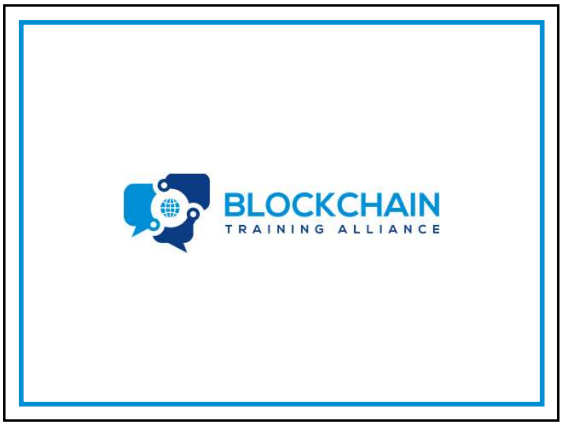
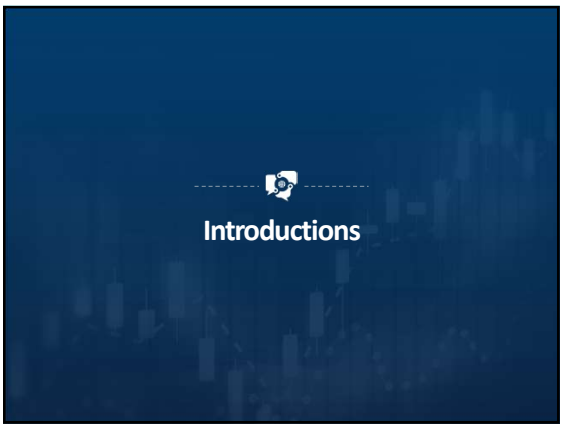

BLOCKCHAIN
TRAINING ALLIANCE

1


BLOCKCHAIN
TRAINING ALLIANCE
Blockchain Architecture
www.blockchaintrainingalliance.com
© Copyright 2018 | All Rights Reserved

2


Introductions

3

4

Who Should Attend?

- Top-Level Management
- Business Decision Makers
- Business Analysts
- Technical Solutions Architects
- Developers and Technical Engineers
- Sales, Marketing, and Branding professionals
- **YOU!!!**

5

What Will I Learn?

- Be able to intelligently answer the following:
  - What is Blockchain?
    - Learn the major concepts of Blockchain
  - How does Blockchain work?
    - Learn the specifics of how Blockchain works
  - How is Blockchain different from what we have today?
    - Learn how Blockchain compliments and contrasts conventional technology

6

What Will I Learn?

- What does a Blockchain app look like?
  - Learn what a real-life decentralized app looks like and what the component pieces are
- How do I design a Blockchain app?
  - Learn the basic principles of designing a Blockchain application
- How do I develop a Blockchain app?
- How do I test a Blockchain app?
- What are use cases for Blockchain?
  - Learn about the many exciting areas Blockchain can be used in

7

What is Blockchain?

Chapter One

8

What is Blockchain?

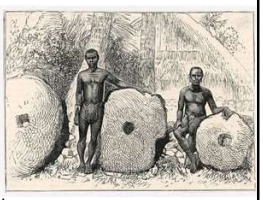
- To understand D.L.T., we need to go back in time
  - 1000 BC
  - Small island in South Pacific
  - Yap Island

- The Yapese people had a very unique form of currency
- Rai Stones – 12' tall, 8,000lbs!!

9

What is Blockchain?

- What happens when money can't be *physically* traded?
  - A ledger is kept
    - A ledger is a recording of all transactions
    - The ledger records:
      - What was exchanged?
      - Who exchanged it?
  - Stones or coins do not have to be physically traded
  - Their ownership can be tracked on a ledger



10

What is Blockchain?

- How did the Yapese manage the ledger?
- **Decentralized Ledger**
  - All tribe members keep a copy of the ledger in their head
  - Everyone knew who owned which Rai stone at any time
- When two parties wished to transact, the would **announce** their transaction to the tribe
- When a transaction was announced, all tribe members updated their mental ledger

11

Let's Review an Example

- Alice agrees to trade Bob her stone by the pond in exchange for all of his cattle.
- Alice and Bob announce their transaction to the tribe.
- Everyone updates their mental ledger. From this point on, they agree that the coin by the pond is owned by Bob until he trades it.

12

What is Blockchain?

Sidebar - A Brief History of Accounting

- Ledgers appear around 5,000 BC
  - Single entry only
- 300 BC – Chanakya
  - First documented accounting standards
- Double-entry ledger appears in 1340 A.D.
  - Track debits and credits
  - Tell the story of a transaction from both / all sides
- Triple-entry ledger appears in 2009
  - aka Blockchain!
  - Debits, credits, and an immutable link to all past debits and credits

13

What is Blockchain?

- The Yapese could have kept a single ledger (a bank)
- One tribe member would keep account of all transactions
- That person would have to be very trustworthy
- The entire tribe would have to the trust the record keeper
- Have to be flexible due to sickness, vacations

14

Let's Review an Example

- Alice tries to corrupt Carol so that Carol's ledger shows that Alice never gave up ownership of the coin.
  - Centralized ledger: only one place to go to cheat.
  - Decentralized ledger: Carol will be outvoted by the rest of the tribe, and her version of the ledger will not be accepted.
- If Alice wants to cheat, she will need a way to convince 51% or more of the tribe to accept an alternative ledger.

15

Centralized vs Decentralized Ledger

- If the single copy of the ledger were changed by any means, wealth would be lost unfairly
- With a decentralized ledger, nobody has to trust anyone else
  - Trustless environment is assumed from the beginning

16

Let's Review an Example

- Some members may have corrupt or incomplete data, this is okay as the majority will not accept it
- Some members may not be present, that's okay. They can get caught up when they come back online

17

Centralized vs Decentralized Ledger

- Lisa own ALL the coins?!?!
- The whole tribe must reach consensus on the truth
  - This is called "Group Consensus"
  - The truth is assumed to be the version of the ledger that 51% or more of the tribe members present agree on

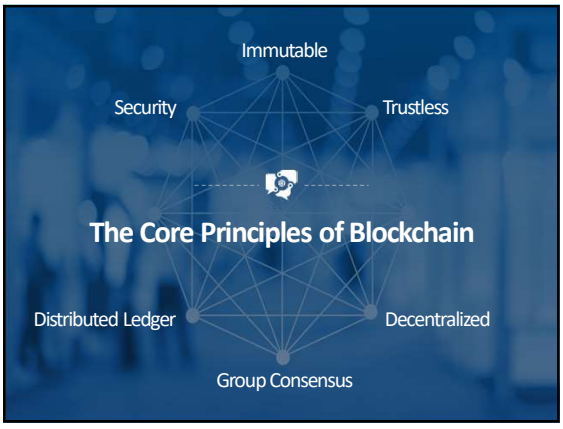
18

- Separation of possession and ownership
  - One day a ship carrying a new coin back to the island sank in the harbor
  - The tribe decided to add it to the ledger and trade it just like any other coin
  - Possession does not equal ownership

19

- Decentralized Ledger == Bank
  - Take deposits, issue credits
  - Decouples possession and ownership
  - Provides a trust-able ledger to all parties
  - Acts a trust broker when two parties who don't trust each other want to trade

20



21

What is Blockchain?

Internet	Blockchain
Email	Bitcoin
Streaming Media	Identity Management
Social Media	???

22

Questions?

•••

23

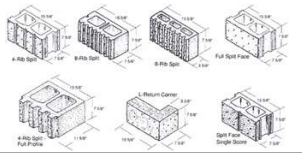
Why not use the Yapese system today?

- **Scalability**
  - Humans can only remember a small number of transactions mentally
  - Even with pencil and paper, limited in how far could scale up system
- Privacy
  - On Yap Island everybody knew the personal net worth of everyone else
  - This is not plausible for modern society

24

Mechanics of Blockchain

- What is a “block”?
  - Let’s say all transactions are recorded on paper
  - Each sheet of paper has 25 lines
  - When a sheet is filled, the tribe will “validate” the transactions on the current page
    - Do we all agree with the data on the page?



25

Mechanics of Blockchain

- Once the page has been validated, it is added to a stack of previously validated sheets
  - Each sheet on the stack can be assumed to be trustworthy
  - Once a sheet is validated it can’t be changed, because computer magic, group consensus!

26

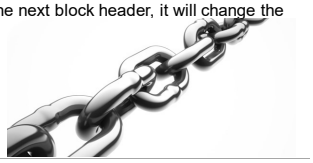
Mechanics of Blockchain

- How are blocks “chained” together?
  - Use of computers & cryptography
  - All data in a block is run through a cryptographic hash
    - We’ll dive into the details later
  - Hashes create a unique output for a specific input
    - Can always recreate if use same inputs
    - Think Black Box
  - Chain is established by embedding the last block’s data into the header of the current block

27

Mechanics of Blockchain

- Changing the data on any block will result in a different hash
  - The new hash will not match the hash in the next block header
  - If you try to change the next block header, it will change the hash of that block



28

What is Blockchain?

- A record keeping system –
  - to record the transfer of "tokens" or "coins" representing wealth (Monetary/currency)
    - Bitcoin and other cryptocurrencies such as Ether, LiteCoin, Monero, etc.

29

What is Blockchain?

- A record keeping system –
  - to record the transactions of importance (Non-Monetary)
    - Update to a medical record
    - Transfer of ownership
    - Training certification on Blockchain
    - Recording important single-party announcements

30

What is Blockchain?

- Three Types of "Transactions"
  - Two or more parties, exchange of monetary value
    - Cryptocurrency
    - Most familiar
  - Peel back the idea of monetary exchange
    - Two or more parties, but no exchange of monetary value
    - Update to medical records, notary services
  - Peel back the idea of two or more parties
    - One party "announcing" an important "event"
    - Supply chain management, business process automation

31

Blockchain is…

- An event tracking system – announcements mark events
- Events can be actionable (Smart Contracts)
- Smart Contracts make a workflow platform
  - Write rules around events

32

What is Blockchain?

33

History of Blockchain

- What's the History of Blockchain?
  - The Byzantine Generals Problem (1982)
    - A number of generals (from the same Army) have surrounded a walled city on all sides. The balance of power is such that if all generals attack at the same time, they will take the city. However, if the generals are not coordinated in their attack, they will lose the city and their campaign.

1982
- Byzantine General Problem

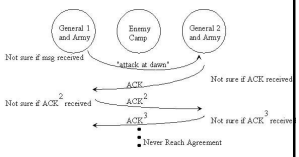
34

History of Blockchain

- What's the History of Blockchain?
  - The Byzantine Generals Problem (1982)
    - The generals use messengers disguised as peasants to carry messages through the city to the other generals. It is critically important that each general be able to trust the content of the message they receive so they can all attack at the same time.

However, the generals have no way of knowing whether a message or a messenger has become corrupt.

1982
- Byzantine General Problem



35

History of Blockchain

- What can the generals do to ensure they can trust the content of their messages?
  - The Byzantine Generals Problem (1982)
    - This was an unsolved problem until 2008
  - Solution:
    - Use cryptography to encrypt messages
    - Ensure more mathematical computation power exists outside the city than inside it
    - This ensures most messages can not be decrypted, changed, and re-encrypted in the time in takes the generals to decrypt the message.

1982
- Byzantine General Problem

36

History of Blockchain

- What's the History of Blockchain?
  - The Byzantine Generals Problem (1982)
    - Blockchain is a digitized solution to this problem
    - "Mathematicians" are financially rewarded for helping "the generals"
    - We refer to this process as "mining", "validating", or reaching "group consensus"

1982
- Byzantine General Problem

37

History of Blockchain

- What's the History of Blockchain?
  - In 2008 a whitepaper is published by "Satoshi Nakamoto" which outlines a solution to the Byzantine Generals problem:
    - **Byzantine Fault Tolerance**

2008
- Satoshi Whitepaper

- Solution is presented as a tokenized currency, Bitcoin is born, starts in 2009
  - Satoshi is anonymous…
    - ...maybe...

2009
- Bitcoin cryptocurrency released

38

History of Blockchain

- What's the History of Blockchain?
  - 2015
    - July 30 - Ethereum goes live
      - Public Blockchain (primarily)
      - Open source
      - Built-in currency – Ether
      - EVM = Ethereum Virtual Machine
        - "global computer" where contracts are executed
        - Nodes & peer-to-peer architecture is largely abstracted away from developers

2015, July
- Ethereum goes live

39

History of Blockchain

- What's the History of Blockchain?
  - 2015
    - July 30 - Ethereum goes live
      - Turing complete Smart Contract programming language
        - Turing Complete – A language which lets developers solve any class of problem in existence given infinite time and resources

Vs

- Turing Incomplete – Can't solve all types of problems, such as problems with loops and iterations (Bitcoin)

2015, July
- Ethereum goes live

40

History of Blockchain

- July 30 - Ethereum goes live
  - Smart Contracts
    - If you're a developer, **Smart Contract == Class**
    - For everyone else….

Workflow with money

A Smart Contract is a set of rules:
- These rules will be called when a certain type of financial transaction is made
- These rules will be called when a certain type of non-financial transaction is made
- These rules will be called when a certain type of announcement is made

2015, July
- Ethereum goes live

  - Much more mature ecosystem
  - Large and very active community

41

History of Blockchain

- What's the History of Blockchain?
  - 2015
    - July 30 - Ethereum goes live
      - Transactions cost "gas"
      - "Gas" is paid for with Ether
        - Gas and Ether are decoupled for price stability
      - Permission-less architecture
        - 100% transparent data, Smart Contract bytecode (source code)
        - Slower than Hyperledger (transaction speed), does not scale as easily
        - Currently used for individual, non-corporate use, decentralized public applications

2015, July
- Ethereum goes live

42

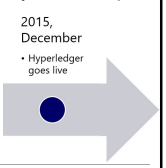
History of Blockchain

- What's the History of Blockchain?
  - December 2015 – Hyperledger goes live
    - IBM and The Linux Foundation
    - Open source
    - Private Blockchain (primarily)
    - "Hyperledger" refers to a suite consisting of multiple technologies
      - Visit https://www.hyperledger.org/projects
      - Fabric (IBM)
      - Sawtooth (Intel)
      - Burrow
      - Composer / Cello
        - Toolset and framework for easily developing business networks

43

History of Blockchain

- What's the History of Blockchain?
  - December 2015 – Hyperledger goes live
    - Modular architecture
    - Chaincode (a.k.a. Smart Contracts)
    - Configurable group consensus and security/membership mechanisms



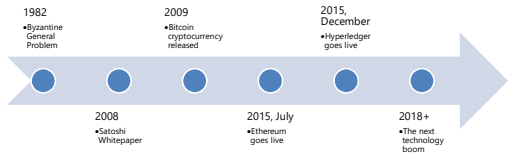
44

History of Blockchain

- What's the History of Blockchain?
  - December 2015 – Hyperledger goes live
    - Designed for use in corporate scenarios, finer grained control over system behaviors
    - Two types of transactions
      - "*Deploy*" and "*Intake*"
    - Client apps communicate with *chaincode* (smart contracts) via an SDK

45

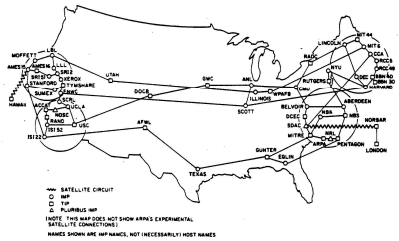
History of Blockchain



46

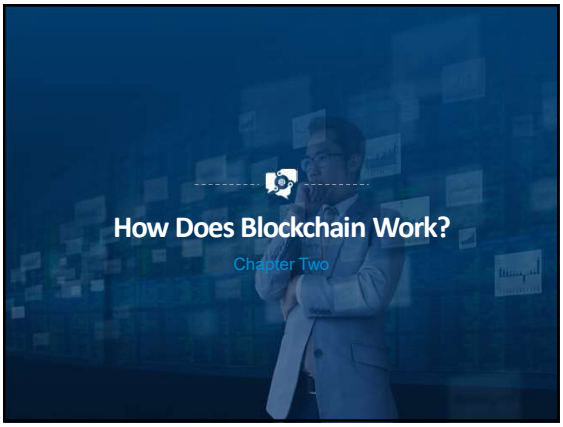
History of Blockchain

Isn't it too late???



47

48



49

Benefits of Blockchain

- What are the benefits of Blockchain?
  - Publicly verifiable
    - Become accountable to customers and end-users (permission-less)
  - Secure
    - Control who sees what data when (permissioned)
  - Quality assurance
    - Track origin of all supply chain components
    - Example – Food safety recalls
    - Smart Contracts as a replacement for middlemen operators
  - Lower transactions costs
    - Removing middlemen reduces cost

50

Benefits of Blockchain

- What are the benefits of Blockchain?
  - Tokenization
    - Create trade-able tokens backed by real-world value
    - Fractional ownership
      - Example - Own 1 car in 1 city, or 100 cars in 100 cities
  - Trade, commerce, and business process automation
    - Smart Contracts as a replacement for middlemen operators

51

Drawbacks of Blockchain

- What are the drawbacks of Blockchain?
  - Very new technology
    - Constantly changing, evolving
    - Not very many trained resources
    - High cost for trained resources
  - Best practices, recommended patterns still being formed
  - Scalability, transaction speed / cost

52

Drawbacks of Blockchain

- What are the drawbacks of Blockchain?
  - Still learning good and bad use cases
  - Stigma and history of Blockchain
    - Cryptocurrency and the dark web
    - ICO/ITO scams
  - Anonymity of origin – Satoshi Nakamoto
  - Data in the blocks

53

Decentralization

- KEY CONCEPT: Decentralization
  - Decentralized - Peer-to-Peer data sharing, hosting hardware owned by many not few, fault tolerant, secure, lower performance
  - Distributed - Solution components spread across hardware assets, solution components and data maintained and controlled by central authorities
  - Centralized - Solution components, data, and control all maintained by a central authority

54

Trustless Environment

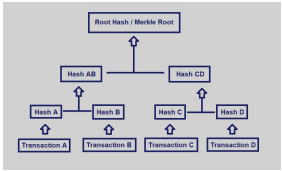
- KEY CONCEPT: Trustless environment
  - Traditional systems assume trust at the beginning, then provide safety measures to block the actions of bad actors
  - Blockchain assumes a trustless environment from the beginning, so requires no additional hardening***

*** - All Blockchain code should be reviewed by a cybersecurity professional!!

55

Trustless Environment

- KEY CONCEPT: Merkle Tree
  - In common Blockchain design, Merkle trees are used to create a **lightweight digital fingerprint** of all the transactions within a block.
  - Provides an 'index' of all transactions on the Blockchain.



56

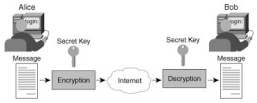
Why not use the Yapese system today?

- **Privacy**
  - On Yap Island everybody knew the personal net worth of everyone else
  - This is not plausible for modern society
- Scalability
  - Humans can only remember a small number of transactions mentally
  - Even with pencil and paper, limited in how far could scale up system

57

Cryptography

- Cryptography can be used to address the issue of privacy
- What is cryptography?
  - The study of how to send information back and forth securely in the presence of adversaries



58

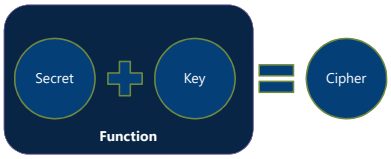
Terms

- Terms:
  - The Secret – The data which we are trying to protect
  - The Key – A piece of data used for encrypting and decrypting the secret
  - The Function – The process or function used to encrypt the secret
  - The Cipher – The encrypted secret data, output of the function

59

Cipher

- The Secret and the Key are passed into the Function to create the Cipher



60

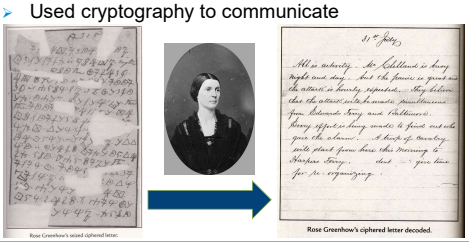
Cryptographic Function

- What is a cryptographic function?
  - A function for encoding or encrypting data to protect the contents from adversaries
  - Simple example function:
    - The Secret - “Blockchain Training Alliance”
    - The Function – Swap each letter in the secret with a new letter according to the Key
    - The Key - “+2”
    - The Cipher = “Dnqemejckp Vtckpcpi Cnnkcpeg”

61

Cryptographic Function

- Real World Example: Rose Greenhow
  - Renowned confederate spy during US Civil War
  - Socialite in Washington D.C.
  - Used cryptography to communicate



Rose Greenhow's ciphered letter decoded.

62

Cryptographic Function

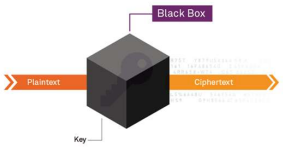
- Public Key Cryptography
  - **Provides identity & transaction approval**
  - Public Key
    - Verify the digital signature of a given key pair
  - Private Key
    - Sign/approve any transaction/action that might be made by the holder of the key pair

Digital Signatures

63

Cryptographic Hash

- What is a cryptographic hash function?
  - A hash is a one-way function, encrypted information CANNOT be decrypted
  - Each unique input generates a unique output



64

Cryptographic Hash

- Why would I want to use a hash?
  - Yap Island – Address privacy concerns by making net worth private with a "digital thumbprint"
    - This would NOT be acceptable:
      - Sally - $418,013.45
      - John - $93,247.89
      - Mary - $9,423.11
    - This WOULD be acceptable:
      - 0376189a740845f75bde8260416b3812ab6d4377 - $418,013.45
      - 5753A498f025464d72e088a9d5d6e872592d5f91 - $93,247.89
      - 94F85995c7492eec546c321821aa4beca9a3e2b1 - $9,423.11
    - Nobody knows Sally's net worth, but Sally can always prove which account is hers

65

Hands-on

- Cryptographic Hashing Hands-on
  - Let's try it out (using SHA256)
  - Go to: www.anders.com/blockchain/hash.html (or: bit.ly/2HnJS6O)
  - Demo:
    - Let's eat, Grandma
      - 45b09eea07b7896d836308e89d01986ea92227ea41d5239dc42650c66393cc01
    - Let's eat Grandma
      - aab9185e406446c4700be80ae8c274778a15e9f2914303ae803ecfa2eca19b5a

66

Cryptographic Hash

- Why would I want to use a hash?
  - Landlord and tenant can compare lease documents
  - Verification of software
    - If there's ANY difference between what should be and what is, it's easy to identify
      - Malware which makes slight changes to the original codebase can be easily detected
      - Important for self-driving cars, automation, IoT, etc..
  - Instantly compare two or more LARGE volumes of data to ensure they're the same
    - Has 1 bit been flipped in a 100TB file?

67

Questions?

68

Summary of Blockchain

- Blockchain = modern day Rai Stones
- "Announcements" are made by nodes/computers
- Each announcement is recorded on the current "block", on all nodes
  - Announcements can be many things: multi-party monetary, multi-party non-monetary, single-party announcement
  - The "block size" determines when a block is full
- Announcements are recorded on a block until it fills up

69

Summary of Blockchain

- When a “block” is filled, it is validated through group consensus and added to the “chain”
- All transaction data on a chained block is assumed to be trustworthy
  - This data has not been tampered with
  - This does **NOT** speak to the validity of the data itself

70

Proof of Work (PoW)

- Proof of Work Consensus
  - When a block is full, each node competes to solve a guessing game problem
  - This problem is non-computational
    - Random guesses are most efficient
    - Example: Guess a number between 1 and 1000
  - Miners try to guess the “nonce”
    - All block data plus the current guess (nonce) are run through a cryptographic hash
    - **If the result matches the current level of “difficulty”, the miner has guessed the right answer**

71

Demo

- Mining demo
  - https://anders.com/blockchain/block.html
  - Block data = “ThisIsMySampleBlockData”
  - Difficulty = find a nonce such that the hash of the block data and the nonce starts with “0000” <- four zeros

72

Proof of Work (PoW)

- Proof of Work Consensus
  - Difficulty example: Find the nonce where the hash output has #### of leading zeros
  - Difficulty is adjusted by the network to correspond to load
  - Any miner who thinks they have the correct answer will share it with all other miners
  - Miners will confirm the answer is correct by using the nonce with their block data to try and get the correct result
  - Guessing is hard, verification is easy
    - Guessing uses computing power, this incurs a real-world cost
    - Example: square root of 9723

73

Proof of Work (PoW)

- Proof of Work Consensus
  - If *51% or more of the miners agree with the proposed nonce, the transactions on the winner's block are considered to be correct
    - *This percentage can vary depending on the Blockchain in question
  - The miner with the correct answer will be rewarded
    - Usually the reward is given in platform tokens
  - If the majority of miners do not agree with the nonce, no reward is given

74

Proof of Work (PoW)

- Proof of Work Consensus
  - Work performed for no reward = cost with no return
    - Miners have no incentive to cheat, strong incentive towards honesty
    - No way to recoup costs when cheating
  - Adding a node to the network increases security by 1/N
    - N = number of nodes on network
    - Increases transaction time by 1/N
      - ALL nodes must validate!

75

Demo

- Mining demo
  - Block Data = “ThisIsMySampleBlockData”
  - Nonce = “24725”
  - Network difficulty: 0000
  - SHA256 hash result of:

00002b1b5e98e4257a504ab55a937ef678de4547fe88
4d19488927c1e69b173b

76

Demo

- Mining demo
  - Let’s try to cheat...
    - Block Data = “ThisIsMyHackedSampleBlockData”
    - Nonce = “24725”
- SHA256 hash of:

eae8ad1ce120fb96c790765efa500f81ae633873d3f4069acc9f2957fe3b6a13

77

Questions?

78

Summary Proof of Work (PoW)

- Proof of Work Consensus
  - When a block is full, each node competes to solve a guessing game problem
    - This problem is non-computational, random guesses are most efficient
    - Miners try to guess the "nonce"
      - All block data plus the current guess (nonce) are run through a cryptographic hash
  - If the result matches the current level of "difficulty", the miner has guessed the right answer
  - The miner with the answer shares it with all other miners, Miners will confirm the answer is correct by using the nonce with their block data to try and get the correct result

79

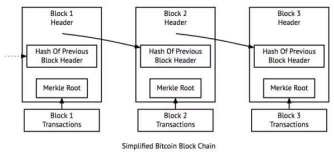
The "chain" of Blockchain

- What creates the chain?
  - Block header
    - Contains information about the block
      - Platform version, timestamp, difficulty level, nonce, etc.
      - Also contains the hash output of the previous block data
  - All data from the current block, including its header is hashed
  - The hash output for the current block will be stored in the header of the next block
  - All data from that block, including the header, will be hashed

80

The "chain" of Blockchain

- The "chain" of Blockchain
  - If I hash the data of any block, and the output matches the hash value from the header of the next block, I can trust the data has not been tampered with



Simplified Bitcoin Block Chain

81

Demo

- Blockchain demo
  - Go to https://anders.com/blockchain/blockchain.html
  - Block Data = “ThisIsMySampleBlockChainData”
  - Nonce = “26050”
  - Network difficulty: 0000
  - SHA256 hash result of Block 1: 0000418e886d6630816c1ee1650ad0ddbad770cae5cb0777b9f5fe0586a4a1fc

82

The “chain” of Blockchain

- To summarize
  - The hash output will be stored in the header of the next block, etc.…
  - Changing the data in any block will result in the hash of that block not matching the hash value stored in the header of the next block
  - The next block will need to be changed too…
    - ...and the block after that, and the block after that, and the block after that…

83

Proof of Stake (PoS)

- Proof of Stake Consensus
  - Proposed as an alternative to Proof of Work
  - Attempt to overcome scalability concerns imposed by PoW consensus
  - Removes the guessing game from consensus
  - Mining no longer requires specialized and powerful hardware
    - Many feel that specialized hardware requirements lead to centralization
      - Blockchain is about de-centralization
  - Less energy intensive form of consensus
    - Addresses concerns about “green” mining

84

Proof of Stake (PoS)

- Proof of Stake Consensus
  - How does it work?
    - Let's pretend there's a new casino game – Honesty
      - Each hand of Honesty costs $10,000 to join (stake)
      - Every honest player will always win $25 each hand (net gain $25)
      - A winning player keeps their earnings and gets their stake returned
      - Every dishonest player will not only miss out on the $25 winnings, they also loose their stake (net loss of $10,025)
        - Small upside for being honest, large downside for being dishonest

85

Proof of Stake (PoS)

- Proof of Stake Consensus
  - Ok, but how does it really work?
    - Block of transactions created
    - When it's time for group consensus, all who wish to participate lock up funds in a stake
    - A random 'player' is selected
    - That player's block data is shown to all other participants
    - Other players stake on the validity of the block transactions

86

Proof of Stake (PoS)

- Proof of Stake Consensus
  - Ok, but how does it really work?
    - If the majority agree with the proposed block, the random player is rewarded, as are all who staked on that player
    - If the majority disagree, the random player gets no reward AND loses their stake!
    - Then a new player is randomly selected to share their block data

87

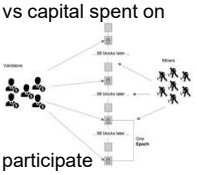
Proof of Stake (PoS)

- Proof of Stake Consensus
  - No “computing” is ever performed during consensus, only staking/wagering
  - Any kind of device can stake, regardless of computing power
  - Some argue that this also leads to centralization as only players who can afford to stake are able to participate in consensus

88

PoW vs PoS

- PoW vs PoS
  - Work for a reward vs make a safe bet for a reward
  - Security vs Speed
  - Centralization vs Decentralization
  - Proven vs New
  - Capital spent on hardware vs capital spent on staking funds
- Ethereum – moving to PoS (test net only)
  - 0.1.0 Released May 2018
  - 2 year roll-out, 1500 eth to participate



89

Other Consensus Mechanisms

- Other consensus mechanisms
  - Proof of Activity
    - Hybrid of PoW and PoS
    - Empty template blocks are mined (PoW), then filled with transactions which are validated via PoS
  - Proof of Burn
    - Coins are “burned” by sending them to an address where they cannot be retrieved
    - The more coins you burn, the better your chances of being selected to mine the next block
    - Eventually, you must stake more by burning more coins

90

Other Consensus Mechanisms

- Other consensus mechanisms
  - Proof of Capacity (aka. Space)
    - Pay to play with hard drive space or memory
    - The most space you 'stake' the better your odds of being selected to mine the next block
    - Consensus algorithm generates large data sets called 'plots' which consume storage
    - Major criticism – this method has no real deterrent for bad actors

91

Other Consensus Mechanisms

- Other consensus mechanisms
  - Proof of Elapsed Time
    - Created by Intel to run on their trusted execution environment
    - Similar to PoW, far more energy efficient
    - Major criticism – requires trust in Intel, places power back in the hands of a central authority
  - Proof of Authority
    - Uses a set of "authorities" – nodes that are explicitly allowed to create new blocks and secure the blockchain
    - Replacement for PoW - Private blockchains only
    - Earn the right to become a validator/authority

92

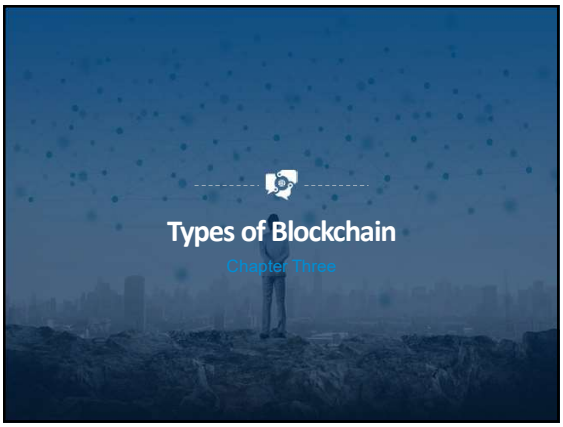
Consensus Mechanisms

- This is the technical weeds!
  - Blockchain consensus mechanisms are the nuts and bolts of validation.
  - Think Internet & TCP/IP – transmission of bytes of data across the Internet
- PoW is the only tried and true (9+ years in use). PoS is coming, rolling out now.
- Rest are just concepts and ideas people are talking about.

93

Questions?

94



95

Blockchain as History

- Blockchain as History
  - Immutable, cannot be changed
    - Remember, each block contains the hash of the previous block
  - Append-only
    - Data on the Blockchain cannot be deleted or edited, only additions can be made
    - This provides a detailed history of ALL events, not just a snapshot of the current state!

96

Types of Blockchains

- Who are the participants in the Blockchain?
  - ANYONE with an important announcement to make can be a Blockchain participant
    - Personal data
  - Groups of people can use the Blockchain to capture announcements that are important to them
    - Supply chain participants
  - Large groups of people can use Blockchain to capture important data which support critical social and economic functions
    - Voting records, land titles

97

Types of Blockchains

- Public vs Private
  - Who can **write** data to the Blockchain?
  - Public – everyone can add a record
  - Private – only certain participants can write data
- Open vs Closed
  - Who can **read** data from the Blockchain?
  - Open – everyone can read Blockchain data
  - Closed – only certain participants can read data

98

Public or Private Blockchain

- Should the solution be a permissioned or permission-less Blockchain
  - Are all participants considered equal, or should some have abilities that others do not?
  - Election chairperson can add candidates to an election = permissioned
  - A digital currency which can exchanged and traded by all = permissionless
- Do customers understand the tech well enough to trust it with their data?
  - Great solutions may not be accepted until they have been socially normalized
  - Credit cards and early e-commerce

99

Public or Private Blockchain

- Hyperledger vs Ethereum
  - These are discussion points, NOT absolutes
  - Ethereum
    - Music and content distribution
    - Digital currency or asset-backed token
    - Blockchain-enabled mobile data service
    - Gambling and on-line gaming
    - Authoring, editing, and amending a piece of legislation
    - Group consensus is needed/required

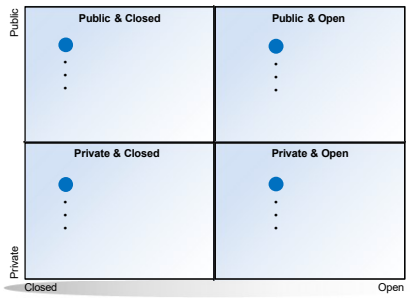
100

Public or Private Blockchain

- Hyperledger vs Ethereum
  - These are discussion points, NOT absolutes
  - Hyperledger
    - Supply Chain
    - Supplier / Manufacturer inventory management
    - Managing internal business processes across geographically distributed locations
    - Allowing elected officials to vote on initiatives without being present

101

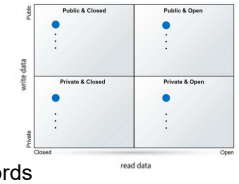
Blockchain Decision Worksheet



102

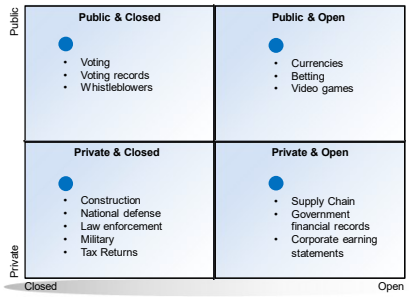
Blockchain Decision Lab

- Currency
- Securities exchange
- Betting
- Video game
- Voting records
- Supply chain data
- Government financial records
- Corporate earnings statements
- Construction tracking
- Defense programs
- Law enforcement agencies
- Others?



103

Blockchain Decision Matrix



104

Public Blockchains

- What is blockchain? (public, open, permissionless)
  - Decentralized ledger
    - Can store any type of data
    - Shared ledger
  - Immutable
  - Anonymous
  - Fully-Transparent
  - Group Consensus
    - Nodes only verify data was recorded correctly
    - No ability to verify truth of the data itself
  - Smart Contracts
    - Ability to automate processes
    - Blockchain as workflow / BPM

105

Blockchain for Business

- Are these properties good or bad?
  - Decentralized ledger
    - Can store any type of data
    - Shared ledger
  - Immutable
  - Anonymous
  - Fully-Transparent
  - Group Consensus
    - Nodes only verify data was recorded correctly
    - No ability to verify truth of the data itself
  - Smart Contracts
    - Ability to automate processes
    - Blockchain as workflow / BPM

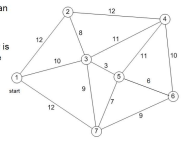
106

The Travelling Salesman

- The travelling salesman problem asks the following question:
  - *"Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin city?"*

The Traveling Salesman Problem

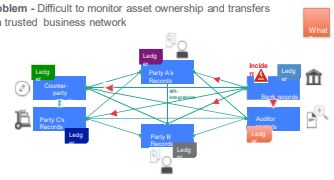
- Starting from city 1, the salesman must travel to all cities once before returning home
- The distance between each city is given, and is assumed to be the same in both directions
- Only the links shown are to be used
- Objective - Minimize the total distance to be travelled



107

Problem

Problem - Difficult to monitor asset ownership and transfers in a trusted business network

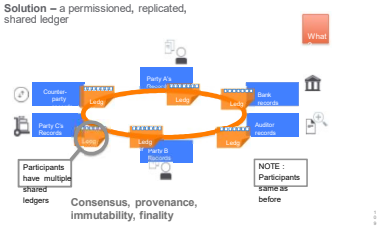


Inefficient, **expensive**, **vulnerable**

108

Solution

Solution – a permissioned, replicated, shared ledger



109

Where are private blockchains a good fit?

- Large business networks / consortium
- Each participant wants to retain ownership/control of their data
- Many different audience types (each of whom will need to see or edit different parts of the data)
- Desire to standardize (and automate) business processes
- Process requires audit, need to prove you did what you said you did
- Need a permanent record of how business assets evolve over time
- Need a system of record between multiple organizations / groups

110

Hyperledger Projects

- Frameworks
  - Fabric
    - A highly modular blockchain platform.
  - Sawtooth
    - Another highly modular platform which offers a unique consensus method – Proof of Elapsed Time.
  - Iroha
    - A blockchain platform offering unique consensus and ordering, and multi-signature support.
  - Burrow
    - Allows an EVM to be embedded in a Fabric / Sawtooth solution.
  - Indy
    - A blockchain platform built for identity management. Used for platform interoperability.

111

Hyperledger Projects

- Toolsets
  - Composer
    - Tool for building blockchain business networks and smart contracts.
  - Quilt
    - Toolset for platform interoperability.
  - Explorer
    - Used to view or query blocks and transaction history data on the ledger.
  - Caliper
    - Blockchain benchmarking and performance tuning tool.
  - Cello
    - Toolset for deploying blockchain infrastructure as a service.

112

Some Hyperledger members

113

Questions?

114

What is a Smart Contract?

- Also known as Chaincode – program rules and decision points into Blockchain transactions and processes
- Automates transactions and ensures they're all following the same rules
- Stored on the Blockchain
- Addresses limitation of Bitcoin protocol

115

Smart Contracts

- Smart Contracts provide:
  - Autonomy: Smart Contracts can be developed by anyone, no need for intermediaries such as lawyers, brokers, or auditors
  - Backup: A Blockchain and Smart Contracts deployed to it can provide a permanent record, allowing for auditing, insight, and traceability even if the creator is no longer in business
  - Efficiency: Removing process intermediaries often results in significant process efficiency gains

116

Smart Contracts

- Smart Contracts provide:
  - Accuracy: Replacing human intermediaries with executable code ensures the process will always be performed the same
  - Cost Savings: Replacing intermediaries often provides significant cost reduction
- Example: A landlord might create a smart contract to automatically collect payment and charge interest on late payments

117

Tokens / Coins

- Tokens / Coins (Ethereum)
  - Some platforms offer built-in token architectures that be used to create your own token or coin
  - Monetary value can be represented in a Blockchain system by coins or tokens
  - Tokens can be backed by real-life assets
    - Allows for fractional ownership of assets in new and creative ways
      - What if a city wanted to purchase its football team?
  - Ownership of tokens is tracked on the Blockchain
  - Trading and exchanging of tokens is managed on the Blockchain

118

Tokens / Coins

- Equity coins vs Utility token
  - **Equity Coins:** represent ownership
    - Football team
    - Owning a piece of multiple self-driving cars
  - **Utility Tokens:** represent usage credits on the solution platform
    - An airline could issue tokens to customers as compensation for delayed or canceled flights
    - Membership rewards programs

119

Tokens / Coins

- Ethereum Token Standards
  - ERC20 – most common standard, only current accepted ERC standard
  - ERC223 – backwards compatible to ERC20, improved smart contract transfers, potentially safer, quicker
  - ERC721 – non-fungible token, asset tokenization
    - Can be non-tradable as well
    - Track certifications, accomplishments, professional milestones

120

Gas

- Gas in Ethereum
  - How users pay for the cost of a transaction
  - Usually decoupled from token or coin so that real gas price cost stays constant while coin prices are volatile
  - Every transaction on the Blockchain must be submitted with gas, unused gas is returned to the user

121

Gas

- Gas in Ethereum
  - Prevents infinite loops and closes certain security vulnerabilities
  - Example: Calculating a hash (Keccak256) always costs 30 gas plus 6 gas per every 256 bits of data being hashed
    - "BlockchainTrainingAlliance" = 26 bytes / 208 bits
    - Cost to hash = 30 + ((208 / 256) * 6) = 34.875 gas
    - ETH Gas Station – https://ethgasstation.info/calculatorTxV.php
    - March 2018 = 0.00000001 ETH / $0.00005

122

Gas

- Gas in Ethereum
  - Gas is only consumed when data is written to the Blockchain, reading consumes no gas
    - Internal state variable value changes
    - Paid equally to all nodes for writing of announcement
  - A function which runs out of gas will be terminated, no gas will be returned to the user
  - Gas Station: https://ethgasstation.info/

123

Gas

Opcodes & Gas

	QUICKSTEP	FASTESTSTEP	FASTSTEP	MIDSTEP	SLOWSTEP	EXTSTEP
Gas cost	2	3	5	8	10	20
	ADDRESS	DUP	MUL	ADDMOD	JUMPI	BLOCKHASH
	ORIGIN	SWAP	DIV	MULMOD	EXPBASE	BALANCE
	CALLER	PUSH	MOD	JUMP		EXTCODESIZE
	CALLVALUE	ADD	SDIV			EXTCODECOPYBASE
	CALLDATASIZE	SUB	SMOD			
	CODESIZE	LT	SIGNEXTEND			
	GASPRICE	GT				
	COINBASE	SLT				
	TIMESTAMP	SGT				
	NUMBER	EQ				
	DIFFICULTY	AND				
	GASLIMIT	OR				
	POP	XOR				
	PC	NOT				
	MSIZE	BYTE				
	GAS	CALLDATALOAD				
		CALLDATACPY				
		CODECOPY				
		MLOAD				
		MSTORE				
		MSTORE8				

124

Questions?

125

How is Blockchain different than what we have today

Chapter Four

126

How is Blockchain Different than what we have today?

- Blockchain is decentralized, not distributed or centralized
- Centralized:
  - Owned by a single entity, resources are concentrated in single location or system
  - Single point of failure
  - Easy to maintain
  - Low fault tolerance
  - Low scalability

127

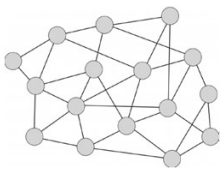
How is Blockchain Different than what we have today?

- Distributed:
  - Owned by a single entity, resources are distributed across locations and systems
  - High fault tolerance
  - More scalable
- Decentralized:
  - Most often, there is no single owner of a Blockchain
  - Ownership and upkeep is shared amongst participants

128

How is Blockchain Different than what we have today?

- Decentralized:
  - Resources exist as independent entities on a peer-to-peer network
  - Most difficult to maintain
  - High fault tolerance
  - Infinitely scalable



129

How is Blockchain Different than what we have today?

- Sidebar: What is Peer-to-Peer (P2P)?
  - Network of equally privileged participants
  - Tasks and workload partitioned across all participants (nodes)
  - Every node is a consumer and a supplier of resources
    - Different from client (consumer) / server (supplier)
  - Remember Napster? (filesharing - 1999)
  - Requests are served by peers around you, not by a central resource

130

How is Blockchain Different than what we have today?

- Sidebar: What is Peer-to-Peer (P2P)?
  - Blockchain is P2P
  - Other use cases:
    - Mesh Networking
      - Sharing network connectivity and network resources amongst nodes
    - Multimedia content distribution
    - Content based addressing (**IPFS**)
      - Known as a ***Hypermedia Distributed File System***

131

How is Blockchain Different than what we have today?

- Blockchain vs conventional tech
  - Blockchain is very new
  - Many of the features and functions taken for granted in other solution stacks do not exist yet
    - String concatenation in Solidity
  - Still highly experimental, many areas still pre-production tech
  - Not many trained resources yet
    - Supply vs demand creates high wages for trained resources and lots of competition

132

How is Blockchain Different than what we have today?

- Blockchain vs conventional tech
  - Not mainstream yet
    - Lots of fear, misconception, misunderstanding
  - Internet and e-commerce vs credit cards in restaurants
    - It takes awhile for new tech to be socially normalized

133

How is Blockchain Different than what we have today?

- Low transaction processing speed (for now)
  - Public Blockchain, approx. 15 tx/sec
  - Visa, approx. 70,000 tx/sec
  - This is being addressed with new consensus mechanisms such as PoS and new architectures such as tangle
    - Each transaction is verified by the previous XX transaction submitters
    - Bandwidth – 14.4k dial up in early 90's, 5G mobile data today

134

How is Blockchain Different than what we have today?

- Blockchain prioritizes security over speed
- Storing all block data on a node requires a lot of space
  - Bitcoin ledger is > 167GB as of May 15, 2018

135

How is Blockchain Different than what we have today?

- Scaling up the network
  - Peer-to-peer architecture increases security, not performance
  - Every node must validate all transactions on the block
  - Adding more nodes makes things safer, but not faster
  - More nodes = greater fault tolerance
  - Increasing performance requires changing platform architecture, NOT increasing network size

136

How is Blockchain Different than what we have today?

- Software vs Firmware
  - Software developers are used to being able to patch and upgrade releases
  - Firmware developers do not usually have this luxury
    - If your oven needs a firmware upgrade, it's not practical to drag it to Home Depot to be re-flashed
  - Smart Contracts are like firmware, not software
    - Once a contract is deployed, it is permanent
    - Cannot be changed – the code can be updated and a new contract instance can be created, but the old one will remain

137

How is Blockchain Different than what we have today?

- Software vs Firmware
  - Smart Contracts are like firmware, not software
    - Once deployed, a contract is always available unless it has been "killed"
      - Killing a contract means it will refuse any new transactions, does NOT delete the contract
      - Kill or Self-destruct is an optional function, does not have to be implemented
        - Developers need to think through the choice to implement this function carefully
        - If a Kill method is implemented, GREAT caution should be taken to ensure it can ONLY be called by the right people or circumstances

138

Database vs Blockchain

- Database vs Blockchain
  - Would a distributed database or other technology be adequate?
    - YES
      - Performance is important
      - Large number of trained resources is important
      - Highly confidential data
      - Don't need a history, just a snapshot of data
      - Ease of maintenance
      - Application logic expected to change frequently
      - Users are physically separated by tens of thousands of miles
      - Maintaining centralized control of resources is important

139

Database vs Blockchain

- Database vs Blockchain
  - Would a distributed database or other technology be adequate?
    - NO
      - Requires transparency and public validation
      - Needs extreme fault tolerance
      - Needs to be infinitely scalable
      - Users physically separated by light minutes
      - Full data history and lifecycle is important to capture
      - No single authority can or should be entrusted with the data

140

Data Sovereignty

- Data Sovereignty
  - You don't own data somebody else puts on Blockchain
  - In a centralized system, all data is owned by the system owner (Facebook, Instagram, Amazon, etc.)
  - In scenarios where you must demonstrate you own and control data, Blockchain may not be a good solution
    - Permissioned Blockchains such as Hyperledger may help ease fears around this

141

Database vs Blockchain

- Blockchain provides:
  - A data layer
  - A business logic layer
  - A messaging layer
  - A token model for trading and exchange
- Blockchain does not provide:
  - A user interface layer
- Blockchain is DDoS-proof
  - Cannot deny service to a P2P architecture

142

Database vs Blockchain

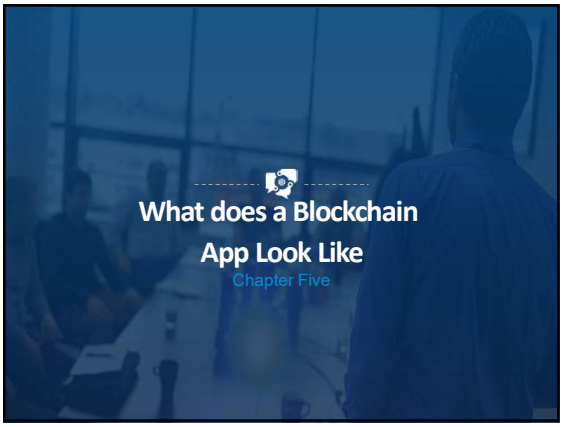
- Blockchain provides almost unlimited fault tolerance
  - Ideal for situations where nodes are expected to go online and off-line frequently

143

Questions?

•••

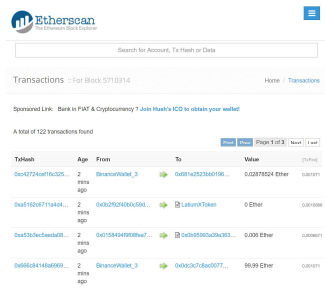
144



145

What does a Blockchain App look like?

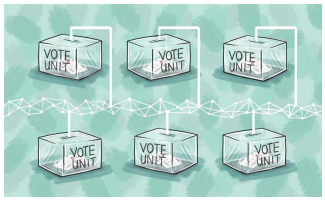
- Anatomy of a Blockchain transaction
  - Etherscan.io



146

What does a Blockchain App look like?

- Demo
  - Votethereum – a sample decentralized Ethereum Voting Platform



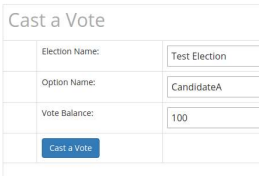
147

Questions?

148

What does a Blockchain transaction look like?

- Voter casts 100 votes for Candidate A in UI



149

What does a Blockchain transaction look like?

- Node.js middle-layer calls castVote() function in Smart Contract

```
Votethereum.deployed().then(function(contractInstance) {
  contractInstance.castVote(electionNameCastVote, optionNameCastVote, voteBalanceCastVote, {gas: 550000, from: web3.eth.accounts[theVoter]})
```

150

What does a Blockchain transaction look like?

- Smart Contract adds votes to the total for the candidate

151

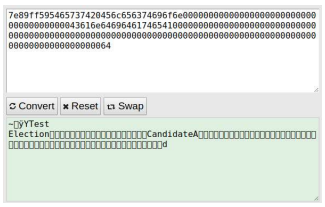
What does a Blockchain transaction look like?

- This transaction / announcement is stored on the blockchain

152

What does a Blockchain transaction look like?

- This transaction / announcement is stored on the blockchain



153

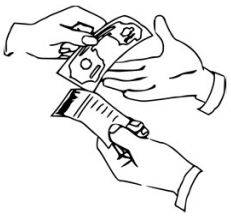
What does a Blockchain transaction look like?

- This transaction / announcement is recorded by all nodes

154

What does a Blockchain transaction look like?

- All nodes compensated for recording transaction with gas submitted by end user



155

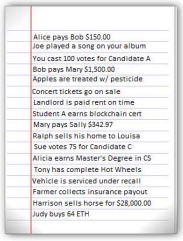
What does a Blockchain transaction look like?

- This transaction / announcement is stored on the blockchain

156

What does a Blockchain transaction look like?

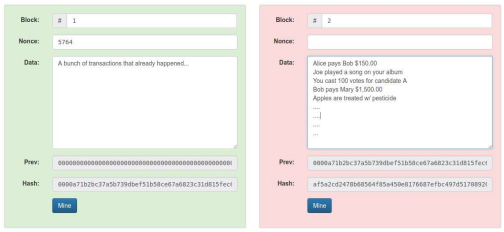
- Block is filled with transactions / announcements



157

What does a Blockchain transaction look like?

- Miners try to guess a nonce to match the current difficulty



158

What does a Blockchain transaction look like?

- One miner will randomly guess an answer that matches the difficulty

159

What does a Blockchain transaction look like?

- This miner will announce its answer to all other nodes

160

What does a Blockchain transaction look like?

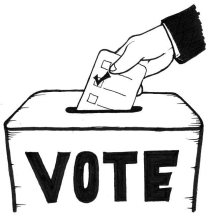
- The other nodes will try the proposed nonce on their copy of the data

Nonce:	21030
Data:	Alice pays Bob $150.00 Joe played a song on your album You cast 100 votes for candidate A Bob pays Mary $1,500.00 Apples are treated w/ pesticide

161

What does a Blockchain transaction look like?

- Did it work? Time for the miners to vote!



162

What does a Blockchain transaction look like?

- If 51% agree, each node adds the accepted block of transactions to the chain

163

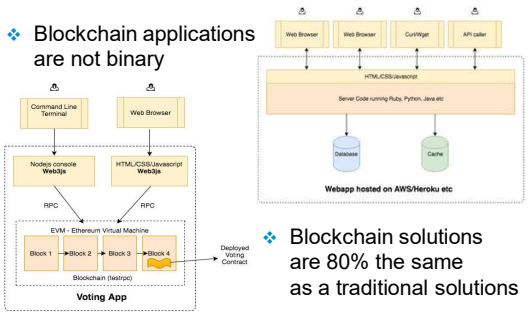
What does a Blockchain transaction look like?

- Miner with correct nonce rewarded

164

A Blockchain Solution

- Blockchain applications are not binary



- Blockchain solutions are 80% the same as a traditional solutions

165

DApp = Decentralized Appliction

- Decentralized app = DApp
  - DApp, dApp, dapp, ĐApp
  - Resembles typical full stack web application
- The front end / user interaction layer
  - HTML / CSS front end (presentation layer)
  - Mobile app front end (presentation layer)
  - No special technical front-end requirements
  - User experience should seek to make users understand and value any new or different features that Blockchain provides

166

DApp Architecture

- Middle (Interface) layer - communication to and from the Blockchain
  - Node.js (application logic and contract interface)
  - Server-side code
  - Brokers communication between the user and the Blockchain
  - All communications to and from Blockchain is bytecode, this layer abstracts this away with JS libraries
  - Packages such as Truffle
  - This layer can contain validation and exception handling code
  - Business logic can live in the Blockchain layer, but should it?

167

DApp Architecture

- Blockchain layer
  - Smart Contract(s)
  - Contracts can call other contracts
  - Contracts can call **Oracles**
    - Smart Contracts do not have access to external data, by default
    - An Oracle provides external data to a Smart Contract
      - Weather info, stock price, news headlines, data from external LOB systems, etc.

168

Oracles

- Oracles in the Blockchain layer
  - Objective is to provide trusted external data that can be used by a smart contract
  - Smart contracts use oracles to resolve/provide details that are not known at the time the contract is written
  - Oracles are third party services which are not part of the blockchain consensus mechanism.
    - Trust in the Oracle owner is critical

169

DApp Development Tools

- Integrated Development Environment (IDE)
  - Visual Studio Code
    - Open source code editor
    - **Plugins for both SOL & Hyperledger**
    - HTML editor
    - CSS editor
    - JavaScript editor
  - Hyperledger Composer
  - Remix (remix.ethereum.org)
    - BEST way to start developing a Smart Contract

170

DApp

- Web3.js frameworks
  - Ganache / TestRPC
    - Allows for testing and development with a local Blockchain simulation
  - Truffle
    - Allows for deployment and testing on Ethereum test Blockchains
      - You will need ETH for test network (no value) just like you will need ETH for the production network (real value)
      - Test ETH can be obtained in various ways depending on the particular test network being used
    - Allows for release to production Blockchain

171

Questions?

172

Demo – Smart Contracts

- Remix demo
  - I just want to write some Smart Contracts

173

Demo

- **Hello.sol code:**

```
pragma solidity ^0.4.19;

contract Hello {

    // internal state variable used to store the current name
    string name;

    // Constructor
    function Hello(string initialName) public {
        name = initialName;
    } // Hello

    // Function to say hello to the current name
    function sayHello() constant public returns (string, string) {
        return ("Hello", name);
    } // sayHello

    // Function to set or change the value of the name
    function setName(string x) public {
        name = x;
    } // setName

} // Hello
```


174

Demo

- **Hello.sol code:**
  - **Indicates version of Solidity to be used**

```
pragma solidity ^0.4.19;
```


175

Demo

- **Hello.sol code:**
  - **Contract keyword + contract name**

```
contract Hello {
    // contract code goes here
} // Hello
```


176

Demo

- **Hello.sol code:**
  - **Internal state variables used to store contract data**
  - **Gas costs incurred when these are changed / updated**
    - **Change or update written to blockchain**
  - **No gas costs to read state variable values**

```
contract Hello {
    // internal state variable used to store the current name
    string name;
} // Hello
```


177

Demo

- **Hello.sol code:**
  - **Constructor function**
    - **Function name must match contract name**
    - **Constructor called when contract is deployed to blockchain**

```
contract Hello {

    // Constructor
    function Hello(string initialName) public {
        name = initialName;
    } // Hello

} // Hello
```


178

Demo

- **Hello.sol code:**
  - **Functions which return data to the caller marked with "returns" keyword**
  - **Returned data types explicitly stated**
  - **No gas required for this function to complete**

```
contract Hello {

    // internal state variable used to store the current name
    string name;

// Function to say hello to the current name
    function sayHello() constant public returns (string, string) {
        return ("Hello", name);
    } // sayHello

} // Hello
```


179

Demo

- **Hello.sol code:**
  - **Functions which update state variables do not require special keywords**
  - **Input parameters must be specified**
  - **This function must be provided gas to complete**

```
contract Hello {

    // internal state variable used to store the current name
    string name;

    // Function to set or change the value of the name
    function setName(string x) public {
        name = x;
    } // setName

} // Hello
```


180

Demo

- **Hello.sol code:**

```
pragma solidity ^0.4.19;

contract Hello {

    // internal state variable used to store the current name
    string name;

    // Constructor
    function Hello(string initialName) public {
        name = initialName;
    } // Hello

    // Function to say hello to the current name
    function sayHello() constant public returns (string, string) {
        return ("Hello", name);
    } // sayHello

    // Function to set or change the value of the name
    function setName(string x) public {
        name = x;
    } // setName

} // Hello
```


181

Questions?

•••

182

How Do I Design & Develop a Blockchain App

Chapter Six

183

Guiding Principles

- Begin by defining Guiding Principles
  - Yes / no questions that you, your team, and your organization must agree on before starting solution design
  - Your answers define the default (assumed) view or behavior
    - Doing the opposite is possible, but requires justification
  - Examples:
    - Feature-heavy or feature-light?
      - Do we try to provide as many features as possible, or do we aim to provide a simple application?
      - Feature Heavy - Microsoft Excel
      - Feature Light - Prezi, Google
    - Collaboration or Security?
      - Do we favor open and frictionless collaboration, or do we want to ensure the highest levels of security?
      - Collaboration - Facebook, Instagram
      - Security - Online banking application, e-commerce

184

Guiding Principles

- Begin by defining Guiding Principles
  - Centralized or Decentralized support?
    - If users have a question or need help will they come to us directly, or should they seek out help from a broad community?
    - Centralized Support - Custom developed LOB application, medical diagnostics application
    - Decentralized Support - YouTube, Instagram
  - Consistency or specialization?
    - Will this solution treat all users the same and grant them the same access and abilities or will this solution allow users to specialize and grow in areas they desire?
    - Consistency - Facebook, Twitter, LinkedIn
    - Specialization - World of Warcraft

185

Guiding Principles

- Guiding Principles will help to define personas
  - Personas are the types of users who will use your application
  - Each person can have multiple personas based on mood, time of day, expectations, etc.
    - Example Persona: Rosa is a busy accounting clerk who is frustrated at the amount of paper involved in her current business processes. She feels she could accomplish much more if the processes she's involved in would simply adopt modern toolsets. She often becomes discouraged and frustrated at work and wonders if other organizations are doing a better job than hers. The expense reimbursement process is particularly painful for her.

186

Guiding Principles

- Guiding Principles will help to define personas
  - A persona is NOT a person!
  - Example Persona: Rosa loves gourmet cooking and on weekends she loves to try preparing new dishes and sharing them with friends. She likes to experiment by deviating from popular recipes to add her own touch. Often she comes up with new dishes she thinks are quite good. She wishes there was an easier way to share these creations and collaborate on cooking ideas with other amateur chefs.

187

Personas

- Personas will help to create user stories
  - User stories are short stories about a persona and the desired interaction they will have with your solution
  - User stories should define how each persona interacts with your solution in a particular use case, and should describe the expected outcome

188

User Stories

- Personas will help to create user stories
  - Example User Story: Rosa is happy the new expense reimbursement system in her company has gone live. Rosa begins her day by checking all the submitted expenses from the last day. She has to approve or deny them all. She can quickly review electronic records, and approve or deny with the click of a button. Any denied expenses can be sent back to submitter for changes, or can be deleted. All approved expense are sent to accounting without Rosa having to do anything extra. Rosa feels this is a far better system than the old paper-based system because of the enormous efficiency gains she has experienced since go-live.

189

User Stories

- User stories will help to create functional requirements
  - A functional requirement describes what the solution should do without focusing on how that thing should be done
  - From the previous user story, we can extract these functional requirements (Rosa needs…):
    - Quickly view all submitted expenses from past day
    - One click approval or denial of each expense
    - A denied expense can be send back to the submitter or can be deleted
    - Approved expenses are routed to accounting for pay-out

190

Functional Requirements

- Functional requirements will drive technical requirements
  - Technical requirements explain how a functional requirement will be fulfilled
  - Our example functional requirement can be broken into these technical requirements:
    - Functional Requirement: Quickly view all submitted expenses from past day →
    - Technical Requirement: A green button should appear in the top right corner of the application window

191

Technical Requirements

- Clicking this button will show Rosa the list of expenses submitted in the last 24 hours which are still in a pending status (not approved, denied, or in-process)
- The list of expenses to approve will be displayed as a modal window. Each expense is a hyperlink which opens the expense detail on a new tab

192

Technical Requirements

- Technical requirements will define tasks
  - Tasks are the explicit steps that must be taken to fulfill a technical requirement
  - Our example technical requirements can be broken into these tasks:
    - A green button should appear in the top right corner of the application window
      - Task 1: Place green command button control at top-right hand side of frame
      - Task 2: Button click event should call GetAllUnprocessedExpenses() routine
      - Task 3: Results should be passed to the DisplayModalExpense() function

193

Tasks

- Tasks will define estimates and required skillsets
  - Once you get to the task level, provide an estimate for each task
  - Each task should also be evaluated to determine the skillset needed to complete it
  - In general, if a task cannot be estimated or skillset can not be identified, the task needs to further broken down into sub-tasks

194

Tasks

- Tasks will define estimates and required skillsets
  - Our example tasks can be estimated and matched to a skillset:
    - Place green command button control at top-right hand side of frame
      - 1 hour, user interface designer
    - Button click event should call GetAllUnprocessedExpenses() routine
      - 1 hour, back-end developer
    - Results should be passed to the DisplayModalExpense() function
      - 1 hour, back-end developer

195

ALWAYS ASK

- Do my personas align to my guiding principles?
  - If not, is there a justifiable reason why not?
- Do my user stories align to my guiding principles?
  - If not, is there a justifiable reason why not?
- Do my functional requirements align to my guiding principles?
  - If not, is there a justifiable reason why not?
- Do my technical requirements align to my guiding principles?
  - If not, is there a justifiable reason why not?
- Do my tasks align to my guiding principles?
  - If not, is there a justifiable reason why not?

196

Good architecture & design questions

- What does this solution need to let users do?
- Will the proposed solution reduce or remove the problems and pain points currently felt by users?
- What should this solution prevent users from doing?
- Do you need a solution ready for heavy use on day 1?
  - No = Blockchain, yes = traditional
- Is your solution idea enhanced by the use of Blockchain? Does the use of Blockchain create a better end-user experience? If so, how?
  - No = traditional, yes = Blockchain

197

Good architecture & design questions

Has your business developed custom software solutions before?

- YES:
  - Do you have the bandwidth to train developers on Blockchain tech, either directly or indirectly?
  - Do you already have a healthy dev ops process & practice?
  - Will you outsource development, project management, engagement management, architecture, or UX design?
- NO:
  - Do you want to develop in house or outsource?
  - Do you have a single solution idea, or do you envision creating multiple solutions?

198

Good architecture & design questions

- What level of support are you going to need?
- How big is the developer community?
- Does your vision of the future align with the project or platform's vision of the future?
- Does the platform aim to make new and significant contributions to the development space, or is it an efficiency / cost play?
  - Which is more important to you?

199

Good architecture & design questions

- If private is preferred, how will membership and access be granted, controlled, and regulated?
- Should the solution be an open or closed Blockchain?
- Who needs to see the data?
- Who should NOT see the data?
- How fast does it need to be?
  - Greater speed (private Blockchain) = lower trustability (smaller network size)
  - Lower speed (public Blockchain) = higher trustability (larger network size)

200

- Create a plan for contract updates and changes!
- Once a contract is deployed, it's permanent unless its Kill method is implemented and called
  - Have you made sure this can't be called maliciously?
- If your contract is updated, a new version is deployed alongside the old contract
- How will your application know to use the proper contract?
  - Calling contract?

201

- Hybrid solutions
  - Use conventional data stores
  - Store hashes or checksums on Blockchain so data can independently verified
  - Best of both worlds - high transaction speed, data verifiable on Blockchain
  - Does NOT allow user to see original data if data has been manipulated or changed
- Monetary exchanges are NOT required!
  - Many good solutions do NOT use a token or coin
  - Make sure the addition of a coin or token will make the user experience EASIER or BETTER than it is today!!!

202

Questions?

203

How Do I Design & Develop A Blockchain App

Continued

204

How Do I Develop a Blockchain App

- AGILE approach pre-release
  - keep team sizes small at first (mythical man month)
  - plan, develop, test, repeat in multiple cycles
  - ability to re-prioritize features
- Define guiding principles up front
  - Guiding principles drive use cases
  - Personas drive user stories
  - User stories drive functional reqs
  - Functional reqs drive technical reqs
  - Technical reqs drive tasks
- Technology third!
  - Worry about the technology LAST!!!
  - Everything should align clearly to a guiding principle, or have a very obvious and well-documented reason not to

205

How Do I Develop a Blockchain App

- Link contracts to share functions
- Be careful to only call contracts you've written, or can trust implicitly
- Use calling contracts to keep contract address the same
  - One calling contract that can determine the latest version of the correct contract to pass the call onto
- CONSIDER No of Users * Avg No of Transactions (state changes) per User

206

How Do I Develop a Blockchain App

- Should a blockless solution be used?
  - IoTA (https://www.iota.org/) - permission-less DLT
    - Tangle

- **Transactions verified by several future transaction posters, not entire network.**
  - NO - High security needed
  - YES - Fast transaction times needed

207

How Do I Develop a Blockchain App

- Performance?
- DO you need acknowledgement or verification?
- In most cases, acknowledgement is fine.
- Security?
  - ASK yourself, who SHOULDN'T see my data?
  - THEN, follow up with WHO SHOULD see it?
  - RECONCILE the two lists.
  - Measure potential attack surfaces in each layer (display, communications, Blockchain)
  - Measure potential attack surfaces between layers / application as a whole

208

How Do I Develop a Blockchain App

- Check identity in Kill function, DO NOT let anyone kill a contracts
- Killed contracts still exist, but do not accept transactions
- Anonymity?
  - Anonymity can increase trust and faith in a brand
  - Anonymity in tech solutions can be viewed as malicious if the tech is not well understood (Bitcoin & Tor as examples)

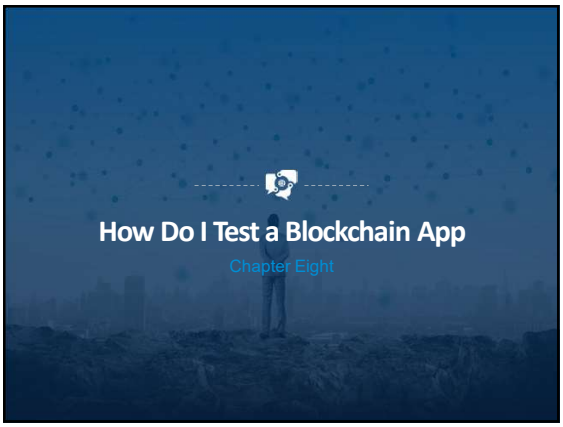
209

How Do I Develop a Blockchain App

- Monolithic vs Modular
  - Modular is more efficient
  - Modular allows for code reuse
  - Modular introduces additional security concerns
  - Modular is generally preferable, but not always
  - Monolithic == potentially greater security
- Sandwich complexity model
  - Keep Smart Contract layer (bottom) as simple as possible
  - Keep presentation layer (top) as simple as possible
  - Keep complexity in the middle of the application

210

211



212

Testing a Blockchain App

- Recommend 5x to 10x traditional application testing time
- Testing FIRMWARE not SOFTWARE
  - Ensure 110% test coverage!!!
- Does the application suit user needs today?
  - Will the application suit user needs tomorrow?
- Zero Defects (ZD) Philosophy
  - Do things right the first time
  - Cost of preventing problems is less than fixing them later

213

Types of Testing

- Unit Testing
  - Developer level testing
- Configuration & Environment Testing
  - Testing of code in a specific environment (Test, Staging, Production)
- Load & Performance Testing
  - Volume/Stress testing
- Regression Testing
  - Testing of all code deployed, even code not changed
  - Reintroduction of old bugs
  - Introduction of new bugs

214

Types of Testing

- Testing tools
  - Mocha / Chai
    - Works on Ethereum using Truffle test
    - Works on Hyperledger Composer

215

Testing a Blockchain App

- SECURITY is #1!
  - This space moves very fast, and new vulnerabilities are found often. This is a lot to ask for a regular developer.
  - Strongly consider hiring a cybersecurity expert for solution review
  - Use caution calling external contracts, favor your own

216

Testing a Blockchain App

- Handle all errors, catch all exceptions in external calls
- Favor pull over push payments or transfers (put amount in 'escrow', inform user they can withdraw)
- Contract address can have a balance before a contract is created. Never assume a zero balance.
- On-chain data is public! Is it protected?
- Contracts can come online and then go offline, repeatedly…
- Networks (Ethereum)
  - TestRPC / Ganache
  - Local testing

217

Testing a Blockchain App

- Create test cases
  - Testing your solution under a pre-defined set of conditions.
  - Example:
    - Can our solution handle 223 concurrent users averaging one transaction every .5 seconds?
    - When a password complexity rule is updated, are existing users with weak passwords notified they should be updated?

218

Testing a Blockchain App

- Provide test cases to developers during development
- Test Driven Development: Write functions from the beginning which can pass defined test cases
  - Run your test cases every build!
- Test the "Happy Path“ FIRST!
  - What you expect most users will do most of the time
- Then wander off the "Happy Path"
  - Are you intentionally testing scenarios that you don't frequently expect? If not, your users will...
  - EXAMPLE: What happens if I register with a user name of: #$$*!~`` >,.\\))(@#*476?

219

Testing a Blockchain App

- Test Coverage
  - How much of your app do you plan to test?
  - 100% is often not practical, but is the ideal standard
- Do not let developers test their own code
  - Developers will subconsciously test the code according to the mental model they used to create it
- Test your application at 2x-3x expected maximum load
  - DO NOT be a victim of your own success

220

Testing a Blockchain App

- Involve testers during the architecture and design phase
  - Letting testers be a part of the entire design and development process will give them a better idea of the project vision
  - Their tests will align to product vision
  - "Shift-Left Testing" = involve tester and QA engineers earlier in the process

221

Testing a Blockchain App

- Separate dev and test environments
  - DO NOT let developers touch the test environment
  - Developers WILL forget to document config steps and misc requirements
  - Not letting them touch the test environment will surface these missing pieces
- Let end users participate in testing
  - End users are going to discover all your undiscovered bugs anyway
  - Opinion isn't biased by project involvement

222

Testing a Blockchain App

- Make your testers and QA engineers TEACH someone else how to use the product
  - Will uncover concepts, ideas, and features that seem obvious to project members, but not end users
- Test your documentation and support materials as well
  - Users who can't find support or information about your solution will perceive it as 'broken'
- Automated testing tools allow you to test faster, not better
  - Make sure you have good test cases from the beginning
  - Tools will NOT improve your test cases

223

Testing a Blockchain App

- Testing is NOT an expense, it is a risk-reduction strategy!!
- Bug reports should be multi-media
  - Not just written
  - Sometimes a picture (or video) is worth a thousand (million) words!!

224

Testing a Blockchain App

- Ropsten / Rinkeby test networks
  - Test public Blockchain
  - Measure gas costs!!
  - Verify transactions are correct!
- Production Blockchain
  - Costs are real!
- Bug Bounties
  - Offer bounties to developers and hackers who can find flaws, security holes, and exploits in your code

225

Testing a Blockchain App

- Hire an attacker
  - ties into bug bounty idea
  - encourage professional hackers to attack your platform, reward them for finding vulnerabilities and unknown attack surfaces
- Test on every device your user will use
  - Don't test on the latest iPhone, test on ALL iPhones!!
  - Explicitly state any untested platforms are not supported!

226

Testing a Blockchain App

- Personas / User Stories should be the start of your test cases
  - Create a set of test cases for each persona
  - Test a function multiple times from the POV of multiple users
- Have a plan to address undiscovered bugs BEFORE they're discovered
  - You users WILL find bugs, prepare for this reality!!

227

Testing a Blockchain App

- Types of bugs:
  - Accessibility – the code doesn't meet spec for accessibility (Americans with Disabilities Act)
  - UI bugs – user interface doesn't meet the design specification
  - Integration – two or more components don't work together as expected

228

Testing a Blockchain App

- Ideal bug reports link the bug being reported to a business value
  - EXAMPLE:
    - Interface is hard to use, very complex
    - The complex user interface will most likely cause most users to feel scared and abandon the app before they understand the value of the product
      - should the product be simplified OR
      - should the value proposition be more clear from the beginning

229

Testing a Blockchain App

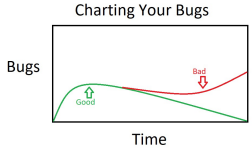
- Types of bugs:
  - Business logic – something isn't right according to business requirements
  - Security – the code is vulnerable to some security exploits
  - Regression – some code updates caused existing features to break
  - Performance – the code is slow or some actions execute extra functions

230

Testing a Blockchain App

- Chart your bugs
  - Keep track of the number of bugs discovered, addressed, and closed for each major functional area over time
    - Make sure you're trending down
    - Rushing too fast will trend up and introduce more bugs than you close

SLOW DOWN!!



231

Blockchain Testing Tools

Some are available, some are coming…

- Ethereum
  - Ethereum Tester – GitHub project
  - Truffle – automated tests for JavaScript and Solidity
  - Ganache – local, in-memory Blockchain
- Hyperledger, Composer, 3 Types of Testing (today)
  - Interactive testing
  - Automated unit testing
  - Automated system testing

232

Questions?

•••

233

What are some use cases for Blockchain

Chapter Nine

234

Use Cases

- Autonomous Organizations
  - DAO (aka DAC)
    - **Decentralized Autonomous Organization** (ex. Bitcoin)
    - Decentralized Autonomous Corporation
  - An organization which replaces typical employee functions with Smart Contracts
  - Initial concept of DAO is to earn a profit for the shareholders by performing valuable services

235

Use Case Examples

Blockchain technology is new but it will touch everything. Let's examine some of the use cases

- Loyalty Reward Points/Programs
  - A customer redeeming loyalty reward points at checkout
  - An airline crediting a customer with frequent flier miles
- Social media platform – Steemit
- Storing your personal credit history and score
- Storing voter registration data and voting records

236

Currency

- Bitcoin
  - The primary interface between fiat and digital currencies
- Ether
  - Foundation of all ERC tokens
  - "Smart Money" - money which can be programmed to make decisions for itself
- Litecoin
  - Light version of Bitcoin, aims to be used for smaller scale transactions
- Monero
  - Privacy and anonymity are number one priority

237

Banking Services

- A digital currency can intrinsically provide many of the same benefits of a traditional bank
  - Take deposits, issue credits
    - Value can be stored in digital currency
    - Individuals can loan money on their own terms and conditions via Smart Contracts
    - Enables micro-lending and micro-payments
  - Decouple possession and ownership
    - I do not have to physically posses anything to hold value in digital currency

238

Banking Services

- A digital currency can intrinsically provide many of the same benefits of a traditional bank:
  - Act as trust broker in transactions
    - I don't have to trust you fully to trade with you
    - Smart Contracts and the open nature of Blockchain fill the "trust gap"
    - CryptNote Protocol for complete privacy
- Money transfer and wire exchanges
  - BBVA

239

Other Use Cases

- Medical Records
  - Store medical records anonymously, unlock and access from anywhere
  - What if you got sick and ended up in a hospital 10,000 miles away?
- Supply Chain / Value Chain – Carrefour SA (Chicken)
  - Track events of significance during product creation and distribution
  - Verifying organic produce…
- Content Distribution – Wemark (photography)
  - Track plays, views, purchases and requests – charge appropriately
  - Pay 'bounties' to loyal fans based on how many of their friends they introduce to your work

240

Other Use Cases

- Verification of Software Updates (cars, planes, trains, etc.)
  - Prevent autonomous devices from being infected with bad software
  - Wired Jeep Grand Cherokee hack – 2015
- Automation via IoT
  - Keep connected devices secure and accountable
  - Smart Dubai 2021

241

Other Use Cases

- Law Enforcement
  - Share records across agencies and departments
  - Reduce tragedies resulting from a lack of good communications and data-sharing infrastructure
- Title and Ownership Records
  - 1/3 of all land titles in US have 1 or more errors
  - Trade property with one transaction on the Blockchain
  - No need for intermediaries and time-consuming processes
- Social Media and Online Credibility
  - Track reviews, popularity, and reputation
  - Why go to Yelp for restaurant reviews, JD Powers for automotive reviews, Consumer Reports for appliance reviews, LinkedIn for employee reviews, Wired for tech device reviews, etc.?
  - Get all reviews in one place
  - Reviews weighted by credibility and reputation of reviewer

242

Other Use Cases

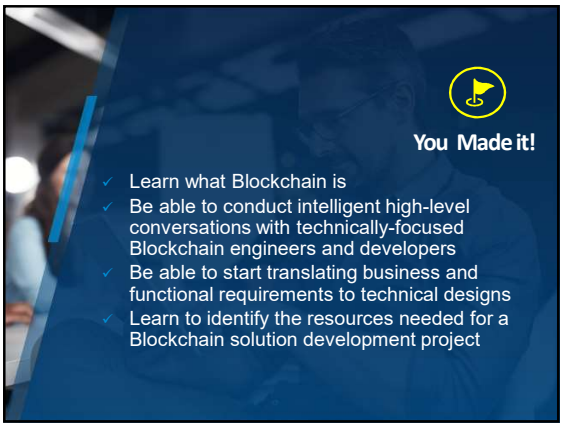
- Fractional asset ownership
  - Owning a car vs owning a piece of car in each city
  - Own a single pizza restaurant or own 1/1000 of every pizza restaurant in the US Midwest?
- Cable Television billing
  - Correlate payments between networks and MSOs according to contract terms without intermediaries
- Campaign finance and political donations
  - I will donate $1,000 to your campaign
  - You can use $250 now
  - You get another $250 if you uphold two campaign promises in the first year in office
  - You get the remaining $500 if you uphold all promises
  - If not, I get the balance returned to me

243

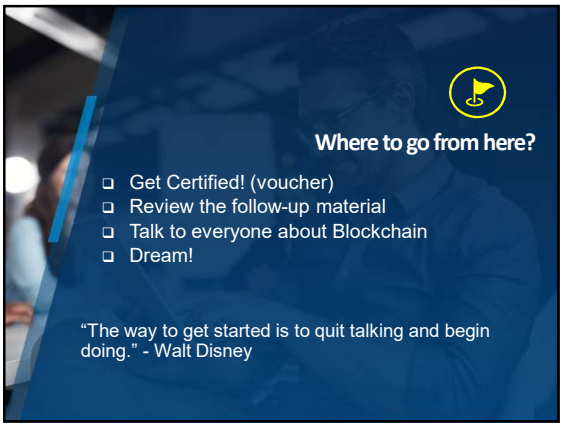
Going Forward!

- Identify two or three use cases in your personal / professional life that may be suited to Blockchain
  - For each, determine:
    - Should this be a public or private Blockchain?
    - Should this be an open or closed Blockchain?
    - Should this be a permissioned Blockchain?
      - If so, what are the various roles, and what can people in each role do / not do?
      - If not, explain why it is important for each user to have equal permissions and abilities
    - Describe any Smart Contracts you envision and the role they will play in your solution
    - Explain the pros and cons of using Blockchain for your solution vs conventional technology

244



245



246

Questions?

THANK YOU

247

instructors@blockchaintrainingalliance.com

LinkedIn -
https://www.linkedin.com/in/kbennett2000/
#BlockchainBeardGuy

248