FR. CONCEICAO RODRIGUES COLLEGE OF ENGINEERING
Department of Computer Engineering

1. Course , Subject & Experiment Details

Academic Year	**2022-23**	**Estimated Time**	**02 - Hours**
Course & Semester	**B.E. (CMPN)- Sem VII**	**Subject Name & Code**	**BCT - (CSDC7022)**
Chapter No.	**05**	**Chapter Title**	**Private Blockchain**

Practical No:	**7**
Title:	Case study: Hyperledger Implementation
Date of Performance:	**26/09/2022**
Date of Submission:	**03/10/2022**
Roll No:	**8953**
Name of the Student:	**Brendan Lucas**

Evaluation:

Sr. No	Rubric	Grade
1	On time submission Or completion (2)	
2	Preparedness(2)	
3	Skill (4)	
4	Output (2)	

Signature of the Teacher:

Date:

HyperLedger Case Study

Name: Brendan Lucas

Roll No. 8953

Batch: C

Class: BE Comps B

Hyperledger Fabric

Hyperledger Fabric is an open source enterprise-grade permissioned distributed ledger technology (DLT) platform, designed for use in enterprise contexts, that delivers some key differentiating capabilities over other popular distributed ledger or blockchain platforms.

One key point of differentiation is that Hyperledger was established under the Linux Foundation, which itself has a long and very successful history of nurturing open source projects under open governance that grow strong sustaining communities and thriving ecosystems. Hyperledger is governed by a diverse technical steering committee, and the Hyperledger Fabric project by a diverse set of maintainers from multiple organizations. It has a development community that has grown to over 35 organizations and nearly 200 developers since its earliest commits.

Fabric has a highly modular and configurable architecture, enabling innovation, versatility and optimization for a broad range of industry use cases including banking, finance, insurance, healthcare, human resources, supply chain and even digital music delivery.

Fabric is the first distributed ledger platform to support smart contracts authored in general-purpose programming languages such as Java, Go and Node.js, rather than constrained domain-specific languages (DSL). This means that most enterprises already have the skill set needed to develop smart contracts, and no additional training to learn a new language or DSL is needed.

The Fabric platform is also permissioned, meaning that, unlike with a public permissionless network, the participants are known to each other, rather than anonymous and therefore fully untrusted. This means that while the participants may not fully trust one another (they may, for example, be competitors in the same industry), a network can be operated under a governance model that is built off of what trust does exist between participants, such as a legal agreement or framework for handling disputes.

One of the most important of the platform's differentiators is its support for pluggable consensus protocols that enable the platform to be more effectively customized to fit particular use cases and trust models. For instance, when deployed within a single enterprise, or operated by a trusted authority, fully byzantine fault tolerant consensus might be considered unnecessary and an excessive drag on performance and throughput. In situations such as that, a crash fault-tolerant (CFT) consensus protocol might be more than adequate whereas, in a multi-party, decentralized use case, a more traditional byzantine fault tolerant (BFT) consensus protocol might be required.

Fabric can leverage consensus protocols that do not require a native cryptocurrency to incent costly mining or to fuel smart contract execution. Avoidance of a cryptocurrency reduces some significant

risk/attack vectors, and absence of cryptographic mining operations means that the platform can be deployed with roughly the same operational cost as any other distributed system.

The combination of these differentiating design features makes Fabric one of the better performing platforms available today both in terms of transaction processing and transaction confirmation latency, and it enables privacy and confidentiality of transactions and the smart contracts (what Fabric calls "chaincode") that implement them. Let’s explore these differentiating features in more detail.

Modularity Hyperledger

Fabric has been specifically architected to have a modular architecture. Whether it is pluggable consensus, pluggable identity management protocols such as LDAP or OpenID Connect, key management protocols or cryptographic libraries, the platform has been designed at its core to be configured to meet the diversity of enterprise use case requirements.

At a high level, Fabric is comprised of the following modular components:

• A pluggable ordering service establishes consensus on the order of transactions and then broadcasts blocks to peers.

• A pluggable membership service provider is responsible for associating entities in the network with cryptographic identities.

• An optional peer-to-peer gossip service disseminates the blocks output by ordering service to other peers.

• Smart contracts ("chaincode") run within a container environment (e.g. Docker) for isolation. They can be written in standard programming languages but do not have direct access to the ledger state.

• The ledger can be configured to support a variety of DBMSs.

• A pluggable endorsement and validation policy enforcement that can be independently configured per application.

There is fair agreement in the industry that there is no "one blockchain to rule them all". Hyperledger Fabric can be configured in multiple ways to satisfy the diverse solution requirements for multiple industry use cases.

What is Hyperledger Fabric?

The Linux Foundation founded the Hyperledger project in 2015 to advance cross-industry blockchain technologies. Rather than declaring a single blockchain standard, it encourages a collaborative approach to developing blockchain technologies via a community process, with intellectual property rights that encourage open development and the adoption of key standards over time.

Hyperledger Fabric is one of the blockchain projects within Hyperledger. Like other blockchain technologies, it has a ledger, uses smart contracts, and is a system by which participants manage their transactions.

Where Hyperledger Fabric breaks from some other blockchain systems is that it is private and permissioned. Rather than an open permissionless system that allows unknown identities to participate in the network (requiring protocols like "proof of work" to validate transactions and secure the network), the members of a Hyperledger Fabric network enroll through a trusted Membership Service Provider (MSP).

Hyperledger Fabric also offers several pluggable options. Ledger data can be stored in multiple formats, consensus mechanisms can be swapped in and out, and different MSPs are supported.

Hyperledger Fabric also offers the ability to create channels, allowing a group of participants to create a separate ledger of transactions. This is an especially important option for networks where some participants might be competitors and not want every transaction they make — a special price they're offering to some participants and not others, for example — known to every participant. If

two participants form a channel, then those participants — and no others — have copies of the ledger for that channel.

Shared Ledger

Hyperledger Fabric has a ledger subsystem comprising two components: the world state and the transaction log. Each participant has a copy of the ledger to every Hyperledger Fabric network they belong to.

The world state component describes the state of the ledger at a given point in time. It's the database of the ledger. The transaction log component records all transactions which have resulted in the current value of the world state; it's the update history for the world state. The ledger, then, is a combination of the world state database and the transaction log history.

The ledger has a replaceable data store for the world state. By default, this is a LevelDB key-value store database. The transaction log does not need to be pluggable. It simply records the before and after values of the ledger database being used by the blockchain network.

Smart Contracts

Hyperledger Fabric smart contracts are written in chaincode and are invoked by an application external to the blockchain when that application needs to interact with the ledger. In most cases, chaincode interacts only with the database component of the ledger, the world state (querying it, for example), and not the transaction log.

Chaincode can be implemented in several programming languages. Currently, Go, Node.js, and Java chaincode are supported.

Privacy

Depending on the needs of a network, participants in a Business-to-Business (B2B) network might be extremely sensitive about how much information they share. For other networks, privacy will not be a top concern.

Hyperledger Fabric supports networks where privacy (using channels) is a key operational requirement as well as networks that are comparatively open.

Consensus

Transactions must be written to the ledger in the order in which they occur, even though they might be between different sets of participants within the network. For this to happen, the order of transactions must be established and a method for rejecting bad transactions that have been inserted into the ledger in error (or maliciously) must be put into place.

This is a thoroughly researched area of computer science, and there are many ways to achieve it, each with different trade-offs. For example, PBFT (Practical Byzantine Fault Tolerance) can provide a mechanism for file replicas to communicate with each other to keep each copy consistent, even in the event of corruption. Alternatively, in Bitcoin, ordering happens through a process called mining where competing computers race to solve a cryptographic puzzle which defines the order that all processes subsequently build upon.

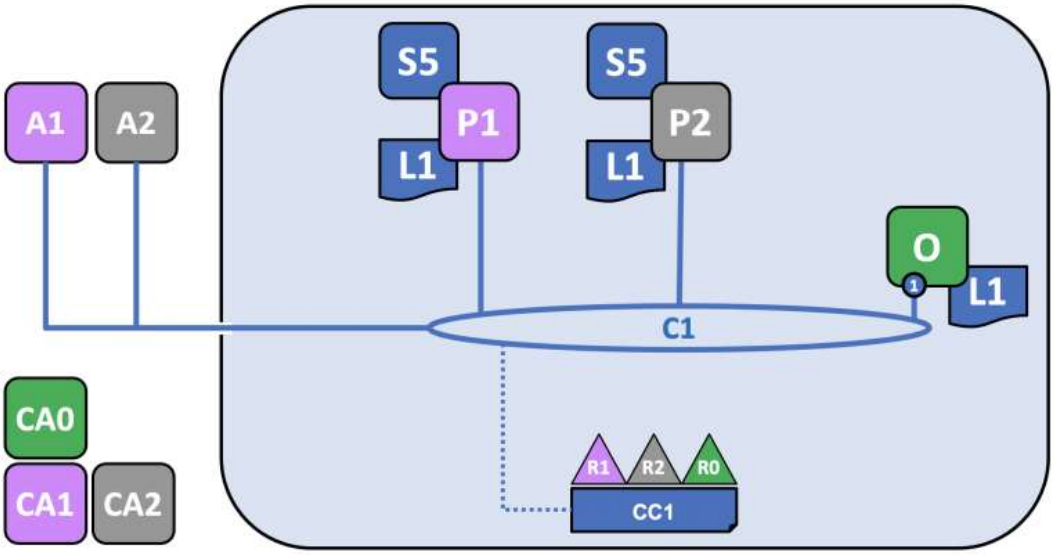
Hyperledger Fabric has been designed to allow network starters to choose a consensus mechanism that best represents the relationships that exist between participants. As with privacy, there is a

spectrum of needs; from networks that are highly structured in their relationships to those that are more peer-to-peer.

The Sample Network

Before we start, let's show you what we're aiming at! Here's a diagram representing the final state of our sample network.

It might look complicated right now, but as we go through this topic, we will build up the network piece by piece, so that you see how the organizations R1, R2 and R0 contribute infrastructure to the network to help form it. This infrastructure implements the blockchain network, and it is governed by policies agreed by the organizations who form the network – for example, who can add new organizations. You'll discover how applications consume the ledger and smart contract services provided by the blockchain network.



Three organizations, R1, R2, and R0 have jointly decided that they will establish a network. This network has a configuration, CC1, which all of the organizations have agreed to and which lists the definition of the organizations as well as the policies which define the roles each organization will play on the channel.

On this channel, R1 and R2 will join peers, named P1 and P2, to the channel, C1, while R0 owns O, the ordering service of the channel. All of these nodes will contain a copy of the ledger (L1) of the channel, which is where transactions are recorded. Note that the copy of the ledger kept by the ordering service does not contain a state database. R1 and R2 will also interact with the channel through the applications A1 and A2, which they own. All three organizations have a Certificate Authority that has generated the necessary certificates for the nodes, admins, organizations definitions, and applications of its organization.

Some Compelling Use Cases

This section describes five concrete examples where blockchain has a clear and compelling use case. These use cases are drawn from different domains and arranged in alphabetical order:

- Banking—applying for a loan
- Financial services—post-trade processing
- Healthcare—credentialing physicians
- IT—managing portable identities
- Supply chain management—tracking fish from ocean to table

In each case, Hyperledger has useful tools available; in some cases, a proof-of-concept has already been developed.

Banking: Applying for a Loan

Banks want to lend, but only to borrowers who are good risks. This motivates the banks to gather detailed, personally identifiable information (PII) from everyone who applies for a loan, such as date of birth, annual income, government ID or passport number, and so on.

Ultimately, the banks use this PII to access an applicant's credit rating. Regulations may demand that certain PII is shared with authorities, for example, to prevent money laundering. But retaining so much PII makes every bank a juicy target for hackers.

Seeking a loan isn't much fun for borrowers, either. The application process is intrusive, and it's hard to "shop around" for the best rates. Every new application multiplies the effort and increases the risk that the applicant's PII will be abused.

HYPERLEDGER INDY CAN STREAMLINE THIS PROCESS

Hyperledger Indy offers a transformative identity solution for this use case.

With Indy, applicants can share only the information the banks need to make a decision, in a way that guarantees truth, builds confidence in the lender, and satisfies pressures from regulators.

Anyone seeking a loan can apply to 100 different lenders in milliseconds, without placing any sensitive personal data into a hackable database.

Instead of disclosing any PII, loan applicants can generate zero-knowledge proofs that they are over 21, that their income on last year's taxes passed a certain threshold, that they hold a valid government ID number, and that their credit score met a certain threshold within the past week.

Strong, distributed ledger-based identity establishes a global source of truth, which delivers value to many parties. Applicants can give consent, and everyone can agree on when and how it was given. Lenders can conform with regulations and show an immutable audit trail.

As a result, the market can operate more efficiently: Banks can offer loans with confidence, while applicants can effectively safeguard their PII.

OTHER HYPERLEDGER PROJECTS ADD FURTHER STRENGTHS

This use case becomes even more compelling when you consider the added strengths of other Hyperledger projects.

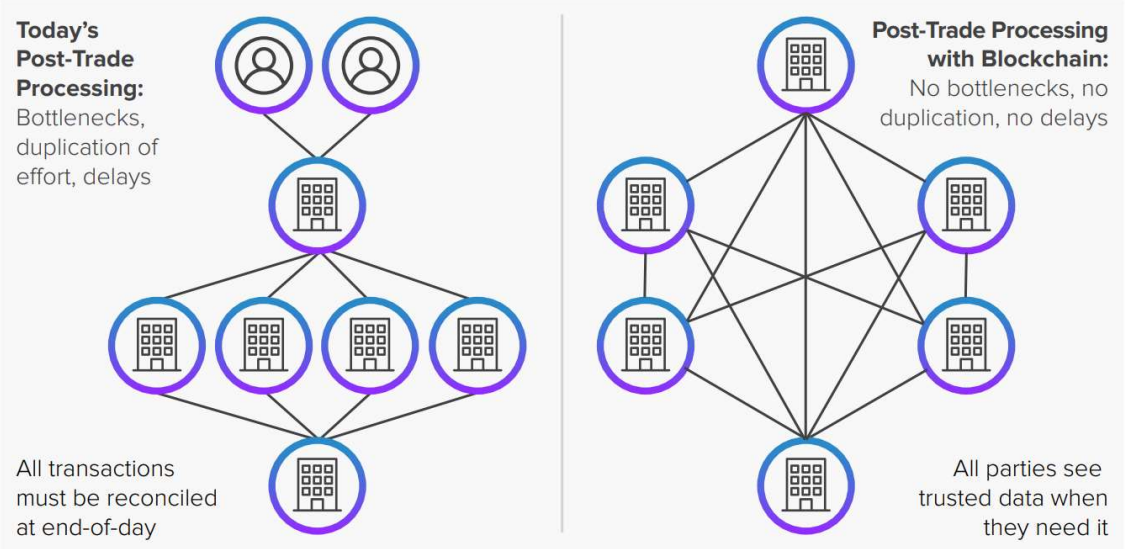
For example, **Hyperledger Burrow** can turn loan applications into smart contracts, and attach them to strong identities as a seamless next step. And **Hyperledger Fabric** can drive a membership system by linking to the pre-existing, self-sovereign identity on the loan application.

Financial Services:

Post-Trade Processing The primary drivers for blockchain in today's financial services are privacy, confidentiality, and accountability.

Compliance guidelines like "Anti-Money Laundering" and "Know Your Customer" require that banks and service providers can verify a customer's legal identity and give them clearance to do transactions. These requirements drive the adoption of permissioned and private blockchains, since public blockchains can risk compromising a participant's privacy and confidentiality.

Together with the large volumes of transactions, these are the main reasons why consortium blockchains are gaining momentum in financial services. Among many possible use cases in this industry—especially in capital markets—post-trade processing can benefit from blockchain.



POST-TRADE PROCESSING, WITH AND WITHOUT BLOCKCHAIN

THE MANY STEPS IN POST-TRADE PROCESSING

Post-trade processing includes all the activities done after a trade is completed. This covers transactions done over-the-counter (OTC) or at an exchange.

On a high level, post-trade processing includes these steps:

1. Trade validation—Validating and confirming the transaction following the execution of the trade.

2. Clearing—Matching the trade instructions and confirmations across the different counterparties as well as the potential netting activities.

3. Settlement—Legally realizing the contractual obligations to reach the finality of the transaction. This includes support processes such as notifying all entities affected by the transaction.

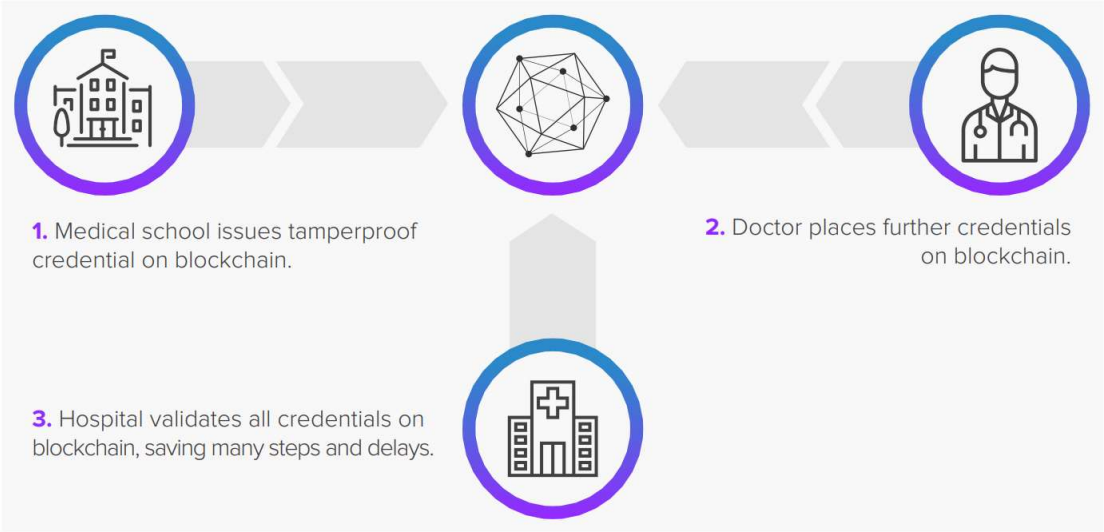
4. Custody activities—Adjusting the positions held with the custodians on behalf of the counterparties.

5. Reporting—Satisfying all reporting requirements for regulatory and internal risk, i.e., the transaction's contribution to the market and credit risk of each counterparty.

Healthcare: Credentialing Physicians

Blockchain technologies promise to reduce one of the great annoyances of modern medical practice: "credentialing''. Hospitals use the credentialing process to make sure that its physicians are competent and trustworthy. In a sense, credentialing is the hospital's way of performing "due diligence'' on a physician.

But today this process imposes a huge burden, both on the physician applying for affiliation and the hospital that must vet the applications.



BLOCKCHAIN MAKES DOCTOR CREDENTIALING FASTER AND MORE SECURE

THE PHYSICIAN GATHERS CREDENTIALS, MANY ON PAPER

Any physician who wishes to become affiliated with a hospital begins the process by gathering copies of all their professional credentials, such as:

- Medical school diploma
- Certificates of any residencies and fellowships completed
- Certificates from any specialty medical boards
- All state medical licenses
- Evaluations from peers
- Proof of meeting requirements for continuing medical education

• Letters from hospitals where the physician was previously affiliated, explaining how and why the affiliation ended

• Details of any malpractice suits

THREE KEY QUESTIONS FOR ANY BLOCKCHAIN SOLUTION

An effective blockchain solution for medical credentialing must answer three key questions about content, identities, and resources.

1. **Will actual content or only pointers to content be placed on the blockchain?**

Credentialing solutions might place public information (such as state medical licensing) on the blockchain itself. However, private information (such as peer reviews) might be better stored off the chain; this would guard against any loss of keys, and enable users to remove—but not change—private information.

2. **What's the best way to manage the identities of many participants?**

An ambitious credentialing solution might include every hospital, every physician, every source of continuing medical education, and so on. This could eventually number thousands of participants. How will so many identities be efficiently and securely managed?

3. What resources are required, especially for storage?

Credentialing solutions may be in service for decades, requiring significant resources for processing, communication, and storage. For example, what if at some point credentialing organizations want video testimony from peers? Storage requirements could skyrocket—and who would cover that added cost?

Supply Chain Management

Tracking Fish from Ocean to Table Ocean fishing represents more than $70B in worldwide trade. But the industry faces many problems.

For example, estimates suggest at least 20% of all fish are caught illegally—yet only a tiny fraction are ever inspected.

A recent study based on DNA testing found that nearly one in three fish were mislabeled by sellers. And a detailed sampling from 674 outlets across the United States found that 87% of snapper and 59% of tuna were mislabelled—and worse, 95% of all sushi restaurants were serving mislabeled fish.

These issues create health risks for consumers, hurt vulnerable fish stocks, rob nations of taxes, and damage the integrity of the whole industry.

MANY CHALLENGES IN SEAFOOD TRACEABILITY

Traceability and provenance are well-managed for certain local catches such as Maine lobster and Maryland crab. But as shown in the Figure, the complexity of the ecosystem makes it challenging to achieve better traceability.

A recent study by the non-profit sustainable seafood organization FishWise11 identified these key problems:

• Many different paths from ocean to table

• Lack of global authority for tracing

• Proprietary tracing systems do not scale

• Most existing processes are paper-based

The supply chain that delivers fish from ocean to table is extremely complex and opaque. It includes many participants from different industries, and regulatory controls that cross national boundaries. That makes this supply chain a perfect opportunity for blockchain technologies.

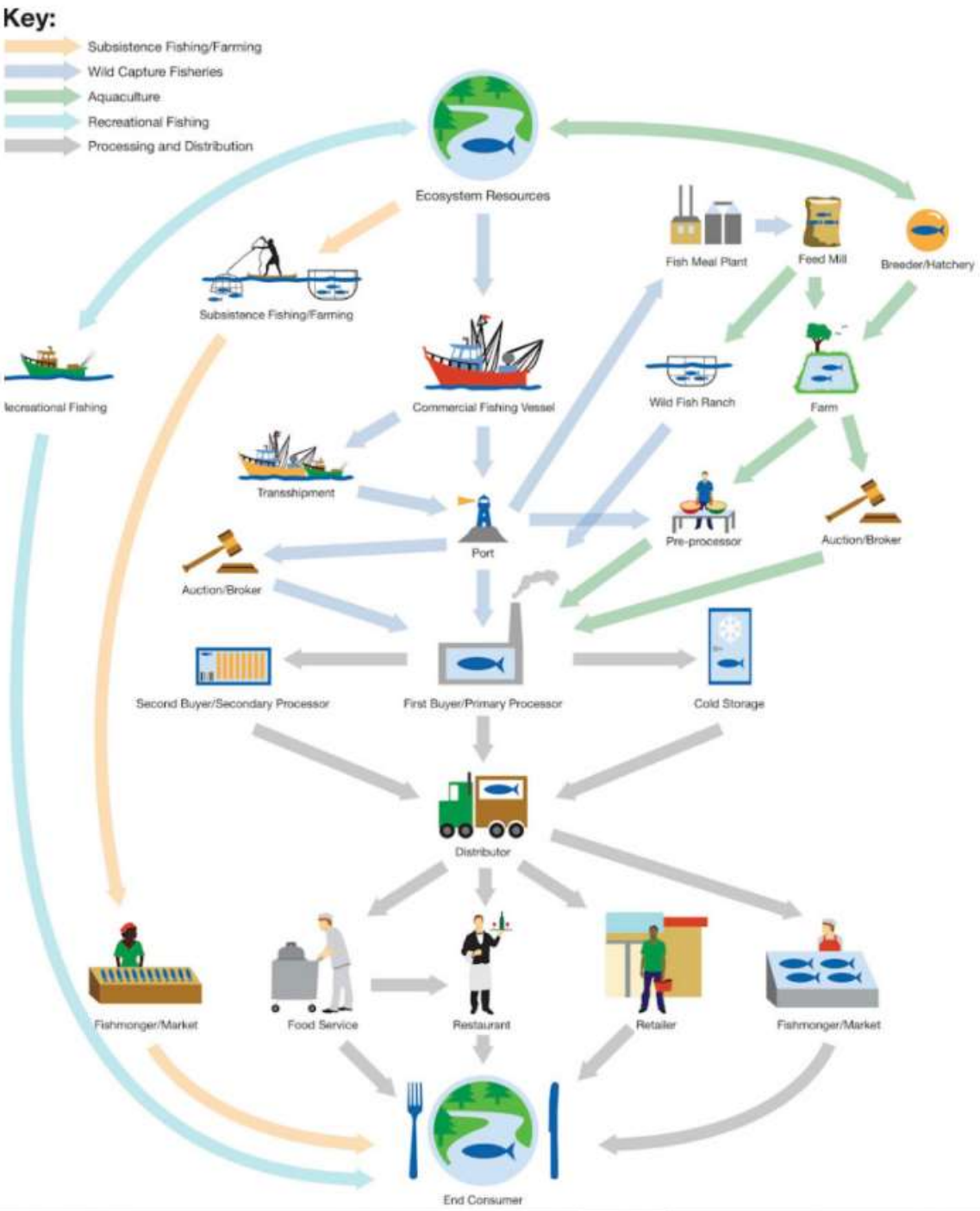
Oceana, an NGO devoted to protecting the oceans, postulated that a shared platform for traceability would help to improve the accuracy of labeling and reduce pirate fishing: “Despite formidable challenges, seafood traceability is well within reach. Simply by keeping track of where our seafood comes from at every step of the supply chain, we can make progress against pirate fishing.’’

A SEAFOOD SUPPLY CHAIN PROTOTYPE

A team at Intel is using **Hyperledger Sawtooth** to build a traceability prototype that combines the distributed ledger, IoT sensors, and advanced communications to track telemetry parameters throughout capture, processing, and transit.

Sensors are attached to the fish when it is caught to record data such as location, temperature, and humidity. This data is recorded in the ledger, along with further events in the processing of the fish: ownership changes, storage temperature range, transport company, and so on. The ledger can also provide analytics for both regulatory enforcement and scientific analysis of fish harvesting and consumption.

This prototype highlights the benefits of Hyperledger Sawtooth as a platform for tracing assets. The lightweight, highly decentralized consensus protocol in Sawtooth (proof of elapsed time or PoET) is particularly well-suited to a diverse, distributed ecosystem where thousands of validating nodes may be required. Broad participation in the ledger reflects the cross-industry nature of the seafood supply chain.



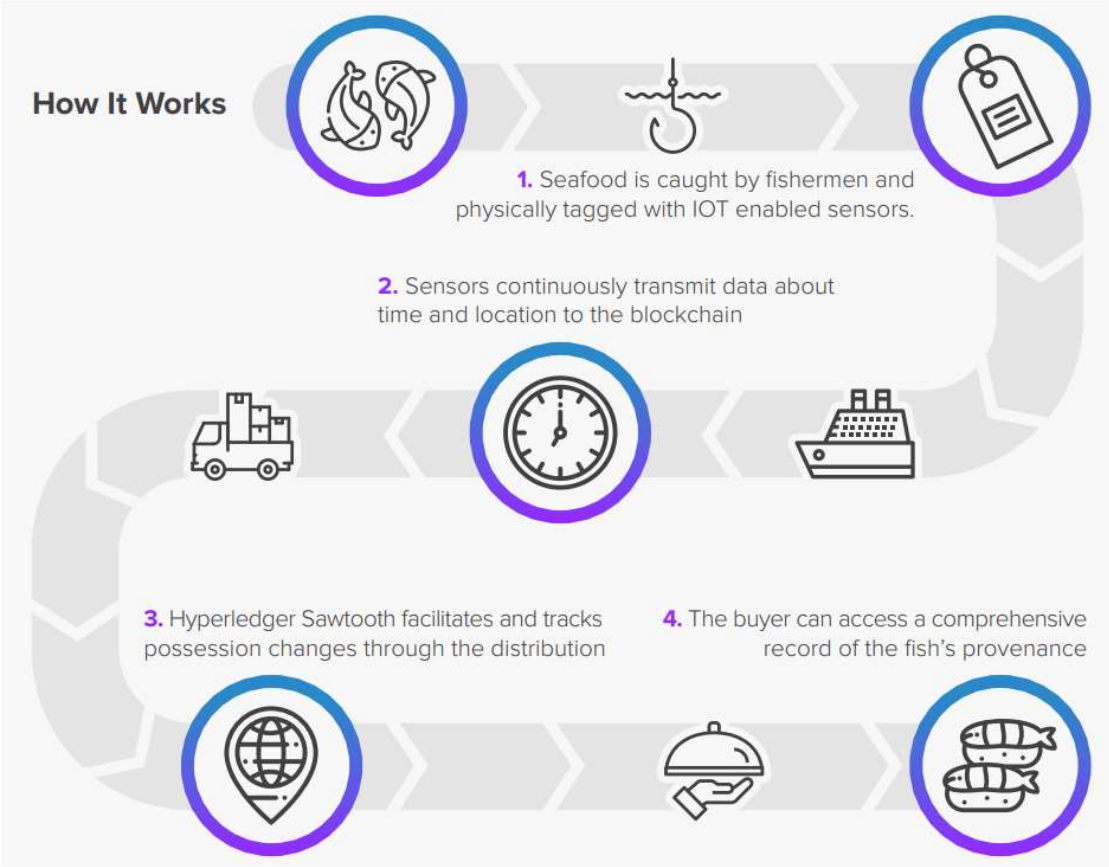
ASSET TRACKING TOUCHES ON DIFFERENT ISSUES

Asset tracking touches on several issues not generally seen in ledgers for financial products For example, asset tracking requires handling diverse data types, such as the composite format required for telemetry and environmental sensing.

Sawtooth accommodates both domain-specific data and the transaction families that operate on it, including data constraints such as verifying the calibration of a sensor.

Blockchain promises a number of benefits for cross-industry traceability. Most of all, these technologies can help establish a community of participants and an authoritative record of provenance. The blockchain's decentralized fault-tolerance enables updates from a wide range of nodes, including fishing boats, trucks, cold-storage facilities, retail stores, and restaurants.

Beyond traceability, digitizing assets opens the door for completely new markets such as, for example, monetization of provenance.



OCEAN TO TABLE FLOWCHART USING HYPERLEDGER SAWTOOTH

Installation

Prerequisites

The following prerequisites are required to run a Docker-based Fabric test network on your local machine.

Git

Install the latest version of git if it is not already installed. To use Fabric binaries, you will need to have the uname command available. You can get it as part of Git but beware that only the 64bit version is supported.

Update the following git configurations

git config --global core.autocrlf false

git config --global core.longpaths true

You can check the setting of these parameters with the following commands:

git config --get core.autocrlf

git config --get core.longpaths

These need to be false and true respectively.

cURL

Install the latest version of cURL if it is not already installed.

Docker

Install the latest version of Docker if it is not already installed.

Once installed, confirm the latest versions of both Docker and Docker Compose executables were installed.

Go

Optional: Install the latest version of Go if it is not already installed (only required if you will be writing Go chaincode or SDK applications).

JQ

Optional: Install the latest version of jq if it is not already installed. (only required for the tutorials related to channel configuration transactions).

Install Fabric and Fabric Samples

Please install the Prerequisites before following these install instructions.

We think the best way to understand something is to use it yourself. To help you use Fabric, we have created a simple Fabric test network using docker compose, and a set of sample applications that demonstrate its core capabilities. We have also precompiled Fabric CLI tool binaries and Fabric Docker Images which will be downloaded to your environment, to get you going.

The cURL command in the instructions below sets up your environment so that you can run the Fabric test network. Specifically, it performs the following steps:

• Clones the hyperledger/fabric-samples repository.

• Downloads the latest Hyperledger Fabric Docker images and tags them as latest

• Downloads the following platform-specific Hyperledger Fabric CLI tool binaries and config files into the fabric-samples /bin and /config directories. These binaries will help you interact with the test network.

– configtxgen,

– configtxlator,

– cryptogen,

– discover,

– idemixgen,

– orderer,

– osnadmin,

– peer,

– fabric-ca-client,

– fabric-ca-server

Conclusion

Any serious evaluation of blockchain platforms should include Hyperledger Fabric in its short list.

Combined, the differentiating capabilities of Fabric make it a highly scalable system for permissioned blockchains supporting flexible trust assumptions that enable the platform to support a wide range of industry use cases ranging from government, to finance, to supply-chain logistics, to healthcare and so much more.

Hyperledger Fabric is the most active of the Hyperledger projects. The community building around the platform is growing steadily, and the innovation delivered with each successive release far out-paces any of the other enterprise blockchain platforms.

Implementation of Code

https://www.freecodecamp.org/news/ultimate-end-to-end-tutorial-to-create-an-application-on-blockchain-using-hyperledger-3a83a80cbc71/