Blockchain Platforms
Course 4 of the Specialization

Contents

Chapter 1

Permissioned Blockchains

1.1 Hyperledger

1.1.1 Intro

We began this journey with the BitCoin genesis. Then worked on problem solving using Ethereum. And then deeper into blockchain protocol with the decentralized application development using Ethereum. In this course we rise up from the rabbit hole of blockchain exploration and look around the ecosystem. We explore the surroundings for other Blockchain platforms. Frameworks, tools, alt and letters to POW contents protocol, and working decentralized application platforms. It is indeed bustling with promising initiatives. We observe hyper ledger, an umbrella organization managed by the Linux Foundation with participation from many prominent businesses, an institution at various levels. We also know the contribution of Microsoft in the form of Blockchain as a service facilitating easy adoption of many Blockchain technologies such as Ethereum, R3, and Hyperledger Fabric, and many more. And then there are hotly debated issues such as the content is protocol, scalability, privacy, confidentiality, and interoperability of the Blockchain technology. And autonomous payment issues such as escrow and multi-signature contracts. And of course we see the emerging alternatives to the Blockchain, such as Hashgraph. Our goal in this last course is to cover these important topics at a high level. I'm excited to venture out into this ecosystem, and I hope you are too to come along for the ride. And completion of this course, you will be able to discuss the permissioned blockchain architectures of hyperledge. Discuss the Blockchain as a service offered by Microsoft Azure. Analyze two representative decentralized application platforms in Augur and Grid+. Explore the challenges in wider adoption of blockchain and solutions for continuous improvement. And explore alternative decentralization models such as IPFS and Hashgraph.

1.1.2 Part 1

The primary purpose of the Bitcoin blockchain is to support a decentralized peer-to-peer payments system. It was meant to be a transparent, permissionless, and a public system, where anyone can enter and leave as they wish, just like in any other bearer payment system such as transacting cash. However, when the use cases for the blockchain expanded beyond the simple payment system into the business areas such as personal healthcare systems and financial systems, privacy and restricted access became imperative and warranted. Even in the public payment system, it became apparent that the whole chain may not be relevant and need not be recorded by all the participants. For example, business transactions in a Buffalo School District may not be relevant to the transactions of Nairobi Tourism Board. Such thoughts resulted in the creation of permissioned blockchain, where only nodes with permission can transact, and take part in the blockchain operations.

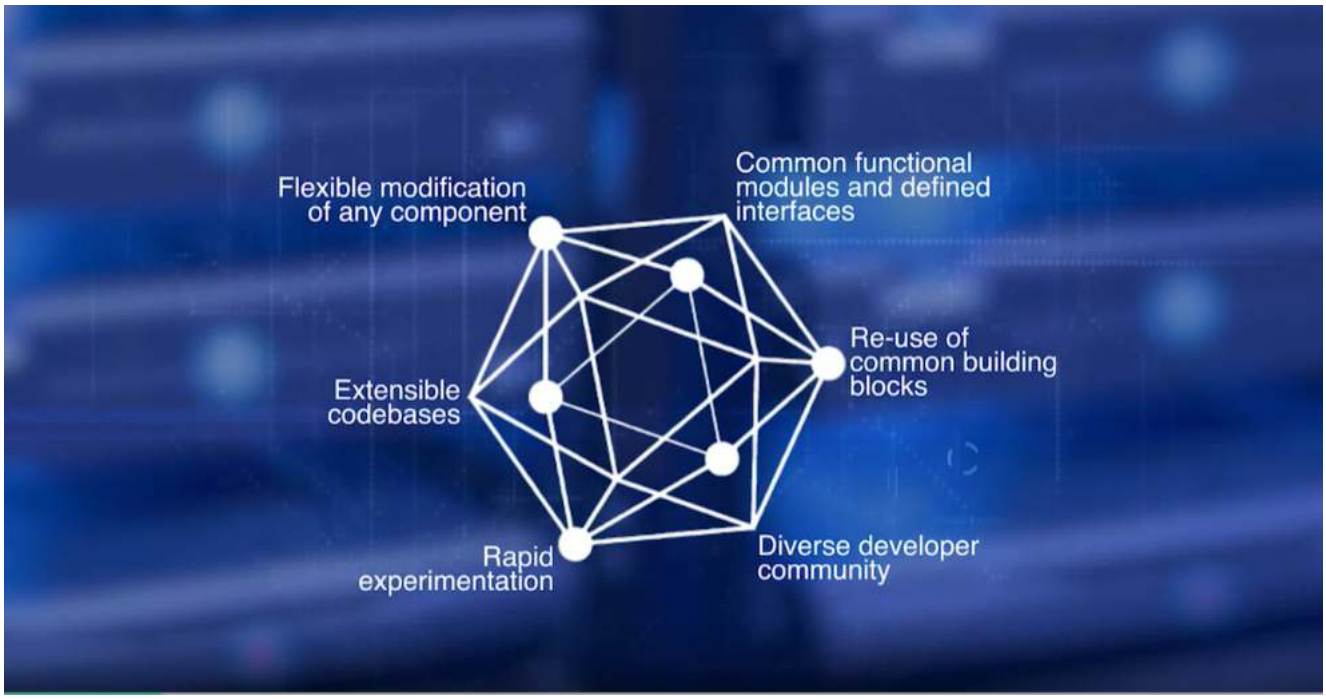
Thus, we have the characterization, the permissioned blockchain. Permissioned blockchain is also known as a consortium blockchain based on its common use cases in specific vertical business domain such as the automobile or food services consortiums. In this module, we'll explore the contributions of two major technology organizations. The Linux Foundation's Hyperledger Fabric, and Microsoft Azure blockchain as a service. The former is a permissioned blockchain platform, and the latter leverages its popular Cloud offering to allow users to stand up several blockchain platforms. These are indeed diverse technologies. Let's go explore.

On completion of this module, you will be able to explain the goals and working of Linux Foundation's Hyperledger project, explain the architecture and working of Hyperledger Fabric, explore tools and application

development for Hyperledger Fabric, and provide a high level overview of Microsoft Azure's blockchain as a service.The Linux Foundation initiated the Hyperledger project in 2015 to promote cross industry collaborations. The goal was and remains to bring together stakeholders, technology providers, and developers to advance the development and adoption of blockchain solutions. Some of us recall a similar situation when Unix became a popular operating system resulting in numerous non-standard commercial versions; Solaris, AIX, BSD, and so on. Finally, culminating in the open-source version of the Linux operating system. A comparable moment is happening on the blockchain front with a Hyperledger effort. This kind of collaboration among industrial partners and community of developers also helps in understanding the collective needs of the users and in minimizing the duplication of the efforts in design and engineering. It can also improve the pace of advancement especially in this nascent technology where numerous challenges are still waiting to be solved. There are many other advantages to this concerted effort, and you can read about them in the references we have provided in the resources section of the course. Upon completion of this lesson, you will be able to explain the Hyperledger initiative and list some implementations of Hyperledger Frameworks and tools. The Hyperledger is an ecosystem supporting not only blockchain protocol, the distributed ledger, and the smart contract, but it also supports the framework and tools for active engagement and collaboration of developers, businesses, and other stakeholders. The overarching goals are to promote the development of a safe, reliable, efficient, innovative, quality driven open-source components and platform to support enterprise adoption of the blockchain technology. The Linux Foundation, the members, and associates have established a home, and an environment for supporting these goals. The basic framework for Hyperledger is defined by the Linux Foundation. The member organization can then design their blockchain by extending this definition. This project also helps to build tools and frameworks and support educational and application development activities. It is expected that under the umbrella of well-defined specifications, blockchain modules created by various members will be pluggable into each other's technology environment.

1.1.3 Part 2

This Hyperledger logo depicts the goals of the Hyperledger Project. These include common functional modules and defined interfaces. Re-use of common building blocks, Diverse developer community, Rapid experimentation, extensible codebases, and flexible modification of any component.



It is two years since Hyperledger launched. And at the time of this recording, there are more than 191 members, four tools, five frameworks, and contribution to the codebase from all over the world. The five frameworks are Fabric, Sawtooth, Indy, Iroha, and Burrows, all incubated by the Hyperledger framework. Of the five frameworks, one has been out of incubation and into production. It is that of Hyperledger Fabric 1.0 from IBM.

Tools are essential for rapid prototyping and testing, and four tools are in incubation, Cello, Quilt, Composer, and Explorer. Later on, we'll demo the use of Hyperledger Composer. That provides a convenient development environment to assemble blockchain applications. Recall that Bitcoin protocol defined the transactions of a blockchain in terms of UTXOs, Unspent Transaction Outputs. An Ethereum protocol introduces smart contract and the concept of an account. The Hyperledger Fabric goes further than this in defining a whole business network with roles and assets and aligning the protocol closer to real-world application. Note that Hyperledger Fabric 1.0 is the only production framework out of The Linux Foundation at this time.

Let's explore significant differences between Bitcoin, Ethereum, and Hyperledger. In Hyperledger protocol there is no cryptocurrency. Hyperledger is designed for handling only business logic, such as the smart contract functionality. The smart contract code is called chaincode in Hyperledger. An even more significant difference between Bitcoin and Hyperledger is that Fabric is a permissioned blockchain, and unknown peers cannot join and leave the network as they wish.

Hyperledger aims to architect solutions for business to business, business to customer, rather than any unknown, trustless, decentralized peers. Mining of the blocks is also significantly different since only designated validating peers will undertake this operation in Hyperledger. This feature helps in the implementation of the consensus algorithm PBFT, Practical Byzantine Fault Tolerance, versus proof of work of Bitcoin blockchain. Summarizing, we learned the goals of Linux Foundation Hyperledger Project and the need for permissioned blockchain. In the next three lessons, we'll explore the Hyperledger Fabric that is the first to mature from incubation to production version in Fabric 1.x.

1.1.4 Resources

About Hyperledger
Hyperledger tutorial
What is Hyperledger? Brian Behlendorf Explains the Linux Foundation's Blockchain Initiative
Hyperledger Fabric
An introduction to Hyperledger Fabric from The Linux Foundation
Microsoft And The Blockchain: MSFT's Big Projects

1.1.5 Quiz

1. What are Hyperledger frameworks used for? 1 po

- ◯ Hyperledger frameworks are primarily used for building public blockchains.
- ◯ Hyperledger frameworks are primarily used for building smart contracts for public blockchains.
- ◯ Hyperledger frameworks are primarily used for building permissioned blockchains for organizations.
- ◯ Hyperledger frameworks are used for only building smart contracts for IBM's blockchain.

2. The cryptocurrency for Hyperledger is called Hyperledger Iroha?

- ◯ True
- ◯ False

3. Smart contracts in hyperledger are called?

- ◯ Chaincode
- ◯ Smart contract
- ◯ Coded contracts
- ◯ Smartcode

4. At the time of this video recording, which of the following is out of incubation and into production?

- ○ Sawtooth
- ○ Burrow
- ○ Iroha
- ○ Fabric

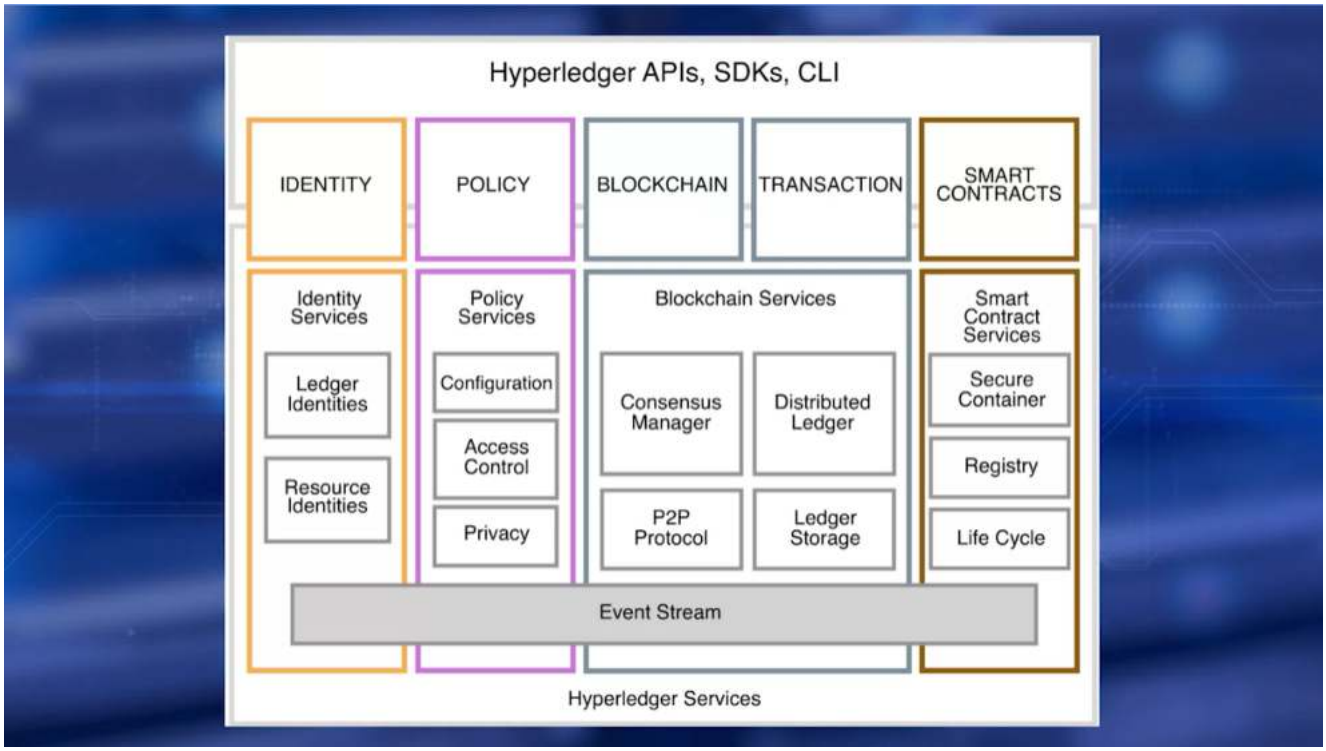
1.2 Fabric Services

1.2.1

Hyperledger fabric services. As explained in the last lesson, Hyperledger fabric is a permissioned business blockchain. Hyperledger fabric has changed many hands from its genesis at IBM, moving on to incubation at Linux Foundation, and is managed by many technical working groups. When we begin the exploration of a new platform, it's a common practice to begin with a whitepaper about the platform. We will begin with the whitepaper that provide the services-based description of the Hyperledger. On completion of this lesson, you will be able to list the services-based architecture Hyperledger framework, list the services and API offered by the Hyperledger framework, explain the various services offered by it. A popular way, and emerging technology, our business is explained is through a whitepaper. Recall, we refer to Bitcoin's whitepaper, and Ethereum's whitepaper in our earlier courses.

Of course, there's a whitepaper explaining the services of the Hyperledger framework. We have provided the link in the resources section. According to the whitepaper, the design of Hyperledger places heavy emphasis on modularity and configurability, pluggable modules. A Hyperledger blockchain application views its domain as

being made up of many interacting blockchains of various capacities. From the Hyperledger whitepaper, here is the services-oriented representation of the Hyperledger blockchain. The reference architecture specifies four different groups of services and the corresponding APIs for the applications to access them. Here is a list of services offered by the Heperledger framework, identity services, policy services, blockchain services, and smart contract services. We'll now discuss the concept covered by each of these groups. Identity services module manages the identities of entities, participants, ledger objects, such as smart contract.



In the case of fabric, a smart contract is called chain code. Policy services module manages access control, privacy detail, consortium rules and consensus rules. Blockchain services module manages the peer-to-peer communication protocol, the distributed ledger maintaining the global state, the global state replicated at many participants, the plugable consensus algorithm, PBFT or POW. The smart contract services module provide a secure and lightweight sandbox and moment for the chain code to execute. Only full nodes call the validating nodes include the smart contract services. This service also provides the secure container equivalent to Ethereum virtual machine, registry, and life cycle management functions. Java virtual machine that runs bytecode typically serves as a computational environment for the smart contract execution. APIs allow application programs to call into the underlying services. SDKs help encode development based on these APIs. CLI is a command-line interface for invoking these APIs for testing purposes. In this lesson, we reviewed the services-based architecture of the Hyperledger framework and the corresponding APIs. In the next one, we'll examine the components and their functions, and the operational details of the fabric.

1.2.2 Resources

Main three components of Hyperledger Fabric
Peer channel-based event services
Architecture Explained
APIs - CLI, REST, and Node.js
Blockchain Development on Hyperledger Fabric using Composer : What is Hyperledger?

1.2.3 Quiz

1. A popular way to explain an emerging technology is through?

   - ◯ Blog post
   - ◯ Research Paper
   - ◯ Press conference
   - ◯ Whitepaper

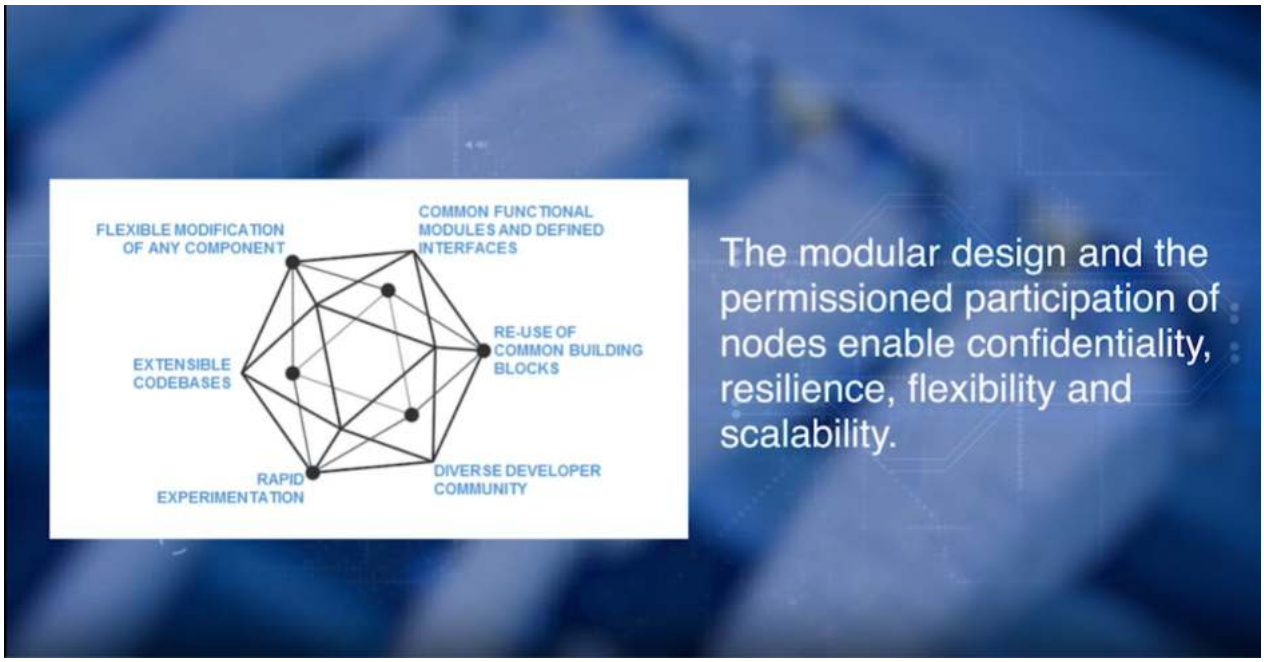
2. The hyperledger fabric framework only offers blockchain services

   - ◯ True
   - ◯ False

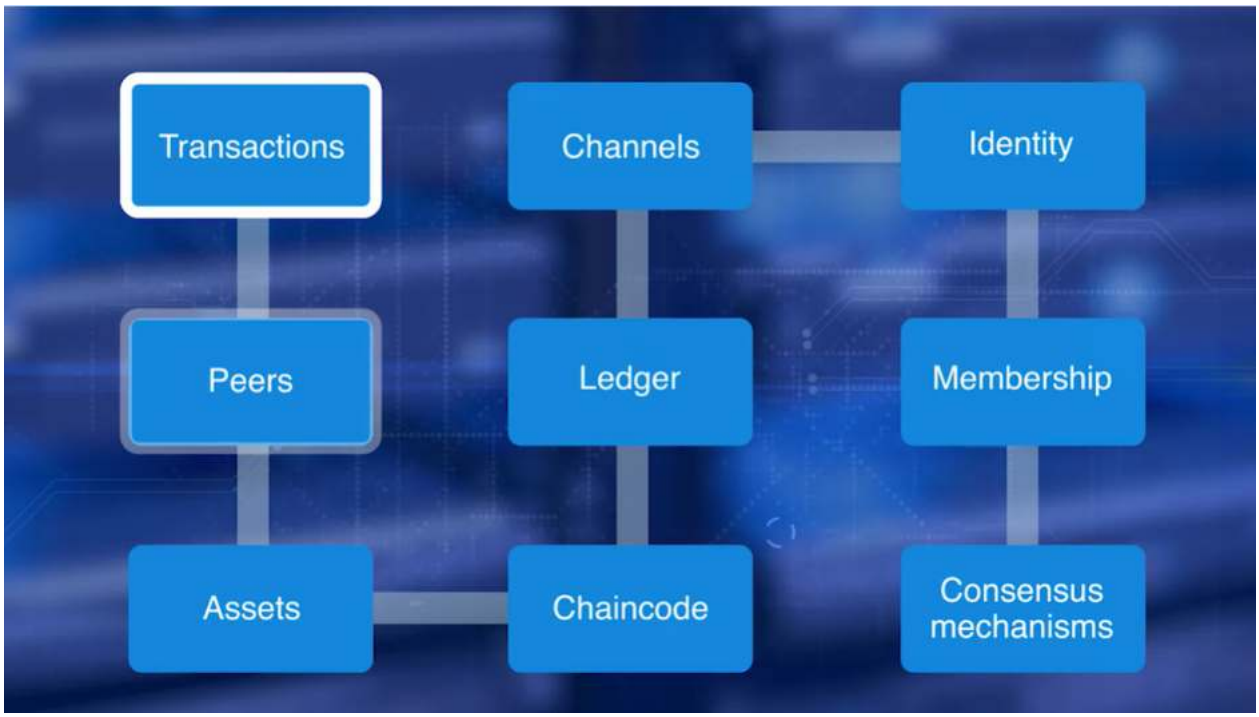
1.3 Fabric Model & Functions

1.3.1 Part 1

Since the Hyperledger white paper was released, many working groups have been formed. Of those working groups, the Hyperledger architecture working group has defined a layered architecture for various services that we discussed in the last lesson. The initial focus of the group is in the plug and play feature of the hyper ledger framework. That allows for various members IBM, Intel, Exit Direct, to implement their own version of a specific feature. The plugability enables an elastic and extensible architecture. Distinguishing it from the earlier Blockchain designs. The modular design and the permissioned participation of nodes enable confidentiality, resilience, flexibility and scalability. A must read documentation can be found at the link shown.



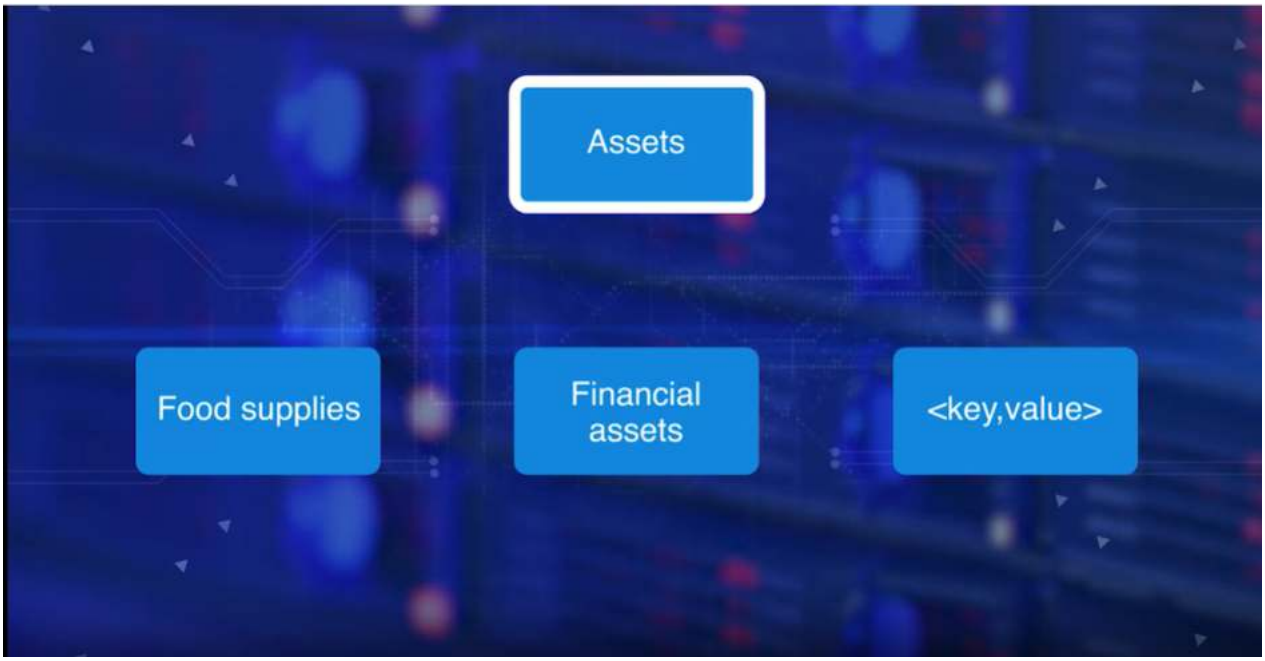
Let's explore the details. On completion of this lesson, you will be able to explain fabric model and its functionality. The fabric model consists of transactions, peers, assets, chaincode, ledger, channels, identity, membership and consensus mechanisms. Here you see an old view of the application that are enabled by Hyperledger Fabric. You see, multiple business blockchains each with its own channel, with its validating nodes, membership services for identity and membership management. Interactions between different independent business networks are enabled by confidential smart contracts. Transaction could be inside a single network or cross network. We'll now examine the various components in detail.



Peers are nodes that initiate transactions and maintain the state of the ledger. There are three types of peer nodes. Endorsing peers, receive and validate transactions, sign them, and return them to the creating application. They're called endorsers. Ordering peers collects signed transaction, ordered them into blocks and

send them to committing peers. This is also known as the ordering service. Committing peers receive the blocks created by the ordering service. Validate condition such as double spending and signature and then commit them to the ledger. Assets represent the tangible items of value that are transacted in the blockchain, for example, food supply and financial assets. Assets are represented in the program as key value pairs in JSON on binary format. Chaincode is a smart contract that defines a set of assets and provides the functions for operating on the assets and changing the states. It also implements application-specific rules and policies.

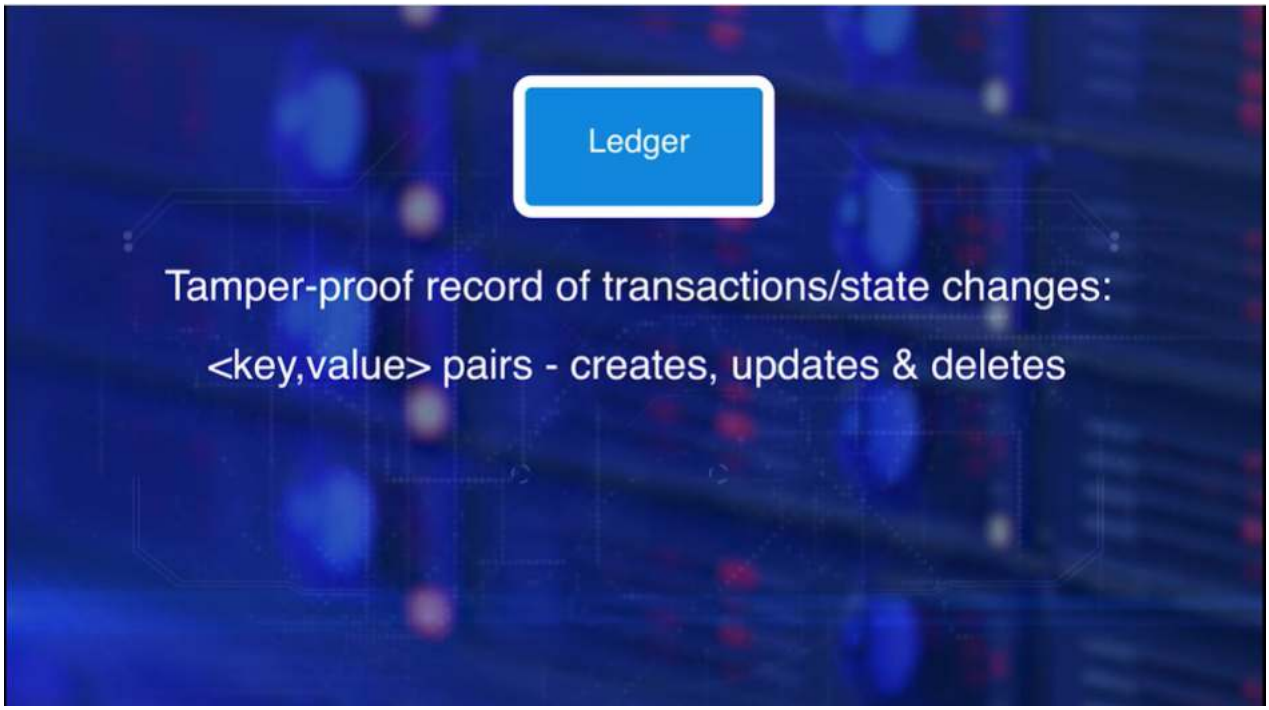
Function execution may result in state changes that are recorded on the ledger. The ledger is similar to other blockchain ledgers it is a tamper proof record of state transactions in the fabric.



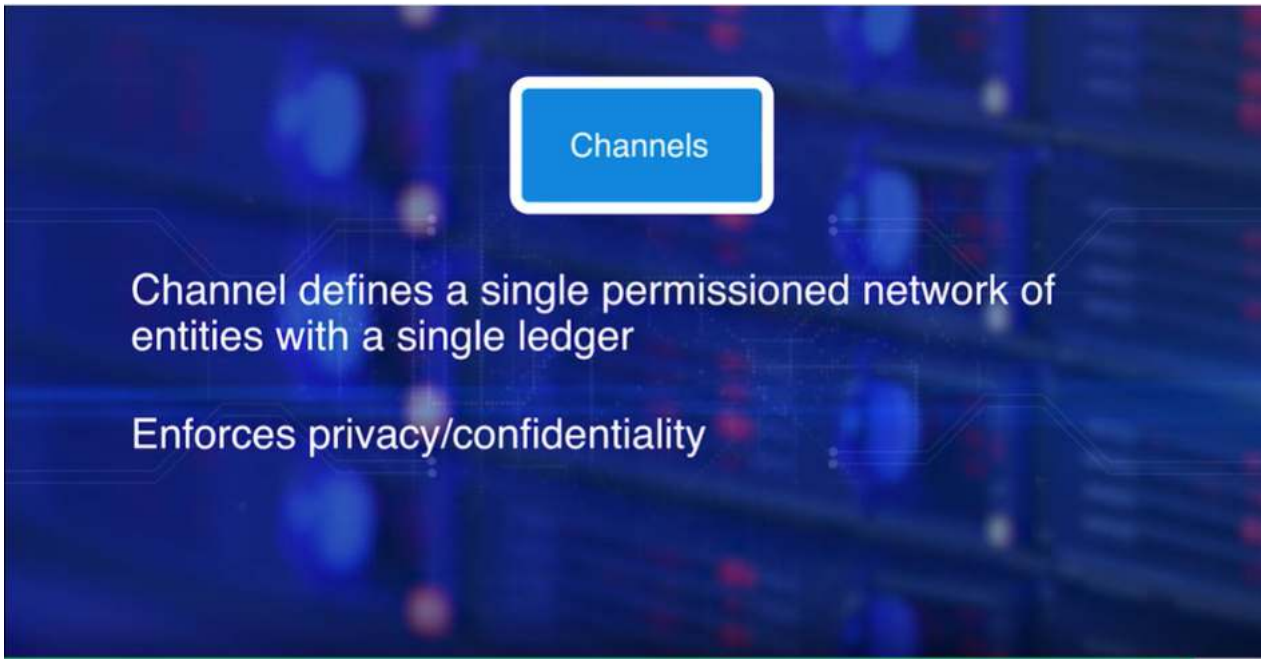
State changes are effected by the in vocations of chain code functions by the transactions initiated by

participants on the fabric. Transactions assigned and all the access control rules set by the permission services are enforced for a transaction to be executed and included in a block for recording.

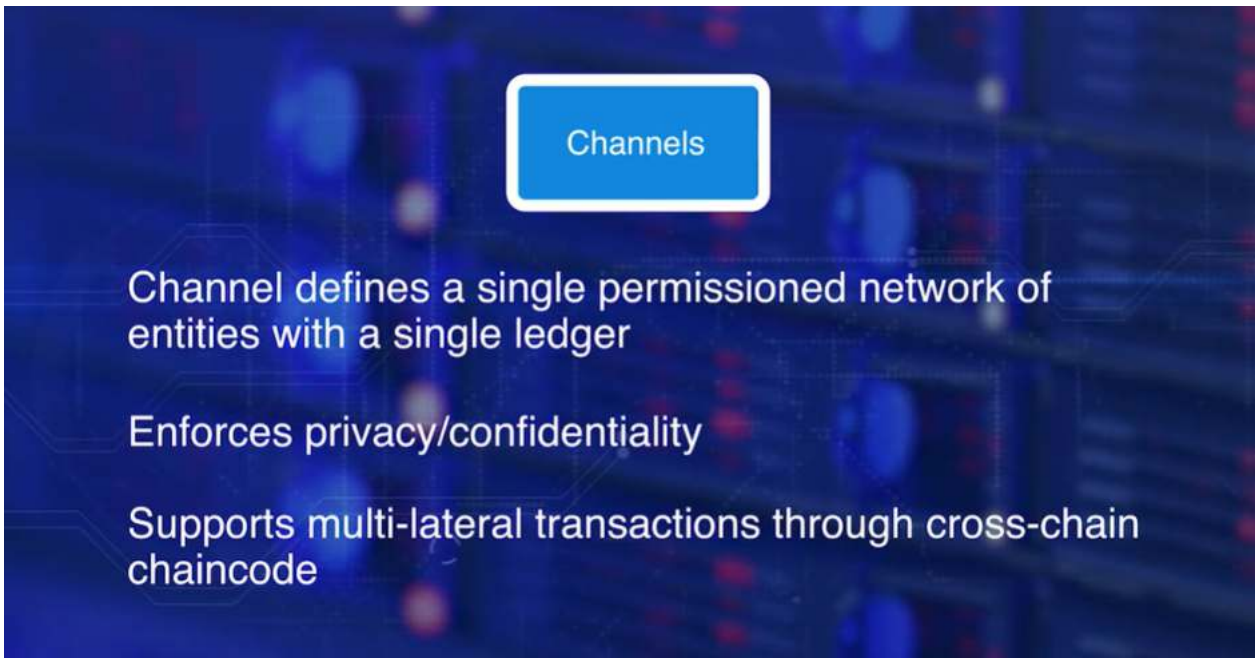
Each transaction results in a set of asset pairs that are recorded on the ledger as creates updates and deletes. Since a ledger is a key value store, it can be easily queried for later analysis and auditing Another element of the Fabric model is channels. Fabric enforces privacy and confidentiality through the channel concept. Channel defines a single permissioned network of entities with one single ledger for all its transactions and state changes.



Channel provides segregated fabric for a group of entities to transact privately. Even within the channel, the data and transaction confidentiality can be achieved by out for skating and by using cryptographic methods.



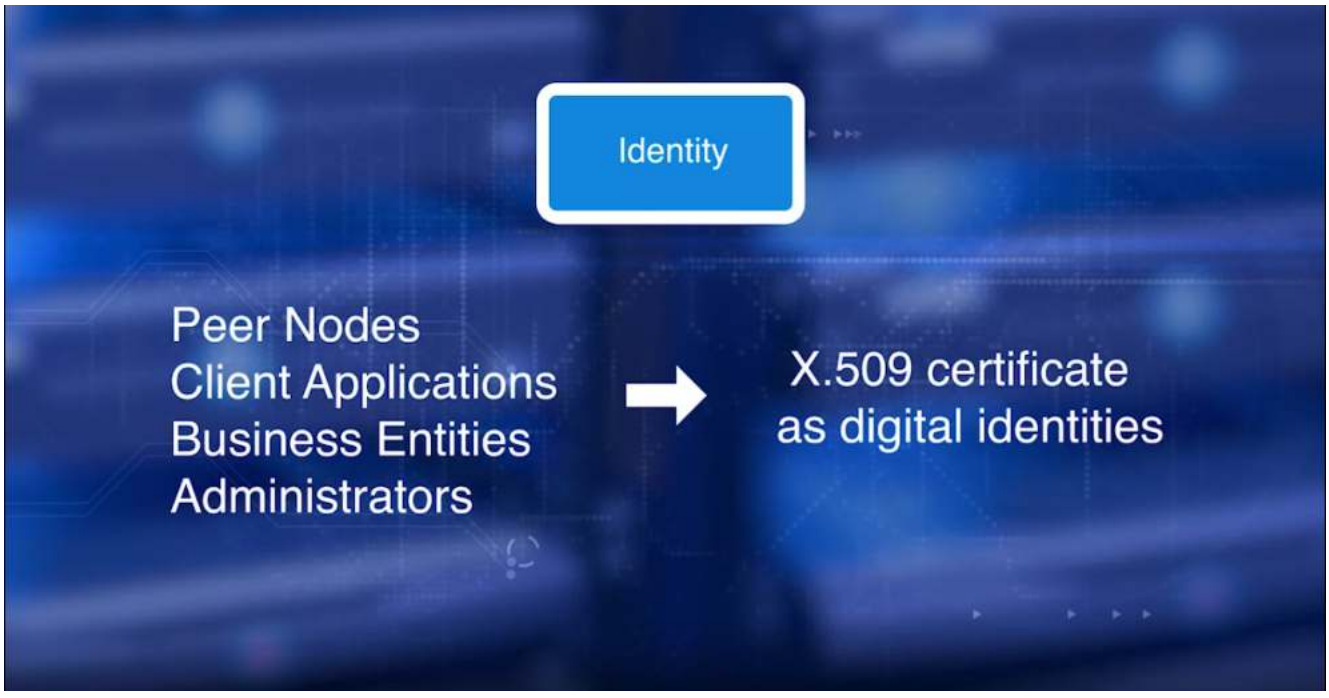
Channels also provide the ability to support multi-lateral transactions among competing businesses and regulated industries through cross-chain chaincode. In this case, each private business consortium operates on its own private network with cross-chain transaction controlled by policies implemented in the chain code.



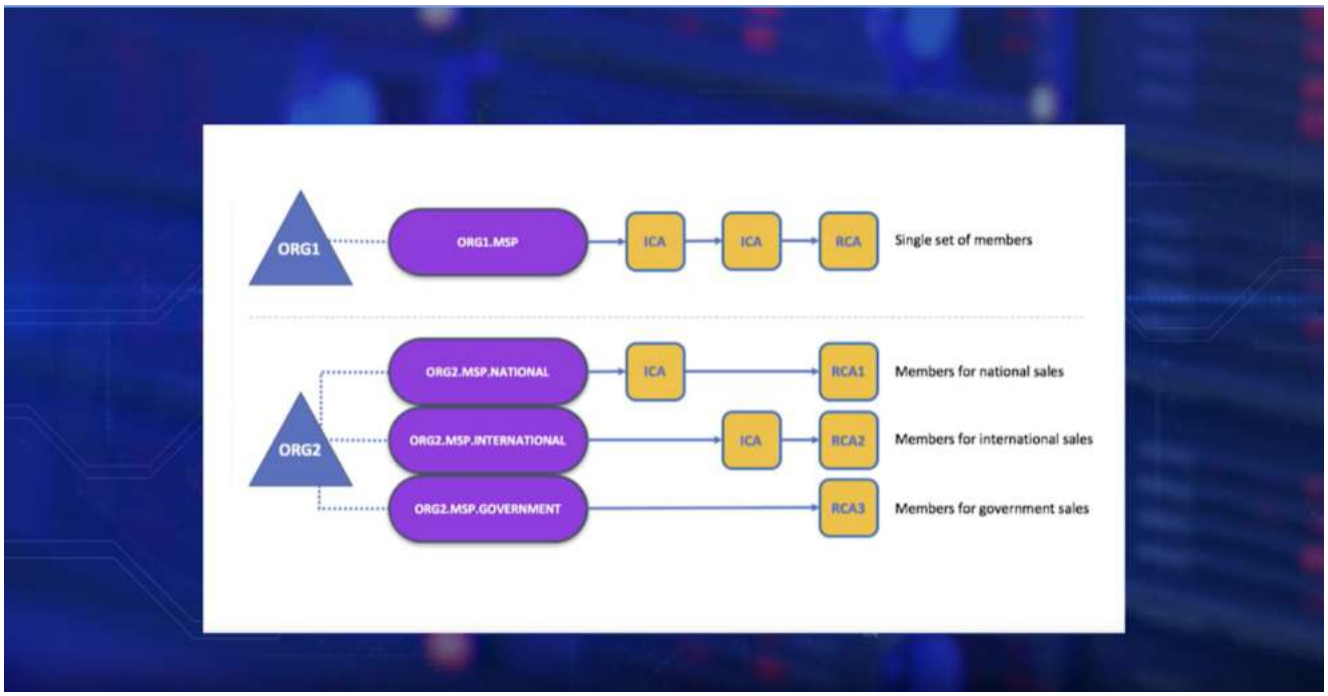
1.3.2 Part 2

Now, let's move on to the membership service provider component of the Fabric. Recall the public blockchain such as Bitcoin, are permissionless and unknown entities can join and leave as they wish. Entities wanting to participate in a Fabric network, enroll through a trusted Membership Service Provider or MSP. An organization manages its membership, and the roles of the participating entities through an MSP.

Participating entities should have verifiable identity. For realizing trust, the default implementation of MSP uses a X.509 certificate as a digital identity. The peer nodes, client application, business entities, and administrators need uniform identities in the Fabric. Each are assigned an identity that is an X.509 certificate.



These identities determine the role of the entities, and the permissions they have for accessing the resources in the blockchain network. This leads us to the next question: Who in MSP assigns them identities? There is a root certificate authority and intermediate certificate authorities managed by the MSP.



Here you see two organization. ORG1 does not have any sub-organizations. ORG2 has two sub-organization, each with root certificate authority and intermediate certificate authorities providing identity certificates. We are now moving into the final element of the Fabric model, the consensus model. Consensus at the high level is the agreement on the next block of transaction to be added to the chain, and the extensive validation and verification of the order and the correctness of the transaction including double-spend and other conditions. Fabric allows for 'pluggable' consensus model.

That means, depending on the characteristic of the channel, the minor of the next block is decided by a round robin policy or a Practical Byzantine Fault Tolerant or anything in-between. If the channel is deemed to be highly trustworthy, a simple consensus model will do. If it is a business-to-consumer channel where there's a potential for malicious operations, even a Proof of Work algorithm can be plugged in as the consensus model. Now, let's explore the functionality of the Fabric model. At the core, Fabric provides the Distributed Ledger Technology and through the various components we discussed in its model, it provides these functionalities; confidentiality and privacy through building a permission blockchain network among trusted business peers.

Fabric also offers segregation through the channels, identity management through membership services. It offers efficiency to various types of peer nodes operating in parallel, validating, ordering, and committing the transaction to the chain. Chaincode functionality helps in implementation of application-specific logic, and configurability through modular architecture of the Fabric. Allows for pluggable modules for anything from consensus model to the internationalization and policy modules for various regions of the world. This lesson examined the various components of the hyperledger Fabric model, and its application model that allows for multiple channels and interactions among those. In the next lesson, we'll illustrate the development of a hyperledger Fabric application.

1.3.3 Resources

Hyperledger Fabric Model
Hyperledger Fabricdocs Documentation
Hyperledger Architecture, Volume 1

1.3.4 Quiz

1. Ledger in the Hyperledger Fabric is a record of state transitions, true or false? **1 point**

- ○ False
- ○ True

2. Assets on Hyperledger Fabric represent tangible items of value that are transacted on the blockchain, true or false? **1 point**

- ◉ True
- ○ False

3. Within a channel, there is no available method to keep the data confidential from participating entities. True or False? **1 point**

- ○ True
- ○ False

4. What does MSP stand for in the context of Hyperledger Fabric? **1 point**

- ○ Member Security Protocol
- ○ Metadata Storage Peer
- ○ Member Selection Protocol
- ○ Membership Service Provider

5. Hyperledger Fabric only allows Proof of Work consensus to be plugged in to ensure a high degree of trustworthiness. **1 point**

- ○ True
- ○ False

1.4 Composer

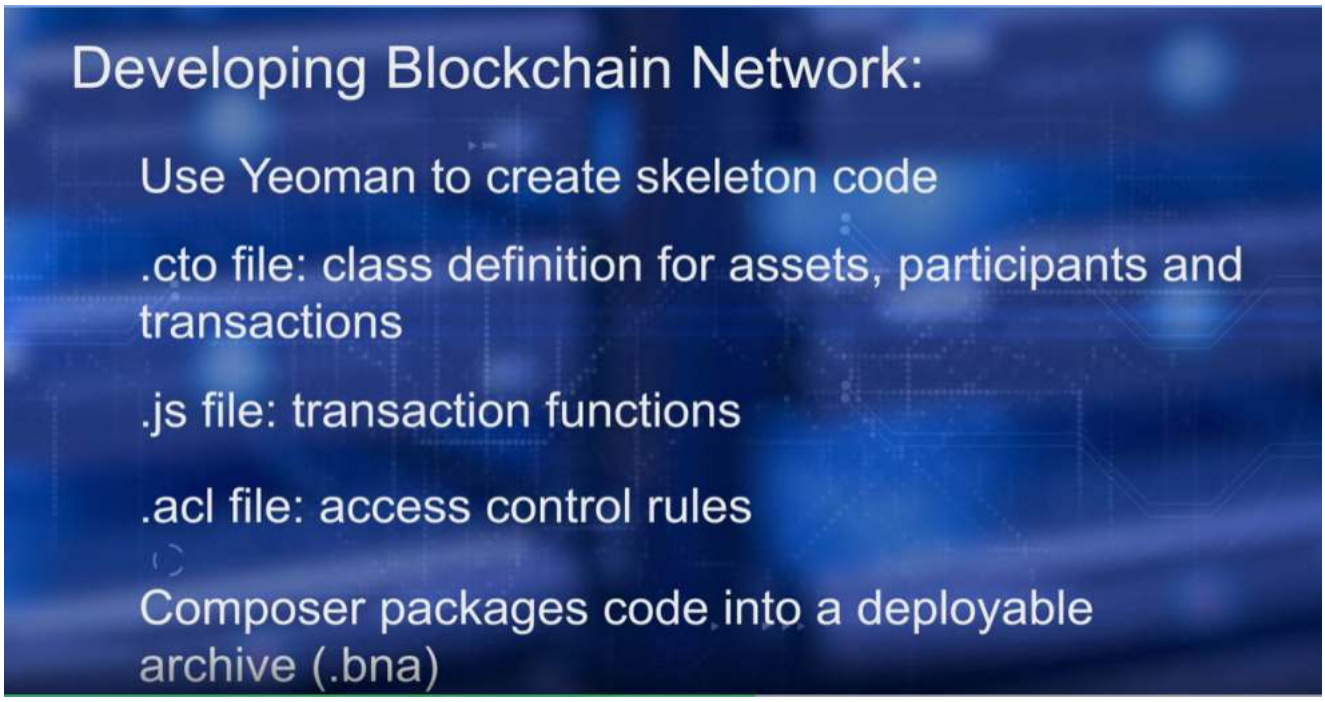
1.4.1

Module one, lesson four; Application development using Hyperledger Composer. The Linux foundation Hyperledger project not only defines block chain frameworks by various partners such as IBM and Intel, but also specify several tools for developing applications. In this lesson, we'll examine some of the tools. The human tool with the command you, that can be used to create the skeleton Business Network, the composer modeling language that is used to define assets of business application, and the composer tool for deploying their application. The composer tool is similar to the truffled tool we used in course three for dApp development. Let's explore the business network development process with these tools. We provide an overview of the techniques by using the tools in the development of a simple business blockchain network. Steps in development are; use human tool to create a skeleton code for the business network. In the CTO file created, you will find the class definition for all assets, participants, and the transactions in the business network. This file is written in Hyperledger Composer modeling language. In the Javascript file are the transaction functions. In the access control file (ACL), are the basic access control rules. After updating the CTO JavaScript and ACL files, the entire directory is packaged using the composer tool, to package the code into a deployable business network archive. Composer is used in setting the credentials, running in their server and deploying the application. The composer playground tool is executed to allow interacting with the deployed business network. Do you wonder about the origin of the type.cto for asset definition file? It comes from the word concerto. That was the earlier name of the composer tool. We'll illustrate these steps, by developing an application for commodities trading. You will define the files needed for setting up the project using the composer tool. The directory structure of the project files are shown here. In the demo, we'll develop the application and the interface to add and update the details of the commodities. Any transaction related to the trading, will be recorded on the ledger. In the last few lessons, we discuss the services oriented architecture of Hyperledger, components and functions of fabric model, and the application development on the hyperledger fabric framework, using Hyperledger composer tool.

1.4.2 Demo (Part 1)

Hello everyone. In this video, we'll demo Hyperleger Fabric and related tools and Composer or Explorer and Playground. This is a very basic demo and this just gives you an introduction into the platform. If you are interested in learning more about Hyperledger, please check out a very nice course on Hyperledger offered by Coursera and IBM. We will offer the demo in four parts: first, creating the business logic and the network in an archive file; second, deploying the project archive using Composer tool; third, exploring the API using the reservoir at port 3000 and Angular app at 4200 port; and a Web interface at port 8080 using Composer Playground. Let's start. First thing that I want to do is I want to install the Fabric infrastructure.

I do have saved all the commands in the file, and we will provide a detailed list of what I'm doing even though I may be skipping through couple of steps. The detail list of how you can do this yourself will be provided though it is not required. It may require you to download a VM, virtual machine image about about three GB, and if you want to do that, you can go ahead and do that or watch the video. First thing is I want, I'm in a base directory of the virtual box and I've installed it and I've imported it, and now I'm going to go into setting up the Fabric server, right. Let's go in there and then I'm going to start them, Fabric infrastructure. Okay. This is the blockchain based, and I'm also going to create some credentials for an admin right after that. So, since I had Fabric running before, it is stopping the previous version and then starting a new one, and I'm going to create- the next thing I'm going to do is I'm going to create some credentials. All right. So, now, the next thing is create and you can use tab complete to run the commands. I created a credential for an admin.



All right, we are ready. Now, the next step is setting up the business network where the code about your logic will be created. I've already created a tutorial network as it was provided by the Hyperledger tutorial, but I'm going to show you how you can create the initial template. This is like your truffle in it. So, I'm going to say we're going to use a very nice scaffolding project initialization tool called UA Apache, and then I've given you all the instructions in the instructions that we have provided along with this demo, and namespace, and it's asking for namespace as my network. I'll just say example, mynet, and it's asking for generated empty network. Let's say no. In response, it creates a whole lot of files, and as you can see, these files should be available in mynet, and this is the template for you that the human tool created and according to the rules specified by the

Hyperledger Composer and Business Network. The ones that are important for us are the models where the logic is defined. Library- inside the library there's a file where- logic.js where the transactions are defined. This is a permission network and so the permissions are defined here. These are the only three files that we're going to be affecting. As a developer, these are the three items that you will be adding to create the basic Business Network. So, we already created that in the tutorial network, so I'm going to go into the tutorial network. Let's look at the models. I have here model and this model has an asset commodity. It has an asset trader, participant trader. So, there is an asset, which is a commodity because this is a business network for commodity trading. It also has a trader who is the owner and identified by the ID, and the first name and last name, and the transaction where the commodity is transferred to a new owner. That's all very simple. Then let's look at the logic where they're defined, where the transaction is defined. I'm going to say logic.js and you can see the transaction defined here. Okay. The third item is the permissions for who can access what and it's a simple one that we're going to do. We have- I can permit all the access and we want- so, we're going to do a demo after all, so we're not restricting the access to it. All right.

1.4.3 Demo (Part 2)

Now that we have created the artifacts required for the project in terms of the model and the logic and the permissions to access the various items, assets and participants and transactions. We can go ahead and create the archive that can be deployed on the fabric. Once again, we use composer tool for this, composer archive is the command. Create using the current directory and place the archive in the current directory. And I'm using a new network that we created, but actually I pre-created all the elements in another network that we are going to move into. Okay, and you can see, this composer archive command created the business network archive type bna. I'm going to go back to the tutorial network that we have already created for you, that pre-filled with realistic trade commodity and trader information. And so in this network I'm going to use, I've already created as you will see, the business network archive, and I'm going to use that to install this, I'm going to install that on the fabric that we deployed. Okay, fine, that is again we use composer tool and we use composer install, we have installed the new business network that we created, now, let's start it. As you can see the business network that we created has been deployed and network admin branches has been established. Now we can start the rest server, once again the instructions give you clearly the steps involved in this, even although I'm missing, maybe missing a couple of steps here and there. You should be able to follow through and run it through. So creating an rest server is and once again we're going to use composer-rest- server. Okay, and it will ask for a few questions, and we want to use admin@tutorial-network, that is the network that we are using. And never use name spaces, I'm just going to use my arrow key to go down, keyboard arrow key, never use name spaces. And then rest of them and then specify, usually the correct options, default options are given in uppercase, select them, No, and it says API. You can say no, and yes that is the default option, pick up all the default options and, it'll go through and install the rest server. The rest server exports the API for your business network archive for other applications to develop on. So let's go to our browser and open up local host. And the 3,000 That should show you the API that is being exposed for. Again, here, you can see Explorer Tool coming into action, the commodity is one of them. And these are all the API, rest API exposed for commodity system, and I'm just going go back to trade, and the trader information, the participant information. These are all the rest API exposed applications to use, external applications to use. Now that we have done that, we are going to further develop an angular interface, angular app. Now that we've deployed the fabric on top of a business network using the composer tools and also a rest server which exposes API, let's see what the composer playground can do for us. It exposes a whole lot of tools for you to look at and review add and edit the business network and so on, so it's quite a versatile tool. And I'm going to start off with that and then go outside the protocol, hyperledger protocol, and start an angular app and also look at the angular app. I think the playground will give you much more impressive interaction field than the angular app itself. Just for completion's sake I'm going to do the angular app also, I'll do the playground first and then the angular app. So composer once again just like init truffle, truffle init and truffle commands compose our playground, we'll set up the web interface at 8080 for you, and you can see that. And this gives you a lot of things to play around with, that's why it's called a playground. Let's wait for a few minutes, and as you can see, and it gives you about a Readme file and the model file so you can look at, ah-ha, very nice web interface you can go and edit there as you can see. Anything you want to add to the business network you can add, and the script file that we added for the transaction is also here. And you can add on any more skips if you want, and then the access schedule. Very clearly the playground gives you the ability to web access these components and edit them and add them. These were the ones that we used in order to create the business network archive. All right, let's see what else can we do, we can go into the admin. In admin, my business network, and connect now

to the business network. Okay, and we did that, and then let's go into the test

1.4.4 Demo (Part 3)

In the test network, we already created a few commodities and also there are a couple of traders. And so, let's see whether we can add a trader and also add a commodity or transact a commodity. So, I'm just going to add a traitor. So, add a new participant. And I'm going to make the number of the traitor so that we can remember. So, I'm going to make it 6666 and I'm going to, first name is, Lolla and Sykes, something like that. And I'm going to say Create New and you will see that Pending here. I think it went through quite quickly. So, otherwise, you would see a little note or notification that it says it's pending. So, if there are some access controls, and things like that. So, we've got a new trader here and I'm going to go into commodity. I'm not going to make, you can always add the commodities just like we did there. And you can see that this commodity is owned by 2, 3, 4, 5. So let's see whether we can transfer this commodity and you can see that we added a participant on the chain. So this is the chain that keeps track of all the transaction and you can see the hash there and the timestamp. And the historical record, you can see that. And then I'm going to submit a transaction. In this case, into the commodity number, I think we have one PQR and I'm just going to make the commodity PQR. And I'm going to transfer the ownership to Lolla, that we created just now, so I'm going to make it 6666 and submit. And so you will see that it's going to take some time because it's pending. Yeah, you can see transaction was submitted successful, so sometimes, you will see pending because it me takes sometime to verify the credentials and so on. And you can also go back, yeah, here you go, the trade was done and lets go back and look at the commodity and you will see that this particular commodity has been transferred to Lolla Trader number 6666. This is a simple test and you can play around with this after all the playground, you can see all the transactions, you can read the traders, and so on. That is the very simple demonstration of the playground. I'm going to stop here, and then let's go back to do the last part of the demo where an Angular app can be created as a web app to transact or to view the Blockchain itself. In order to create the Angular app, once again, you're going to use the Yeoman scaffolding tool and Hyperledger Composer, Angular as a generator, as you can see on the screen here. Yeomanhyperledger-composer-angular is the project type and I'm going to hit Return. And it'll go through and it'll create, it'll ask you for a few questions and you can take the default answers. Yes, I want to connect the running business network, and Angular app. Yes, author name, Apache, that's fine, and then I would say. Take the, mostly all the default answers, I'm using the up arrow keys and then it creates. It just goes through a set of install that is required. And, once it starts, once it completes, go the app, Angular app directory and start mpm start to start the Angular, to the simple server, to support your Angular app. And then we can go interact with the Angular app using the web port 4200 where it is found. Takes a few minutes, be patient. It's installing all the Node.js item that I required. All right, it's done. Well, let's see where we are, and I'm going to go into the angular app, and, npm start Sober and now it is ready to go. I'm going to go in here and I'm going to go into 4200, may or may not be ready let's be ready for that, 4200, Some few minutes maybe it's still going on, okay, it's still going on, it can take a few minutes, so let's go back up there. It's trying to connect, yep, we should be able to see it now. There you go, that's your angular-app. Again, assets, commodity, it's possible that you could add, you can see what we created there now we are looking at it. So, now you can create an asset here much easier because we have very nice template here. Trading Symbol is abc and description is TV company or something. And the main exchange is NYSE I don't know. And quantity is 1,000. You can see the angular provided you very nice interface for you to create and then it will show up there instantly. There you go, that's your, abc. And the participants, you can see trader and we have three of them, Lolla That we created in transactions. I know we did the transaction trade. There's no interface created but in case you want to do it, you can do it. Okay, so this is all the, tools that the composer, and the explorer, and the playground and the Hyperledger fabric, and under the umbrella of Hyperledger fabric. Please do test it yourself, okay? There is a VM provided, the testing is optional. But when you have some time, this pre-installed VM should help you learn about Hyperledger by doing it and going deeper into creating Hyperledger block chain frames.

1.4.5 Resources

Hyperledger Composer
What is Hyperledger Composer from The Linux Foundation with Dan Selman
Hyperledger Composer Modeling Language
Model and test your blockchain network

1.4.6 Quiz

1. What is the main tool used for building business applications in Hyperledger Fabric?

- ◯ Sawtooth
- ◯ Cello
- ◯ Composer
- ◯ Quilt

2. What was the earlier name of the Composer tool?

- ◯ Confetti
- ◯ Consento
- ◯ Concerto
- ◯ Cornetto

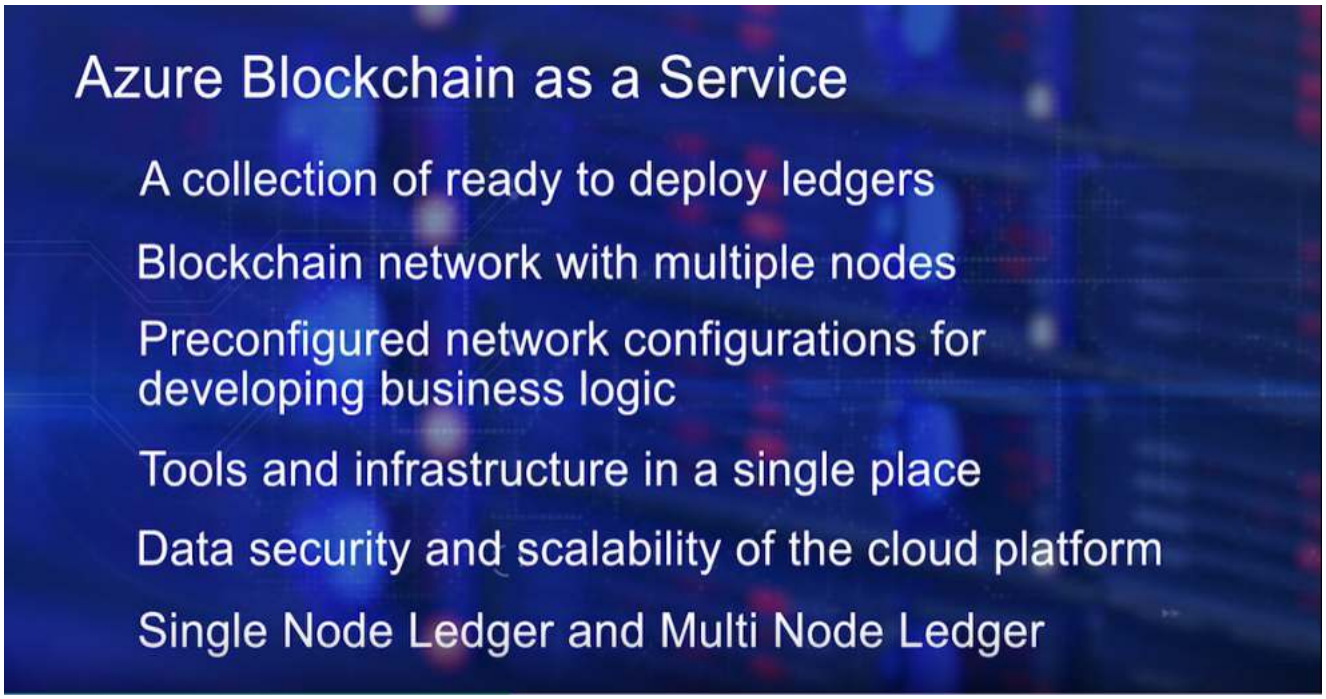
1.5 Microsoft Azure

1.5.1

The blockchain innovation has not left any industry untouched. In this lesson, we'll explore the contribution of the tech giant Microsoft to the blockchain movement. Microsoft is developing its own platform in CoCo blockchain in cooperation with Ethereum and financial companies. Simultaneously, it has also created a framework for hosting different industry blockchain in its cloud offering. Let's explore Microsoft Azure's blockchain as a service model. On completion of this lesson, you will be able to discuss the vision of Azure's blockchain effort. Explain the building blocks of Microsoft Azure blockchain as a service, and list the various platforms supported by Azure. Let's start with the vision. Microsoft Azure's top level vision is to accelerate blockchain deployment. It does this by offering tools and development environments to rapidly deploy end-to-end solutions. Providing integration capabilities through plug-and-play among multiple platforms. Enabling rapid deployment of ledgers and network and provisioning the blockchain on the cloud or at the local infrastructure of the users. Azure blockchain as a service feature includes: a collection of ready to deploy ledgers, a blockchain network with multiple nodes with hashing, mining, the consensus among the nodes and the distribution of replicated ledger to all node, preconfigured network configuration for developing business logic, tools and infrastructure in a single place, data security and scalability of the cloud platform, and Single Node Ledger and Multi Node Ledger. Azure blockchain offerings

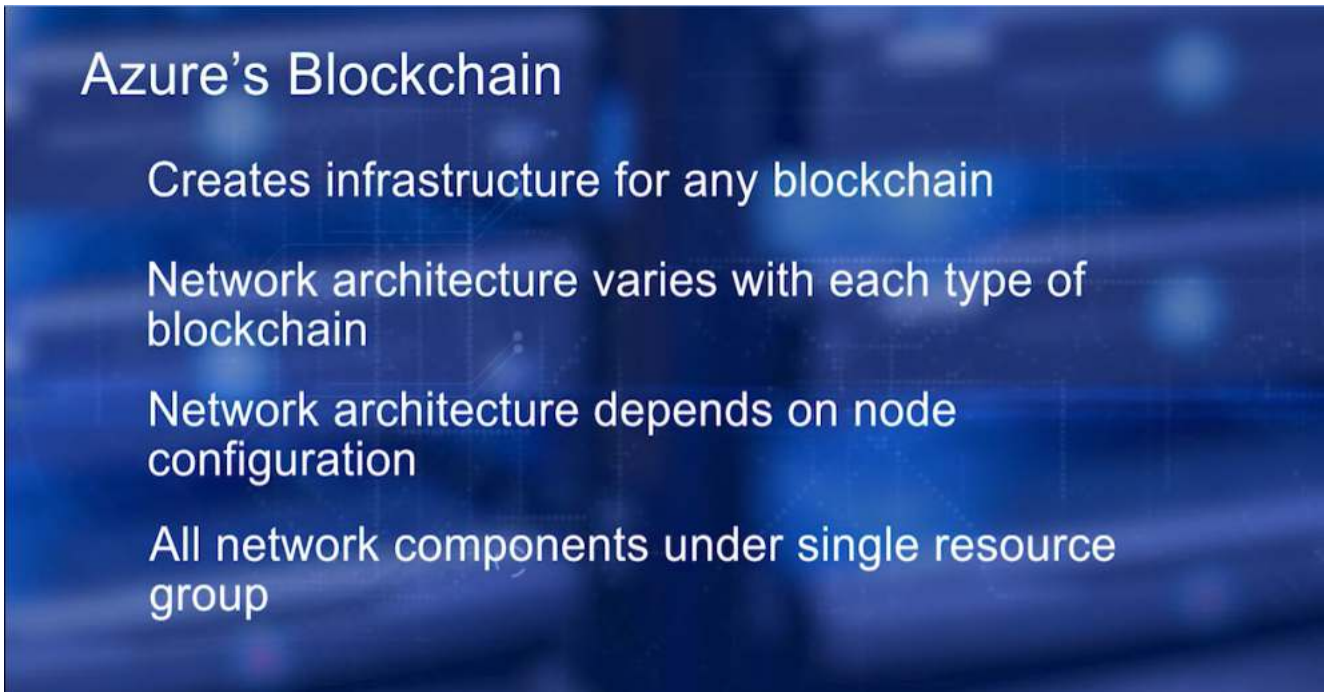
at the time of this recording are Ethereum and Enterprise Ethereum, Hyperledger fabric, R3 Corda, Chain and Quorum. Let's look at Azure's Blockchain architecture.

Azure creates the infrastructure needed for any blockchain platform and maintains the network for each. The network architecture varies with respect to each blockchain platform. Ethereum would have a minor and non-minor nodes, whereas R3 Corda will have notary nodes and participating nodes. The network architecture also depends upon the node configuration you specify. All network components will be under single resource group.



How will you use Azure blockchain as a platform service? First, you need to choose the platform. For example, Ethereum, Corda, or Hyperledger. Next, based on your application model you need to choose the type of ledger, Single or multi-node. Once decided, you need to enter the network configuration needed. Say for Ethereum, number of nodes, virtual machine storage type, virtual machine size et cetera.

After entering the configuration details, Azure will create a resource group containing the resources needed for your network based on the input configuration you provided. The connection, the maintenance, and the infrastructure will be provided by Azure. You can ssh into the nodes to interact and all the connected nodes will automatically be synchronized. In addition to these, you could spin up additional tools for development purposes like truffle IDE. In this lesson, we introduced Microsoft Azure's blockchain as a service to quickly set up and prototype several different blockchains. We discussed the vision and main features of the service at a high level.



We suggest that you explore some more on your own especially if your work environment is Azure. Summarizing this module, we began the discussion on Hyperledger framework as a project under Linux Foundation and the participation of the wide range of industries. We then discussed the services based architecture of Hyperledger and the various components of the fabric model.

We then illustrated application development using the composer tool. We also reviewed briefly the contribution of Microsoft to the blockchain revolution in the form of blockchain as a service. It is quite obvious that there's enormous interest in this emerging blockchain technology from various industries. Now is your chance to participate.

1.5.2 Resources

What is Microsoft Azure, Anyway?
Tour of Microsoft Azure
Simplifying blockchain app development with Azure Blockchain Workbench

1.5.3 Quiz

1. What is the main contribution of Microsoft Azure to the blockchain ecosystem? **1 point**

- ○ Public blockchain powered by Cortana
- ○ Private blockchain powered by Cortana
- ◉ Blockchain as a service

2. What is the top level aim of Microsoft Azure? **1 point**

- ○ Provide oracle services to public blockchains
- ○ Integrate Microsoft services into public blockchains
- ○ Provide cloud support to host Dapps
- ○ Accelerate blockchain deployment.

Chapter 2

Decentralized Application Platforms

2.1 Augur

2.1.1 Intro

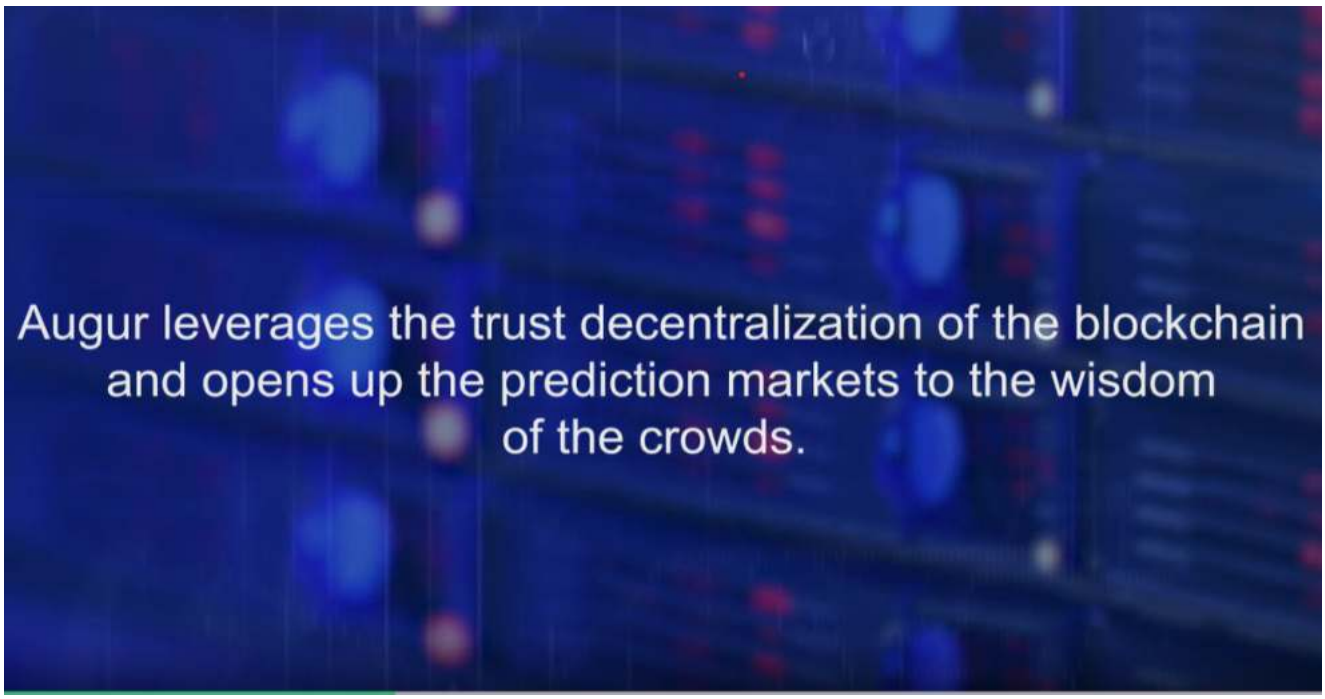
Course four, module three, Decentralized Application Platforms. We've been exploring various blockchain platforms, challenges in the vigorous development activities advancing the blockchain technology. Have you wondered about the practical application that illustrates the special features of blockchain technology? Namely, Decentralization, Disintermediation, and Distributed Immutable Ledger? Are their application that affect you and me? Yes there are. We'll explore two Dapp platforms; Auger and Grid Plus, which are both implemented on Ethereum.

Ideas and plans for these applications have been in the development side by side with Ethereum. Auger and Grid Plus are layers above the smart contract, each opening up the blockchain and the smart contract layer to the vast expands of vertical domains. In this case, it is the prediction markets and the energy retailing, respectively. More importantly, you can be a participant in these Dapp platforms, if you wish to do so. Let's venture out and see how the blockchain technology is applied and utilized in these two Dapp platforms. So, where does the Dapp platform module fall in our specialization roadmap? Recall material from earlier courses.

We've been developing these layers shown. Previously we've developed simple Dapps, whereas here, we'll explore a comprehensive platform for a vertical business domain and understand how smart contracts form a central role in these applications. Why did we choose these particular Dapp platforms? Auger and Grid Plus are chosen specifically to illustrate two very diverse uses of the blockchain. These applications have extended history, complex components, and operations. Provenance of transactions are critical in both applications. They are addressing real world problems, not hypothetical ones. They offer innovative models for other domains to adapt. On completion of this module, you will be able to discuss the architecture and working of Auger platform, list use cases for Auger, explain the architecture and working of Grid Plus platform, and explore how Grid Plus can be used.

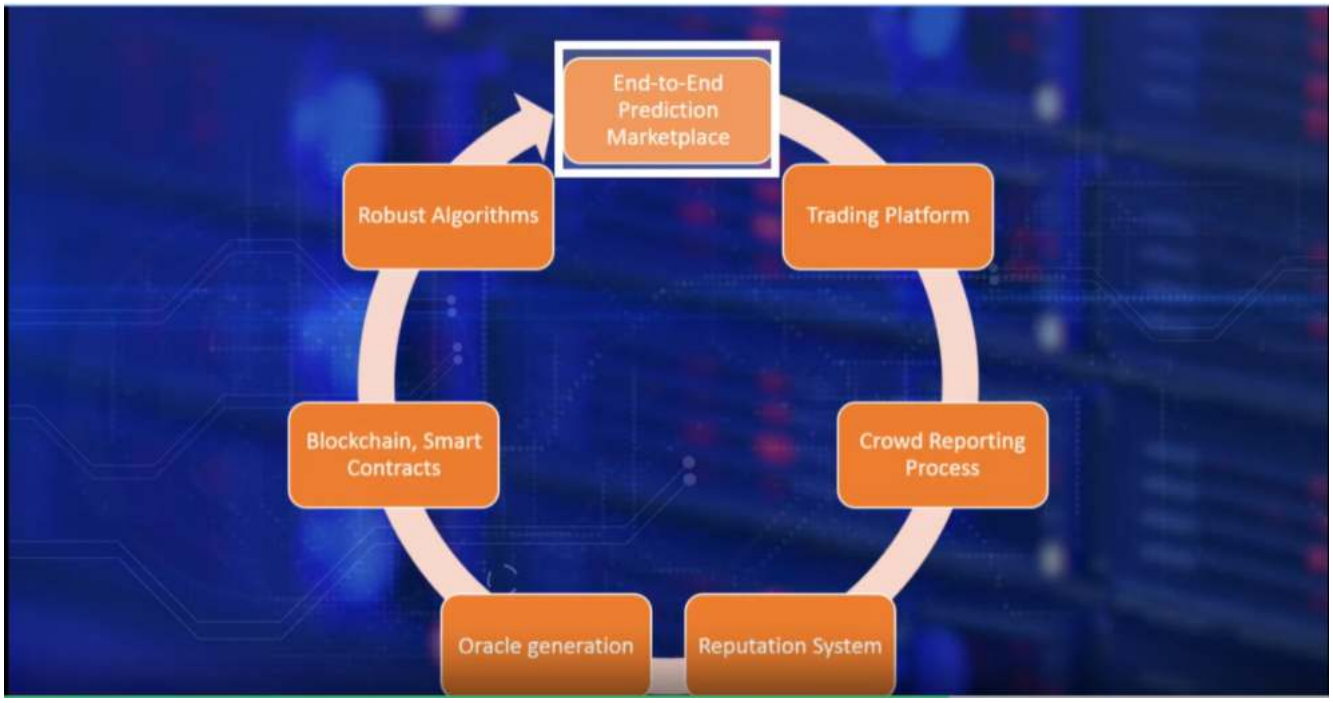
2.1.2 Part 1

The Decentralized Prediction Markets. Prediction markets are not new. They've been there since the dawn of history. Consider the web at Web 3.0 and Urban 2 Linux at 17.X. In comparison, decentralized prediction market would qualify to be prediction 108.X. They are that ancient. All the surveys you seen in the mail, the poles and the betting you may do on sports outcomes are all examples of prediction markets. The central idea of these markets is that you bet an outcome of an event. Though these are in existence for a long time, most of these prediction markets are centralized. There has always been a ledger, but traditionally, it has been localized to an organization. Decentralization with an immutable distributed ledger of the blockchain throws open the prediction markets to enormous opportunities.



In this lesson we'll explore the details of Augur and how it uses blockchain technology. Augur leverages the trust decentralization of the blockchain and opens up the prediction markets to the wisdom of the crowds. This is indeed innovative. On completion of this lesson, you will be able to discuss the components of the Augur, and roles to the participants, explain the working of Augur Dapp and list use cases for Augur. What is Augur? Augur is a trustless, decentralized prediction market platform based on blockchain technology.

Participants can join and leave as they wish. That's why it is trustless. It is decentralized since its underlying technology is based on the decentralization feature of the blockchain. The participants are not bound by any central authority. In a typical mode of operation, a market creator post a market event for which she seeks the wisdom of the participants. Market reporters speculate on the outcome of this event. Once the outcome is known, the participants who forecast the outcome correctly, win a payback and others lose the money they put in it. The end-to-end process is carried out automatically by the Ethereum Smart Contracts. This is a very simplistic explanation. There are a lot more details to this in Augur implementation. Let's look into the details. What are some key ideas?



End-to-end Prediction Market, Augur facilitates an end-to-end process from creation to settlement of a prediction on anything, from an idea of project, proposal, stock, stock futures, pouts and experiment in others. This is set up by a smart contract. The next key idea is the trading platform. To give just a few examples, Augur allows decentralized global trading on any idea, a proposal and event outcome or the effect of a policy. Trading patterns can be used to forecast outcomes. Since all of these are decentralized, anybody anywhere can participate. The next idea is reputation system. Augur provides a platform for decentralized participants to speculate on an event outcome based on their belief, then the crowd reporting process. Augur features an incentivized model for materializing the wisdom of the crowd using its own token rep or reputation token. Oracle, Augur captures the wisdom of the crowd in a secure Oracle that could be accessed by the underlying smart contracts for resolving the outcomes. Prediction markets is indeed a complex system that's why we need an automated auditable smart contract based system to implement Augur in a decentralized context.

2.1.3 Part 2

We just reviewed the key ideas of Augur without going into the theory or the mathematics. Let's look at these ideas to get a picture of the ideal interplay of the Augur application and the blockchain features, the smart contract, immutable ledger, and the oracle. Let's examine how you can participate and what are the roles in the prediction market. There are three main roles or types of participants. You can be a market creator, who places the market event and sets the expected outcomes, pays fees and escrows, and establishes the rules, and designates the initial set of reporters. You can also be a trader, who places the bets on the expected outcomes and takes part in the pre-reporting phase of the process. Traders buy and trade shares that bet on the odds of the outcomes. The trading currency is currently Ether, that of Ethereum blockchain.

In the next role, you can be a reporter, who reports on the outcomes. Understand that outcomes do not have to be binary, as we all know. The reporter can be a designated reporter, or an open reporter, based on the phase of the process. The prediction process runs it's course. Do not expect just one round of crowd reports to arrive at a prediction outcome. Most of them may take years by nature of the idea. Outcomes may not be binary. It can be scalar with many possible outcomes. Take a look at the market events posted at this Augur DApp interface. For example, one of the markets is science, and the market event is how many drugs will be approved by FDA in 2018? Range of outcomes is between 0 and 30. Another market is space. The prediction event is will SpaceX land on Mars the end of 2018? Outcome here is binary, YES or NO. We explain the key ideas of Augur Decentralized Prediction Market, then we explain the different roles of the participants. Now, let's get a high level understanding of the process. Some of them may take years by the nature of the idea.

A stakeholder creates a marketplace. This results in configuration and deployment of smart contracts on the Ethereum network. Rep tokens are used in this process and Ether for smart contract executions. Trading begins as soon as a prediction event is posted. The designated reporting period begins. After the designated reporting period, a ten-to-two outcome prediction may be available. If not, the first prediction after this is assumed to be the predicted outcome. It can be disputed by the reporters by staking rep tokens. In the worst case, the dispute phase made it result in a fork. Augur protocol has robust methods to address forks and the disputes. Next, the prediction phase ends with a prediction. The event happens. The next phase of reporting begins. At the end of the reporting period, the collective reported outcome is a availble as an oracle. It is accessed by the smart contract to finalize the outcome and settle the trades and fees.

Now, let's look at the Augur web interface. You can see various markets from crypto to temperature. This screen also features a market event. Will Augur's live release happen by the end of 2018? At this time, 35% of the reporters believe so. You will know the prediction outcome when you're watching this recording. Can you tell the percentage now? Augur is built on top of Ethereum blockchain. And it is a public network, since it is indeed dependent on the decentralized traders and reporters. The Augur organization will manage the blockchain and provide the access to the markets through its webpage. A user can create a market, post market events, trade, and/or report. The trading token is Ether, and the reporting token is rep.

Systems already exist for financial market and sports speculation, so these are not use cases for Ether. It is highly relevant for science experiments, policies, agriculture, insurance, and the like. It is highly relevant to micro and macro level systems in economics. Augur automates the prediction market to a wide range of other use cases and may recognize significant value. Augur provides accessibility to these markets through web user interface and a blockchain ledger so that anyone can participate. It also provides a model for incentivizing good reporting and a participatory economy.

2.1.4 Resources

What Is Augur?
Augur white paper
Augur: The World's Most Undervalued Crypto Project
Augur in 2 Minutes

2.1.5 Quiz

1. The token used in Augur in the context of reporting is called ___

- ○ Eth
- ○ REP
- ○ Coin
- ○ Gas

2. Augur's REP Token is ERC 20 compliant. True or false?

- ○ True
- ○ False

3. What is the blockchain technology on which Augur is based?

- ○ Hyperledger
- ○ Bitcoin
- ○ Dash
- ○ Ethereum

4. Wisdom of the crowd is captured by a service called ____.

- ◯ Oracle
- ◯ Event
- ◯ Browse
- ◯ Mine

5. Augur is a private blockchain based decentralized app, True or False?

- ◯ True
- ◯ False

6. The payback to and from customers is handled automatically by smart contracts. True or False?

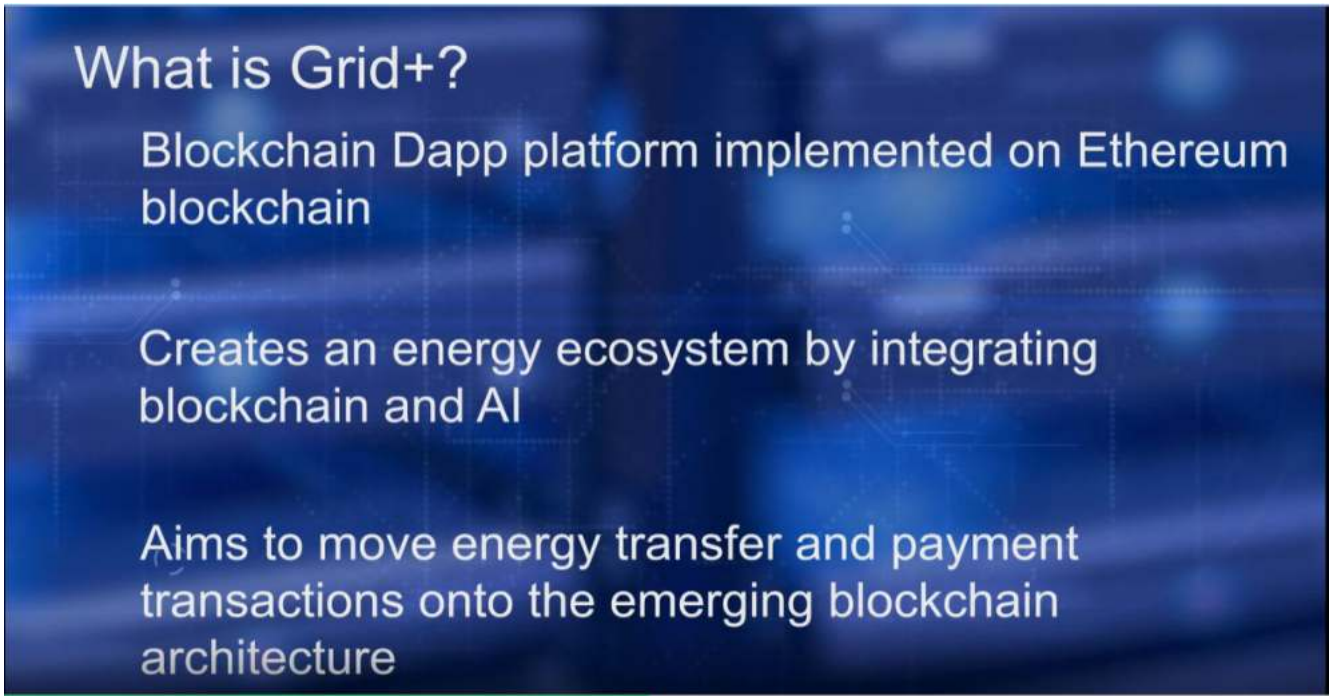
- ◯ True
- ◯ False

2.2 Grid+

2.2.1 Part 1

The energy grid of the future. In the past 10 years, how many times have you seen a solar panel installation on a personal dwelling, a windmill on a country farm, advertisement for geothermal energy possibilities in a new housing development? These are called distributed energy resources or DERs. You're witnessing the proliferation of DERs. You're indeed seeing energy production decentralization. We're also witnessing the growth of major end-use electric devices. A significant one is electric car, another is a major development in Internet of Things, IoTs and the advances in smart homes. How do they manage the diverse demand at the user level, to the decentralized production enabled by the abundance of DERs?

In the last lesson, we added a production layer on smart contracts. In this lesson, we add Grid+ that is an accounting layer or Dapp platform for the energy ecosystems on the smart contracts layer. Shown here are four phases of the energy supply chain: production, transmission, distribution, and retail. Of these, Grid+ focuses on innovating the energy retail sector with the integration of the blockchain technology. There are opportunities for modernization, improving efficiency, updating payment system, and cost cutting for the end-user. Field experts suggest that about 50 percent of the end-user energy cost is not for electricity but for the administrative warheads. Grid+ aims to minimize this with better technology and make electricity grid cost effective, reliable, and robust. On completion of this lesson, you will be able to discuss the need for a smart energy grid, explained architecture of the Grid+ system, and explain the working of a Grid+ system. What is a Grid+? Grid+ is a blockchain Dapp platform implemented on Ethereum blockchain. Grid+ has created an energy ecosystem by integrating the blockchain and AI thus leapfrogging the energy economy into the modern era. Grid+ aims to move energy transfer and payment transaction onto the emerging blockchain architecture. Grid+ pushes market signals to the end-users who can retain control of their energy assets and make smart decisions. This improves energy usage efficiency and also reduces costs to the end-user. Grid+ provides a seamless payment system, leveraging crypto tokens, thus improving efficiency in the energy payments system. Have you noticed, it's all about disintermediation of the energy system. Recall blockchain is about decentralization, disintermediation, and distributed ledger.

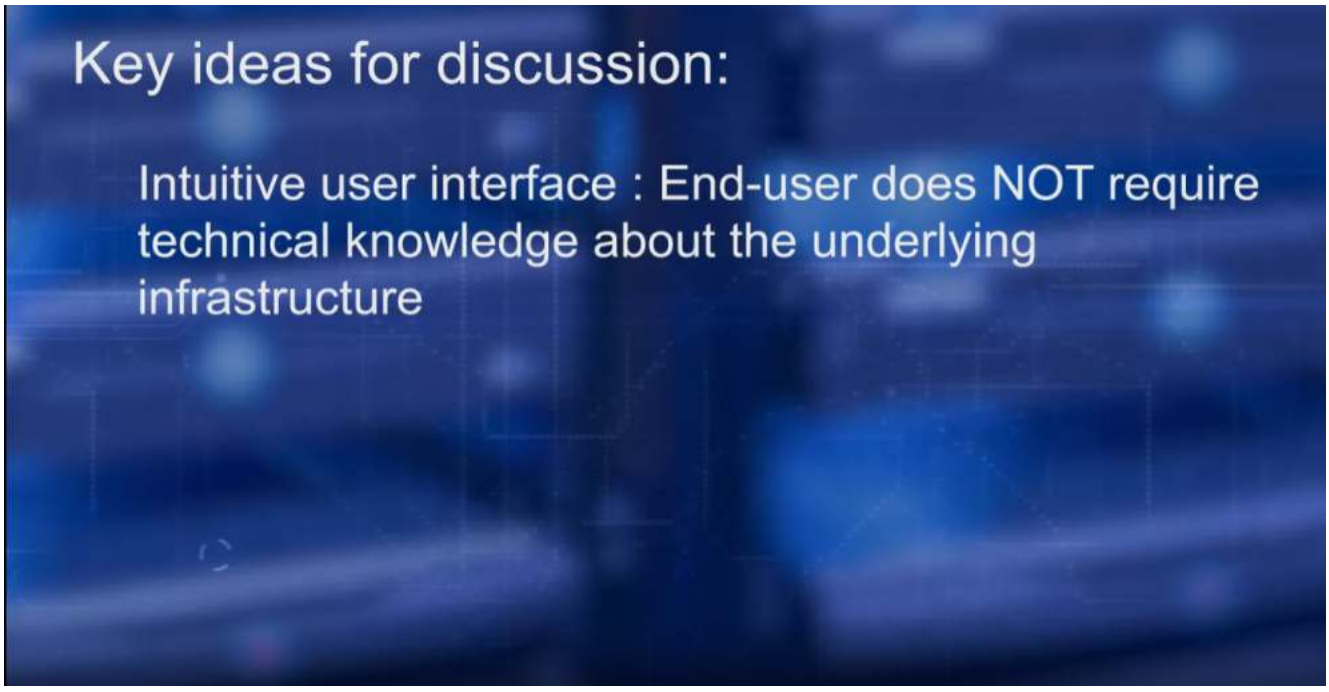


What is regulated and deregulated markets? In a deregulated electricity market, market participants other than the utility companies own power plants and transmission lines. In such contexts, the companies that generate electricity, sell electricity into a wholesale market, and the retail energy suppliers purchase this electricity to sell it to the customers. This deregulated electricity market offers an opportunity for Grid+ to test its operational capacity and commercial viability. These are the key ideas for Grid+, energy retailer. Grid+ will operate as a commercial electricity retailer in deregulated markets.

The second key idea is that of a smart agent at the user or the customer household. A Grid+ Smart Agent is a computing device that hosts a software for blockchain transaction, multi-signature crypto wallet with PKI security, and off chain payments for faster confirmations.

Next key idea is the intelligent electricity usage. Electricity trading is a complicated process with many intricacies, Grid+ manages these by coding efficient price options for the customers using smart contract. ERC20 Token payments: A special ERC20 compliant token called BOLT has been created for payment purposes. Integration to Internet of Things devices, IoT devices: A smart agent can be integrated into other intelligent agents in a household like NEST and electric batteries such as Tesla Power Wall.

Remote control: Grid+ enables integration of mobile phones and computing devices to allow remote control of its operation.

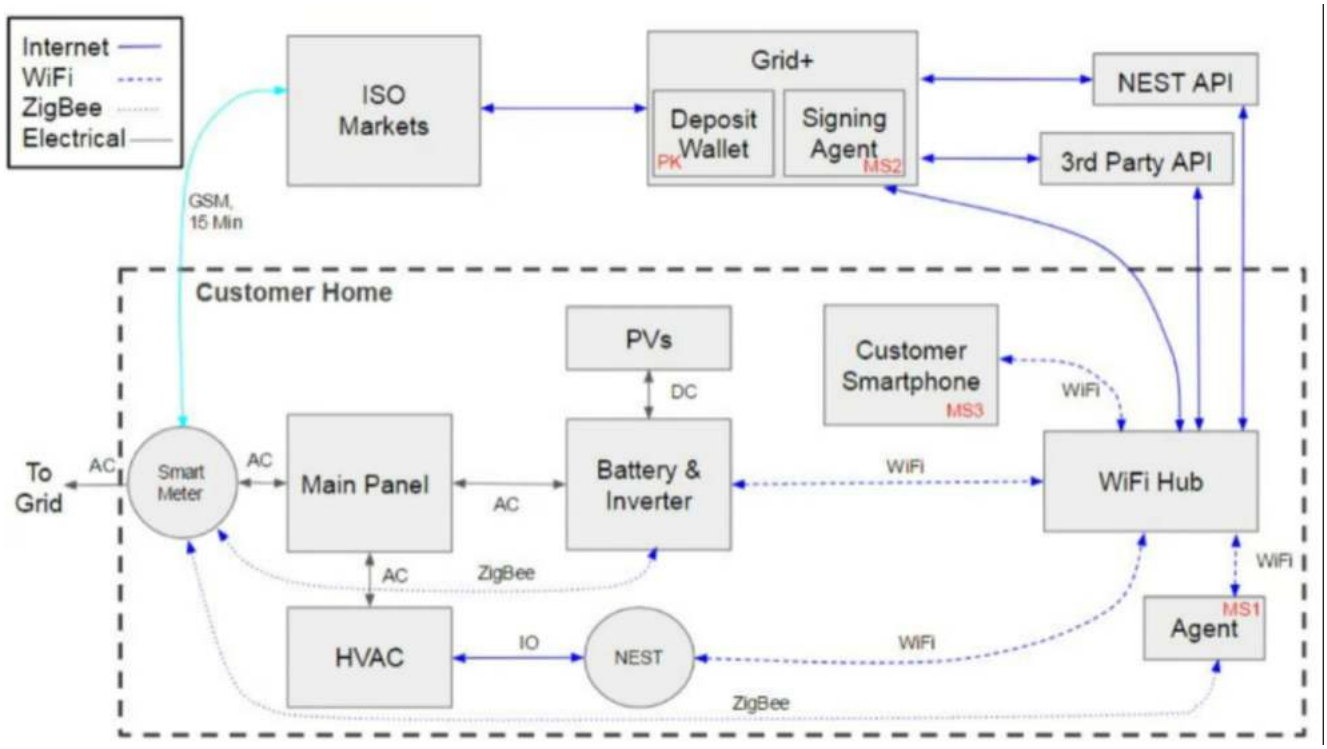


Intuitive user interface that allows easy interaction with the system. Even though blockchain and crypto token are involved, the end-user does not require any technical knowledge about the underlying infrastructure

2.2.2 Part 2

Here you see, the grid smart agent prototype, that is an embedded real-time system. It's a dedicated computing system that represents a node on the Ethereum blockchain. Do you wonder what type of blockchain? It is indeed

a permission blockchain made up of a consortium of users of the Grid+ in a particular state, and the business that support the delivery of electricity and payment systems.



Observe the keypad on the smart agent for users to interact with the device for configuration and signing transactions if needed. Here is the architecture of Grid+ platform showing the various components. The components inside the dotted rectangle are at the consumer's location, maybe a residential house. It has a smart meter, main panel, several electricity-based devices, heating ventilation, air conditioning, HVAC, battery, PVs or photovoltaic devices that could be, solar panels generating electricity. On the right side are the wireless hub devices such as NEST, customer smartphone, and finally at the bottom, you see the Grid+ smart agent, SA. Smart agent is connected via the Internet to the Grid+ retailer shown at the top. Grid+ under smart agent are deployed on the blockchain. You can also see three signatures, MS1, MS2 and MS3. Grid+ is indeed multi-sig with two out of three sign for triggering a transaction. Users can use smartphones. MS3 and MS2 are used to control and configure smart agent. Even if the smart agent is stolen or compromised, it cannot be misused and coins in it cannot be misused because it is multi-sig. Grid+ working. Compare the Grid+ architecture with dApps we developed. The former is indeed a complex interplay of many systems. The smart contracts and the blockchain are the new system components in the electric grid, automating the de-centralized usage and the energy generators. Grid+ is about disintermediation.

It pushes the management of the usage to the edges, to the user's smart agent. Recall from course one, we are moving the concerns to the periphery or to the asset holders. Users hold the payment tokens boat and directly send the payment to the Grid+ retailer of energy by signing an appropriate transaction, no intermediaries such as credit card or bank. The signing could be as simple as pushing a button, the user can be a agnostic to the underlying cryptoeconomics. Now, let's explore further. The user buys the smart agent. Using the serial number on the device that is unique, two keys are generated. Based on these and a registry smart contract, the smart agent is configured. User then buys BOLT coins as tokens provided by the Grid+ using regular money, credit card, or bank card. A smart contract keeps track of the BOLTs as escrow in the smart agent. Users now can use electricity and can use smart devices such as NEST to optimize usage. Based on the usage, BOLT is automatically transferred to the Grid+ retailer using a off-chain payment channel. User also has a BOLT deposit at the retailer for any overuse than earlier configured in the smart agent. Grid+ estimates the energy required for a period for all its customers based on their smart agent configuration. It buys the amount in wholesale and sells it at retail. All this automated using AI, artificial intelligence, and smart contracts. This model cuts down on the intermediate administrative cost, saves users money, enables efficient use of electricity.

In summary, Grid+ leverages smart contracts and blockchain technology to eliminate intermediaries in the energy market. This is accomplished by automation of the payment process and optimizing the usage patterns to supply patterns. This cuts down nearly 50 percent of the customer cost on the administrative cost of the energy delivery. This business model of energy we discussed for Grid+ can be adapted to many other vertical domains. Look around you, you should be able to identify products for applying decentralization. If you do, the discussion in this module Auger and Grid+ white papers should provide you enough information you need to proceed.

2.2.3 Resources

ConsenSys Introduces Grid-Based Solution For Energy Inefficiency
GridPlus - What is Grid+?
GRID+ white paper

2.2.4 Quiz

1. What are the four phases of the energy supply chain?

- ◯ Mining, Transmission, Distribution, Sales
- ◯ Production, Storage, Distribution, Retail
- ◯ Production, Transmission, Distribution, Retail

2. What are the two types of energy markets?

- ◯ Private, Public
- ◯ Black, White
- ◯ Regulated, Deregulated

3. BOLT, the Grid+ token, is ERC20 compliant. True or False?

- ◯ False
- ◯ True

4. Which blockchain technology does Grid+ use?

- ◯ Ethereum
- ◯ Dash
- ◯ Bitcoin
- ◯ Hyperledger

5. Payments in Grid+ system are done using BOLT tokens. True or False?

- ◯ True
- ◯ False

Chapter 3

Challenge & Solutions

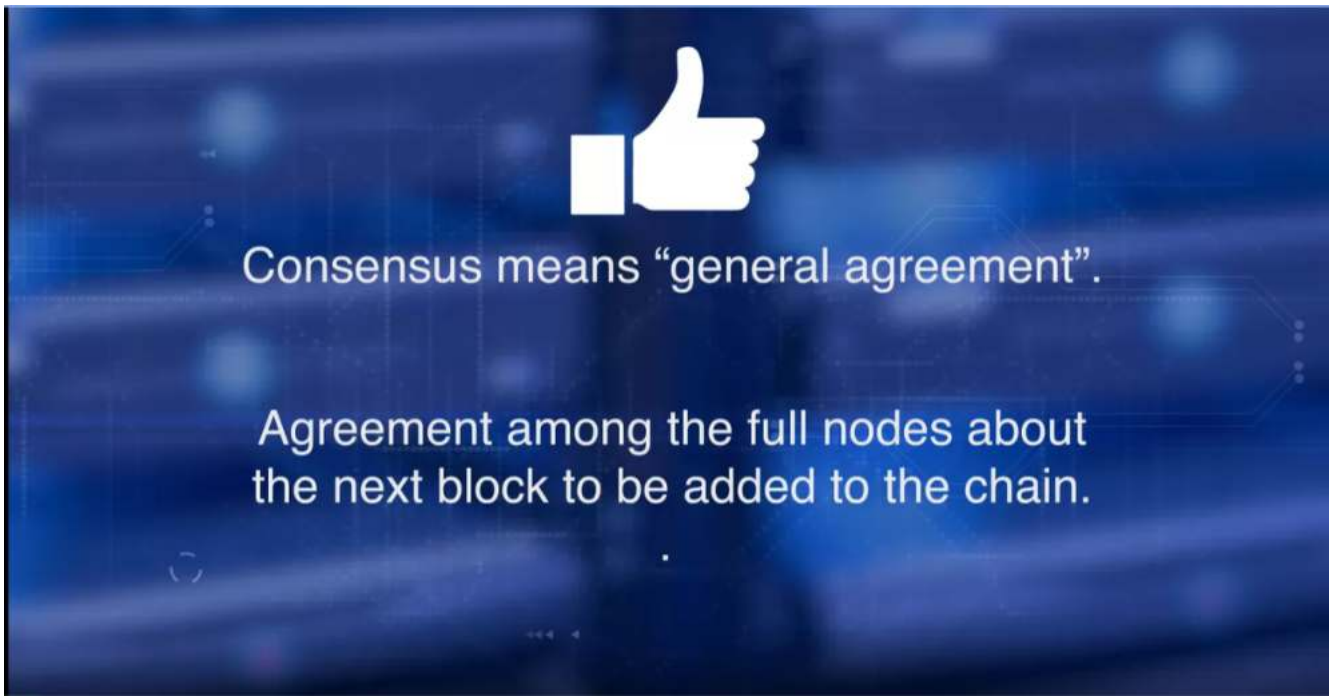
3.1 Consensus

3.1.1 Intro

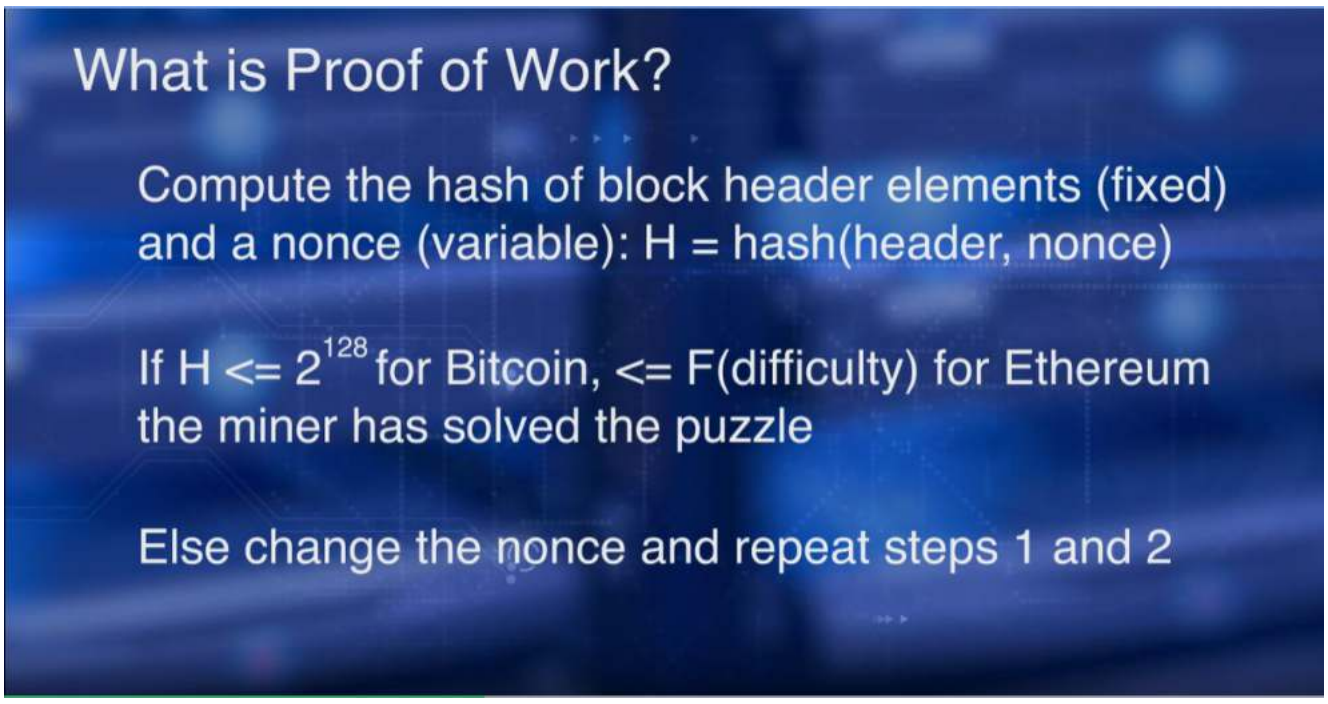
Course for Module 2: Challenges and Solutions. Any emerging technology will experience challenges as it is maturing. Blockchain is not an exception. The field is churning with activities and initiatives in a quest for continuous improvement in the Blockchain technology. There are many challenges impeding its widespread adoption. We'll explore just a few of the important challenges and solutions that are continuously innovating this emerging technology. On completion of the module, you will be able to: describe the consensus problem and some of the existing solutions, explain scalability of the Blockchain and possible solutions, explain the escrow and multi-sync solutions in the context of smart contract, and discuss privacy and confidentiality and its solutions.

3.1.2

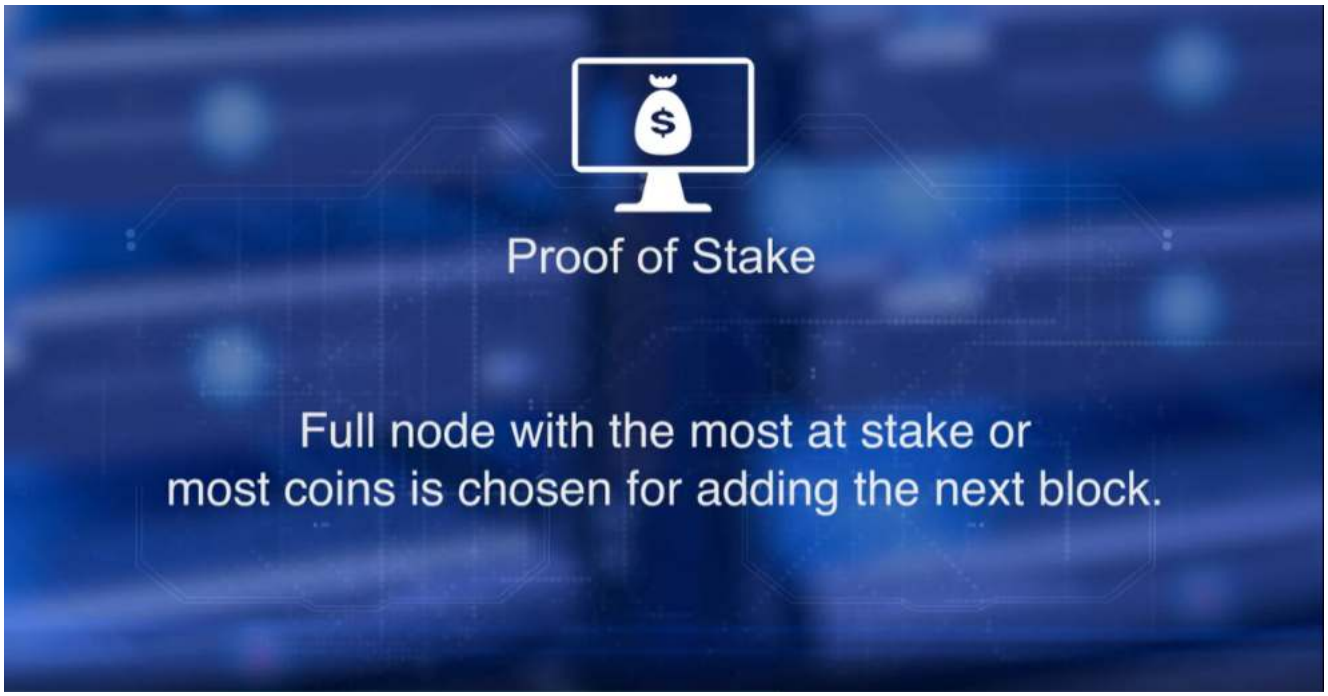
Consensus means general agreement. In the context of blockchain operation, consensus is an agreement among the full nodes about the next block to be added to the chain. This ensures the integrity of the chain. Even among the known permissioned entities, we need consensus. Different consensus protocols are used in various blockchains. Bitcoin uses proof of work for consensus. It is computationally intensive, and results in enormous power consumption in the massive racks of ASIC computers used in solving the POW puzzle, for the right to mine the next block.



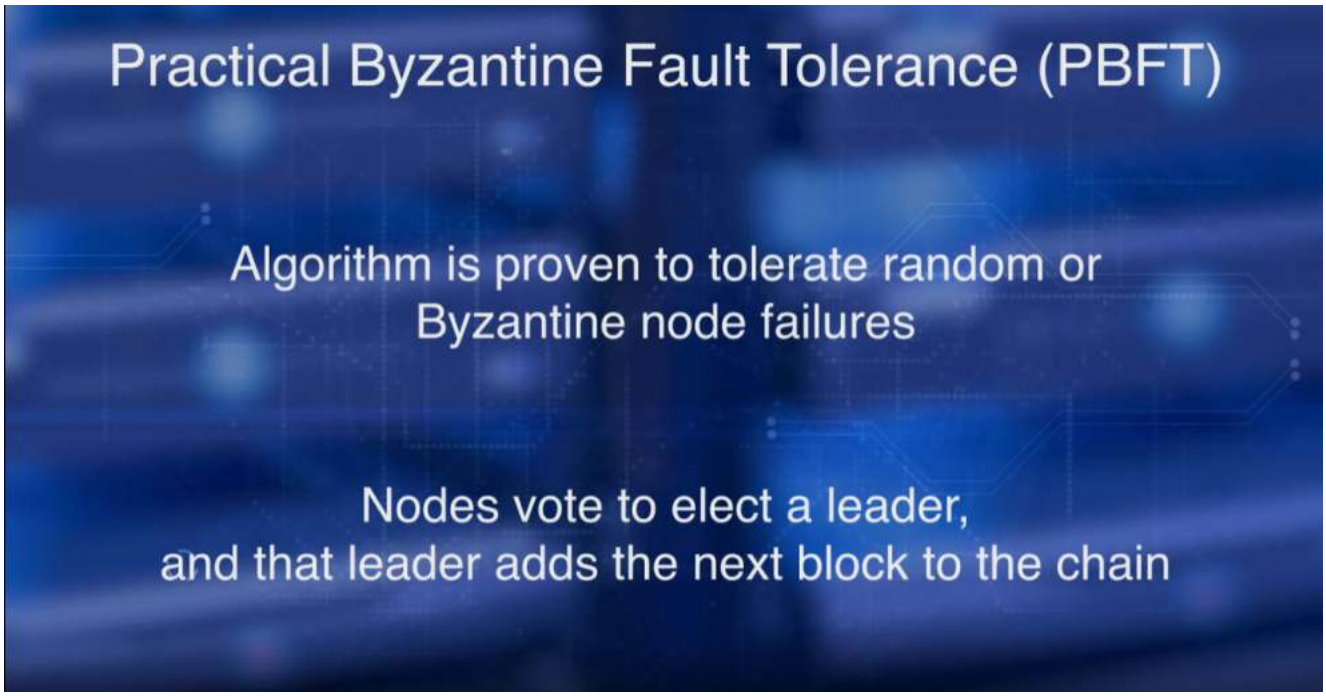
It is estimated that Bitcoin mining consume as much energy as the country of Ireland, per day. There are many other alarming statistics about Bitcoin mining you can read about in the link we have provided. Therein lies the problem, the enormous energy used by POW for mining a block. We discussed the POW method and two alternatives. Our lesson goals include explaining the importance of consensus, listing two alternatives to Proof of Work consensus algorithm, and explaining Proof of Stake. What is Proof of Work? Proof of Work works like this. Compute the hash of the block header elements that is fixed and a nonce that is a variable. H is the hash of the header and nonce. If the hash computed is less than 2 to the power of 128 for Bitcoin and less than function of difficulty for Ethereum, the miner has solved the puzzle.



Otherwise, change the nonce and repeat the steps above. While it is difficult to find the combination of header and nonce that solves the problem we described about, it is easy to verify. How do you verify that hash is less than or equal to 2 to the power of 128? Check if the leading 128 bits of the hash H are 0. Bitcoin used Proof of Work, or POW, and Ethereum followed it with slight modifications. In the current version, Ethereum Metropolis, it still utilizes Proof of Work. However, Ethereum Metropolis protocol uses a memory based proof of work that is not energy intensive. In the upcoming fork, Ethereum Constantinople, the plan is to switch to POS, or Proof of Stake. What is POS? In Proof of Stake, the full node with the most at stake or most coin is chosen for adding the next block.



That is why it is called Proof of Stake. The idea is, that node with the most stake will not be malicious and risk its stake for forking the network. To avoid monopoly by the node with most stake, an age based round robin policy is used in addition to the basic POS. The minter fee is paid by the transaction fees, and there'll be no minor fee such as in POW. In fact, as Ethereum is moving to its POS, the miner fee has been reduced from five ether to three ether. The POS approach is environment friendly and efficient, and is being popularly adopted by many blockchain platforms. The next consensus algorithms we'll review is Practical Byzantine Fault Tolerance, PBFT. This algorithm is proven to tolerate random or Byzantine node failures, including malicious nodes.



In PBFT, nodes vote to elect a leaderc and that leader adds the next block to the chain. This is the block of validated transactions. Each node maintains the state of the network. The nodes exchange messages. The messages, along with the saved state, are used to reach a consensus in the presence of random independent faults, or bad nodes. The chosen node adds the next block of validated transactions. PBFT is popular in permissioned block chains, such as hyperledger fabric. Later in the course, we'll learn an emerging consensus algorithm called Hashgraph that is a variation of Byzantine fault tolerance. It is called asynchronous Byzantine fault tolerance, or ABFT. In summary, consensus is a core component of blockchain protocol, and an efficient algorithm is essential for the integrity as well as scalability of the blockchain. There are many consensus algorithms being explored for blockchains. We reviewed three of them. When you study a new blockchain platform, here is a question to ask. What is the consensus algorithm used and how does it work?

3.1.3 Resources

Basic Primer: Blockchain Consensus Protocol
Consensus Achieved Using Proof-of-Work
Consensus and Mining on the Blockchain
How nodes reach a consensus on a blockchain

3.1.4 Quiz

1. What is the common consensus protocol used in Bitcoin and Ethereum Metropolis?

- ◯ Proof of stake
- ◯ Proof of work
- ◯ Proof of weight
- ◯ Proof of authority

2. What is a major problem with Proof Of Work?

- ◯ It is unreliable
- ◯ It is difficult to implement
- ◯ Multiple miners have to be rewarded
- ◯ It is CPU-intensive and consumes enormous amount of power.

3. Which of the following is used in solving a POW puzzle?

- ◯ Sharding
- ◯ Encrypting
- ◯ Hashing
- ◯ Fragmenting

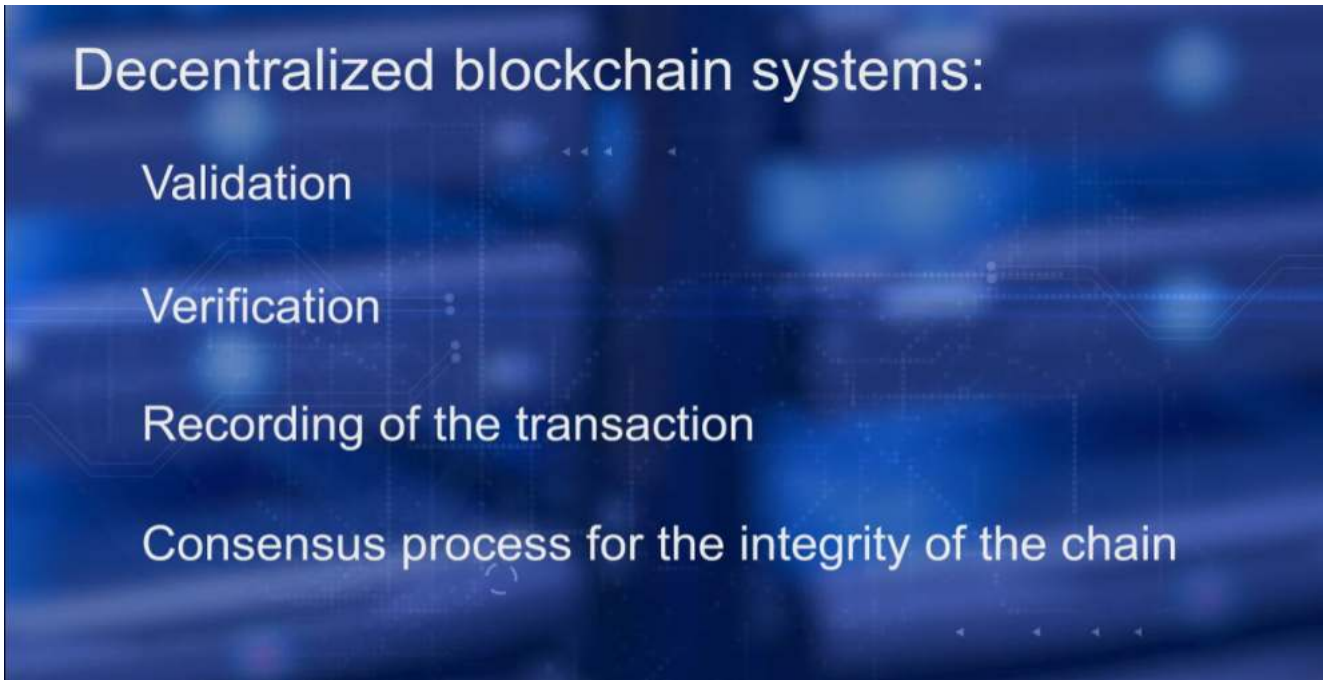
4. In Practical Byzantine Fault Tolerance, ___.

- ◯ the node with most coins is chosen for adding the next block
- ◯ the nodes elect a leader and that leader adds the next block
- ◯ a master node selects the next node that adds the next block

3.2 Scalability

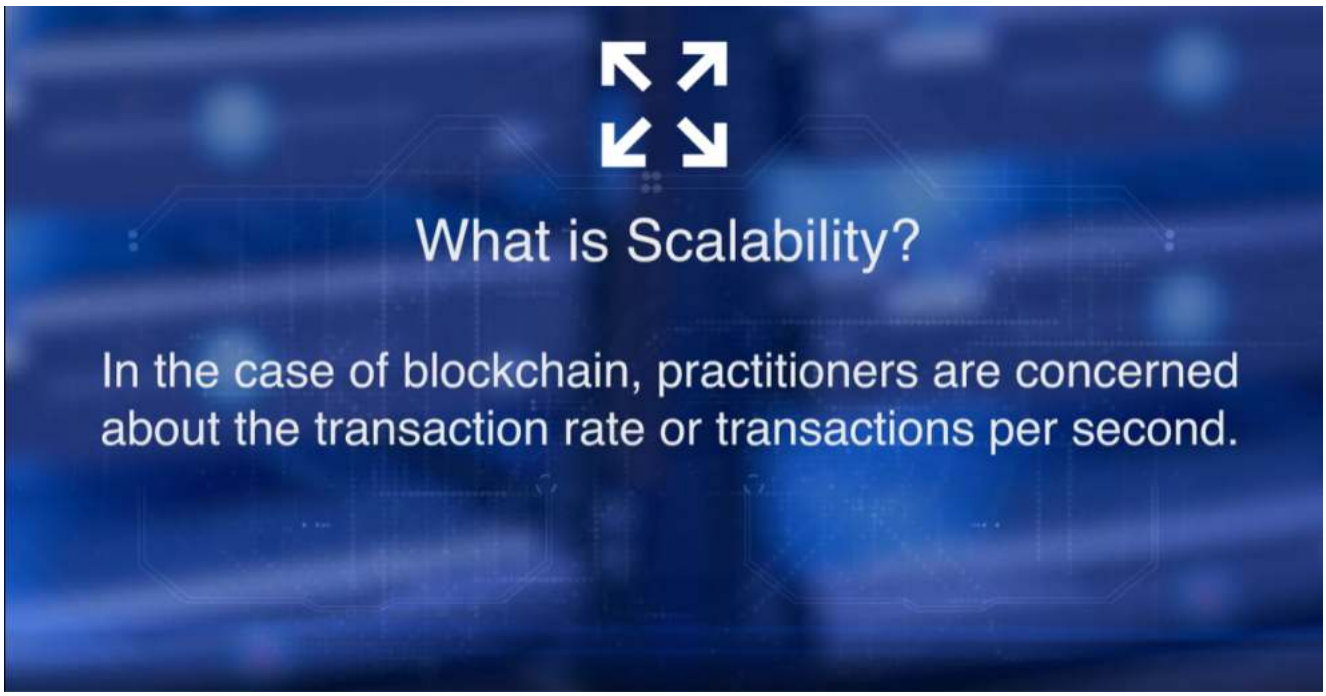
3.2.1

Module two, lesson two: Scalability. Let's compare decentralized blockchain systems with centralized systems. Blockchain has taken on the responsibility of the intermediaries in the form of validation, verification, and recording of the transactions, and of course, the consensus process for the integrity of the chain. All these functions take time and result in significant overhead in the confirmation time of transactions, when compared to the centralized system. Transaction in Ethereum, for example, execute on all nodes and execute sequentially, not in parallel. Every full node stores the entire chain of blocks. All these impede the scalability of the blockchain applications.

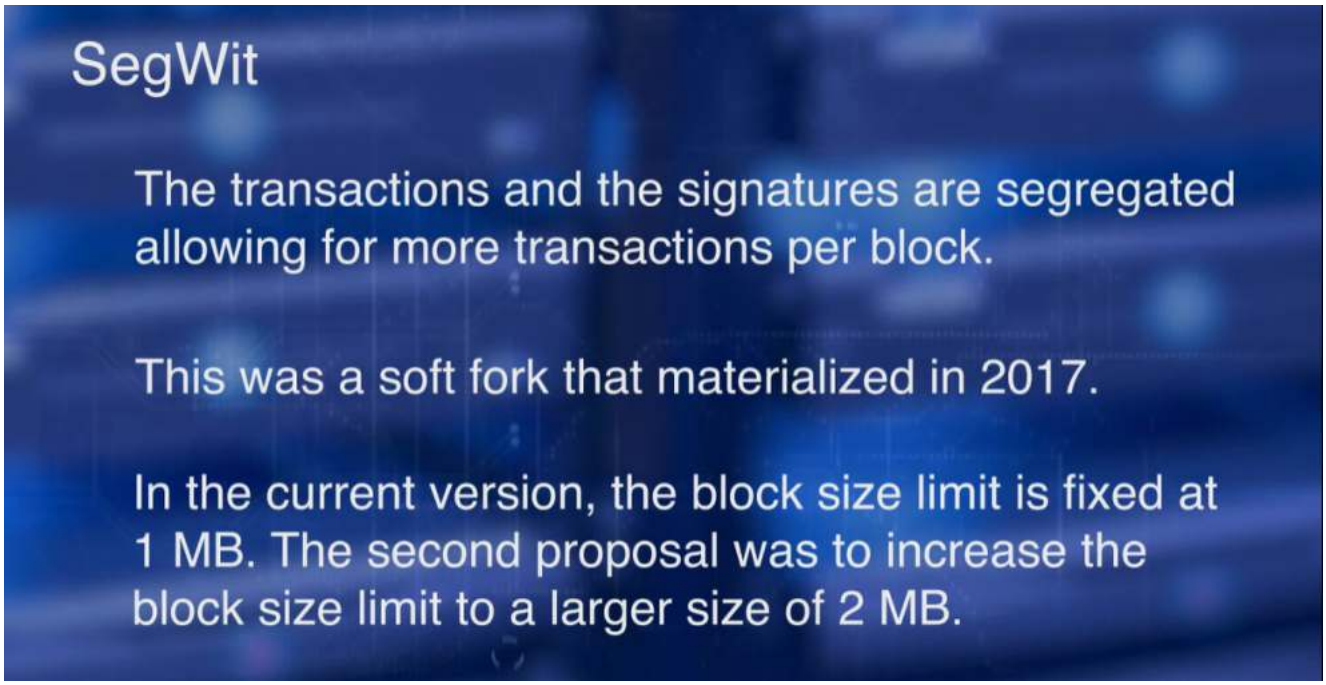


Transaction rates are not satisfactory compared to current centralized application. This is the challenge. In this lesson, we'll examine some solutions addressing scalability. What is scalability? Scalability is the ability of a system to perform satisfactorily at all practical levels of load. Load in the context of the blockchain could be transactions, number of nodes, number of participants and accounts, and other attributes of blockchain.

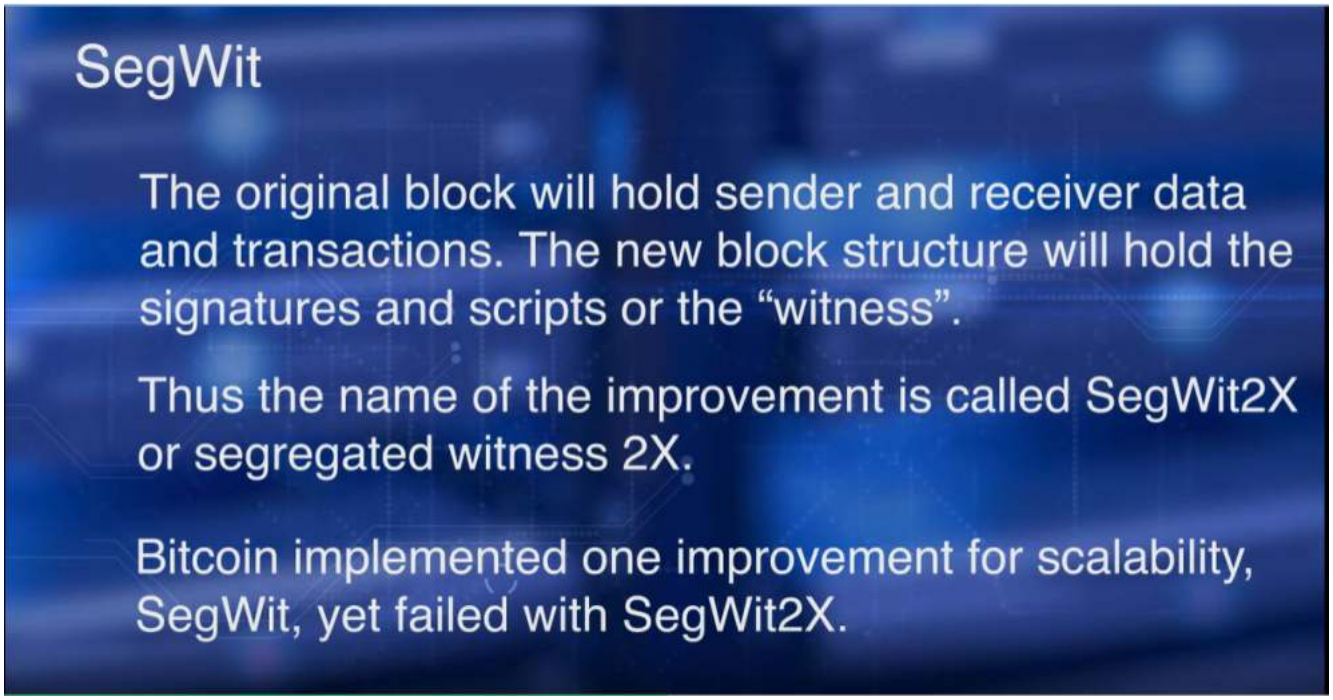
In the case of blockchain, the practitioners are concerned about the transaction rate or transactions per second. This is a critical metric for many applications from payment system to supply chain management to perform well.



So, we focus on transactions per second as a metric for scalability. Let's explore scalability further. On completion of this lesson, you will be able to: discuss an on-chain solution in improving scalability using larger variable block sizes, and explore off-chain transaction in state channels for improving scalability. The most obvious method to improve transaction rate is to increase the number of transactions per block. Bitcoin addressed it in two ways, segregated witness and increase in block size limit.



In segregated witness or Segret, the transactions and the signatures are segregated for allowing for more transactions per block. This was a soft fork that materialized in 2017. It's working in the current version of the Bitcoin blockchain. However, the current version, the block size limit is fixed at one megabyte. The second proposal for scalability was to increase the block size limit to a larger size of two megabyte. In this case, the original block will hold the sender and receiver data and transactions, the new block structure will hold the signatures and script, or the witness. Thus, the name of the improvement is called Segret2X, or Segregated Witness 2X.



It was a planned hard fork that was to have taken place in November 2017, but did not go through.

At this time, Bitcoin has implemented one of its improvement for scalability namely the Segret, it failed with Segment 2X. Now, let's explore how Ethereum addresses the block size. In Ethereum, block size can vary, and is limited by the gas limits specified in the block header. It can be increased by miners working on it. The number of transactions in a block dictates the block size. Here, we see four blocks mined with 30 transaction, 294 transaction, 55 transactions and 183 transaction with the average block time of eight seconds. This variable block size along with ASIC-resistant Pure W Consensus Algorithm have improved the transaction rate. It is expected to improve further when Ethereum moves to proof of state Consensus Protocol in an upcoming hard fork in 2018. We just discussed an on-chain solution to improve transaction rate. Next, we'll discuss an off-chain method. One of the solutions to address transaction rate is to unload some of the transactions off-chain. This is carried out between trusted transacting parties. Once the unload has concluded, a transaction confirming this off-chain activity is recorded on-chain on the blockchain DLT. This feature in Bitcoin is called payment channel. Payment transactions can be carried out with minimal fees at considerable higher rates between trusted parties. This leverage is the Segret improvement we discussed earlier. Lightning network is a well known blockchain implementing this protocol. An Ethereum solution extends this payment channel concept to Smart Contracts. The state channels are an extension of the payment channel concept to any arbitrary application level transaction. It is called the State Channel since the state of a node on the main chain is locked when transacting off-chain, and it is synchronized periodically with relevant updates on the main chain. The transaction of off-chain channels take place at much higher speeds than the blockchain network since no consensus or recording under DLT is required. Thus, offering a scalable solution. The radar network is an off-chain solution for Ethereum-based main chain. The popular grid plus energy platform uses this off-chain payment channel for real-time energy payments from a Smart Contract to the grid plus retailer Smart Contracts. In this case, there is a one to one trusted transactions between two known parties that have secure channel pre-established. In this lesson, we looked at two solutions for scaling, one on-chain, the adjustment of block size, and the second solution that is off-chain, the state channels. Both these approaches aim to improve the scability of the blockchain. Other solutions include sharding and panel processing. Now, the question to ask when you are studying a blockchain is, what approaches are used for improving scalability?

3.2.2 Resources

Blockchain Scalability: When, Where, How?
Scalability Tradeoffs: Why "The Ethereum Killer" Hasn't Arrived Yet
Can Blockchain Solve Its Scalability Problem?
Brock Pierce from Blockchain Capital explains the scalability problems for Bitcoin

3.2.3 Quiz

1. What is the size of the bitcoin block since 2010 ?

- ○ 1 KB
- ○ 1 MB
- ○ 1 GB
- ○ 2 MB

2. Transaction rate on bitcoin blockchain is a main concern for many practical applications?

- ◯ True
- ◯ False

3. List two solutions for improving scalability ?

- ◯ Larger block size and off-chain transactions
- ◯ Smaller block size and on-chain transactions
- ◯ Larger block size and on-chain transactions
- ◯ Smaller block size and off-chain transactions

4. Every Ethereum block holds a fixed number of transactions. True or False?

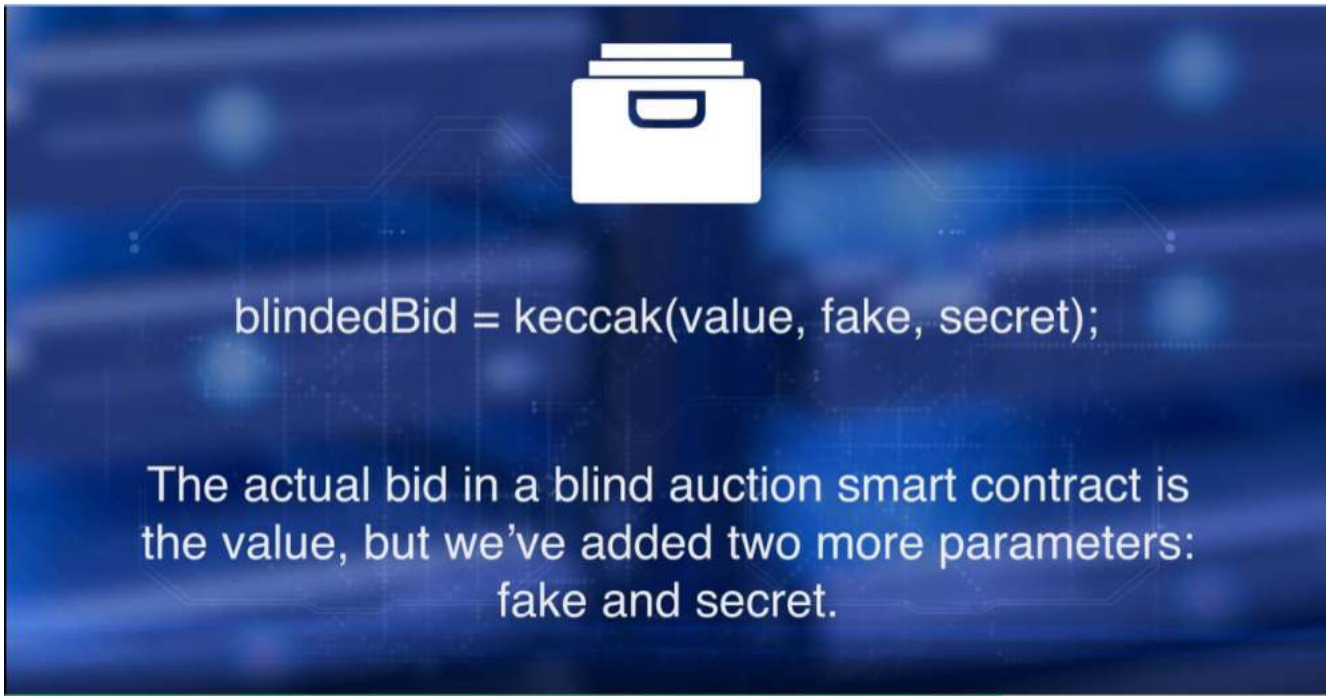
- ◯ True
- ◯ False

3.3 Privacy Confidentiality

3.3.1

Privacy is the right to protect the data, attributes, and assets of an entity from observation by unconsented parties. Confidentiality of data ensures that only authorized entities can have access to the data. These are two important concept especially for blockchain where the transactions can be viewed and analyzed using custom-built tools. In this lesson, we'll look at methods for addressing privacy and confidentiality. On completion of this lesson, you will be able to discuss the approaches to ensure privacy, explain the methods for realizing confidentiality, both in the context of blockchain.

By now, we all know very well that Bitcoin is a public blockchain. You can see all the transactions happening on the Bitcoin blockchain. Many blockchains including Enterprise Ethereum and Hyperledger Fabric are permissioned blockchains. They limit access to the blockchain to permission entities only.



This ensures privacy of the network to a large extent. Within the network, data access privileges such as create, read, update are assigned to various entities. These are enforced with every transaction. This adds to the basic privacy provided by permissioned networks. Some blockchain platforms such as MultiChain ensures that the transacting nodes are indeed in a permissioned list. It also authenticates the sender and receiver using key exchange for every transaction. We just discussed the basic privacy currently implemented in blockchains. Next, let's look at the confidentiality. For confidentiality, a straightforward method is to encrypt the data transacted and digitally sign it. It is also possible to obfuscate the data by adding extra elements. We illustrate the encryption and obfuscation with a smart contract data, blindedBid, where the actual bid in a blind auction smart contract is the value, but we've added two more parameters called fake and secret. In this case, the blindedBid sent by the bidder is obfuscated with the value fake and a value secret as parameters, and then encrypted with keccak secure hash algorithm. This provides confidentiality of the bid in a blind auction application. We explored at high level some simple methods for ensuring privacy and confidentiality. There's extensive research on privacy including zero knowledge proofs and homomorphic encryption algorithms. You may explore these using the links in the resources section. Now, when you're designing an application, the question to ask is, what are the encryption and obfuscation method you're using in your design?

3.3.2 Resources

SOLVING BLOCKCHAIN'S PRIVACY PROBLEM
Smart Contracts: Privacy vs Confidentiality
Can This Man Build a Better Bitcoin?

3.3.3 Quiz

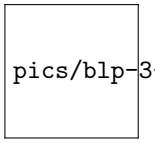
1. What is a simple way of achieving privacy?

- ◯ Using role management
- ◯ Using end-to-end encryption
- ◯ Using a permissioned blockchain

3.4 Escrow & Multi-sig

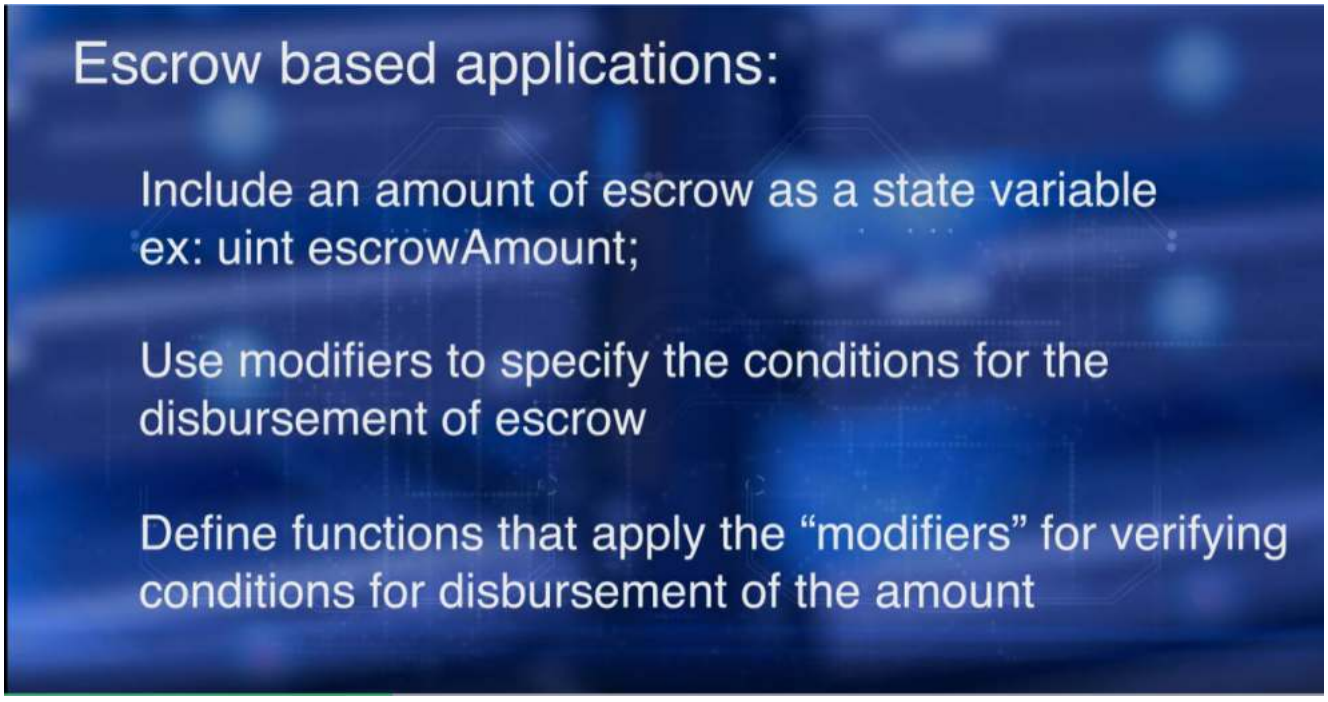
3.4.1

Module two, lesson four: Managing Escrow and Multisig. In this lesson, we discuss two related concepts that are already in use within the context of decentralized applications and smart contracts. Many contracts and legal documents including digital documents are error prone. For example, a single comma placed incorrectly in a written contract cost a million dollars to the Rogers telecommunication company. Escrow and multisig are two techniques that have enormous potential in addressing business contract related issues to transform business processes.



On completion of this lesson, you will be able to explain, the features of escrow within smart contact and features of multisig. Wikipedia defines escrow as a contractual agreement in which a third party receives and disburses money or document for a primary transacting parties, with the disbursement dependent on conditions agreed to by the transacting parties. A smart contract can do all this. It can hold money and all artefacts in an escrow, and allow transacting parties to negotiate on it, and verify conditions to be met before the disbursement of the items in the escrow. What is innovative is that, it can be done among parties that are operating beyond boundaries of trust in a decentralized world. Recall that a smart contract has an address that can hold a balance. You can transact, send or receive cryptocurrency from the smart contract.

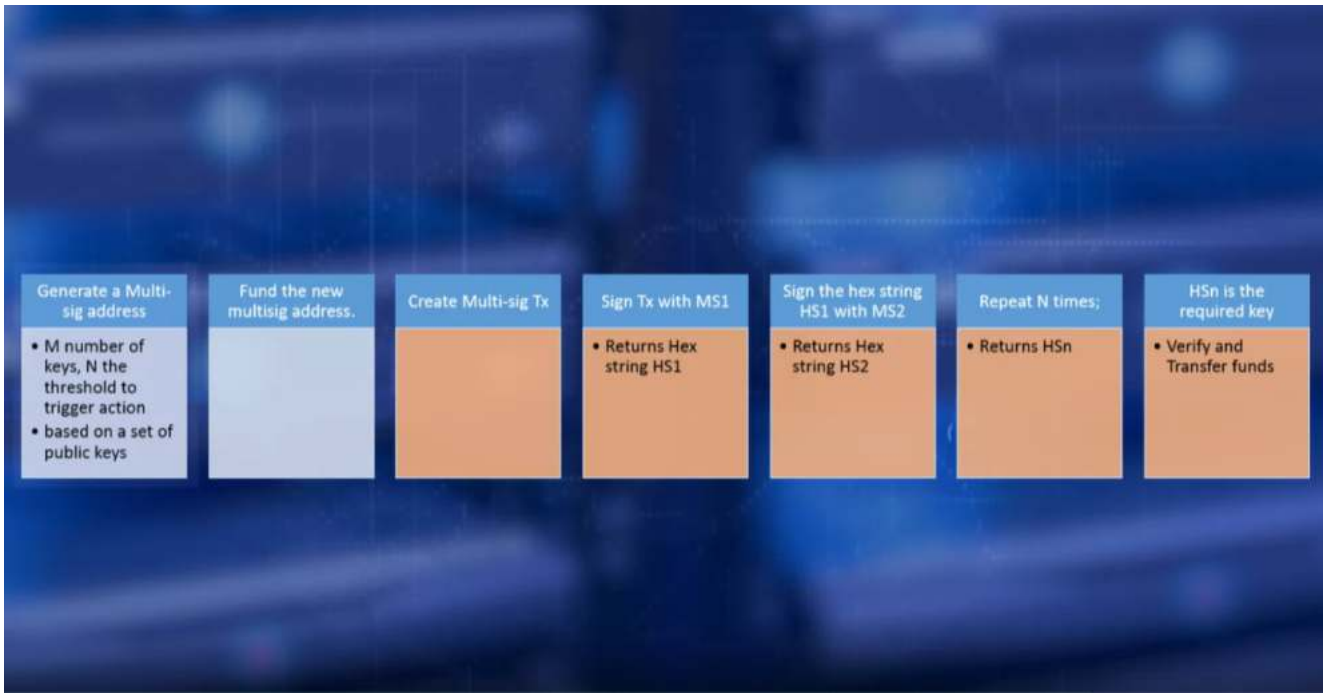
You can use three specific features of smart contract to develop escrow-based application. Include that mount of escrow as a state variable, example; you and escrow amount. Use modifiers to specify the condition for disbursement of escrow. Define functions that apply the modifiers for verifying conditions for disbursement of the amount. Recall grid platform. Grid place energy application platform users, replenish both coins as a deposit escrow in the smart contract agent, that is used for payment conditional on the energy expended by the user.



Based on the energy consumption, the smart agent sends only the payment for this consumption to the grid place retailer. Grid place business model also requires a monthly deposit by the consumer. In case, the smart agent runs out of bonds for the payment of energy consumption, the additional amount is deducted from

the escrow. This is another use of smart contract escrow.

It helps avoid interruption in electricity service. Holding escrow funds commonly works in conjunction with multiple signatures. This is known as the multisig feature. Many real world contracts how multiple signatories for security, verification, and protection of assets, businesses, and individuals involved in the contract. In many cases it may be required by law that multiple parties sign a document. For example, once healthcare proxy has to be signed by the inducer and two witnesses. Many business transactions warrant multiple signatures to the CTO, I.T. manager, and so on. This is about access control.



A signature in the crypto world is a private key of the address, transactions are signed by this address for verification of the sender's authenticity. Security of the assets held by this address is only as good as the security of the private key. If the key is compromised anyone with private key can drain the funds at the corresponding address. These concerns can be addressed by enforcing multiple signatures for execution of certain important transactions. The idea of multikey or multisig is not new. We have seen two-key solution in bank safety deposit boxes. You need two keys to open it. One held by you, and the second held by the bank. A thief cannot get to the content of the safety deposit box without both keys having been compromised. Similarly, bitcoin has used this idea successfully since 2012. We examine bitcoins multisig solution. Bitcoin has an alternative to a single key address. This is defined by the type of address. This is defined by a type of address called pay to script hash, a multisig address. Multsig addresses in the bitcoin transactions begin with the digits three, whereas regular addresses begin with the digit one. A pay to script hash address can support a set of n keys, any of m which are required to transact m of n. In practice, most multisigs are two out of three, two out of two, though larger values of m and n can be used for such use cases as legal documents. How do you use multisig? Generate a multisig address, it has n number of keys, m of which are required to trigger an action. Then fund the new multisig address with a deposit. Now, you're ready to transact. Create a multisig transaction. This transaction will be signed with the first signature ms one, resulting in the hex string, hs one. Then they can be signed by the second signature ms two, resulting in hex string hs two. This is repeated m times for the m signature resulting in hsm. This is the required multisigned key. It is verified by the protocol and the funds are transferred. This explains the multi-sig operation. We learned two closely related features of blockchain that are critical for business adoption of blockchain, escrow and multisig. We looked at some methods for realizing consensus, scalability, privacy, and confidentiality. We also explored methods for escrow and multi-signature in smart contracts, aligning them with real world contracts. Now you are ready to use them in your decentralized application, and contribute to the block chain protocol implements.

3.4.2 Resources

Two party contracts
How A Smart Contract replaced An Escrow Company in a $60k deal
Ultimate Intro to Ethereum app Development [Part 8] - Smart Contracts - Escrow
BURST Decentralize Escrow Service Smart Contracts
Toward an Ethereum Multisig Standard

3.4.3 Quiz

1. Smart contract accounts in Ethereum can hold a balance. True or False?

- ○ False
- ○ True

2. What is an escrow?

- ○ Escrow is an agreement in which assets are held and distributed when conditions are met
- ○ Escrow is cost of execution of smart contracts
- ○ Escrow is a permissioned blockchain
- ○ Escrow is payment for smart contracts

3. What is the common Multi-sig feature of Bitcoin?

- ◉ Pay-to-miner hash (P2MH)
- ○ Pay-to-owner hash (P2OH)
- ○ Pay to Script hash (P2SH)
- ○ MD5 hash

4. What is Multi-sig in the context of smart contract?

- ○ All the nodes on the blockchain have to sign the transaction
- ○ A transaction has to be authorized by multiple signatures
- ○ A transaction has to be authorized by the owner of smart contracts

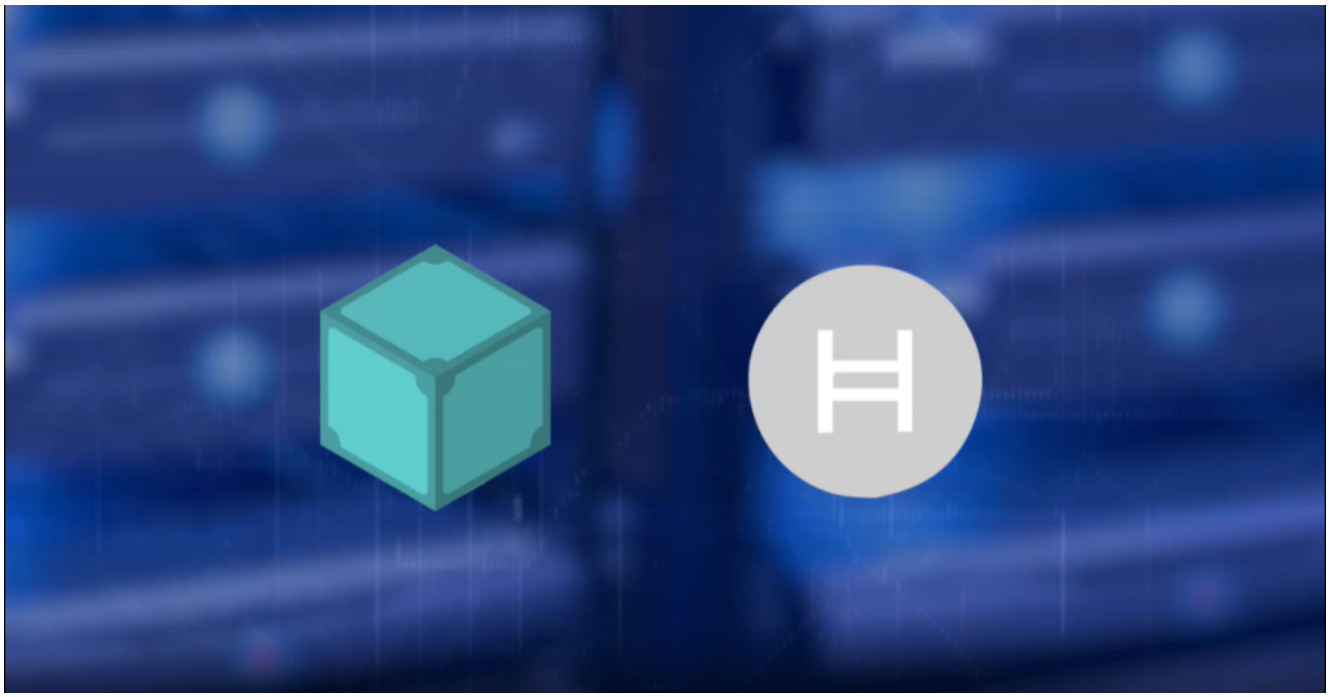
Chapter 4

Alternative Decentralized Solutions

4.1 Interplanetary File Systems (IPFS)

4.1.1 Intro

Since the advent of the blockchain, several alternative solution for de-centralized system, have been proposed for various purposes. This vigorous and enthusiastic support from the community is strengthening the notion of de-centralized system. Here are some prominent contributions. Interplanetary File System IPFS, that aims to address a de-centralized data storage problem and a hashgraph, that aims to solve the decentralized consensus problem.

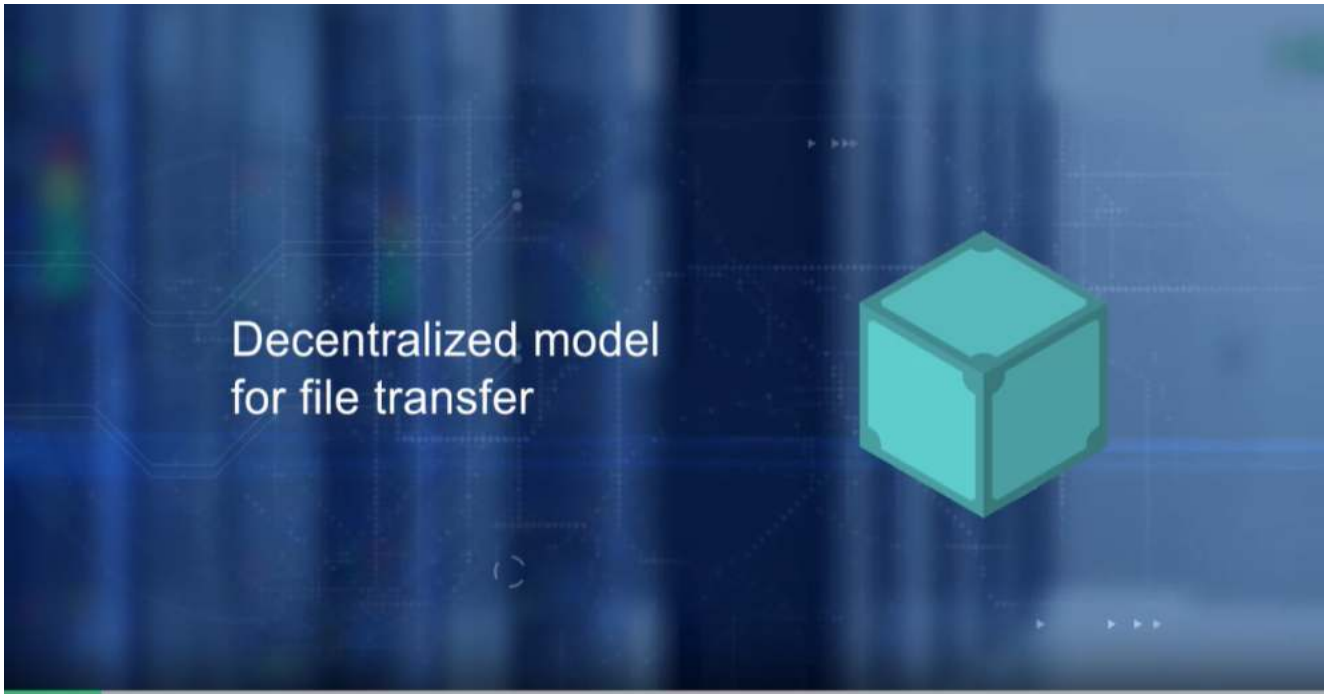


On completion of the module, you will be able to explain the structure of IPFS, describe the operations of IPFS, discuss the structure of a hashgraph, explain the Asynchronous Byzantine Fault Tolerant ABFT consensus protocol of the hashgraph.

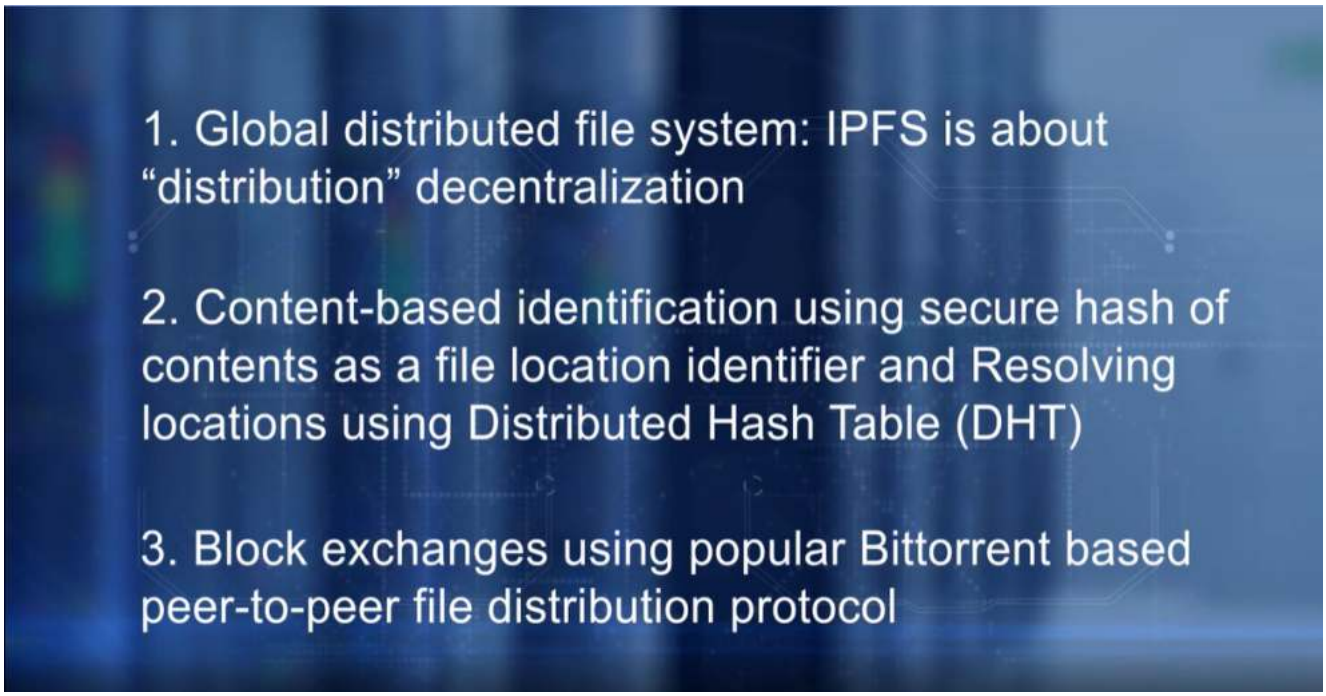
4.1.2 Part 1

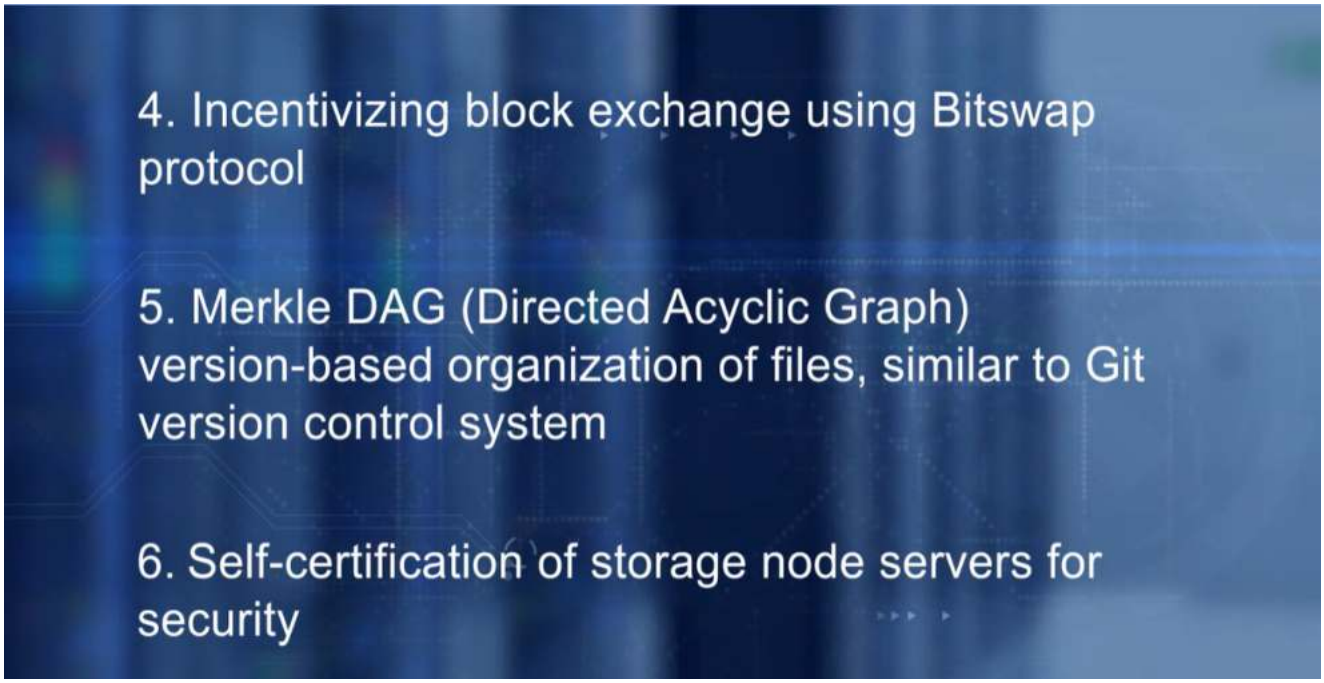
Think about this, a block chain is fundamentally a decentralized system. Alternatively, a decentralized peer-to-peer system can be realized independent of a blockchain. Interplanetary Files System, IPFS, is a fine example of

such a system. IPFS is a decentralized model for file transfer in contrast to centralize namespace and transfer provided by HTTP family of protocols.

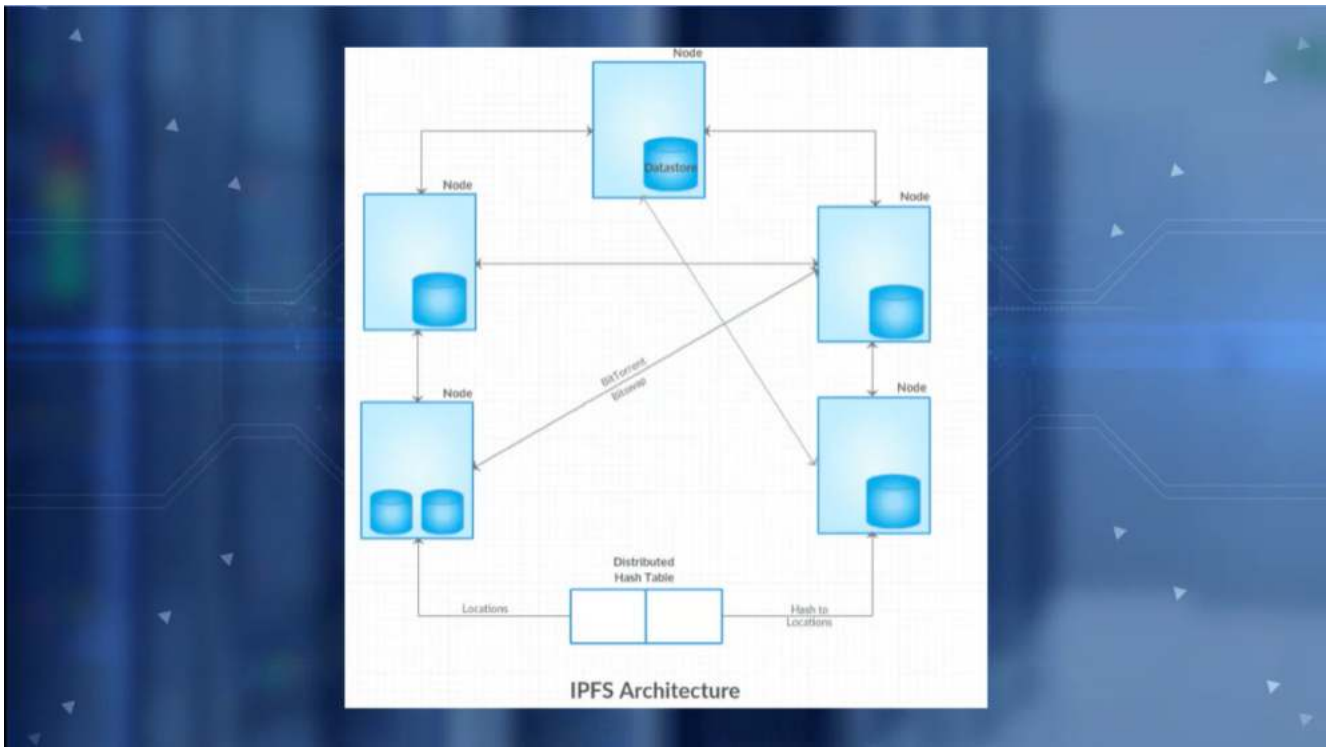


HTTP or Hypertext Transfer Protocol operates in a centralized hierarchical namespace. For example, a website is identified by http://www.cse.buffalo.edu, where each item is resolved hierarchically by the Domain Name Service, DNS. Peer-to-peer transfer of data is not new, it has been an age-old quest. Recall the Napster and Gnutella media sharing services, and the Bittorrent service that is underlying many of our current data seeming services. Juan Benet, the creator of IPFS, described IPFS it is whitepaper as Content Addressed, Versioned, P2P file system. In this lesson, we'll look into the details of IPFS, a decentralized file sharing protocol. On completion of this lesson, you will be able to discuss the architecture of IPFS, explain the operations of IPFS, list the benefits over blockchain based solution, discuss IPFS + Blockchain solution. Once again, similar to Bitcoin, IPFS leverages, combined many successful peer-to-peer system ideas.





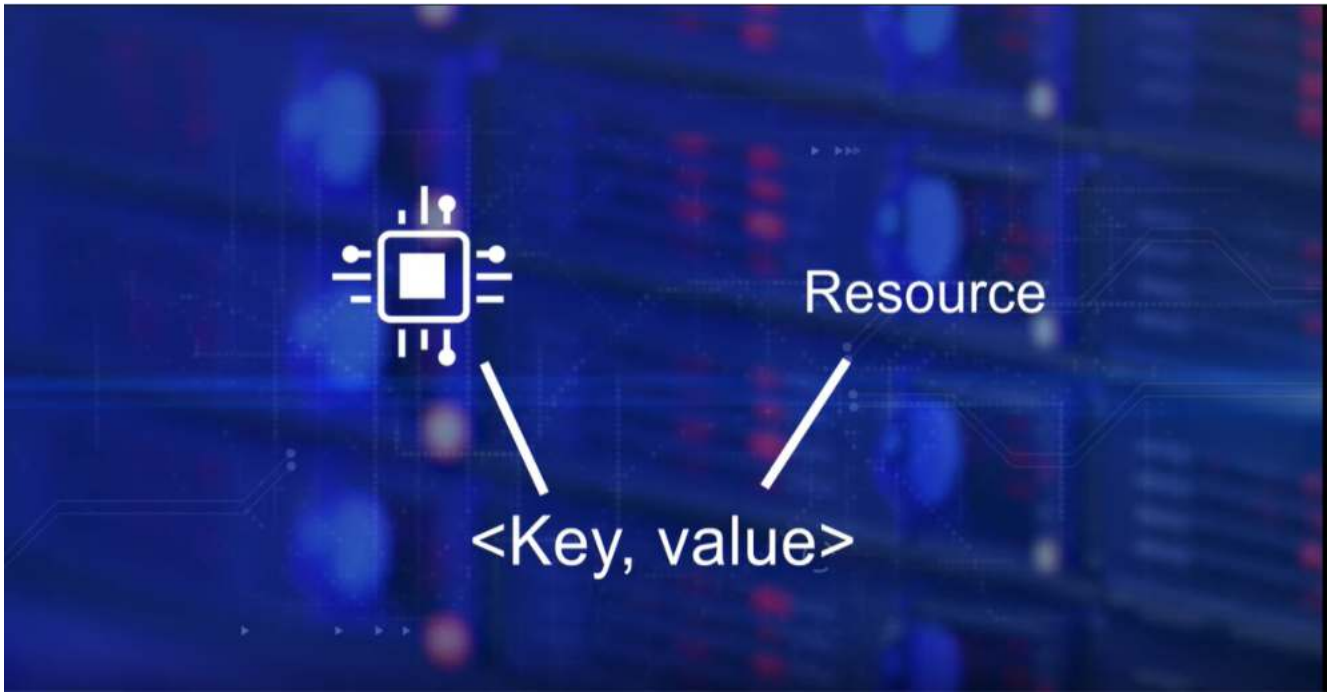
These ideas are: Global distributed file system, IPFS is about distribution decentralization, content-based identification using secure hash of contents as a file location identifier, and resolving locations using Distributed Hash Table.



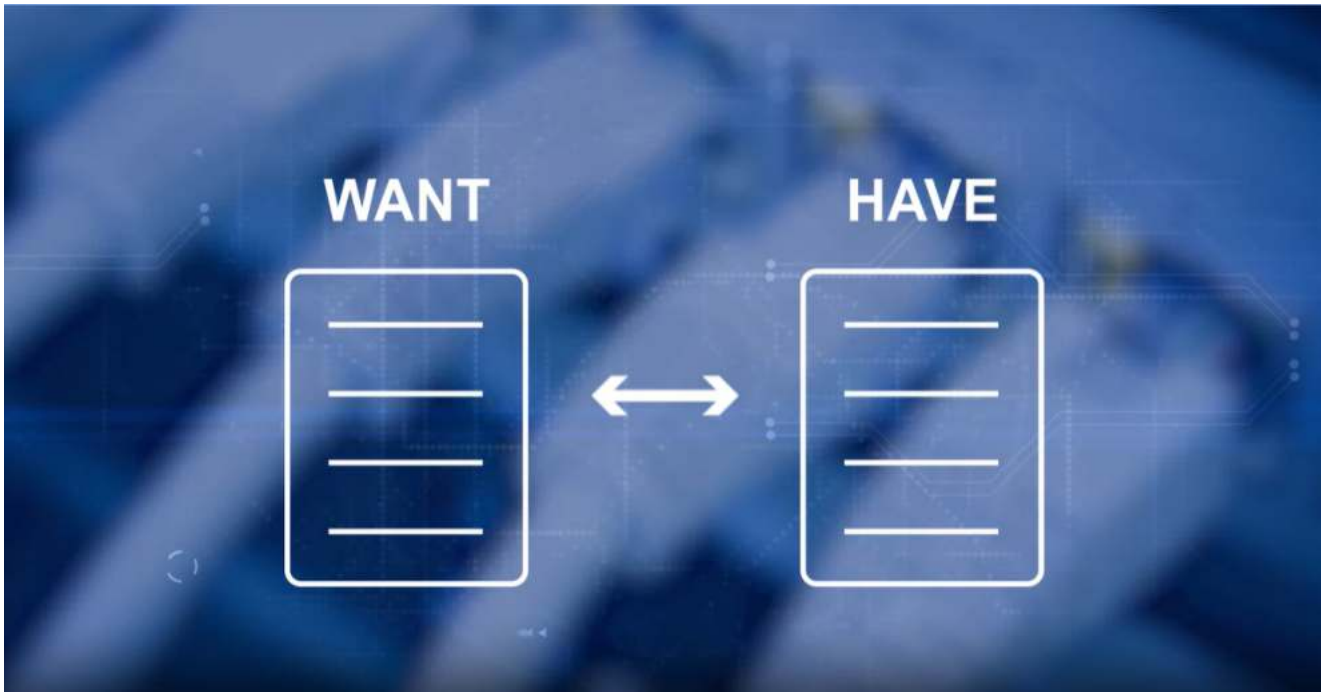
The next idea is the Block exchanges using popular Bittorrent based peer-to-peer file distribution protocol, and Incentivized block exchange using Bitswap protocol, and Merkel DAG, Directed Acyclic Graph, version-based organization of files, similar to Git version control system. Finally, self-certification of storage node servers for security. This is the high level view of the architecture of IPFS. Files are located in a distributed system. A distributed hash table maintains a location of the files. Application use the hash as the key in the DHD that returns the location of the file. Once the location of the file is determined, the peer-to-peer transport takes place. The nodes of computers that hold the decentralized file objects that formed the global file system. They hold the objects that formed the files to be exchanged. Nodes are identified by cryptographic hashes of its public key. This is similar to our blockchain nodes. File objects are identified by a secure hash and any object may contain sub-objects, each with its own hash that is used in the creation of the root hash of the object. Recall the Merkel Tree from course one, the blockchain basics? In the current world wide web protocol, we typically refer to a web resource or a data by the server on which they are stored. For example, https://www.coursera.org/ actually refers to the server on which the Coursera page is hosted, and to a particular directory and file on that server. This is a centralized approach. What if the resources are available in a number of distributed locations? IPFS offers a decentralized solution for this. Then, how does IPFS identify the resources? Well, hash of course. Instead of identifying the resource by its location as an HTTP, IPFS identifies the resource by its content or by the secure hash of its contents. In this case, the file is addressed by a universally unique identifier instead of by its location.

4.1.3 Part 2

How do you resolve the location? Just like you have a URL or a link of a website, you start with the hash identifier of the resource. You send around a request for anyone with the resource with this identifier. On success will respond, access it peer-to-peer. The hash is the key, the key, value pair, and the value is the location of the file or the resource. The routing part of the IPFS protocol maintains distributed hash table or DHT for locating the nodes as well as the file objects. A simple DHT holds a hash as the key, location as the value.

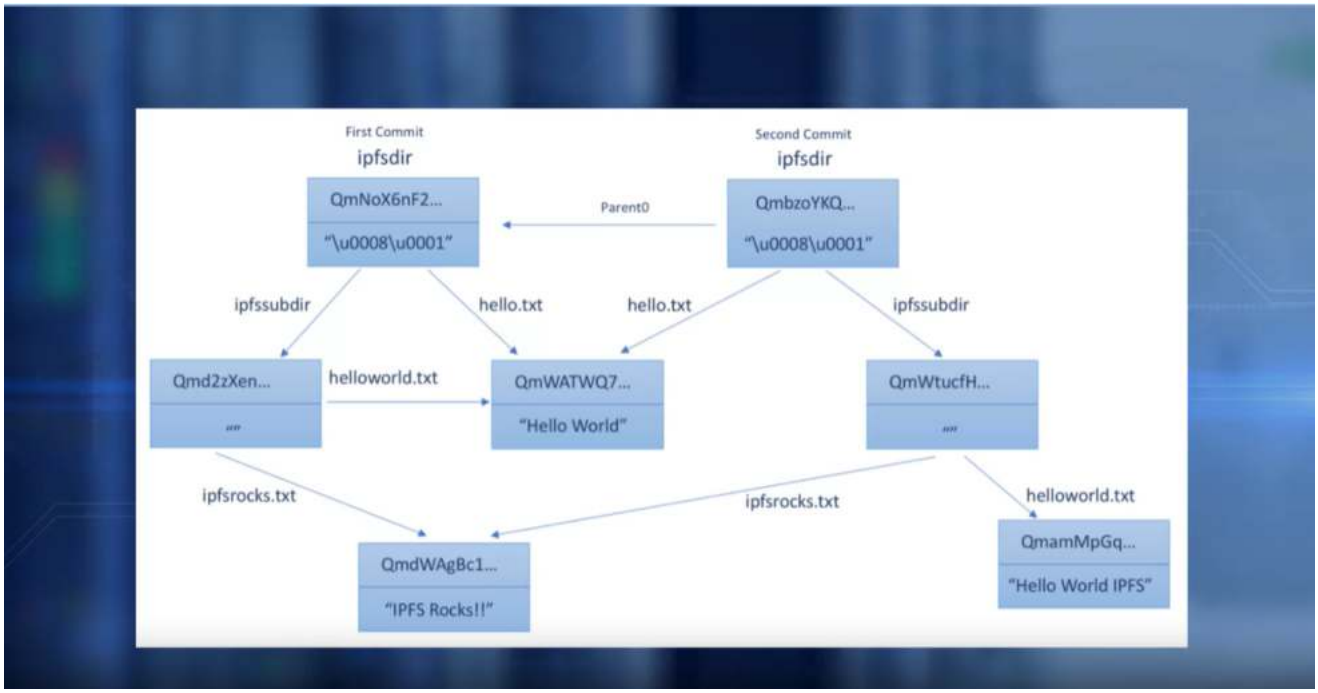


A key can directly map onto the location stored in DHT. Now that we have located the node and the location of the object, how do we exchange the blocks of the file? In a typical IPFS system, DHT resolves the closest location to the key-value. The peer node holding the data blocks are incentivized by a protocol called bit swap. Peer nodes have a want list and a have list, and some form of barter system is formed. The want list is a list of all objects that node wants, and have list is a list of all objects it has available in its position for sharing with others. Any imbalance is noted in the form of a bit swap credit or debt. Bit swap protocol manages the block exchanges involving the nodes accordingly.



The nodes of the network does have to provide a value to the network in the form of blocks. Do you see

that this could be an ideal use case for a digital token? If you send a block, you get an IPFS token that can be used when you need a block. A bit swap protocol has provisions where handling exceptional situation such as freeloading nodes, and nodes wanting nothing, nodes having nothing, and so on. How a multiple versions of files maintain? Multiple versions of a file are maintained using Merkle directed acyclic graph data structure on top of the file system. The basic elements of a version are: the block, list of blocks, tree of blocks representing the file instance, and the commit that is the snapshot of the tree.



This Merkle then also helps to check for any tampering and also to avoid any file duplication. You can observe two commits and of course four directory, for nodes on the left from the first commit, and the three nodes on the right the second commit. This is a dag instead of a Merkle tree we saw in etherium state route. You can also observe the duplication of the Hello World file here. It means that the file is shared between two commits, you see two shared files in this picture. What are the use cases of IPFS, and how about the relationship to the blockchain decentralized system? IPFS can be a stand-alone decentralized file system. It can be complimentary to the existing HTTP based centralized system. We discussed IPFS in the context of blockchain system because it can serve a significant role of decentralized storage for blockchain application that have a lot of data. In this case, the volume and this data can be stored on the IPFS and its metadata on the block chain. In this case, it's of a centralized store. IPFS can be the store that works in tandem with a decentralized ledger technology of the blockchain to create a powerful solution for many storage rich business use cases. We discuss the details of a decentralized storage system that can be used for storing the off chain data for the blockchain. It is used in many genomic data application for storing large genomic data and in dApps such as open law for document storage

4.1.4 Demo

Hello, everyone. This is a gentle introduction to IPFS or Inter-Planetary File System. The purpose of this demo is to get you started with IPFS. This is not an exploration. I would say, you can refer to a fantastic video by Juan Benet, the creator of IPFS. We have linked that in our optional resources section, optional reading section. Please do go through that video. I have already downloaded the tar file, it's here, and I'm going to untar it. Then, I'm going to go into IPFS, and then install it, I'm installing it, and it's as simple as this. So we have it here, and I'm going to initialize it, and when you initialize it, you get a lot of details. You can see that identity, which is about 32 bytes, and you can see it's a base58, just like in Bitcoin, so that we can accommodate more information in there. And then, let's see whether our IPFS system works. All the IPFS commands start with IPFS, and I have here IPFS cat, or just concatenate of the Unix, and that simply displays the file name and IPFS file system. This is the reference to the folder web with the current folder web readme for the IPFS. I'm going

to copy that, and throw it in here, and you will see the readme file being listed. There you go. So, we have been successful in installation. What else can we do? We can add files to the IPFS file system that you have created, and we can also, lets create a file, and I'm going to copy from the desktop. I do have a file created already instead of typing it in. So, I have it here, I'm just going to copy it here, and I can see that, yes, file is there, and let's get concatenated, and see whether, what is in there. I have got two lines and now, in order to make it into an IPFS file, of course, we have to start with IPFS. I'm going to add that file, so that it is added to the IPFS file system. When this resource gets added to the file system, it has got a hash created for it, instead of the path through the your file system. It is going to be a one single hash that exposes it universally and you can see, this 32 bytes is the hash for that particular one. So, I can reference this particular file now by the hash, and you will see that. So, that is the identity for that asset and I can do that, and there you go, did it listed. Okay. So, I have to say IPFS, obviously, and it didn't understand the hash, but when I put the IPFS prefix in front of it, there you go, it is listed. Now, what else can we do here? I'm going to start another term where I'm going to expose the files that I have on the local IPFS to the external world. So, to do that I'm going to start another term, and in that term, I'm going to start an IPFS daemon. So, IPFS, and that will expose the file system to the external world to look at. When I do that, my IPFS daemon is started, and it's available at certain port. I can go now, this starts a gateway, and I'm going to open up a term. In here, I'm going to look at the file that I have. Since we have the daemon, we should be able to look at it from outside, even though it is like a regular URL, it is actually accessing the IPFS file system through them. I'm going to put the hash of the file that we created just now, and you should be able to see. Yes, you're able to see the file system. Let me move the screen, so that you can see it, there you go. So, I have it here, it simply lists the file. This file should be available to all the others to see. Let's now, look at the Web UI. Web UI, instead of the command line, can we look at these things through the Web UI, and that's the next thing that I'm going to show you. The Web UI, it gives you the Peer ID of the node that you created, Agent Version, Protocol Version, Public Key of the node that we created, and some network addresses. One of the interesting things is that, once we've deployed this, you should be able to see the connection, you can see all these are a peer node that are available. You can see that from Houston Texas, and there's one from Ann Arbor, and these are working together, and you'll see these increasing more and more as you go around. You can see that more peers appear here, and you can always go into their node, and see what is available for you to use. When you go here, you can drag and drop files into this, I'm just going to drop this file in here, and it is available to you, so for the whole world to look at and use it. There are many more commands that are available, and also, I'm going to add that hash here of the same file that we've created, it's available for the whole world to use it, it's of no use right now, but if it is a movie or a photograph, the whole world can use it. I'm going to stop the demo at this point. I am going to hand you over to the link that we are provided by Juan Benet, which takes you from here and elaborates on more commands that are available. You can see the config here in Toronto, but I'm going to stop here.

4.1.5 Resources

What is the InterPlanetary File System?
IPFS Alpha Demo
IPFS white paper
An Introduction to IPFS
An Introduction to The Interplanetary File System
The next Internet Revolution — Juan Benet — TEDxSanFrancisco
What's Yeoman?

4.1.6 Quiz

1. Who created IPFS?

- ◯ Juan Benet
- ◯ Joseph Carl Robnett Licklider
- ◯ Satoshi Nakamoto
- ◯ Bram Cohen

2. The IPFS hash depends upon the _____.

- ◯ File name
- ◯ Content of the file
- ◯ Node and file name

3. What command is used to spin up IPFS in your system?

- ○
```
1  ipfs start
```

- ○
```
1  ipfs daemon
```

- ○
```
1  ipfs run
```

- ○
```
1  ipfs init
```

4. IPFS is a ____.

- ○ Content-addressed system
- ○ Blockchain implementation
- ○ Centralized File System

5. IPFS supports version controlling just like git. True or False?

- ○ False
- ○ True

6. How is the location of the file resolved?

- ○ Centralized Hash Table
- ○ Distributed Hash Table
- ○ Merkle DAG
- ○ Location Resolving Protocol

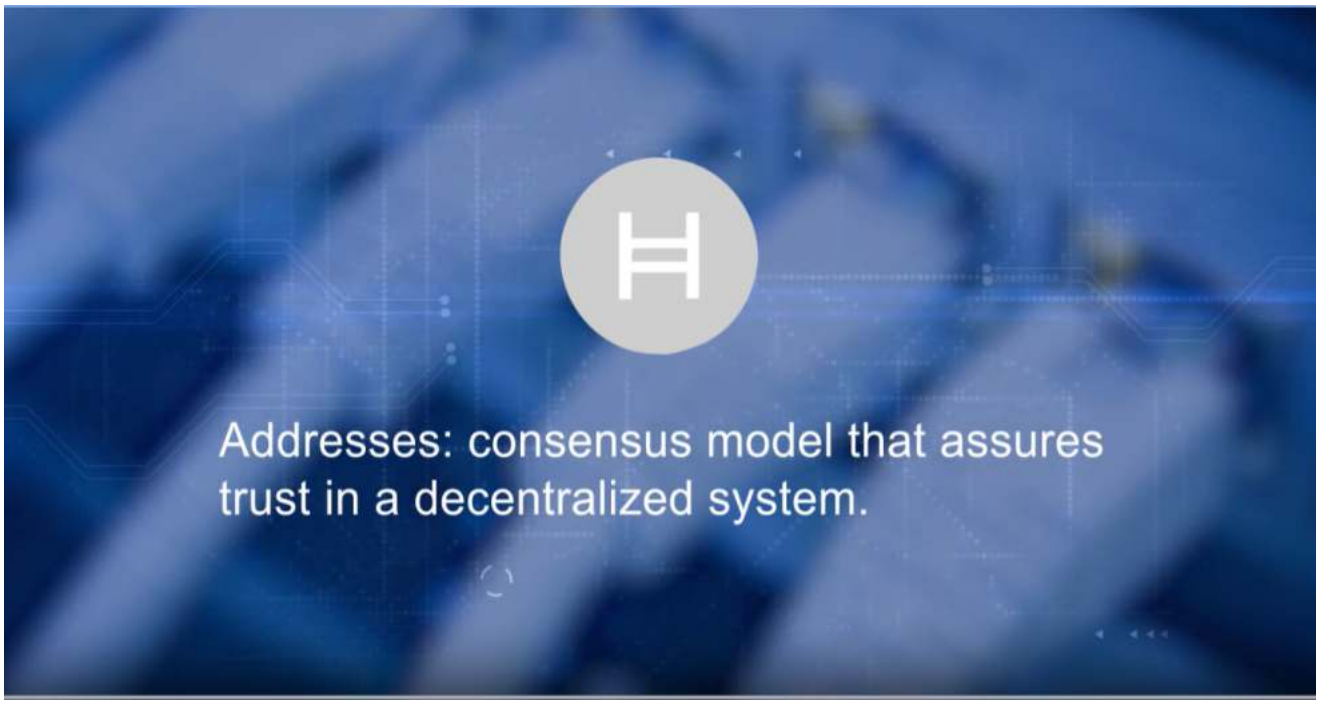
7. Peer nodes are incentivized using ____.

- ○ BitSwap Protocol
- ○ Self-Certification Servers
- ○ Bit Torrent File Distribution Protocol
- ○ Merkle DAG

4.2 Hashgraph

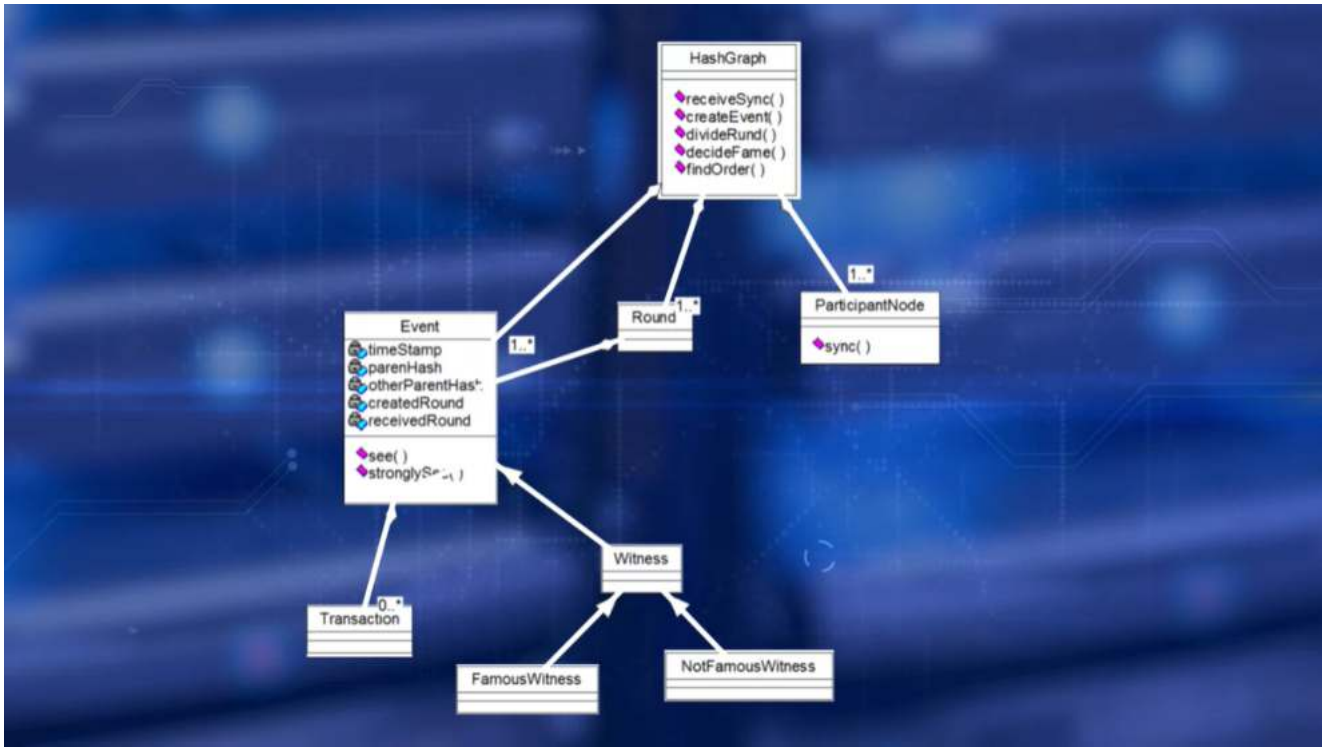
4.2.1 Part 1

Hashgraph aims to address one of the most contentious issues the current public blockchains, the consensus model that assures trust in a decentralized system. Recall that in the case of BitCoin and Ethereum metropolis consensus is achieved by proof of work, POW. What is the cost of consensus with POW? It is indeed quite high. It takes about 78 minutes to confirm a transaction at the time of this recording. And adding a single transaction to the Bitcoin blockchain wastes about 250 kilowatt of energy that can equate to a power used in a typical house for nine days.



Hashgraph is a trust model that provides a consensus layer that addresses the transaction latency and energy wasted fairness and also provides a computationally strong algorithm for besetting full tolerance, and eventual consistency. We'll discuss the high-level details of hashgraph in this lesson. On completion of the lesson you will be able to explain at high level the data structures and algorithms defining the hashgraph algorithm and explain the working of the consensus protocol implemented by the hashgraph. Recall that in a block chain protocol, miners can gather any transactions from the mempool in any order they desire and form a block to validate and then add that block to the blockchain. This results in the competition or race that is solved by the proof of work puzzle, POW. That consumes a lot of computational power. This is all done to reach concensus on the set of transactions and order of transactions. Can we do this ordering of transactions better? In terms of fairness, eventual consistency 100%. At low latency faster transaction confirmation with minimal power consumption. Yes, that's what hashgraph aims to do. The goal of hashgraph is to order the transaction in a decentralized network, addressing fairness, security, latency and energy concentration and also percent and fault. Let's explore it. Here is a visualization of the elements of the HashGraph using our traditional class diagram. Hashgraph is generated by the participants synching or gossiping as you see on the participating node class. Hashgraph is made up of events. It can also be viewed as a rounds of events, as indicated by the rounds class. Each event instance is made up of a timestamp, two hashes or two gossips, and zero or more transactions. An event can be one of the two types of a witness, a famous witness or a non-famous witness. We'll explain all these as we discuss the algorithm.

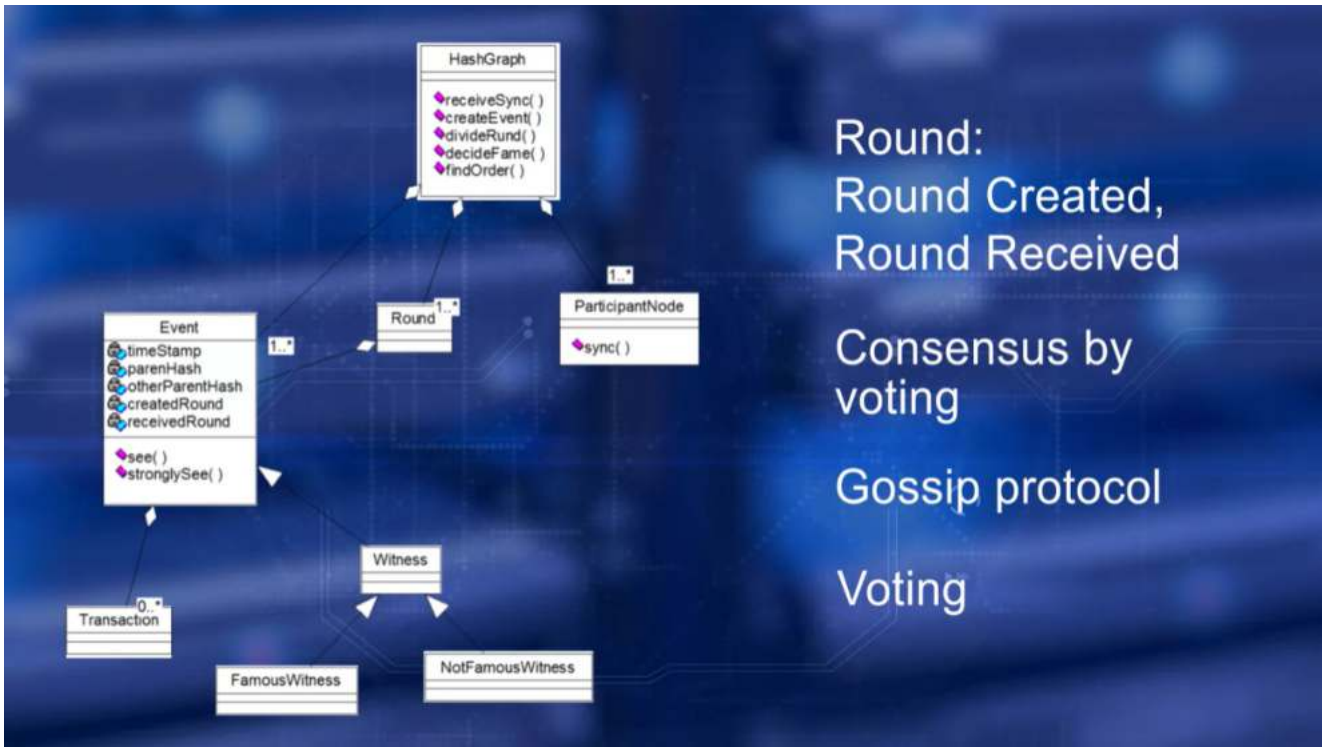
This is a general overview picture to aid your understanding of this complex algorithm. Elements of the hash graph include event, transactions, directed acyclic graph DAG, the hash graph. The witness, famous witness, not famous witness, round, round created round received, consensus by voting by witnesses of the next round casts a protocol. And most important to note, if the voting is derived or computed by the structure of the hashgraph stored in every node.



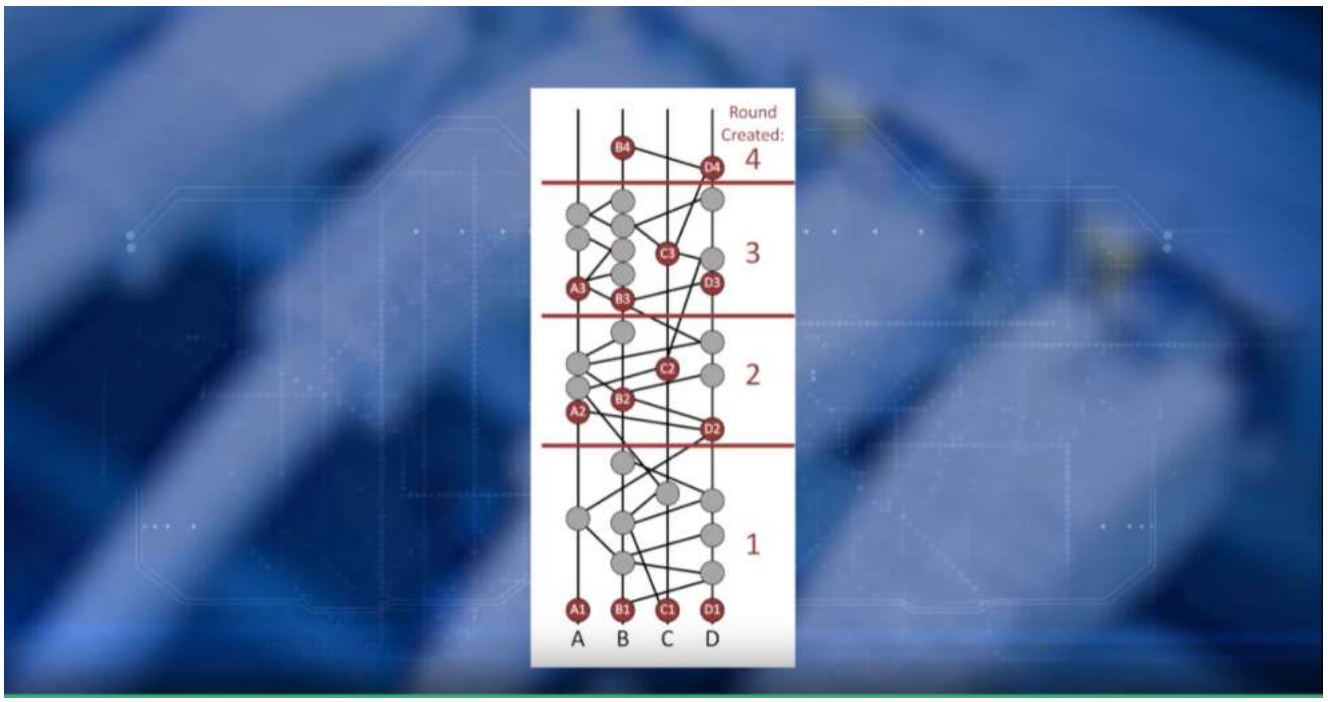
The votes are not transmitted by traditional means. The odds are implied in the graph structure. Since

all the nodes have the hashgraph, they all concur on the vote count and the consensus. Let's discuss the details of a hashgraph. Theoretically, a graph is defined by a set of vertices or nodes and edges connecting them.

The hashgraph has an added element of time. There is a timeline for every participant node in the decentralized network. This is indicated by a vertical line from the bottom to top for every participant on the hashgraph picture. The node of a graph is a data structure called the event and the transmission of an event from one participant to another represents the edge. This transmission is called the gossip sense of transfer or what is known by one node to another node.



This gossip or knowledge gets retransmitted resulting in the concept of gossip about gossip. This information is used for observing the attributes of the graph. This aids the eventual, virtual voting and ordering of events and transactions within the set of events. The graph is also viewed as being made up of rounds of events. You can think of rounds of events as rounds of card game.

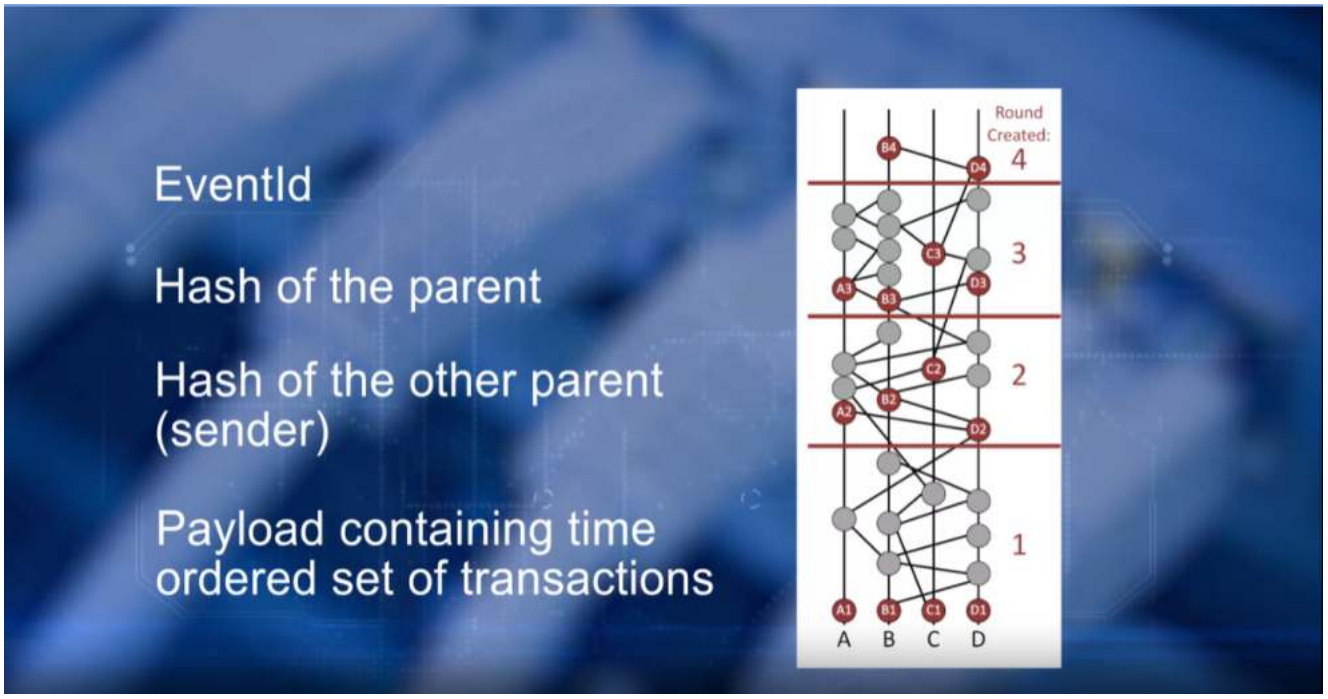


A round defines and delimits an adjacent subset of hashgraph events that can be independently ordered. When you combine independently ordered rounds of events, they will still retain the global order of the transactions

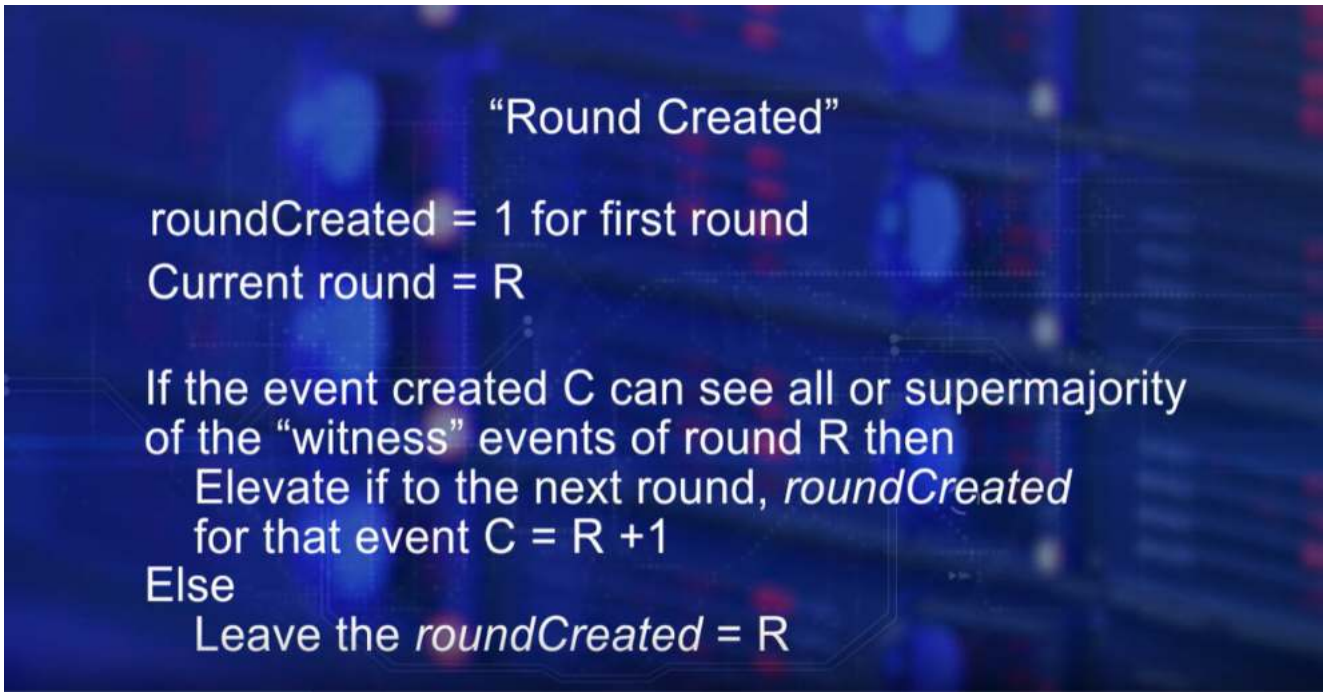
and events. A round consists of all events between the oldest event and the youngest event of the round.

4.2.2 Part 2

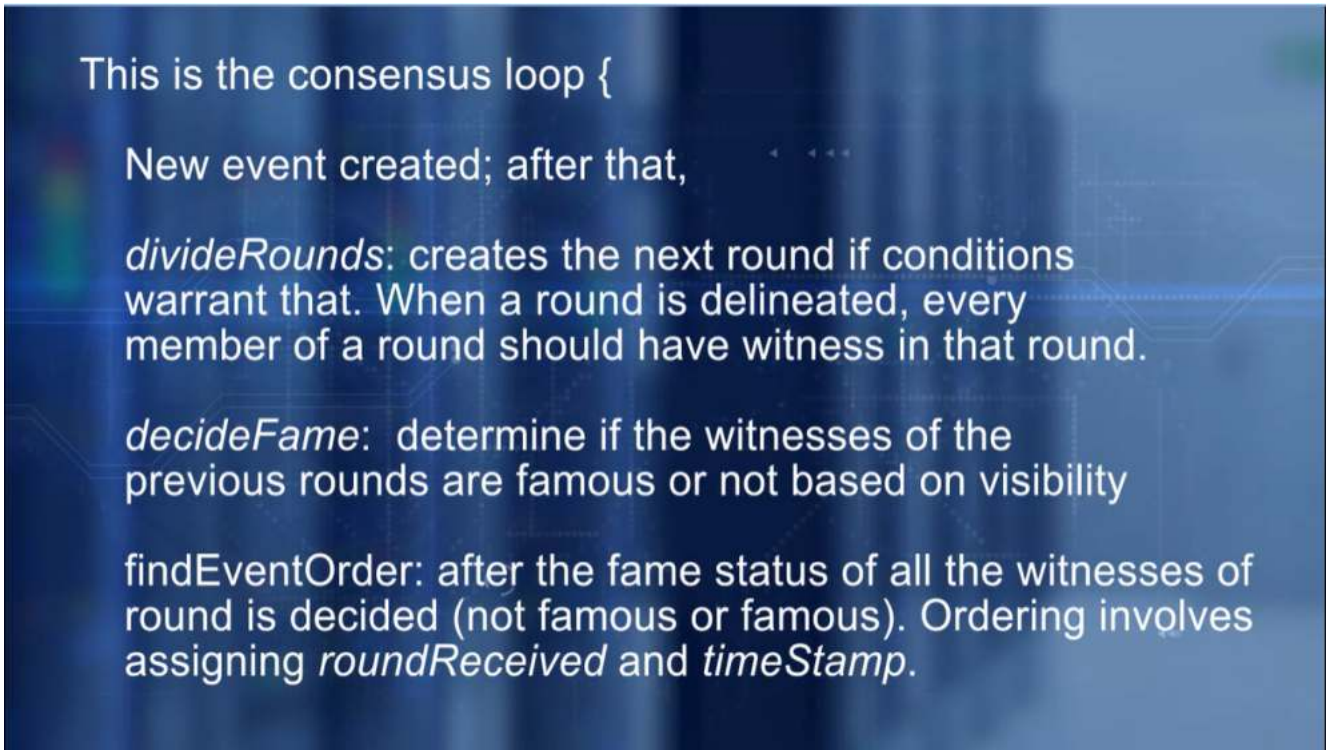
Let's look at the participants, events, and witnesses. Every event has exactly one standard participant and one receiver participant. The event in the graph is represented visually by a filled circle, containing these items or attributes. EventId, hash of the parent, hash of the other parent sender and the payload containing the time ordered set of transactions.



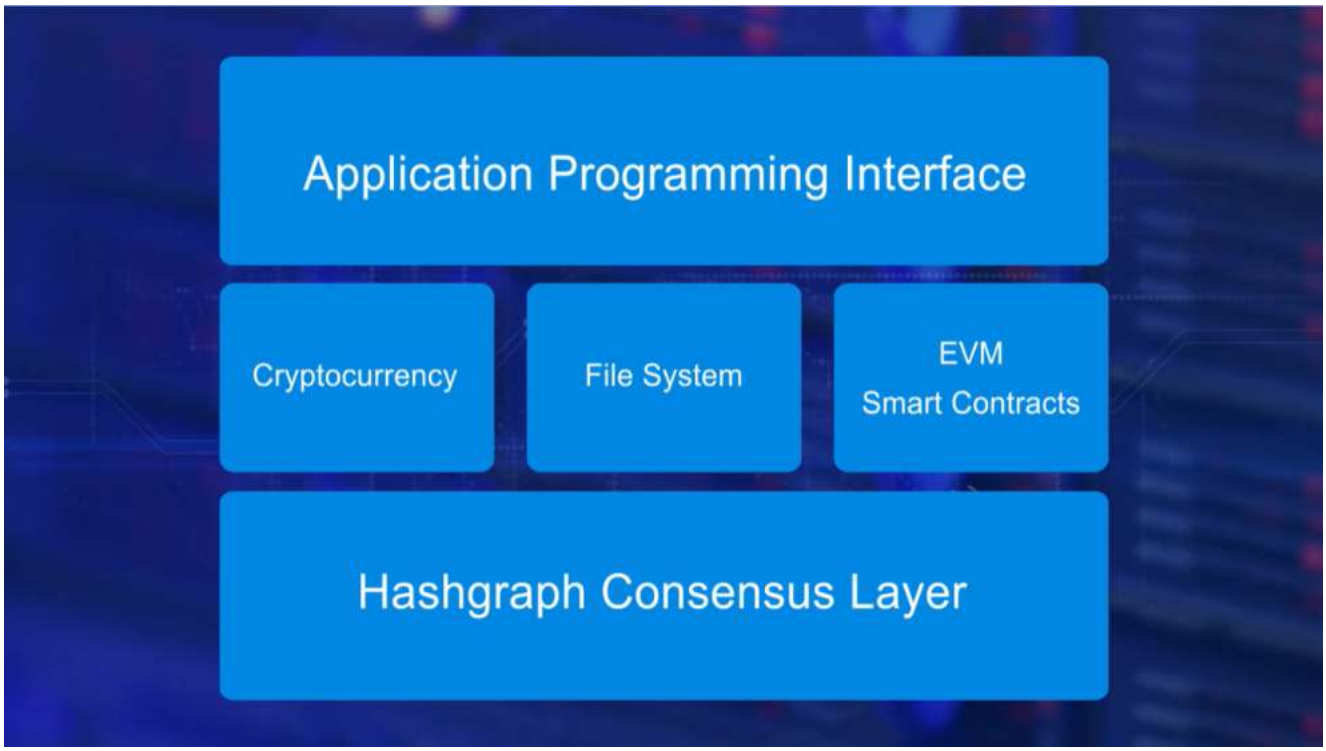
First event in a round, created by each participant, is called the witness, since this event is the witness to the state of the graph. It is also possible that a participant may not have an event or a witness in a round. These witness events get qualified in the future rounds as famous or not-famous witnesses based on their being seen by subsequent events. Every event has associated with it the attribute and integer value called Round Created.



Round Created has a value 1 for very first round. Know that we are using camel notation for the variables. We determined Round Created as follows. Let the current round be R. If the event created C can all or supermajority of the witness events of the round R then elevate it to the next round. And roundCreated for event C has a value of R + 1. Else, leave the roundCreated attribute value at R, and leave the event in that round. We'll be discussing roundReceived attributes of an event, later at which time the consensus on the event is reached. Limenberg's depiction of an instance of a hash graph is a small network of four participants A, B, C and D. A1, B1, C1, D1 are the base events for round one. So they are the witnesses for round one. Recall that witness event for a participant is the first event in particular row. For example for row 3, A3, B3, C3, D3 will be the witness events for the four participants A, B, C, D for that row. Events are created and exchanged among the participants, peers participants updated still within the round one. Consider the oldest event in the first round. It can see the witnesses B1, C1 and D1. Since there is a downward path from the top event to B1, C1, and D1. However, it does not strongly see D1 because it does not cross the event from supermajority of the participant in the path to D1. There is no path from B to C to D or B to A to D in this context. This explains the concepts of strongly C. Now consider D2, the newly created event of the second round. It sees A1, B1, C1, and D1, because there's a downward path from that to these nodes. Among these, it strongly sees B1, C1, and D1. This form the super majority of the witnesses. Thus, D2 is starting a new round. The graph consists of many wrongs and evens within the wrongs are used for computing the vote.



The oldest computed base and the gossips of all gossips and the graph structure, the concept of seeing is equal to voting and the concept is strongly seen is somewhat equal to counting the votes. There is an algorithm for categorizing the witnesses as famous and not famous witness, based on seeing and strongly seeing by the members at the higher level rounds. You should consider other nodes in the hashgraph provided and work on these concepts of sees and strongly sees before proceeding any further. Next, without going to going into too many detail we provide the consensus detail for the hashgraph. This is the continuous, consensus algorithm loop. A new event is created. After that, function divideRounds is executed. It initiates that next round. If the conditions for this node are met, decideFame function is executed. That determines if the witnesses of the previous rounds are famous or not based on the visibility to the current round of witnesses. findEventOrder function is executed. After the fame status of all the witnesses of a round is decided, the votes are counted which really using the graph structure, fame status of the witnesses, and the outcomes of seeing, and strongly see. Ordering involves assigning roundReceived and computation of the median timestamp of the events.



At the end, we get the consensus ordering of the events. That, in turn, determines the ordering of the transactions according to hashgraph creators. This algorithm, though looks complex, enables a transaction rate of hundreds of thousands of transactions per second, thus addressing the scalability issue. The Hashgraph Consensus Layer It runs in the TCP/IP layer of the Internet. It contains an implementation of all the data structures and the algorithms of the hashgraph protocol. On top of this layer are modules for cryptocurrency, file system, and smart contracts. Smart Contracts module currently supports a virtual machine. And the solidity language we learned in course two, smart contracts. On top of these modules is a set of APIs for development of decentralized applications. Hashgraph is the hottest topic in decentralized system currently. Challenge yourself. Compare the consensus model of the blockchain platform we studied with hashgraph, so you too can take part in these discussions.

4.2.3 Resources

Hashgraph Consensus: Detailed Examples
What Is Hedera Hashgraph?
Hashgraph wants to give you the benefits of blockchain without the limitations
Demystifying Hashgraph: Benefits and Challenges
Hedera Hashgraph Thinks It Can One-Up Bitcoin And Ethereum With Faster Transactions
What is Hashgraph and how will it replace the Blockchain?!

4.2.4 Quiz

1. Who invented Hashgraph?

- ◯ Leemon Baird
- ◯ Juan Benet
- ◯ Bram Cohen
- ◯ Satoshi Nakamoto

2. What are the goals of Hashgraph?

- ◯ Order the transactions with high latency.
- ◯ Order the transactions with low eventual consistency.
- ◯ Order the transactions with low latency and minimal power consumption.
- ◯ Better ordering of transactions than blockchain protocol but with high energy consumption.

3. How are Hashgraph edges generated?

- ◯ Edges are generated by gossiping of events between nodes
- ◯ Edges are generated randomly
- ◯ Edges are generated by connecting nodes based on their hash.

4. What are the attributes in an event in Hashgraph?

- ○ Event ID, Timestamp and hash of the parents
- ○ Event ID, Timestamp and hash of all the nodes
- ○ Event ID, Timestamp and hash of current node

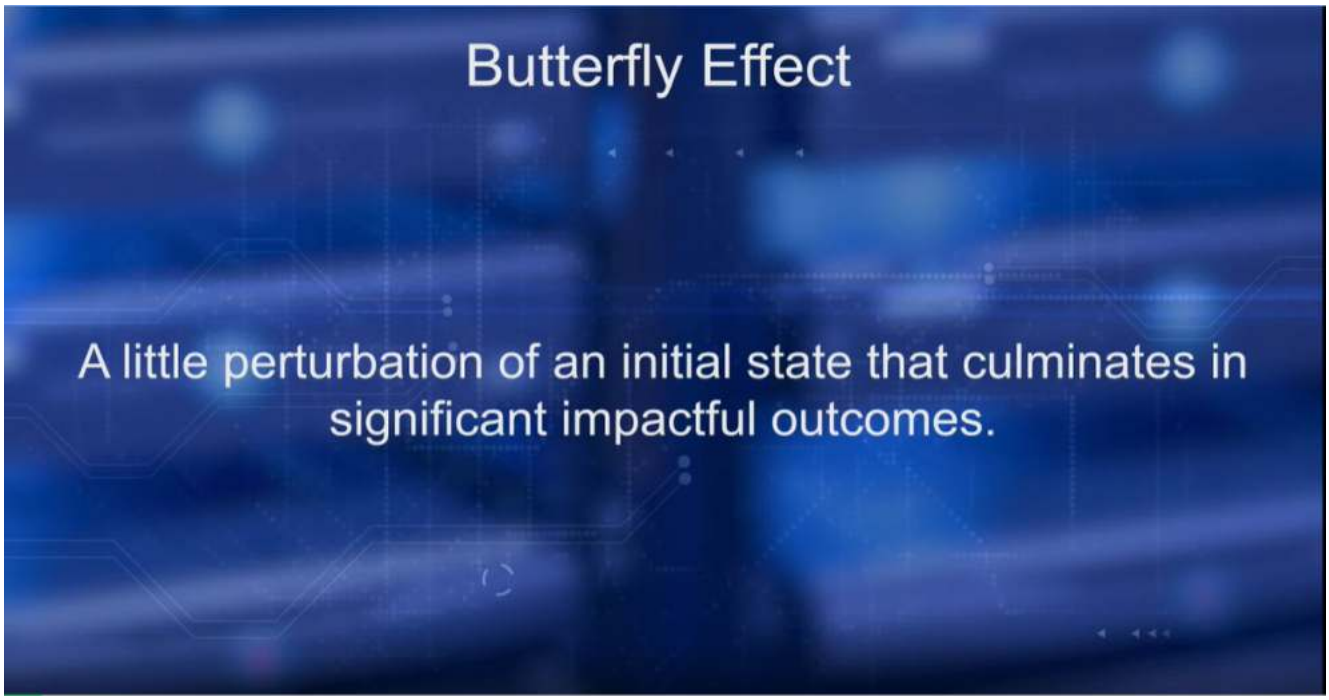
5. How is consensus in Hashgraph achieved?

- ○ Proof Of Stake
- ○ Gossiping information about new transactions
- ○ Practical Byzantine Fault Tolerance
- ○ Proof Of Work

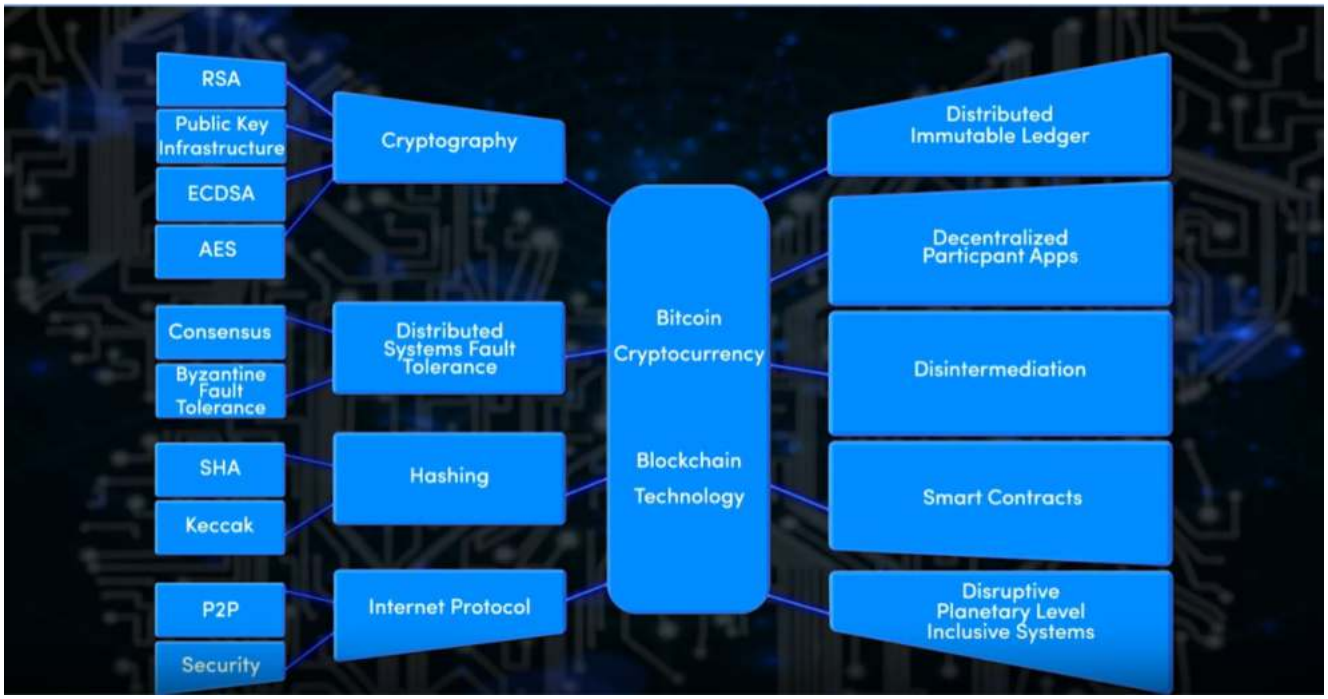
4.3 Blockchain: Social Imperative

4.3.1

Lesson three, blockchain, the social imperative. Butterfly effect is defined as a little perturbation of an initial state that culminates in significant impactful outcomes. This is defined in the context of chaos theory. Bitcoin has had a butterfly effect on technology, literally. Bitcoin creatively synthesized the research result from the last four decades to release and in a way to working model for a peer to peer digital payment system. This singular idea has opened up a Pandora's box of technology, and ushered in efforts and possibly a social revolution. Blockchain innovation of the Bitcoin draws from the research in cryptography and hashing, such as PKI, ECDSA, SHA, and tremendous advancement in Internet technology for P2P transfer. This enable the decentralization, disintermediation, and the DLT of the blockchain. Finite state machines in object-oriented programming languages are used for the development of smart contracts.



Web technology was leveraged for DApp development for blockchains. From these, you can see the transformation brought about by the Bitcoin blockchain. As you can see here, blockchain has numerous diverse applications. You can take part in the blockchain application development and education, design, development tools, and frameworks. Engage in disciplinary research in investigating applications of blockchain, legal, healthcare, government, Fintech and so on. Propose and develop blockchain protocol improvements. Recall, decisions are made using on-chain wording. Explore blockchain alternatives for decentralized systems. Perform fundamental research in blockchain algorithms. Opportunities are endless and there is a role for everyone of you to play. Don't miss these opportunities.



We are going through watershed moment in the evolution of Internet technology. We are witnessing the transformation from centralized to decentralized application enabled by the blockchain technology and cryptocurrencies. These emerging technologies are expected to culminate in newer applications, That transcend demographics and national barriers. It is a social imperative that we educate and enable the creation of an inclusive society where every one benefits from the blockchain applications.

We have created a meaningful set of courses on blockchain to address the need for blockchain education. With the knowledge and skills provided by the courses in this specialization, you should be able to choose your blockchain passion and path to pursue. Participate in a role of your choice, and contribute to a decentralized, inclusive society.

4.3.2 Practitioner's Perspective: Market Adoption

Yeah in 2015, when we really started getting behind watching, we were able to do that because at that point suddenly, and it happened over a course of a couple of months, people, executives, big companies, CEOs, governments started to realize that there was this difference between blockchain and cryptocurrency or bitcoin specifically. Once that occurred, it became safe for companies, like IBM and Microsoft and Intel and others, to get behind the technology as the next evolution of the Internet, as opposed to taking a stand on something like cryptocurrency which a lot of companies in 2015 were not ready to embrace. So, suddenly we were able to go out and say, "We're working on a blockchain, just not on cryptocurrency", and that made sense to people. Three months before I started working on that, you couldn't say blockchain. You couldn't because it would cause mayhem among clients who would say, "No, what are you doing? You're going to ruin our business". Suddenly, a couple of months later after we started, CEOs were saying, "Oh, blockchain is wonderful. We want blockchain. We're not sure about this other stuff, the cryptocurrency but we love blockchain. That's when we started working in earnest in 2015, in the summer, on blockchain technology. Then, we had a decision to make, would we get involved the way we did around Java in the 90s? Say well, we didn't invent Java, but we want to get behind it. It's open source. We're going to get behind that technology. We worked for several months to do that on Ethereum. We said we're going to work on Ethereum and we're going to make it the next Java. Then we ran a ground on that strategy because in 2015, Ethereum's leadership, the community we're really working on some hard problems around the notion of cryptocurrency and public networks that weren't the same problems as we, in industry, were looking to solve. Specifically, the public network side of this schism, this divide. We needed

to focus on scaling, and privacy which now we see in the forms of Sharding, and ZK-Snarks and other kinds of protocols that are coming out. So, for the last two or three years, the public side of this conversation has been working on these really deep hard problems around those issues. So, the best resource, other than this course, is more stuff like this and it's all over the Internet. So, it's not hard to find the information. The most important thing when getting into this blockchain technology is again, focus on a problem to solve and then use blockchain out of the corner of your eye, peripherally to say, "Is that going to give me any insights into that problem that I wouldn't have had if I hadn't had this lens? I am going to dial up that lens". If you look straight at it, you're going to go full blockchain. Going full blockchain means that you're going to start imagining things that don't even need to be solved. You need to focus on the problem. I had an investor in a company I ran years ago that was really smart. I wish they had told me this 20 years before they did. They said, "We don't fund people who are in love with their solution to a problem, we fund people who have an authentic story about a problem because they're the ones that are going to keep on figuring that problem out even if their first idea didn't work." That saying, "Hey, I've got a authentic story about curing cancer or drug use or opioids or anything like that". If you have an authentic story about a problem, then focus on that or at least pay attention to what you pay attention to.

4.3.3 Resources

Commentary: How Blockchain Could Replace Social Security Numbers
Blockchain and U.S. state governments: An initial assessment

4.3.4 Quiz

1. What is the butterfly effect of the Bitcoin blockchain? **1 point**

- ○ An permissioned peer-to-peer currency system
- ○ An ecosystem of innovative inclusive applications
- ○ A cryptocurrency system that will replace fiat currency

2. According to industry experts when did the major businesses realize that blockchain is not just about cryptocurrency? **1 point**

- ○ 2008
- ○ 2017
- ○ 2015
- ○ 2013

4.4 Blockchain Platforms: Key Takeaways

Below you will find a number of key points from this module. Defined terms are underlined. Week One: Permissioned Blockchains Permissioned Blockchain allows only nodes with permission to transact and take part in the blockchain operations.

Permissioned blockchain is also known as a consortium blockchain based on its common usecases in specific vertical business domains such as the auto or food services consortiums.

Linux Foundation's Hyperledger is an ecosystem supporting not only blockchain protocols but it also supports the framework and tools for active engagement and collaboration of developers, businesses and other stakeholders.

The goal of the Hyperledger Project is to promote the development of a safe, reliable, efficient, innovative, quality-driven, open-source components and platforms to support enterprise adoption of the blockchain technology.

Hyperledger has five frameworks

Fabric

Sawtooth

Indy

Iroha

Burrow

There are no cryptocurrency In the Hyperledger protocol

Chaincode is the smart contract code in Hyperledger that defines a set of assets and provides the functions for operating on the assets and changing their states. It also implements application specific rules and policies.

Hyperledger Fabric is a permissioned business blockchain

Here is the list of services offered by the Fabric:

1. Identity services
2. Policy services
3. Blockchain services and
4. Smart contract services

Identity services module manages the identities of entities, participants, and ledger objects such as smart contracts. In the case of Fabric it is called chaincode.

Policy services module manages access control, privacy details, consortium rules and consensus rules.

Blockchain services module manages

the peer-to-peer communication protocol,

distributed ledger maintaining the global state,

global state replicated at many participants, and

pluggable consensus algorithm (PBFT, or POW)

Smart contracts services module provides a secured and lightweight sandboxed environment for the chaincode to execute.

APIs allow application programs to call into the underlying services. SDKs help in code development based on these APIs. CLI is the command line interface for invoking these APIs for testing purposes.

Peers are nodes that initiate transactions and maintain the state of the ledger. There are three types of peer nodes:

1. Endorsing peers receive and validate transactions, sign them, and return them to the creating application. They are called endorsers.

2. Ordering peers collect signed transactions, order them into blocks, and send them to the committing peers. This is also known as ordering service.

3. Commiting peers receive the blocks created by ordering service, validate conditions such as double spending and signatures, and then commit them to the ledger.

Channel provides segregated fabric for a group of entities to transact privately. Channels also provided the ability to support multi-lateral transactions among competing businesses and regulated industries through cross-chain chaincode.

An identity determines the role of the entities and the permissions they have for accessing the resources in the blockchain network.

Consensus is the agreement on the next block of transactions to be added to the chain and the extensive validation and verification of the order and correctness of the transactions including double spend and other conditions.

Microsoft Azure's main goal is to accelerate blockchain deployment. Azure BaaS features include:

· A Collection of ready to deploy ledgers

· Blockchain network with multiple nodes, with hashing, mining, the consensus among the nodes, and the distribution of replicated ledger to all nodes

· Preconfigured network configurations for developing business logic

· Tools and infrastructure in a single place

· Data security and scalability of the cloud platform and

· Single Node Ledger and Multi Node Ledger

Week Two: Decentralized Application Platforms Augur is a trustless, decentralized prediction market platform based on blockchain technology.

Roles participants can play in the prediction market:

1. Market creator who places the prediction query, sets the expected outcomes, pays fees and escrows, establishes the rules, and designates the initial set of reporters.

2. Trader who places the bets on the expected outcomes and takes part in the pre-reporting phase of the process. Traders buy and trade shares that bet on the odds of the outcomes. The trading currency is currently ETH.

3. Reporter who reports on the outcomes. Understand that outcomes do not have to be binary (Yes or No). The reporter can be a designated reporter or an open reporter based on the phase of the process.

Grid+ is a Dapp platform implemented on Ethereum blockchain that has created an energy ecosystem by integrating blockchain and AI.

Energy Retailer: Grid+ will operate as a commercial electricity retailer in deregulated markets.

Smart Agent: At the user household, Grid+ smart agent is a computing device that hosts the software for the blockchain transactions, multi-signature crypto-wallet, with PKI security and off-chain payments for faster confirmations.

Intelligent electricity usage: Electricity trading is a complicated process with many intricacies; Grid+ manages these by coding the efficient price options using smart software.

ERC-20 Token payments: A special ERC20 compliant token called BOLT has been created for payment purposes.

Integration to IOT devices: A Smart agent can be integrated into other intelligent agents such as NEST and electric batteries (Telsa Powerwall)

Remote control: Grid+ enables integration of mobile phones and computing devices to allow remote control of its operation.

Week Three: Challenges and Solutions In Proof of Stake (POS), the full node with the most at stake or most coins is chosen for adding the next block. That is why is it is called Proof of Stake. The idea is that the node with most at stake will not be malicious and risk its stake for forking the network.

In Practical Byzantine Fault Tolerance (PBFT), nodes vote to elect a leader, and that leader adds the next block to the chain. This leader adds the block of validated transactions.

Scalability is the ability of a system to perform satisfactorily at all practical levels of load. Load in the context of the blockchain could be: transaction times, number of nodes, number of participants and accounts, and other attributes of the blockchain.

Escrow is "a contractual agreement in which a third party receives and disburses money or documents for the primary transacting parties, with the disbursement dependent on conditions agreed to by the transacting parties.."

Week Four: Alternative Decentralized Solutions IPFS is a decentralized model for file transfer in contrast to the centralized namespace and transfer provided by the http family of protocols.

Bitswap protocol manages the block exchanges involving the nodes accordingly.

Hashgraph is a trust model that provides a consensus layer that addresses the transaction latency, and energy wastage, fairness and also provides a computationally strong algorithm for Byzantine Fault Tolerance, and eventual consistency. Forks are mechanisms that add to the robustness of the Blockchain framework.

Elements of the hashgraph include:

Event

Transactions

Directed acyclic graph: DAG The hashgraph

Witness

Famous Witness

Round: Round Created, Round Received

Consensus by voting by the witnesses of the next round and

Gossip protocol

A round consists of all the events between the oldest participant and youngest participant of the round.

Bitcoin has had a butterfly effect on technology, its concept has opened up a Pandora's box of technology and efforts ushering in a technological and possibly a social "revolution."