Smart Software Management Tool using Ethereum

M R Ashrit[1], Udayaditya V R[2] , Akshay G

Abstract - *Creating an application using ethereum blockchain, which can be used to manage software project development. Smart contracts help in making the application as per the conditions of the organization. Also, this property implemented in ethereum blockchain makes the application trustworthy, eliminating the involvement of the third party. The application can be accessed and monitored by various users, and the data is not stored in a single place. This decentralizing is achieved with the help of distributed property in ethereum blockchain. Casper in ethereum has proved its efficiency in making all kinds of transactions within the application valid. Different stakeholders of a particular project being built using the application are effectively specified and monitored in ethereum. Comparing hyperledger and ethereum, we used ethereum because the application is essentially business to customer-based. Different developer teams can use the software to build and share the rewards using the application collaboratively.*

Key Words: Ethereum, Smart Contract, Blockchain, Casper, Software Project Management

1. INTRODUCTION

Ethereum is an open-source public distributed ledger system that is based on the original core blockchain idea. Between Bitcoin and ethereum, the concept or the fundamental idea which bought these implementations out in the world was to bring decentralized digital money that was not governed by anyone and give the people the complete freedom to decide and work around with each other. There were a lot of challenges with respect to the existing monetary system, at the same time when you come down to ethereum although it is a derivative of Bitcoin blockchain, which is often referred to as the second generation of blockchain because people have understood what blockchain can be used for and have started using the various capabilities beyond the cryptocurrency mar- ket. As well, the core idea of ethereum built a world computing system which would be a decentralized computing system where all the resources would be shared across as well as the computing effort in itself would also be shared. So it's something that would be the world's first to be centralized distributed computing system. However, the method of release of initial blockchain coins were different whereas in bitcoins you had the mind to get bitcoins and in each ethereum they came up with a monetary system which was basically through an initial coin investment System. Apart from this the other key factors would be the amount of transactions each of them can process. Bitcoin blockchain can process only about 3 to 4 transactions per second and ethereum blockchain can process about 15 transactions per second. And again the time that is taken for creating a block also is slightly different. An average block in blockchain gets created every 10 minutes whereas in ethereum it's about 12 to 15 seconds and also the reward system was also quite different whereas in blockchain it's about 12.5 bitcoins and in ethereum it's only about 5. Some of the key features which make it ethereum so popular or key factors which make ethereum so effective and popular is very important to discuss. There are the four foundational pillars which make ethereum so effective, firstly ethereum is a blockchain based implementation thereby it does require any form of monetary system thereby you have its own internal cryptocurrency which is ether. Apart from that it came up with the idea of smart contracts which is basically an application that would run on blockchain and process all the information present on the blockchain aspect. It also bought in the idea of a decentralized organization that is an organization that proved the foundation knowledge of a governing organization as in the modern society. It also helped to introduce the concept of smart property wherein one could digitally transfer your property without having to face any of the hassles for exchanging the property or validating the documents [12].

Ether is the cryptocurrency which is now the most common is ethereum's cryptocurrency out there is ether which is listed as ETH on most cryptocurrency exchange and that's gaining a lot of popularity of recently. The idea here is to provide for the transaction fees as well as the computational service that you would run on ethereum network itself. This is the difference between the previous cryptocurrencies like bitcoin and ether. There are various applications that will be running on this network and people put a lot of information and one will customize the ethereum network for that one needs set necessities. So for all the operations that one would be running there will be some amount of tokens that are going to be consumed to perform these computations. These tokens are generally referred to as gas. Basically the transaction fees that you would be charged would be for purchasing the gas. Any sort of computation that you want to perform one needs to buy gas. So if you want to perform any operations on ethereum you need to buy gas which is basically going to be deducted in form of ether. If you're providing a high transaction fees then basically the chance of your operation getting completed faster would be high because this is something that miners would take as profit. But if you provide too little a gas then the transaction in itself may fail. It's completely left up to you to decide on the gas use for your computational process. A standard definition the smart contract basically is a computerized transaction protocol which in turn executes a term of a protocol. In terms it's basically an application wherein you write the standard contract rules and it gets executed without

having any change because anything on the blockchain is completely immutable. This is something that is the core foundation of blockchain so if tried to even manipulate a contract it is not possible. For example, say A and B have formed a physical contract and they are trying to put the information on a permission blockchain. This in turn would actually lead to a smart contract. Say if A gives the money and finally B decides not to transfer the property (like a house) then it becomes a loss for A because firstly there could be a challenge to say that B do not receive the money or any situation like that. This can be completely avoided by putting it on blockchain because the transaction that B would be taking money from A. This would be completely visible and it is ensured. Blockchain itself is immutable so this cannot be rejected or it cannot be falsified. Smart contract that says once B has received the money from A and A has transferred the property it would get automatically executed when B would have a confirmation of payment. If someone writes the condition that only if he gets five thousand ethers he will transfer the property, then what would happen is until and unless he gets that five thousand ethers the transaction would not take place. That is the contract would not get executed. This in turn also ensures that falsification from both ends are not met. This is realistic and since it's out there in open and no human intervention can manipulate it. This becomes a trustless system very you don't have to put your trust of someone else. And by making it quite easy for operations to work and quite transparent it is considered reliable. Smart contracts is written in a programming language called solidity.

Decentralized autonomous organization Organizations or group of people who exist entirely on the blockchain are governed by the various protocols of blockchain. Blockchain is an open-source technology which can be manipulated as per the user's requirement. If one is forming an organization consisting of all the stakeholders, he can ensure that they come to a common understanding and then build this organization. One defines these protocols and then create an understanding and then create this organization. Create a unison of multiple long-term smart cracks which are present between these people and any decision that needs to be made or any operation that needs to be done would all be written in smart contracts. Basically they're designed to hold on to the asset and use it in form of a voting system and manage their distribution. If there is someone who holds major assets then he would have a higher priority and again two or more entities within the decentralized autonomous organisation can interact with each other in a fully decentralized and automated manner. Any communication thereby is completely possible and this is something that definitely happens on a regular basis. Each action or a vote is represented by some form of transaction in blockchain. In case if any operation needs to be done it will be in for a transaction. A group of people come and write smart contracts that govern the organization then people actually add funds to this organisation and are given tokens to represent their ownership aspect. So more you invest more tokens you hold and when the organisation begins to operate each member proposals on how to spend this money as per and based on the votes of the members the proposal status is decided. This is how organizations will definitely work out with because this makes it quite easier and also at the same time one does not need to depend on a third party. There is no problem of trust factor thereby making it quite easy and effective as such.

Ethereum moved from consensus based on proof of work to one based on proof of stake. The main differences be- tween proof of work and proof of stake is important to be discussed in order to understand the changes implemented in ethereum. Mining the process through which new coins are released to the network. Proof of work requires nodes on the network to perform a complex mathematical puzzle also called mining as a way of verifying the legitimacy of transactions on the network. This mathematical puzzle has a key feature asymmetry, the work must be hard for the miner to solve but easy for the network to check. All the network miners compete to be the first to find a solution through brute force that requires a huge number of attempts. As a reward for verifying these transactions they're paid cryptocurrency from the network. Once verified, the transactions are placed in a block and appended to the public blockchain and the difficulty of this puzzle increases proportionally to the amount of computing power in the network[8]. That is working on the puzzles that is to say the more miners are there, the more difficult the network makes it to verify transactions and earn the reward. The competitive nature of the difficulty increase incentivizes miners to optimize their ability to solve the puzzle and thus optimize their ability to verify transactions. This makes maintaining the integrity of the network a competitive system with rewards for those that do it well in order to take over the network. One would have to control 51computing power of the network given how large major block chains are that would be incredibly difficult though not impossible. Proof of stake is still an algorithm and the purpose is the same as proof of work but the process is quite different unlike proof of work where the algorithm rewards miners who solve a mathematical problem that creates a new block. The creator of a new block is chosen from a pool of users that have staked a certain amount of cryptocurrency. This means that in the proof of stake system there is no puzzle to complete and so no reward for doing so instead the miners take a feat from every transaction. This also means that because nobody is competing to solve every block there's no massive energy requirement and the penalty for trying to harm the network is the possibility of losing the money you've staked which could easily be upwards of Rs 700,000 in order to take over a proof of stake network. One would need to own 51 percent of the supply of the cryptocurrency on that chain this would be a prohibitively expensive undertaking for any major network and is less likely to happen. Controlling 51 percent of the computing

power something that can already be achieved by mining pools. Proof-of-work miners need a lot of energy to solve their puzzle verifying one Bitcoin transaction. Proof of work requires the same amount of electricity as 1.57 American households use in a day proof of stake lifts the massive energy requirement from the network in favour of a monetary penalty[7]. Developers worried about this energy problem want to switch to the proof of stake method for a greener and a cheaper form of consensus.

2. BACKGROUND INFORMATION

2.1 The structure of a block in block chain

The basic structure of a block in blockchain consists of four sections have been shown in (fig 1).

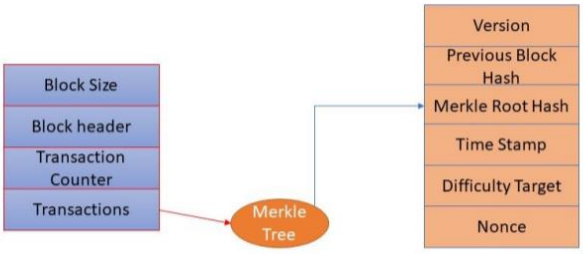

Fig -1: Structure of a block in blockchain.

1) The first section is the block size and the block size is just the size of the block in bytes.

2) Second is the block header and this is formed of several fields.

3) Third is the transaction counter consists of many transactions that are there in the block.

4) Fourth is the Transactions that are processed within this block of the blockchain.

The block header is formed of several fields. There are six fields. The first field is the version and the version is just essentially a number to track software protocol upgrades. Next is the previous block hash which is just a hash of the entire previous block. A merkel root and merkel root is a hash of the root of a merkle tree. All of the transactions are pumped into a merkel tree and it comes out with a Merkel root. If one of the transactions with modify chains removes any change to the overall transaction which were to occur then this local tree will result in a different merkle root hash. Next section is a timestamp and this is just a approximate creation time of the block. Next is the difficulty target and the difficult to target is essentially a proof-of-work algorithm. Finally we have nonce and on top of that it's a fundamental concept in cryptography[6]. The nonce is essentially a random number count application used for a proof-of-work algorithm plus a random no map which just helps to prove the algorithm. Multiple blocks link together. We have free blocks and the number of blocks doesn't actually matter and in these blocks shown in (fig.1) there are four sections which is simplification of what is mentioned above. All these blocks contain the information from above and within these sections the things we are concerned with is the previous hash, the time stamp the random nonce, the root hash, and linked to this we have all of the transactions. The way they link together is actually by the hash of the previous block and that forms the previous hash. This is like a chain, when blocks are linked together that's how blockchain is formed. The entire blocks hash form the previous hash, so this is the structure of the blockchain.

Genesis block First block in block chain is called the Genesis block. Genesis block is statically created just as the starting point. The Genesis block has all the sections of a regular block and the only two things that it doesn't have is the previous hash so the previous hash field is just left blank. The time stamp that has data. Nonce and root hash has some data. The transactions are also non-existent so we have no transaction and this is just empt. Hence genesis block this is the first block in any blockchain implementation and it is statically coded. It is this that is hashed and it forms the previous hash in block number 2.

2.2 Hyperledger

Hyperledger is a project with a lot of different frameworks and tools available. One of the best framework for blockchain is hyperledger fabric. The prominent thing when it comes to blockchain is bitcoin, however blockchain is not just about cryptocurrencies, it basically stores data ie transactions.So programming cannot be implemented here. So building a decentralized internet is the need , which requires software's that will run on this decentralized internet. This requires decentralized software's which are known as dapp's. So ethereum gained popularity because it ensured that it's not just data that can be written , we can also write software's which will run on blockchain.But both etheurium and bitcoin are public blockchain ie all the nodes in the network can access it . But to ensure that the added node is not malicious , we have consensus algorithm. Among them are the Proof of Work (PoW) and Proof of Stake (PoS). Entreprise companies now want their own solutions for blockchain but they face 2 major issues:

1) Expensive mining process

2) Privacy in data sharing so this is where etheruem fails, thereby requires the need of a new blockchain project known as hyperledger. Hyperledger has one framework called fabric. Fabric is a private blockchain. Fabric has the feature of subnets which are called as channels. A new node can be a part of network by enrolling through MSP(Membership Service Provider). Now if some node wants to make some changes. If node A sells an asset to node B. The asset now belongs to node B, so all this data storage and the initiation of this transaction is maintained by an application .So an application alone

cannot change the data of the ledger .This requires some programming ie or a code has to be written in the network, which is possuble by smart contracts. Using smart contracts, an application can send a request to transfer the asset from node A to B .We can write smart contracts in hyperledger fabrics with the help of chaincode. Chaincode can be written in JavaScript, java or go. But fabric is built in go language, so by default go language is preferred for writing the chaincode [2]. We know that ledger is a database which holds data. But fabric has divided this ledger into two parts

1) World state

2) Transaction log JOURNAL OF LATEX CLASS FILES, VOL. 14, NO. 8, APRIL 2020 4 World state fetches the real state of the transaction(who currently owns the asset) whereas transaction log has the history of the entire transaction (who all previously owned the asset) World state can be stored in RDBMS whereas transaction log can be stored in a simple plain file.

2.3 Comparing ethereum and hyper ledger

Ethereum is a decentralized platform that enables business logic to be implemented on the blockchain with the aid of smart contracts. These contracts are executed on a decentralized computer called the Ethereum Virtual Machine which lies at the heart of the ethereum architecture. The developers at ethereum have always advertised themselves as the future of the internet and a lot of people seem to agree too with its simple and generalized protocols. An easy to non scripting language, ethereum has become a platform thriving with use cases of decentralized applications. Hyper ledger is an open source project under the Linux Foundation. The hyper ledger project has various frameworks under its license like fabric aroma etc. All these frameworks enable developers to provide blockchain based solutions for industrial and business-related problems Comparison based on Network types. There are basically three different types of blockchain networks.

1) Public

2) Private

3) Consortium public

1) Public: Block chains are accessible by everybody with access to the Internet. These networks need every ladder to run consensus to validate every transaction that occurs on the network.

These are the main points to be noted about public blockchain

• Participant identity is almost impossible to infer.

• Transactions are viewable by anyone.

• Anyone can be a part of it.

• Open to the entire world Example: bitcoin 2) Private: Blockchains are much more different, they are only accessible by a limited number of participants. As the name clearly suggests, they may or may not even have a cryptocurrency involved with the network as the network is made to be tender to very specific needs.

These are the main points to be noted about public blockchain

• Network members are known.

• Transactions are secret.

• Limited to one company.

• No of nodes validating the blockchain is controlled.

• Number of nodes adding the blocks is also controlled. Example: hyperledger

3) Consortium blockchain: Also known as a permission blockchain is similar to a private blockchain aside from its limited accessibility. It also has levels of permission levied on the network by leveraging trust.

Some use cases require anonymity, while some require privacy and some require a mixture of the two, depending on the characteristics of each participant. However, most business use cases require private, permissioned blockchains-

• Network members know who they are dealing with(required for KYC, AML, etc)

• Transactions are usually confidential between the partic- ipants concerned.

• Membership is controlled.

In the private space, certificates and certificate authority en- ables us to know with whom we are working with. Let us consider a blockchain user, who is going to submit a transaction. So initially, they will have to request a certificate from the certificate authority and that certificate authority will then issue the certificates to the blockchain user's wallet, which will be used by the user while submitting the transaction. When this transaction reaches the required destination, that user will use the certificate which they have got and request the information from that, and then get the information which they are allowed to decrypt, thus seeing what they are allowed to see.

Ethereum is a public blockchain network, if you have a computer and an internet connection.We can participate in the ethereum network which means that if we order to execute some secret transaction, it will be

visible by everybody else on the network and thus the secret doesn't really remain a secret anymore. Hyper ledger on the other hand is a consortium blockchain. It was developed in such a way that we could execute those really secret transactions in complete privacy and confidentiality. Ethereum has its network specific coin called ether ether is mainly used to pay for transactions which is in turn used to pay for the computations committed. New ethers are mined by miners who in turn validate the committed transactions[10]. A ledger has no such specific coin. If a business beams, it is necessary that they need a token or a coin for their purposes. In consensus mechanism ethereum uses a proof-of-work mechanism for its consensus in a proof- of-work based algorithm. A hash function is used to create conditions under which a a single participant is permitted to announce their conclusion about the submitted information and then those conclusions can be verified by all other people in the system. False conclusions are prevented by parameters of the hash function which ensure that false information will fail to compute in an acceptable manner. In the ethereum system, the participants who publicly verify the transaction on behalf of the entire network are rewarded for their participation with newly created ether. A consensus mechanism algorithm called practical Byzantine fault tolerance is used which is also used in other popular blockchain platforms like ripple. In this algorithm, each node on the network maintains an internal state regarding ongoing specific information or status of the network. When the node receives a message, they use the message in conjunction with the internal state to run a computation. This computation in return helps the node to arrive at a decision regarding the validity of the message. After reaching its individual decision about the message, it shares it with all other nodes in the network. A consensus decision is determined based on the total decision submitted by all the nodes. The key difference between proof-of-work and PBFT is that in proof-of-work only one node rather the first node to solve the problem shouts out the answer unlike PBFT which requires all the participating nodes in the consensus to return a decision moving on to smart contracts. Ethereum smart contracts are generally written in ethereum specific scripting languages like solitaire or serpent - logic [13]. Contracts on the other hand are written in chain code aside from that to run smart contracts on ethereum computation. Hypo ledger uses a combination of: and Java Helium as a platform is maintained by the ethereum developer community while hyper ledger comes under the Linux foundation projects.

Consider the following use case:

Voting can be made completely transparent. The voter will submit his information to an identity verifier running on ethereum with the help of smart contracts. After the identity has been verified, a token is generated with the water will enter into the ballot interface This token is unique and solves the problem of double voting in today's world. The voter cast his vote and the sport is registered in a block on the blockchain network, thus maintaining transparency and integrity of the voting system. Ethereum is preferred here as a platform over high polish due to its public nature which is integral for voting. A truck racing scenario: which will enable pharmacists to know where the medicines are being manufactured for Quality Assurance purposes. It is important to note that there is no need for it to be public as that would mean revealing confidential applications between the pharmacist and the man- ufacturers. Every change in the medicines location storage type etc are stored on the blockchain. These can also be automated with the help of Internet of Things sensors, the pharmacist and the patient both can verify the origin of the product and quality is assured. A very similar business model is being used under the name hyper ledger, tool to track fishers after harvest till the time they served on the table at a restaurant. It is believed that in the future, most business-to-business franchisee will run on hyper ledger as they need to execute confidential obligations without passing everything through a central authority. On the other hand ethereum will rule the world of business to customer or b2c enterprise due to its global reach and public nature. Table 1 shows Ethereum Clients

Client	Language	Developers
Go-Ethereum (Geth)	Go	Ethereum Foundation
Parity	Rust	Ethcore
Cpp- Ethereum	C++	Ethereum Foundation
Pyethapp	Python	Ethereum Foundation
Ethereum	JavaScript	Ethereum Foundation
Ethereum(J)	Java	Ether.camp
Ruby- Ethereum	Ruby	Jan Xie
EthereumH	Haskell	BlockApp

Table -1: Ethereum Clients

2.4 Smart Contracts

Smart contracts are fairly straightforward to understand as the term explains itself. With normal contracts, you have to trust PEOPLE to enforce the contract. However with smart contracts the RULES enforce the contract, you don't have to trust anybody. Instead you are placing your trust in mathe- matics. The word "smart" simply means the rules are executed automatically. By normal contracts we mean the contracts that are part of everyday life, for example real estate contracts, insurance contracts, employment contracts etc. We will then compare how the normal contract is different from a smart contract using the same real life example. In normal contract, say Akshay lives in Bangalore books a weeks holiday in Ooty. However one week before he is due to leave, he sees on the TV news that a hurricane might hit ooty the week he is on holiday there. Worried, he goes online and buys some travel insurance from a well known travel agency. The insurance

contract covers natural disasters like hurricanes. Due to the possible hurricane the contract is quite expensive at Rs 11000, but if the hurricane ruins his trip he will be able to claim back the cost of his holiday. At least, that is what the contract says. Unfortunately for Akshay, the hurricane does strike ooty, and he spends 3 days inside his hotel watching rain roll down the window. Once back in Bangalore he puts in a claim on his contract, but it is rejected. He had missed the part of the contract that said he must initiate the claim within 24 hours of a natural disaster occurring. This is the problem with normal contracts. People have to trust other people, and unfortunately people can be untrustworthy, especially when there is money at stake. Now let's look at how a smart contract would help Akshay. Remember that with smart contracts trust is placed in mathematics, not people. Before going on holiday, Akshay buys a travel insurance contract, but this time he buys from a cryptocurrency powered travel insurance company which uses smart contracts. The travel insurance company uses the Obyte cryptocurrency platform to sell insurance via smart contracts. Akshay puts 11000 worth of cryptocurrency into his digital Obyte wallet. The travel insurance company puts Rs70,000 worth of cryptocurrency into their digital Obyte wallet. They put more money in (70,000 vs Bens 11000) because they believe the hurricane is unlikely to hit ooty. The smart contract states that if there is MORE than 1.5cm of rain recorded AND sustained wind speeds of at least 74 mph for 1 minute in Ooty for any 24 hour period during the dates of Akshay's holiday he will be paid 70,000 of cryptocurrency. The money will be made available to Akshay inside his digital Obyte wallet, without anybody making a decision. On the other side, this means that if there is LESS than 1.5cm of rain and there are NOT sustained wind speeds of 74mph or greater recorded in ooty in any 24 hour period during the dates of Akshays holiday, the insurance company will be allowed to take Akshays 11,000 of cryptocurrency. Again, the money will be made available inside their digital Obyte wallet, without anybody making a decision. Unlike a normal contract, this contract is using RULES and a data source which feeds into the Obyte distributed ledger to decide on whether the contract is enforced. This is why it's called a smart contract. Remember that normal contracts use PEOPLE to decide whether or not a contract is enforced. Because neither Akshay nor the insurance company control the data on the weather, they can do business without having to trust each other. Instead, they only need to trust that the data source the contract uses is reliable. In summary, remember that the main difference between a normal contract and a smart contract is that normal contracts rely on people being trustworthy, but smart contracts rely on data being trustworthy. It could be quite useful because it can automate certain things and so if I have some sort of insurance contract which says that if certain event happens then I'm gonna get paid something then we can sort of make that automatic so it just comes right into my account and I don't have to check on anything or call up the insurance company. The problems I've been particularly concerned with contracts that are written for the long term and where people are in long term relationships and economic relationships and they're trying to anticipate what might happen in the future. Which is very difficult to do and to write a contract which takes into account these eventualities they can't do it because the future is very uncertain. Many things can happen that we don't really expect or predict and one don't see how some smart contracts are going to solve that problem. To be more concrete if we have a long term economic relationship it's important that we're on the same page and we understand each other if something unexpected comes up we must have some reasonable way of dealing with it. That makes us both comfortable and that's much more about communication between them. At the time we write the contract than it is about any sort of automated device such as the types of incomplete contracts and the bad incentives that those create. Example of a power plant that locates next to a coal mine and wants to use the coal to burn to make electricity. This is a real example, there's empirical work on things like this and once you've located next to the mine you really want that relationship to work out because they're very costly to ship the coal in from somewhere else. But many things can happen during the course of this relationship so just deciding ahead of time you know exactly what kind of coal how much one should pay and all that is very difficult. Given that the world's going to be changing new sources of energy such as solar powers are going to come along and that's going to affect the industry but we probably can't anticipate that and write that into the contract and so later on we may get into some argument about I want a different kind of coal. In such cases the question of how much should one pay for it and this can be prove to be costly to decide argument. Here is a snap of the smart contract written in solidity for adding the tasks in Dapp (Fig 2.)

```
pragma solidity >=0.4.20;

contract Tasks {
    // Model a Candidate
    struct Task {
        uint id;
        string name;
        string description;
        string time;
        string deadline;
    }

    // Store accounts that have voted
    mapping(address => bool) public voters;
    // Store Candidates
    // Fetch Candidate
    mapping(uint => Task) public tasks;
    // Store Candidates Count
    uint public tasksCount;

    // voted event
    event votedEvent (
        uint indexed _candidateId
    );

    constructor() public{
        addTask("UI Design","Design robust and beautful UI for the application","50", "27th April 2020");
        addTask("Tasks Implementation","Implement taks feature for the app","60", "10th May 2020");
    }

    function addTask (string memory _name, string memory _description, string memory _time, string memory _deadline) private {
        tasksCount ++;
        tasks[tasksCount] = Task(tasksCount, _name, _description, _time, _deadline);
    }
}
```

Fig -2: Smart Contract

2.4 Comparing Proof of Stake with Proof of Work

Mastercard bank transactions are centralized system meaning you must trust one company or entity since they

have all of the history of your Transactions in a centralized ledger. When a transaction goes through a centralized ledger it is verified by the central server because of double entry systems. If you have Rs 7000 in your bank account we can't send 1400. This is because the central server has your transaction history and knows exactly how much you have with the blockchain. Everything is decentralized meaning you don't have to trust anyone and everyone has a copy of all transactions. The decentralized ledger is when a transaction goes through a decentralized ledger it must be verified by the network also called distributed consensus. However your transaction lives with all the other transactions happening in real time in a block. But all transactions on the block must be verified through something called mining.

1) Proof-of-work: Each transaction as a cryptographic puz- zle cleaner and thousands upon thousands of transactional puzzle pieces come together to form a block. This entire block means we solve the process of solving that puzzle or solving the block is called mining. Miners are all trying to solve these blocks using vast computing power. Some miners have a ton of computer solving the blocks, this is the most computers solve the blocks and it results in money. However there's downside in the process of mining miners are unique tons of resource first they're buying expensive hardware called ASIC for application-specific integrated circuits and then they use electricity that burns off at hue. They can't let the computers overheat so in some cases fans are installed to cool down the hardware so you end up using even more electricity. It's a vicious cycle but all of that energy is used to solve the block and after that block is solved that block gets added to the public blockchain but some miners have even located their mining operations in Iceland to take advantage of the cheap geothermal energy so it's a vicious cycle of resources. Wasting all these resources just to solve the block, first it's to validate each transaction but what's the point of having a cryptocurrency if you can't use it as a currency. One gets rewarded when a miner solves the resource intensive task of solving a block that are handsomely rewarded with coinage or Bitcoin say, 12.5 bitcoins and for an ethereum it's 5 ETH. This is also the process in which new bitcoins and ether is created. We must solve these block puzzles because it's all about security. If more the number of miners more security is implemented. So in general proof of work involves the process of mining so that we could validate transactions and the miners get their reward and another block gets added to the public blockchain. Proof of stake In proof of stake miners are instead called validators and there is a block that needs to be generated and there are four validators. Each validator deposits their money to the blockchain to get the opportunity to validate or sign a block. Later one that has the most money he takes up 38 percent of the block elevator, second as a 25 percent stake validator, third has 21 percent and fourth validator has 16 percent with mining. The chances you had of solving the block was dependent on the hardware that you have but the bigger your stake the bigger the chance you have of solving the block so if your validator one have a 38 percent chance of solving the block after some random calculations validator one. With the largest stake wins and gets assigned the block validator one is rewarded not with new coinage but with transaction fees so the rich get richer. So it's dependent on how much you're willing to stake to solve the block. Proof of stake is actually a lot more environmentally friendly than mining with proof of work, all the hardware that you're using to compute in mining actually burns up a lot of energy in order to secure a blockchain. It's estimated that both Bitcoin and an ethereum burn over 1 million dollars worth of electricity and hardware costs per day as part of their consensus mechanism. While proof of work requires miners to effectively burn computational power on useless calculations to secure the network. Proof of stake effectively stimulate with burning so no real-world energy or resources are ever actually wasted.

With proof of work new coins can only be generated by solving blocks and for Bitcoin the coin supply maxes out at 21 million which is supposed to happen in a hundred years from now. While ethereum doesn't have a supply capital. They're planning to partially burn transaction fees like ether deflationary so that it becomes more valuable over time because with proof of stake no new coins can be generated or mined.

2) Proof of stake : It discourages centralized cartels right now bitcoin has a lot of mining cartel problems which is causing the hard fork crisis. If you go to blockchain info slash pools as of today you will see that the top ten mining pools control 83.1 percent of the Bitcoin mining power and out of the ten mining pools eight of them are located in China. There are way too many mining cartels that have too much power proof mistakes would be prevented.

Proof of stake makes a 51 percent attack is virtually impossible. However with proof-of-work mining this is done by having more raw computing power than 51 percent of the entire network. That's a very large energy expense, with proof of stake a validator would have to control at least 51 percent of all of the digital currency in existence which would make it very expensive. Ethereum is also planning to implement steep penalties for people who are trying to duplicate blocks, they'll actually destroy your ether in your stake. 51 percent attacks are extremely expensive so that even a majority of validators working together cannot rollback or finalize blocks without undertaking an extremely large economic loss. A loss so large that a successful attack would likely on that increase the price of the underlying cryptocurrency. The rich get richer the idea is that the validator with a large stake can a contribute to the security of the cryptocurrency and will not endanger his large stake by manipulating the blockchain by doing so he will devalue his fake or even

lose it all together and this all leads back to making proof of stake much more secure and much more stable.

So there you have it the benefits of proof of stake Few problems and possible attacks on the approval stage.

1) Nothing at stake Problem.

2) Initial distribution problems

3) Long-range attack

4) Bribe attack

5) Coin age

6) accumulation attack.

7) Free computing attack.

So proof of stake is by no means perfect but ethereum foundation is developing a distinct POS system called Casper. That's a whole advanced version because it's a mixture of proof of work and proof of stake.

2.5 Delegated Proof of Stake

Delegated Proof of Stake uses reputation systems and real-time voting to create a panel of limited trusted parties. They are called witnesses. Witnesses have the right to create blocks to add to the blockchain and prohibit malicious parties from par- ticipating. Think of it as a representative democracy. Citizens electing officials to represent them when making decisions. In the model, people's vote strength is determined by how many tokens they hold. This means that people who have more tokens will influence the network more than people who have very few tokens. The voting for witnesses is a continuous process, therefore, witnesses have an incentive to carry out their function to the highest standard or they risk losing their position. Overalls, Delegated Proof of Stake is a decentralized consensus model with high transaction rate and low energy consumption.

2.6 Casper

Casper will be a smart contract on the Ethereum network that will implement and monitor proof of stake which is a form of consensus algorithm that builds consensus on the blockchain from owners that stake their clients. Ethereum is using proof of work which is a consensus algorithm that has miners compete for the right to add to the blockchain because only the one that solves the computational puzzle first gets the right to add to the blockchain and the rewarded either with it. This causes an arms race for computing which drives up the need of electricity and server farms dedicated to mining. The average daily use of electricity for ethereum right now can power almost a half a million US homes which is a lot and the ever looming threat of a 51 percent attack makes some programmers worry of a malicious group controlling and gaming the blockchain. In 2014 work began on proof of stake and the first proof of stake with Casper is a smart contract that allows users with about 1,000 ether or in a to be developed validator pool to transfer their stake to Casper. These users will then become validators that will have two functions prepare and commit. These votes are weighed by the amount is either staked and all validators can only vote once per position on the blockchain. Casper will implement two rounds of voting and also Casper will take reports of cheating and penalize bad validators. These two rounds of validator voting is what builds consensus on the blockchain. Let's say there are two ideal validations of three pending blocks. Beginning with prepare, a to be determined proposal method will select a pending block or choose a validator to select a pending block to be prepared. Once that block receives two thirds of staked ether, it is a prepared block. That prepared block will then be voted on in the commit round where two thirds of staked ether will then finalize the block to the blockchain. The process then repeats again with a to be determined proposal method partatizing a blocker validator and a 2/3 vote of stake ether will then prepare the proposed block. The prepared block will then enter the commit stage where a two-thirds vote of staked ether will finalize the prepare block to the blockchain. This two-stage consensus building process will continue on any new pending blocks on the ethereum network. Here we look on how Casper is enforced and to understand enforcement we look at how validators vote and what are the slashing conditions. When validators vote on a pending or prepared block they're paying to make a vote which is essentially a bet. This bet will get paid out and if the block is added to the blockchain. Having validators paid vote incentivizes them to choose the block most likely to be added to the blockchain and discourages pointless voting. Also when validators vote they must reference previous blocks. For preparer validators reference their last prepare and last commit block. For commit validators reference their last prepare block. These references are the links of the blockchain. Casper is a smart contract with game theory based rules that allow for proof of stake consensus. But Casper doesn't enforce its own rules and there aren't any special validators with authority to slash bad validators[9]. Instead both validators and Casper play a role in enforcement as validators can report bad validators and Casper will check and deen the judgment if said validator broke any of the slashing conditions. There are four slashing conditions.

1) The commit requirement Where to commit a block there must be at least two- thirds prepares.

2) Prepare requirement In order to have two thirds prepares at least two-thirds of validators must reference the same prepare block.

3) Prepare Commit Consistency Also in making a prepare validators must reference their last commit block

which should be consistent with the reference of their last prepare block.

4) No double prepares Validators cannot prepare more than once per imposition on the blockchain.

These slashing conditions are all that is needed to ensure proof of state consensus with at least two-thirds of validators votes referencing the previous prepare and commit blocks creating the blockchain.

3. TRANSACTIONS VALIDATION

In the application of the cryptocurrency it is extremely vital that the new information we add is legitimate. We know that blocks are used to store information and information in previous blocks are immutable so then we need to create new blocks if we need to add new information. It's difficult to create a new block. If we recall that blocks with different data must have different hash codes theoretically maybe not but for all practical purposes we can assume that. So there are restrictions hard-coded into the blockchain system and every blocks hash must start with for example five zeros, then if there is already data and one wants to add on to the new block controlling the hash is vital. Controlling the hash comes from a preset algorithm so controlling what the algorithm gives is the real task. Information that we have in a block we have data, the hash of the previous block and time stamp. This is the data we want to store in the new block once decided that one wants to add this specific data to the block one cannot change this plus we have something called nonce. Each block has a special piece of data called the nonce, its sole purpose is for us to have the ability to manipulate the hash code to for example begin with five zeros. Then we give the nonce a random value at first and we see that the hash code does not start with five zeros so we change its value and try again. Essentially this is using raw computing power to guess the correct value of the knots. So we can have hash code that starts with five zeros. People who try to create blocks as finances miners. Many miners could be simultaneously trying to create a block whoever computes the correct nonce first will broadcast it to the network so the other miners hash the blocks data and nonce to see if it returns the right hash with five leading zeros. For example once verified at over 50this new block onto their blockchain. This is a general way of adding data to a blockchain and so far we cannot control what miners decide to add to the blockchain. Hence here's a problem if miners can just add whatever they want how do and we apply this blockchain to crypto currency like Bitcoin then it would be disastrous. If any miner can unilaterally decide to record something like BLTs sent 5,000 bitcoins to me whenever they find the correct knots for this piece of data so each application of blockchain has some additional rules that dictate how data is added onto the blockchain. When miners want to update the blockchain with important data like transactions they must take extra steps to make sure what they're adding is legit. For example you need to make sure that blt actually has 5000 bitcoins in its balance and also that blt actually wants to send those 5000 bitcoins, this is where keys and signatures come in. Each cryptocurrency account has its own unique private key and public key. Public keys are basically accounts address on the blockchain they can be thought of as the user name of the account transactions relating to this account explicitly mention this key. Private keys are only known to the owner of its corresponding public key it's like a password to the public key say BLT wants to send miner bitcoins first blt will need to know miners public key. BLT wants to send the public key of the miner. So basically this just means that BLT wants to send miner 5000 bitcoins and now BLT takes this message and it's private key which is actually another string of numbers, add letters but will use this key to represent it. Hence we add them together, add hash to it to get the signature. The signature is essentially the hashcode of message and the private key.

BLT would need to announce that it would like to make this transaction so it takes the message and the signature which was hash from the message. The BLT prime key and the BLT public key and packages it together, remember miners are the ones with the computing power to create blocks so BLT we need to announce two miners that would like to add this transaction to the blockchain. This transaction is added to the pool of other transactions that other people want to make. None of these transactions are on the blockchain yet. This is just the pool of transactions that want to go onto the blockchain and it is up to the miners to actually add these transactions onto the blockchain. Let's say BLT is lucky and a miner decides to add the BLT transaction to his new block before this miner puts this transaction on his new block he must first verify that this transaction is legitimate. He simply looks at the message and makes sure that BLT has sufficient funds to make the proposed transaction then he runs it through his blockchain software that has algorithm to determine if the signature message and public key in the path package is legitimate. This algorithm is one of blockchains most ingenious things, it is able to verify two crucial pieces of information what the signature was created using the corresponding private key of the provided public key in the package and who the signature was created using the same message as the provided message in the package. Notice how in this entire init process the miner is able to verify that the transaction is indeed the will of BLT without ever seeing BLTs private key. Hence a miner can add multiple transactions on to his new clock, the miner must validate each transaction in the way just described. The miner finds the correct nonce that announces this block with information about all the transactions that he added to it to the network and other miners first verify that the nonce is correct then verify that each transaction described in the proposed new block is legitimate. If all this is correct other miners add this new block to their own blockchains. A new block that contains BLTs transaction has now been added to the blockchain

3.1 Solidity

When the word Ethereum comes up, we try to assume its use of the smart contract. Users who haven't come across regarding these smart contracts are necessary programs that are present inside the Ethereum blockchain. They perform various tasks like sending or receiving Ether or ERC-20 tokens, among other tasks. On the contrary, it is how smart contracts are bind together. Similar to all computer programs, Ethereum contracts are all written in the solidity programming language. While other programming languages are suitable for smart contracts, Solidity is the language. Many of the experienced programmers nowadays use one or more of a bunch of popular new programming languages. Some examples of this type are C (and C++, C , and so on), Python, Java (and JavaScript), Perl, or some other languages. Solidity is created to be easy to grasp for programmers that are already familiar with one or more popular programming languages. Solidity uses an outsized number of programming concepts that exist in other languages. For instance, Solidity has variables, functions, classes, arithmetic operations, string manipulation, and so on. While during a language like C, a programmer would likely create some sort of a "main" function, like "int main(arg1, arg2) //code ", Solidity is developed on a "contract" that's created during a similar manner. Usually, if one wants to find a replacement programming language, they will buy one among the various books available on the topic. For instance, if you would like to find out JavaScript, there are dozens or many books available which will provide the required self- learning training. As Solidity remains entirely new, there are only a little few books available. Therefore the ratings for those copies on Amazon.com seem to be quite mixed if not concluding as total negative. Solidity provides an extensive amount of documentation for a way the code works. However, for somebody new the language, and not already a master programmer, diving directly into the literature might be very daunting, if not a nearly impossible task for many. Practically, the documentation has to be used more as a reference within the same way that one would use a dictionary to seem up a word. One wouldn't read a dictionary from cover to hide to find a speech. If one is devoted and patient enough, though, someone could use the available documentation to find Solidity without an in-depth programming background. It might never be easy, though.

4. ROLE OF BLOCKCHAIN IN SOFTWARE PROJECT MANAGEMENT

Block-chain has proven to be the leading interest to many companies around the world. They are looking to develop next- generation technology and value using blockchain. It ensures secure real-time sharing of data, automation of transactions using smart contracts and solving crucial security problems in many fields including software project management(Fig.4).

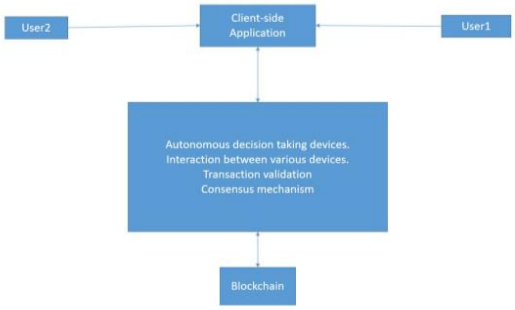

Fig -4: DApp architecture

Block-chain being popularly known for transparency and secure storage of data using the decentralized ledger for storing information with the property of read-only capacity makes it immutable and helps to bring a lot of trusts. With the help of this system is promising in enabling controlling of software development process continuously. Continuity of information flow within the software development teams and between the different stakeholders contributing to the software product. Allowing the process of information flow through consensus mechanism, distributed applications, and storage overlay networks using block-chains help inaccessibility, code of conduct violations, and the link between information flows. There are two types of keys in blockchain technology. They are the public key and private key. The private key is used by individuals to access the network and carry out transactions. The private key is used to generate digital signatures that verify each user. The public key is used by all the users to access the information in the blockchain network. As a result of this, a user can use both the public and private keys in the blockchain technology to increase collaboration and exchange of information efficiently between the different members of the team and also between the various groups in the company. Hence the private key is used by a user to register transactions in the blockchain, whereas public key can be used to derive addresses where information will be sent or received. As a manager, you can assign roles and manage permissions for your team members by appointing a private key to them. It makes the process of software management fast, secure, and cost-effective. Blockchain by managing task dependencies, defining roles and responsibilities of team members, ensuring the safe transfer of data decreases the cost of management to a large extent. It also reduces the risk taken during software development by automating transactions such as sending re- ports, releasing funds and payments, sharing information, task completion and validation, and much more. By automating such processes, we can reduce the risk by reducing the chances of malpractices, theft of information, and hacking. We can track every transaction happening in the company; this helps us in tracing the information flow and monitor the activities of each individual in the process of development. This also helps us in catching the individual indulging in malpractices of

any kind. This spreads awareness of being honest and fair pays off. This also helps in troubleshooting problems in software project management such as a sudden change in demand from the clients, lack of communication between different units of development during the process and also control the rapid changes in product from the beginning of the process till the product has been delivered to the client-side. It helps software process management by supporting the financial network, managing raw materials, and human power to address traceability and visibility challenges. It also contributes to lowering the capital and investments required in developing the software product or managing the development process. Block- chain helps in maintaining every transaction with the help of a distributed ledger, making it more secure, reliable, and permanent. Block-chain is known for representing a trusted economy where the investment process is transparent, and there is no involvement of third parties during any transaction. Smart Contracts stored in the blockchain network have proved its application in validating bank loans, insurance, and portal services. It has exhibited its potential and exponential value in automating payments to different contributors, keeping track of project milestones and on-going activities, and helping in the approval of projects and course of work. This helps us believe in the development process and gain trust in it by transparent multi-functional units in the system. It is using solidity, the language in the Ethereum framework help in building the validation techniques used in smart contracts. This makes it easier to initiate and validate any transactions within the software, developing stakeholders.

Two types of blockchains have different applications based on their authentication procedures. Permissionless blockchain requires authorization between the nodes, which means a node needs to be certified in non-permission, so the requirement of proof-of-work rises. There are other consensus mechanisms such as proof-of-state, proof-of-communication, and many more used in authentication, integrity, and decentralized gov- ernance. In the permission-based blockchain, there is no need for authorization or proof-of-work because nodes are already verified using different permission-based protocols. Ethereum is used to develop DApps (Decentralized Applications) for the integration of different planning processes enabling the small sellers to come into the market and help them sell their products or software on a global platform. A supply-chain network with blockchain helps the developers to reach the outside world and build secure connections between different units to contribute and share information efficiently, by de- veloping software developing assisting applications with all the functionalities like assigning roles, making plans for the project, Designing the project structure, Testing the software, and evaluating the results. It also helps in redefining the prod- uct description and integrate the system efficiently. Managing the process of development becomes a lot easier than before.

Developments such as biometrics through facial recognition and artificial intelligence can authenticate and save the data, which is proven to be lost or corrupted. It keeps the data safer by securing the system using blockchain. Storing relevant data using a shared distributed ledger builds trust in new orders. This helps in managing the various nodes in the supply chain network. Stakeholders and other individuals can also be authenticated using biometrics, and this helps the developers to build advanced mobile applications to verify identity. It is very well used to store high profile identity internationally and proved its reliability on secure and protected systems.

5. CASE STUDY ON ETHEREUM BASED APPLICATION

Decentralized applications (Dapps) are applications without a centralized owner. You'll have heard of Bitcoin or Ethereum, maybe even Augur or IPFS. they're all samples of Dapps. This section deals with the Dapp-AuctionHouse. AuctionHouse may be a decentralized auction platform for any on-chain non-fungible asset. It's decentralized because the business logic is an Ethereum smart contract. For people conversant in web development, there's no "server-side" code apart from the smart contract that's written by us and deployed/ran on the worldwide Ethereum network. Any digital asset that implements the "Asset.sol" prototype is often auctioned off on the platform. An example here might be a registered name on the Ethereum Name Service (think of it as a website name), or the ownership of another smart contract that generates revenue. AuctionHouse may be a straightforward auction app that allows users to make auctions and bid on active auctions. The working of the application is shown in the Fig 5.

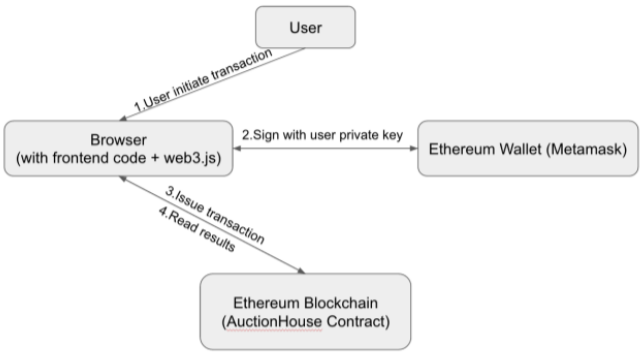

Fig -5: How the AuctionHouse works.

1) The user initiates a transaction (in Ethereum-speak, any state-changing command may be a transaction). For instance, create a replacement auction.

2) Browser signs the transaction with the user's private key stored in an Ethereum wallet. This step requires a further piece of client-side software, usually an Ethereum wal- let. The Ethereum wallet manages the user's private key, so you'll prove that you simply have enough Ethers to put a bid. We use MetaMask. (more on MetaMask later)

3) The transaction gets put onto the Ethereum blockchain. This will be a local blockchain or a hosted blockchain. (An example of a hosted blockchain is Infura)

4) The transaction gets confirmed by the network, and that we can read the result after.

It's very almost like how an internet application works, except the "service-side" is replaced by the blockchain, which talks to the client-side code with help from the Ethereum wallet.

They are big believers in test-driven development. If tests are important for web development, it becomes essential for smart contracts. Security is of the utmost importance, and writing tests may be a good way to form sure your smart contracts are secure. Truffle provides a testing framework built on top of Chai. The TestRPC provides a comparatively fast environment to run your tests, though you've got to restart it sometimes to reset stats like account balance. The most important pain point comes from exception handling. If the smart contract throws an exception, you can't easily catch it within the test code — the test suite simply stops running. This suggests you can't test any exception cases and leaves potential bugs/security holes within the code. Another annoyance is that, since transactions require gas, you've got to recollect to fetch the gas usage and add it to the top result if your test condition depends on the account balance. While TestRPC is great for development, we found the extra step of testing on an area Geth node gave us more confidence. Running an area Geth node isn't that tough, but you ought to remember the safety concerns like unlocking accounts before doing it. you'll read more about it here. One annoyance with local testing is that the changing contract address whenever you are doing a migration, especially if you're collaborating with somebody else. We ended up using Fieldbook as an easily configurable registry and used different network IDs on our localhost, so we will just change the address during a web UI and cargo it dynamically into the front-end code. Truffle gives you an excellent thanks to deploying to the TestNet. you'll simply run an area Geth node that syncs with the TestNet and run 'truffle migrate.' confirm you record the addresses of the deployed contracts because you'll get to ask them in your front-end code. Once you deploy to TestNet, you ought to consider getting to Etherscan and verify your contract. The front-end code is often deployed during a few alternative ways. The foremost decentralized deployment method is through IPFS. It's a decentralized file storage system that also functions as a CDN. However, we tried to try to do this with only limited success. you'll only 'write' (there is not any 'delete'), and therefore the content takes an extended time to propagate through the network. We ended up choosing the more centralized approach of deploying to S3 for now. Security may be a massive topic in smart contract development. Their most significant learning developing AuctionHouse was to modify from a "send" model to a "withdraw" model. This is often because addresses on Ethereum can be both users and contracts. A malicious user can make a bid, and cause the "send" action to her address to fail whenever. This effectively blocks everyone else from making a bid. The fix to the present problem is to vary to a "withdraw" model, where the contract keeps track of the accounting and await users to initiate the "withdraw."

6. RELATED WORKS

Blockchain technology has huge advancement within the sec- tor of research, educational and developing by making use of the platform to innovate, adapt and make a world better place out of it. The workflow of blockchain architecture is shown in Fig 6.

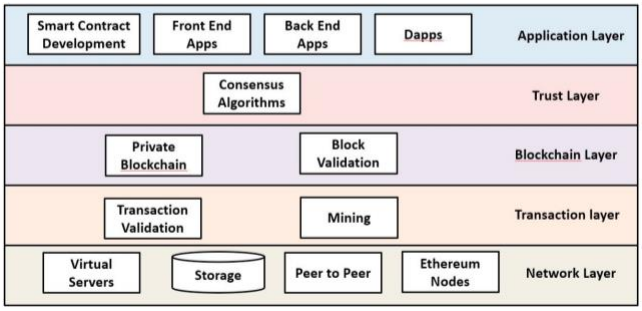

Fig -5: Blockchain Architecture.

Blockchain technology provides several platforms like ethereum, hyper ledger, bitcoin, multichain then on. Hyper- ledger, ethereum are most generally used due to open source, and it supports for various use cases. The fig shows the blockchain architecture working process within the sort of a digital wallet. A block in blockchain mainly consists of knowledge, the hash value of the block, and also the hash value of the previous block. Data stored in each block relies on the sort of blockchain technology. For instance, in Bitcoin, the block stores data about the sender, amount of coins, and, therefore, the receiver information [2, 3]. The hash key's generated using hashing algorithms (SHA 256) this hash key helps in easily identifying each block within the blockchain structure. Once a block is made a hash key's assigned thereto, any changes finished block will intern affect the hash value. The ultimate and important component within the block is that the hash value of the previous block, using this value we build a sequence of blocks, and it'll be the main element for blockchain architecture security. If any block gets corrupted and attempts to impress the blocks to change, then all the opposite blocks which are connected to a sequence will carry misinformation, and the whole

blockchain system is going to be invalid. It's also possible to regulate all the blocks using the proof-of-work protocol. This will allow users to hamper the new block creation process. In bitcoin, it approximately takes 10 min to seek out the required proof-of-work, and to feature a new block to the chain, this task is performed by miners. Miners got to store the transaction fees information received from the block, which they need to be verified as a reward. AA replacement user node joining the peer to see the network will get the complete information of the system. If any new block is made that it's being sent to all the nodes within the blockchain system, each node verifies and validates the knowledge. If the knowledge is correct, then the block is added to the local blockchain in each node. All nodes which are within the blockchain architecture create a consensus protocol that holds the set of rules information; if all the nodes comply with this, then all the blocks are going to be secured[4]. In [6] Clack et al. specialize in the management of the entire the lifecycle of 'smart' legal contracts, whereby they study the creation of legal contract templates and their subsequent use between contracting parties. they are doing not enter analyzing the management of the smart contracts itself, which has been accomplished during this work.In [7], Gervais et al. introduce a novel blockchain simulator to research the safety and performance of proof of labor blockchains like Ethereum. This work differs in being an analysis of a management plane for Ethereum applications employing the utilization of smart contracts through programmed scripts and developed tools interacting with the important Ethereum public test net. The work [8] provides a GUI-based tool for users to simply create safer smart contracts. They also provide a group of design patterns as plugins for developers to reinforce the security and functionality of the smart contract. Our work is complementary, where we offer data-filtering and monitoring templates for developers, which can be used for implementing a management plane in applications using smart contracts. Ethereum programming seems like hardware or financial services programming. The value of failure is high, and you've got to be very careful about the choices you create. You've got to be defensive, assuming every external call can fail. Finally, you ought to only use the blockchain for parts of the app that absolutely require decentralization. This will help simplify your smart contract to reduce security risk.

7. METHODOLOGY

In classic web applications, we use a web browser and connect to a centralized server over a network. All the code and logic we implement lives in this server and all the data resides in a central database. Building our app on the web would give rise to a few problems:

1) The data can be changed on the database, multiple considerations or deletions might occur.

2) The source code residing on the server might also be changed at any time.

Hence, we don't want to build our app on the web. We want to build it on the blockchain so that anyone connected to the network can participate and contribute to the process.

We look at blockchain as a network and a database all in one, instead of having a network, a central server, and a database. A blockchain provides a peer-peer network of computers(nodes), where data and code are shared in. So we just have a bunch of computers that communicate with one another on the same network. Instead of having a network, a central server, and a database, the blockchain is a network and a database all in one. A blockchain is a peer-to-peer network of computers, called nodes, that share all the data and the code in the network. So, if you're a device connected to the blockchain, you are a node in the network, and you talk to all the other compute nodes in the network. You now have a copy of all the data and the code on the blockchain. There are no more central servers. Just a bunch of computers that talk to one another on the same network.

As a substitute for a central database, we can share all the transaction data across all nodes and contain them in bundles of records called blocks, chained together to create a public ledger. It contains and represents all the data in the blockchain. All the data is secured by cryptographic hashing, and a consensus mechanism validates it. Al nodes collectively ensure that data distributed across all the nodes the same. That's one important reason why we are adopting blockchain for our application because we want to ensure that the action performed was correct without any changes to it.

We use this decentralised architecture efficiently in our build build the backend microservices for our application. Solidity provides a strong base to develop and deploy all our backend logic in the form of smart contracts. We then extend the same microservice architecture into the frontend of our application by making use of Components provided by or frontend framework- ReactJS. We now have all the features of a full stack application with all the mentioned advantages of decentralized blockchain integration.

8. RESULTS

As you can see, we have built a full stack decentralized application on our Ethereum blockchain. Here are a few outcomes of the same:

1) Gas is the fee or pricing value required to carry out a transaction(execute a contract on the Ethereum blockchain).

2) It is the network's fuel and the native digital currency of Ethereum block gas limit is the total amount of gas units that can fit into a block. The gasPrice is the gas price per unit in 'wei' units. This is on the Ganache side.

On the truffle side, gasPrice is the price that we will be paying per gas unit to deploy our contracts. gas is the maximum number of gas units the EVM can use to transact the contract. Once the smart contracts are deployed, the backend is fully deployed onto the blockchain. We can see the task being assigned to the team member on redirect of the page. We can verify and refer all accounts, their balance, transactions and contract deployment statuses on the ganache UI. We now have built a fully robust blockchain enabled decentralized implementation for our software project management application. The decentralized application acts as a normal backend with the additional distributed feature. We can deploy our business logic onto our backend by writing smart contracts and deploy- ing them to our local Ethereum backend through our Ganache UI which is our ethereum framework. The decentralized nature of the constructed DApps provide our application with all the mentioned benefits of the blockchain technology. Any relevant actions performed on the webpage creates transactions on our blockchain consuming available ether. This can be parallelly mapped to http client-server communication architecture of our classic RESTful applications.

Smart contracts is our means to enable communication between our client side application and our locally established blockchain. Once migrated they create an immutable transaction entry into our blockchain. This registers an event carried out on the app devoid of errors.

We are currently providing features like tasks, payments and rewards management. These are some of the essential and common features which we come across in all software project management tools.

8. CONCLUSIONS

Development of decentralized application using blockchain (DApps) helps in integrating blockchain with different soft- ware project management applications. Ethereum blockchain is particularly helpful in building applications that enable autonomous transaction validation. It has enabled the secure and reliable interactions between the various devices involved in the network. There is establishment of good pathway between the clients and the developers hence disrupting the major barriers in software development. Autonomous decisions are taken in terms of payment and rewarding hence disabling the involvement of third party. Transparency between the various contributors to the software project and also between the customers and the product managers is made secure with immutable system. The task completion and controlling the development process has effectively being managed in ethereum network. Also the detection of bugs and other trouble shooting problems is properly communicated to respected teams or contributors. This also would be benefical in building a environment where all the developers would be encouraged to increase their productivity as they would have a strong belief in the automated and secure block chain based software project management system. Lack of software management centric consensus protocol is the need of future research. Devices end security has be made more secure because it is prone to security breaches such as DDoS attacks. It has to be implemented and tested before it is delivered to the customer side. The ReactJS as the front-end framework enables us to ef- ficiently manage and control the client-side of the application. It seamlessly gives us the power of a full-stack application bringing in the good will of the powerful, robust NodeJS. We are able to manage the microservices efficiently on the client side using react's efficient Components. We are able to map the backend contracts to the components on the frontend, keeping the microservice architecture intact throughout the application. This further gives us all the power of a full-stack application along all the decentralization discussed in the paper.

REFERENCES

[1] H. Kopka and P. W. Daly, A Guide to LATEX, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[2] D. Vujicic, D. Jagodic and S. Randic, "Blockchain technology, bitcoin, and Ethereum: A brief overview," 2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH), East Sarajevo, 2018, pp. 1-6. doi: 10.1109/INFOTEH.2018.8345547

[3] Vishnu Prasad Ranganthan, Ram Dantu, Aditya Paul, Paula Mears, Kirill Morozov, "A Decentralized Marketplace Application on the Ethereum Blockchain", Collaboration and Internet Computing (CIC) 2018 IEEE 4th International Conference on, pp. 90-97, 2018

[4] Maximilian Wohrer, Uwe Zdun, "Smart Contracts: Security Patterns in ¨ the Ethereum Ecosystem and Solidity", University of Vienna. Wikipedia, "List of controversial elections," 20 September 2016. [Online]. Available: https://en.wikipedia.org/wiki/List of controversial elections.

[5] Ethereum white paper, "A next-generation smart contract and decentral- ized application platform", https://github.com/ethereum/wiki/wiki/White-paper

[6] C. D. Clack, V. A. Bakshi, and L. Braine, "Smart contract tem- plates: foundations, design landscape and research directions," CoRR, vol. abs/1608.00771, 2016. [Online]. Available: http://arxiv.org/abs/1608.00771

[7] A. Gervais, G. O. Karame, K. Wust, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in Proceedings of the 2016 ACM SIGSAC Con- ference on

Computer and Communications Security, ser. CCS '16. New York, NY, USA: ACM, 2016, pp. 3–16. [Online]. Available: http://doi.acm.org/10.1145/2976749.2978341

[8] A. Mavridou and A. Laszka, "Designing secure ethereum smart contracts: A finite state machine based approach," CoRR, vol. abs/1711.09327, 2017. [Online]. Available: http://arxiv.org/abs/1711. 09327

[9] DongchaoGuo, JiaqingDong, KaiWangGraph "Graph structure and sta- tistical properties of Ethereum transaction relationships" Available: https://www.sciencedirect.com/science/article/pii/S0020025519303159

[10] Q. Xu et al. Blockchain-based decentralized content trust for docker images. Multimedia Tools and Applications, pages 1–26, 2017.

[11] S. Porru et al. Blockchain-oriented software engineering: challenges and new directions. In Proceedings of the 39th International Conference on Software Engineering Companion, pages 169–171. IEEE Press, 2017.

[12] G. Destefanis et al. Smart contracts vulnerabilities: a call for blockchain software engineering? In Blockchain Oriented Software Engineering (IWBOSE), 2018 International Workshop on, pages 19–25. IEEE, 2018.

[13] P. Zhang et al. Applying software patterns to address interoperability in blockchain-based healthcare apps. arXiv preprint 1706.03700, 2017.

[14] J. Bell et al. Advancing open science with version control and blockchains. In project management for Science, 2017 IEEE/ACM 12th International Workshop on, pages 13–14. IEEE, 2017.

[15] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels,A. Kosba, A. Miller, P. Saxena, E. Shi, and E. G un. On scaling decentralized blockchains. In Proc. 3rd Workshop on Bitcoin and Blockchain Research, 2016.

[16] A. Kiayias, A. Russell, B. David, and R. Oliynykov. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. Cryptology ePrint Archive, Report 2016/889, 2016

[17] Zhentian Liu, Jing Liu, "Formal Verification of Blockchain Smart Contract Based on Colored Petri Net Models", Computer Software and Applications Conference (COMPSAC) 2019 IEEE 43rd Annual, vol. 2, pp. 555-560, 2019

[18] Kevin Delmolino, Mitchell Arnett, Ahmed E Kosba, Andrew Miller, and Elaine Shi. 2015. Step by Step Towards Creating a Safe Smart Contract: Lessons and Insights from a Cryptocurrency Lab. IACR Cryptology ePrint Archive 2015 (2015), 460.