain, it is challenging to alter, enhancing data integrity.
5. **Global Reach:** Smart contracts can be accessed and executed globally, enabling businesses to interact with international partners and customers seamlessly.
6. **Immutable Records:** Transactions recorded on a blockchain are tamper-resistant and cannot be altered once confirmed, ensuring a reliable and permanent record

Smart Contract Challenges



1. **Coding Errors:** Smart contracts are only as good as the code written to create them. Coding errors or vulnerabilities can lead to significant financial losses or breaches of privacy. The code must be thoroughly audited and tested.
2. **Regulatory Uncertainty:** The legal and regulatory framework for smart contracts varies by jurisdiction. The lack of clarity in some areas can lead to legal challenges or compliance issues.
3. **Irreversible Transactions:** Once a smart contract is deployed, its actions are irreversible. If there's a mistake or dispute, it can be challenging to resolve without external intervention.
4. **Oracles and External Data:** Smart contracts often rely on external data sources (oracles) to trigger actions. If these sources provide incorrect or manipulated data, it can lead to undesirable outcomes.
5. **Scalability:** Blockchain networks, especially Ethereum, face scalability challenges, resulting in slow transaction times and high fees during periods of high demand.
6. **Privacy Concerns:** While blockchain provides transparency, it may not be suitable for contracts that require complete privacy or confidentiality.
7. **Human Element:** Smart contracts can't account for all real-world scenarios. Human intervention may still be needed for complex negotiations or unforeseen circumstances.
8. **Upgrades and Forks:** Blockchain networks can undergo upgrades or forks, potentially impacting the functionality or compatibility of smart contracts.
9. **Cost of Deployment:** Deploying smart contracts on some blockchain networks can be costly due to gas fees (transaction costs) and development expenses.

What Smart Contracts does not promise to do ?



Ease of
Correction



Cases of
Loophole



Third Party
Elimination



Legal Unclarity



Management of
Vague Terms &
Conditions

Blockchain based Smart Contract Integration Platforms

Platform	Blockchain	Smart Contract Language	Consensus Protocol	Cryptocurrency	System Complexity	Scalability
BitCoin	Public & Private	IVY for Bitcoin Language	PoW	BTC	Medium	Block size 3-7 Tx/Sec
Ethereum	Public & Private	Solidity	PoS	Ether (ETH)	High	Block size 5-20 Tx/Sec
HyperLedger Fabric	Private	Java, Node.js and Go	PBFT/SIEVE	None	High	Block size 100-3000 Tx/Sec
NEM	Public	Java and Node.js	PoI	XEM	Medium	Block size 1,000-10,000 Tx/Sec
Stellar	Public	Stellar SDK & Go	PBFT / FBA	Lumens (XLM)	High	Block size 1,000-1,500 Tx/Sec
Waves	Public	RIDE	LPoS	WAVES	Medium	Block size 100 Tx/Sec
Lisk	Public	Lisk JScript	DPoS	LSK	Medium	Block size 25Tx/Sec
NXT	Public	TC Script	PoS	NXT	Medium	Block size 5-20 Tx/Sec
Monax	Private	Monax SDK and Solidity	PoS	MultiAsset	High	-
Qtum	Public	QSCL and Solidity	PoS	QTUM	Medium	Block size 70-140Tx/Sec

Smart Contracts by Complexity Level

Use case examples

Digital value exchange



A family member sends some bitcoin to another family member

Smart right and obligation



Consumer buys a digital content stream

Basic smart contract



Landlord remotely locks nonpaying tenant out of apartment

Multiparty smart contract



Seller lends buyer funds to buy a house

Distributed autonomous business unit



Unit of a corporation issues its own bonds, and buyers monitor payments via a shared ledger

Distributed autonomous organization



Self-driving trucks make P2P deliveries, pay local toll road fees, and buy local electricity

Distributed autonomous government



Settlers of a previously uninhabited area code their own self-enforcing government services

Distributed autonomous society

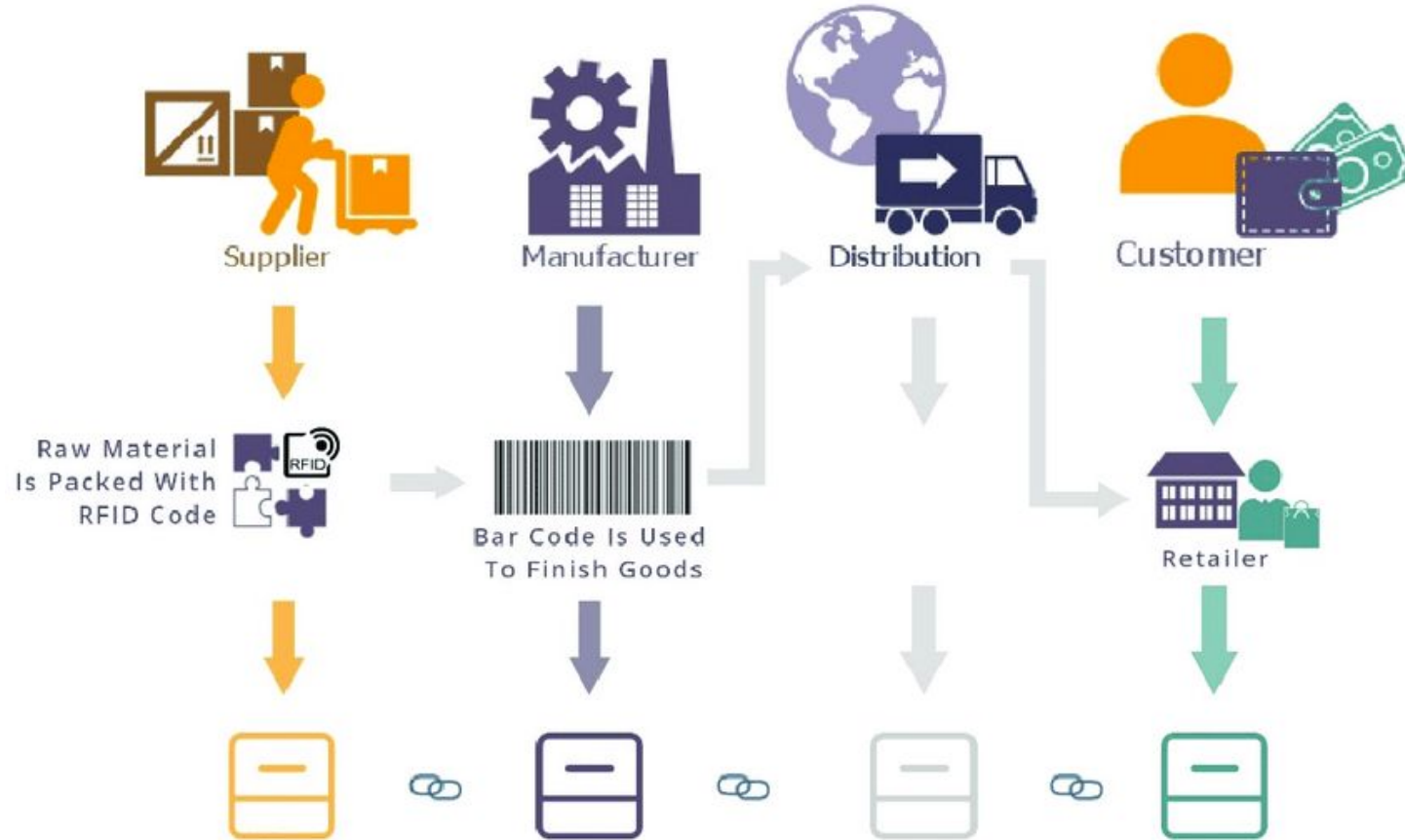


Groups of settlers from different areas establish self-enforcing trade agreements

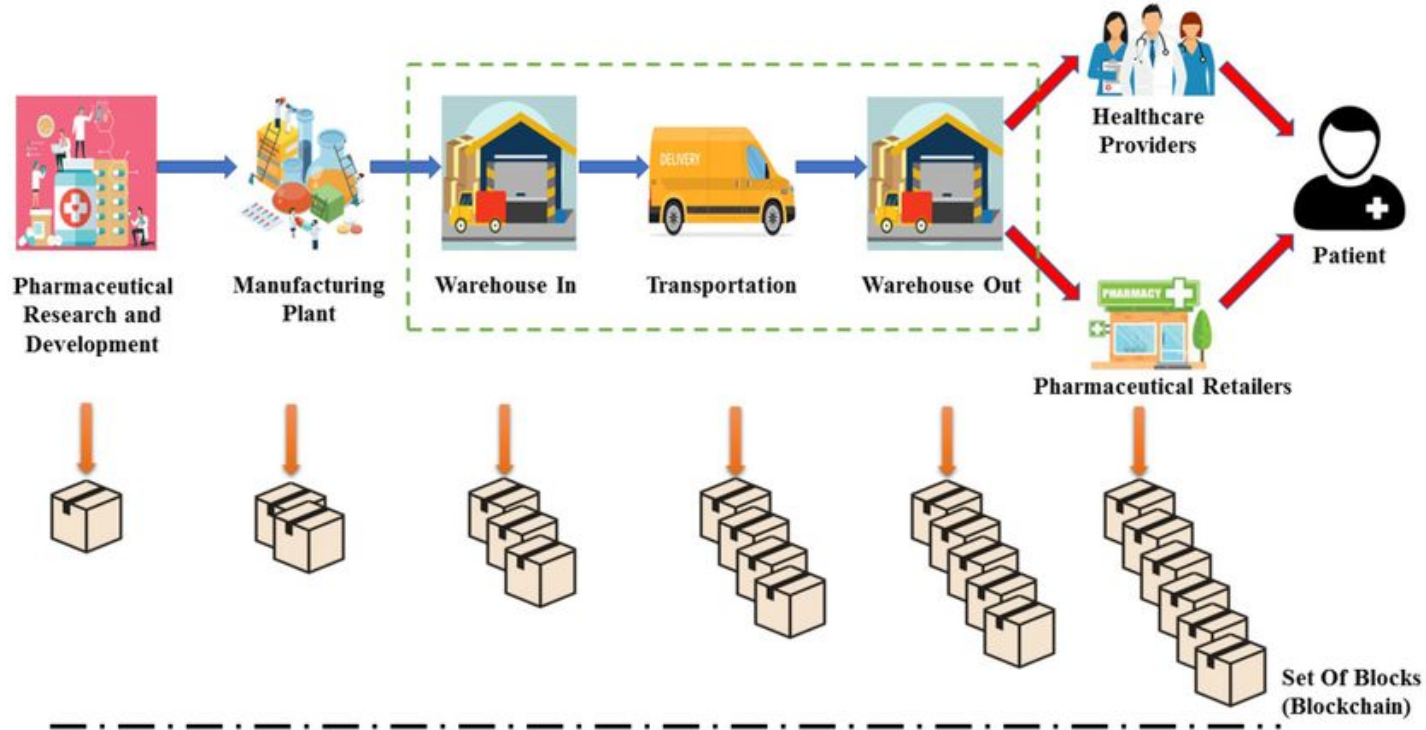
Simple

Complex

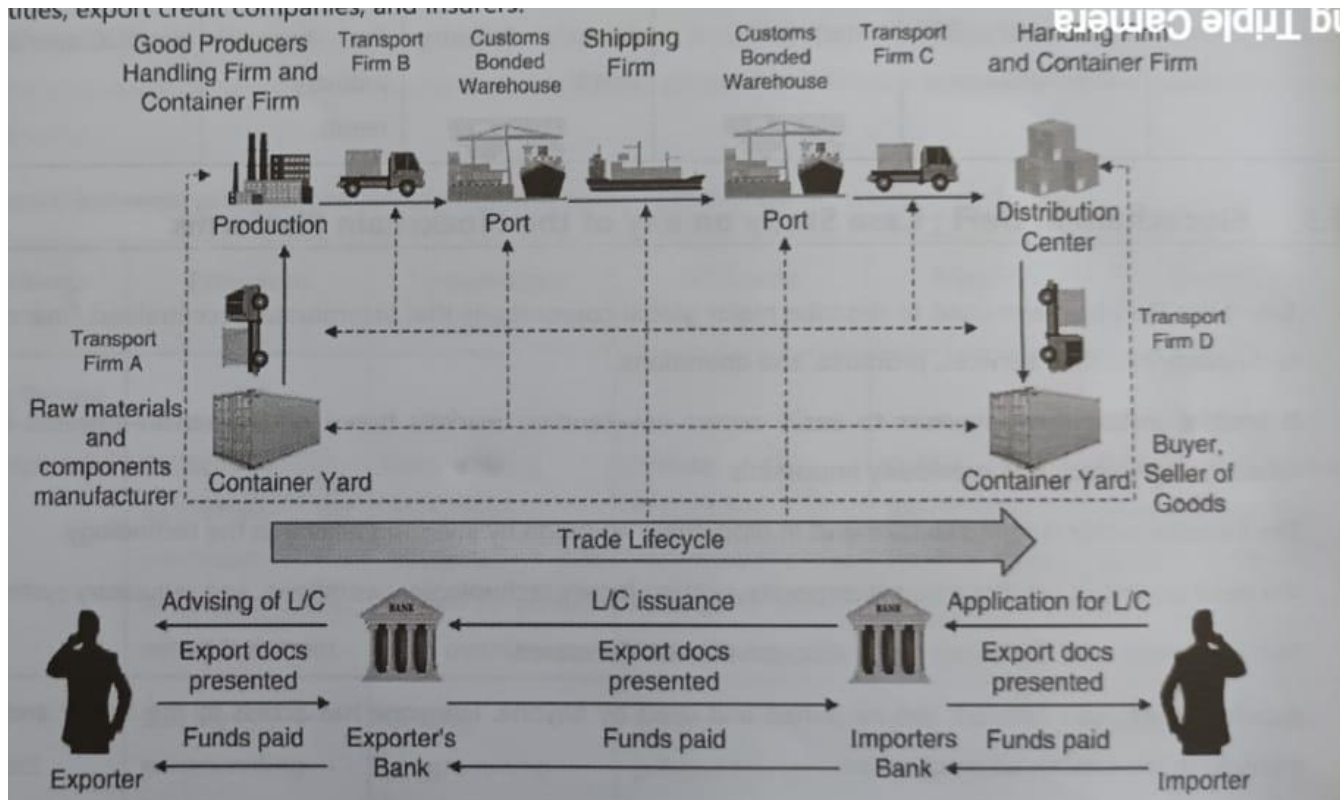
Case Study of Blockchain in a supply chain



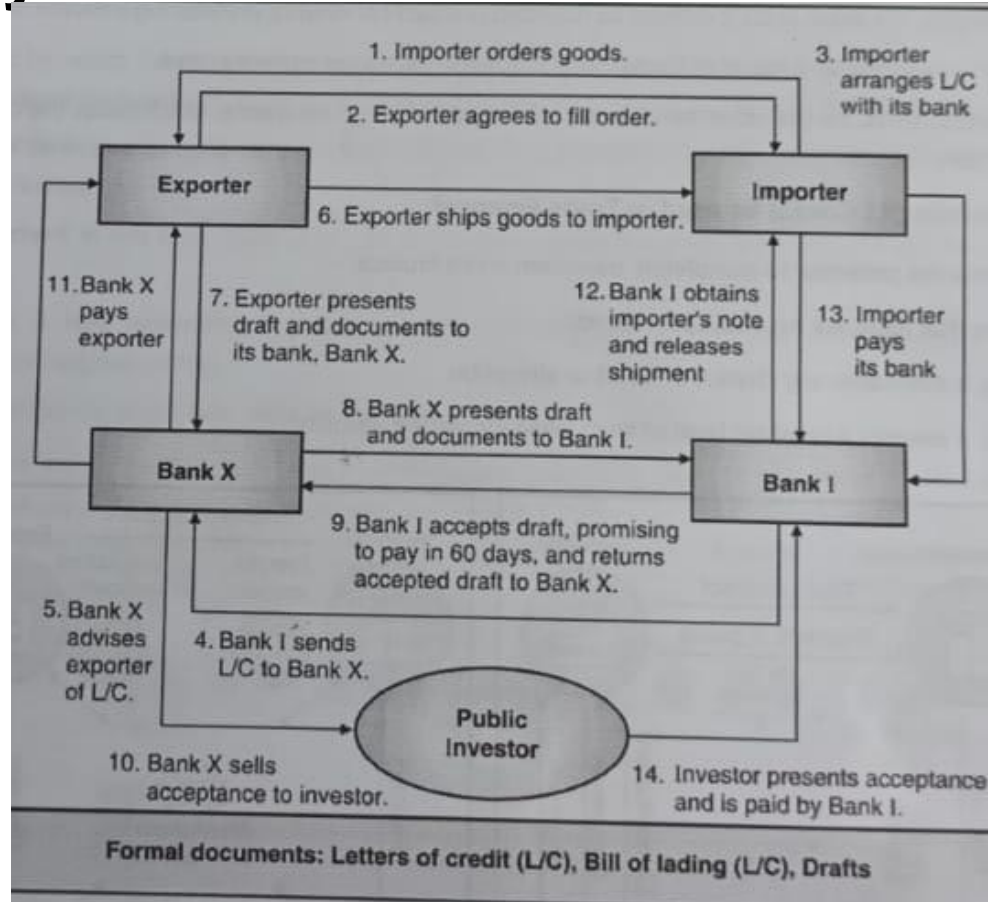
Blockchain in Healthcare SCM



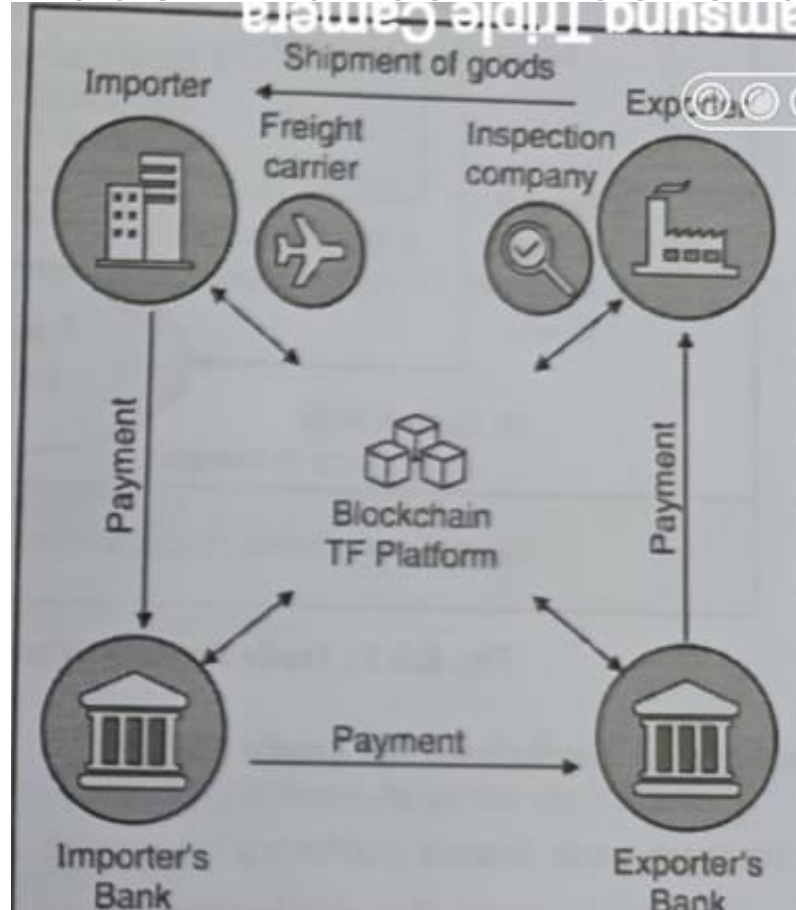
Case Study of Trade Finance : Traditional Setup



Case Study of Trade Finance : Traditional Setup



Case Study of Trade Finance : Blockchain Setup



Case Study of Trade Finance : Blockchain

The Advantages of Blockchain for Trade Finance

- The blockchain trade finance platform's documents can be checked for authenticity at any time! It helps to save time for all parties involved. This also implies that no fraudulent activities take place.
- Transparency is also provided by blockchain in trade finance.
- There are no middlemen, which improves the overall system by reducing fraud. This increases the banks' ability to do trade financing without risk or dispute.
- Blockchain in trade finance can prevent double spending.
- Smart contracts could indeed help to automate and eliminate time spent on paperwork. Before smart contracts can be executed on the network, they must be finalized. Before the smart contract agreement is drafted, both partners must meet and finalize the details.
- Furthermore, the smart contract is only executed when a predefined condition or set of rules is met.
- Blockchain enables buyers and sellers to have proof of ownership, ensuring transparency and tracking the location of the shipment.
- Regulations could be managed in a single location. KYC/AML solutions are included. Additionally, it facilitates trade between nations by allowing for the management of regulatory differences between them.
- Besides this basic use case, blockchain is used in tons of trade finance use cases like, international trade-Letter of Credit, Maritime Trade, etc.

Online References

- <https://www.spiceworks.com/tech/tech-general/articles/top-five-blockchain-platforms/>
- <https://aws.amazon.com/what-is/blockchain/?aws-products-all.sort-by=item.additionalFields.productNameLowercase&aws-products-all.sort-order=asc>
- <https://blogs.opentext.com/blockchain-platform/>
- <https://101blockchains.com/best-blockchain-platforms/>