Blockchain: The Code Law

SEMINAR @ SOFTUNI
15.03.2017

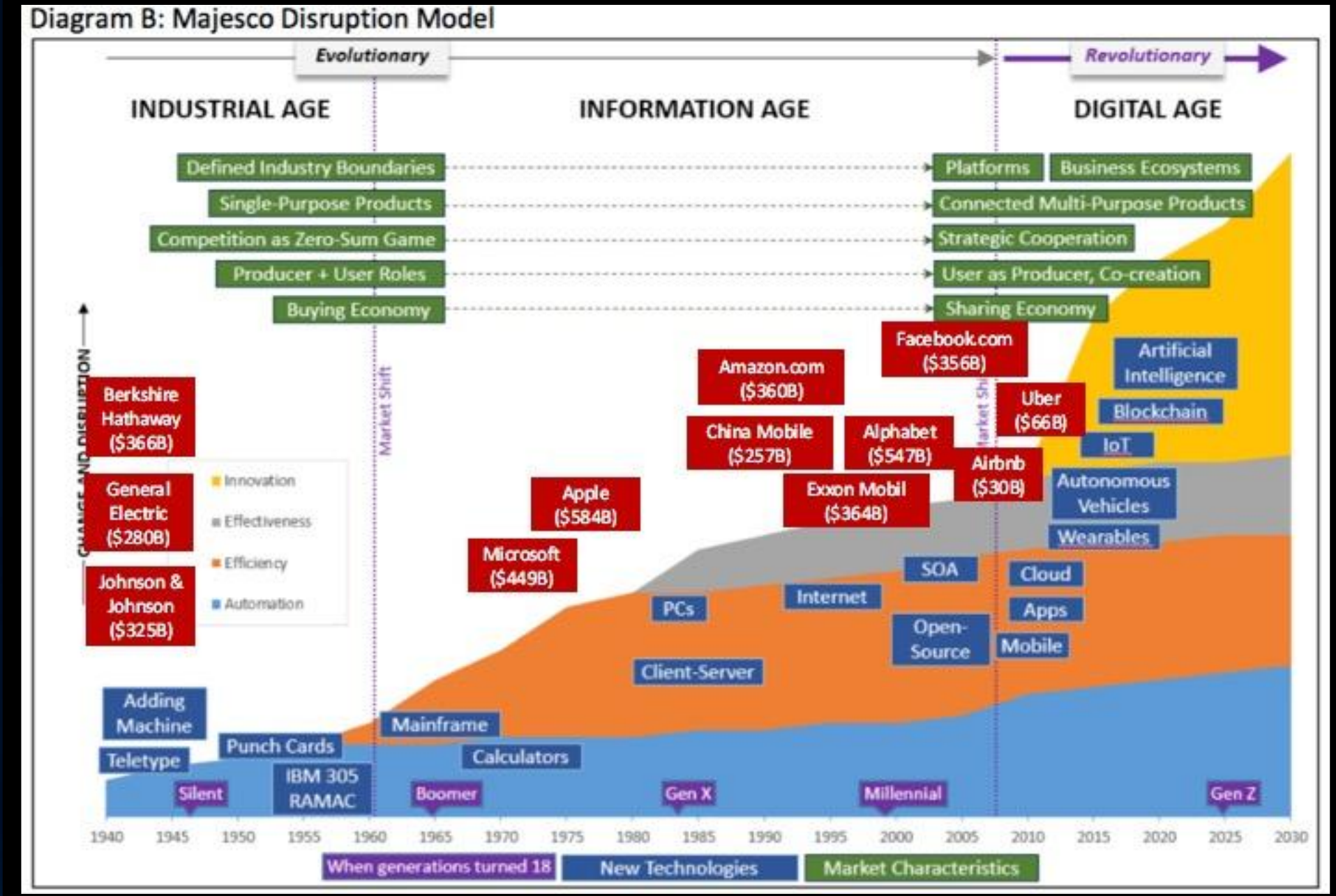
LECTURER: HRISTO HRISTOV
CHRISSTOV.COM

Agenda

1. Seminar opening
2. Introduction
   - Blockchain 101 – what is (a) blockchain
   - Blockchain data – what are blocks and how are they created
   - Blockchain network – some major differences between a blockchain and a traditional DB
   - Blockchain consensus – the process of confirming transactions

<short break>

3. Comparison
   - Core differences between the bitcoin & ethereum blockchain
4. Blockchain Applications
   - Use-cases and implementation scenarios
   - How to create your own blockchain instance
5. Q & A



Source: Majesco (NYSE MKT: MJCO), a global provider of core insurance software

What Is a Blockchain?

- A distributed database
- Maintains a growing list of ordered records - blocks.
    - Each block has a timestamp and a link to a previous block.
- Trustless but trustworthy system

Why the fuss?

- Eliminating the need for an intermediary of any sort
    - recording transactions, establishing identity and establishing contracts
- can be used to store any kind of digital information, including computer code. (not only bitcoins)

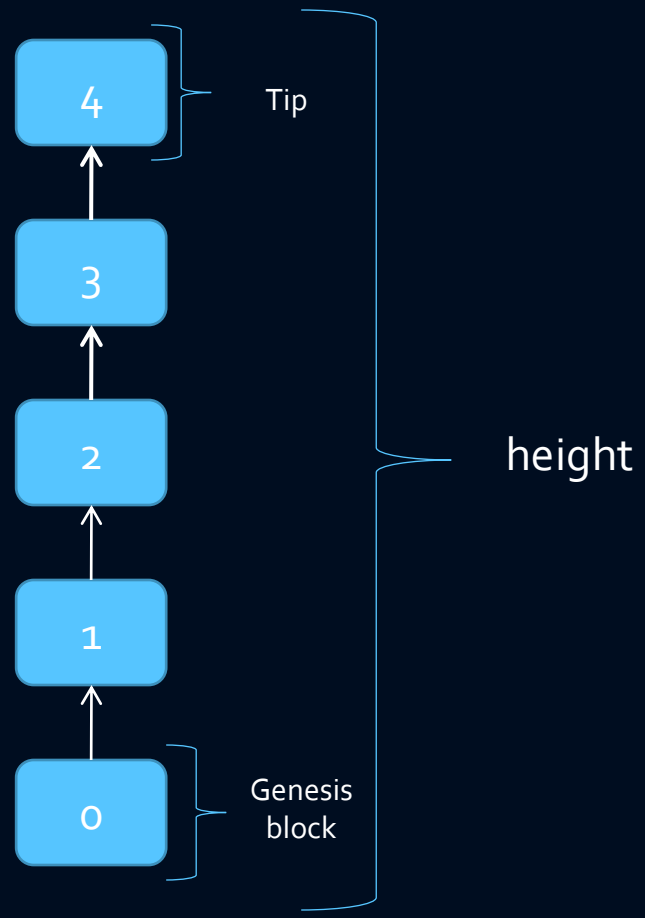
Most important features:

- no middleman required
- anonymous/private
- distributed
- cryptographically secure

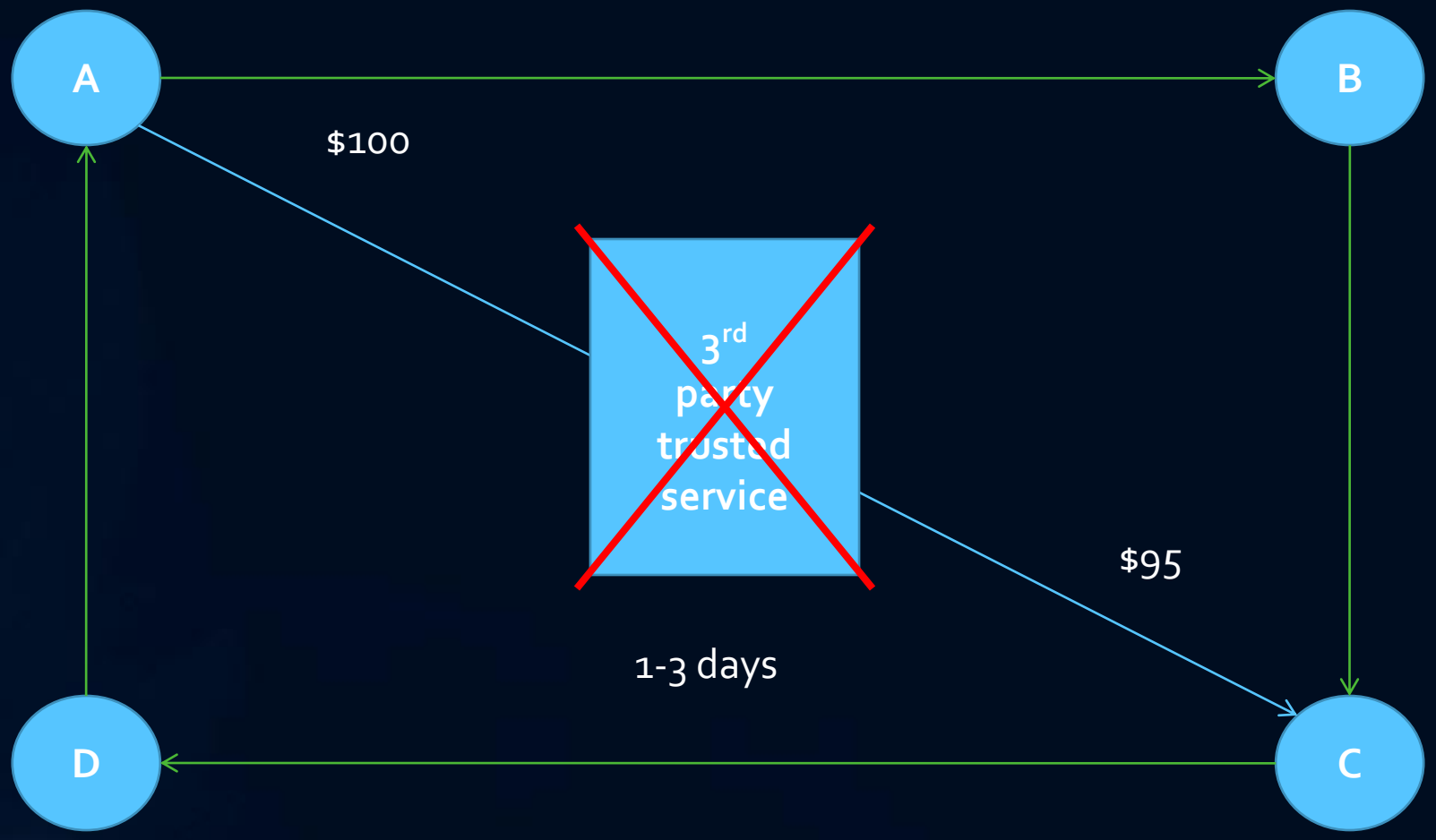
Let us get into the details:

- back-linked structure of ordered data
- often visualized as vertical stack with each block on top of the other (remember this please)
- Some terms:
  - blockchain height
  - blockchain tip
  - genesis block

- Each block is identified by a hash (SHA256 - bitcoin)
- Each block references a previous one – the parent
- Each block contains its parent hash in its header
- **The sequence of hashes linking each block to its parent creates a chain going back all the way to the first block ever created, known as the *genesis block*.**
- When the parent is modified in any way, its hash changes.
- **This change affects the child block and all subsequent blocks**
  - *Think of a big pot filled with soil – the surface is soft, but the deeper you go, the more stable it gets*



An Example

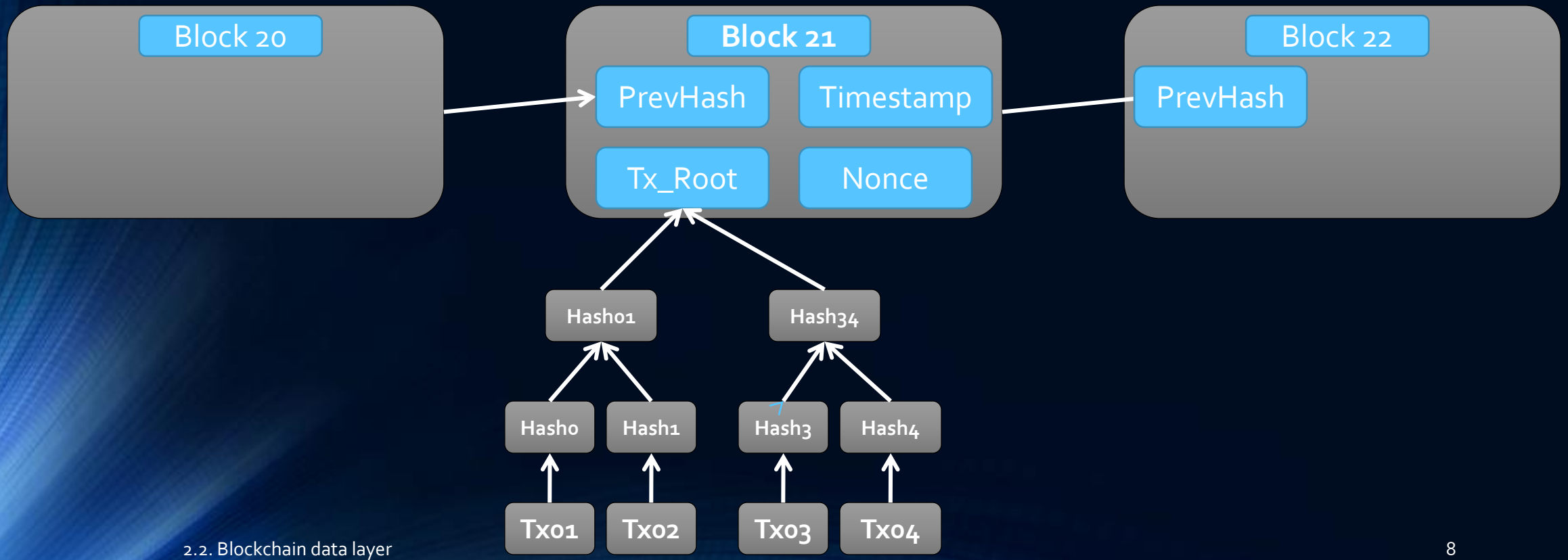


Digital ledger = blockchain

- Think also of pages in a book
- Blockchains by definition are linear and grow in chronological order
- Blockchains are P2P networks, with each peer holding a copy of the ledger (contrast to client-server)

Data Layer

- Data is stored in blocks
- Blocks are chained together via a hash
- Hash = a program that processes data(transactions) and outputs a piece of data. This process is also called mapping.
- Nonce – makes it impossible to duplicate block. In essence it is a random number.

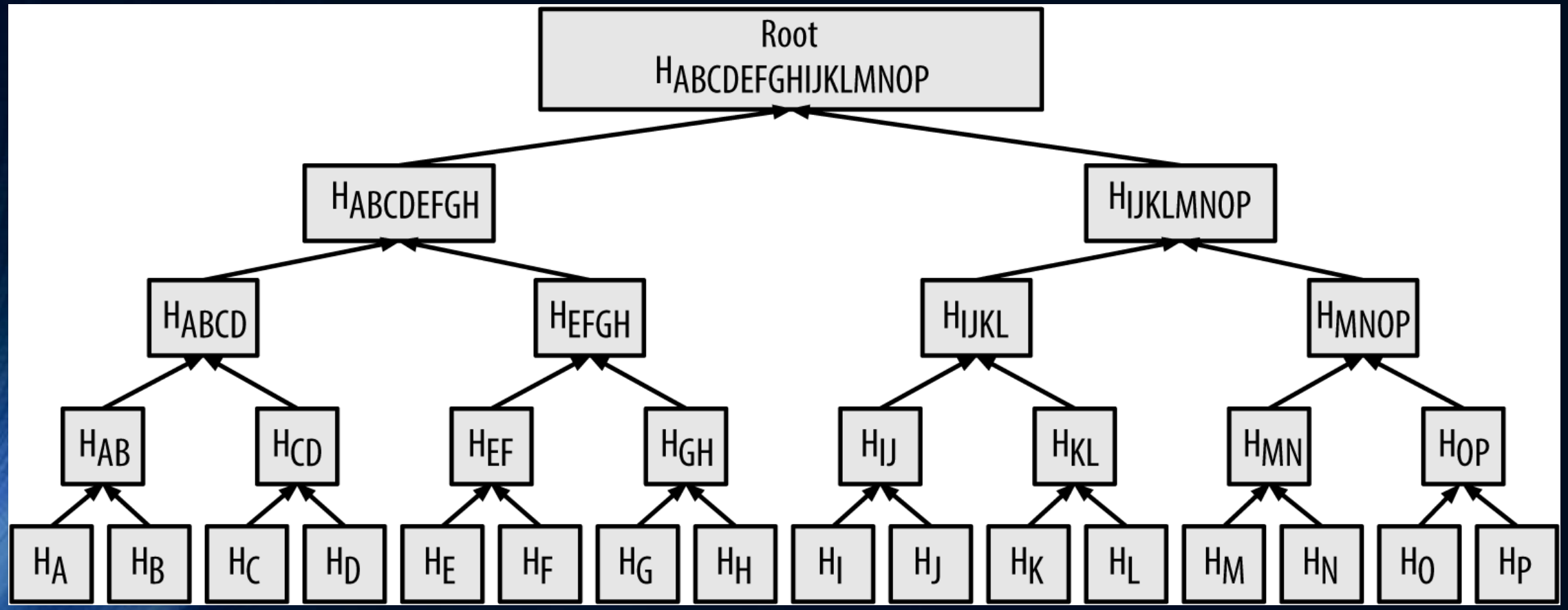


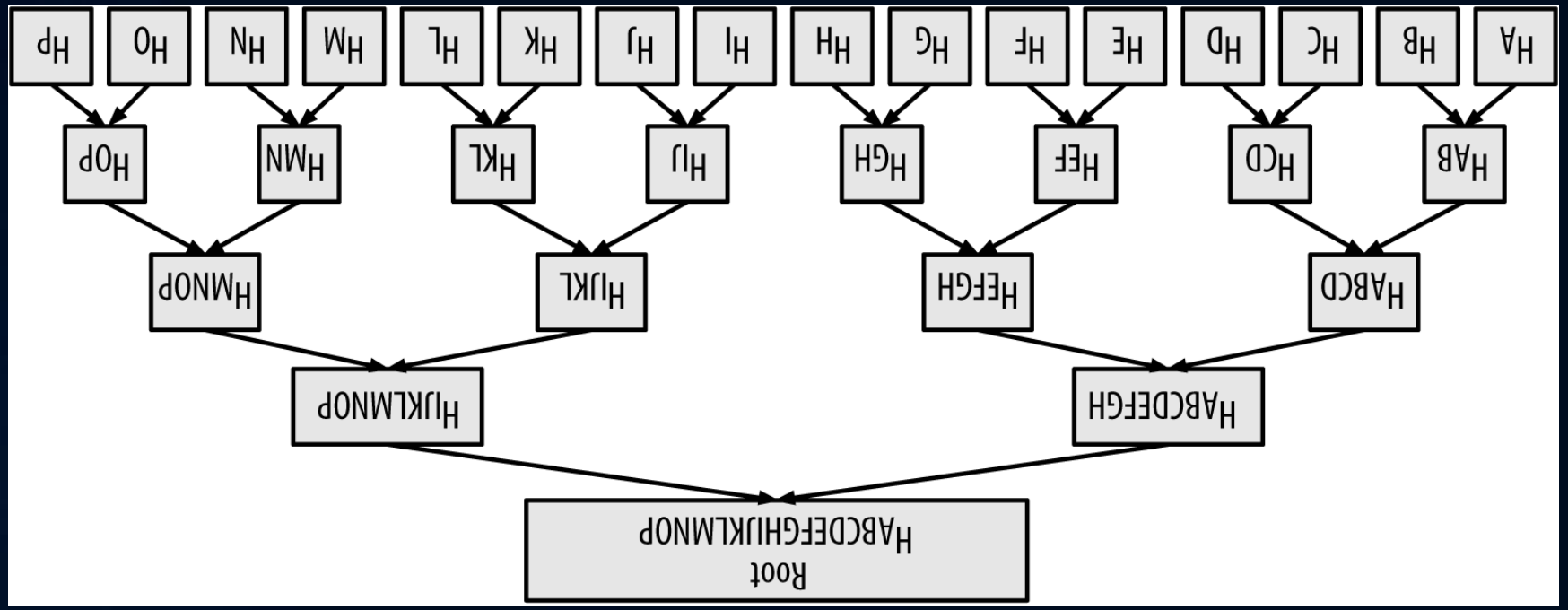
Briefly About Hashing

- Hashing:
  - taking data of random size and mapping it to data of fixed size
- Hash
  - The output of the hashing functions
- One-way hashing function
  - Easy to check if true BUT impossible to recreate the input data
- Hashing ≠ encrypting
  - Encryption supposes someone will decrypt something, with hashing you can only check values

Block Data Structure

- Merkle tree = binary hash tree
- has the ability to make a large piece of data very small, allowing quick access (reference) and provability
- "I had certain data, the hash is the proof"




Root
HABCDEFGHIJKLMNOP
HABCDEFGH
HIJKLMNOP
HABCD
HEFGH
HIJKL
HMNOP
HAB
HCD
HEF
HGH
HIJ
HKL
HMN
HOP
HA
HB
HC
HD
HE
HF
HG
HH
HI
HJ
HK
HL
HM
HN
HO
HP

Network Layer

- by definition blockchains are ***decentralized*** and ***distributed***
  - i.e. a list of transactions is replicated across a number of computers, rather than being stored on a central server.

- If a certain blockchain does not posses this attributes it is:
  - Either a centralized/private blockchain
  - Or a database incorrectly termed as a "blockchain"

- Full nodes vs Miners in a network
  - Simplified Payment Verification (SPV)

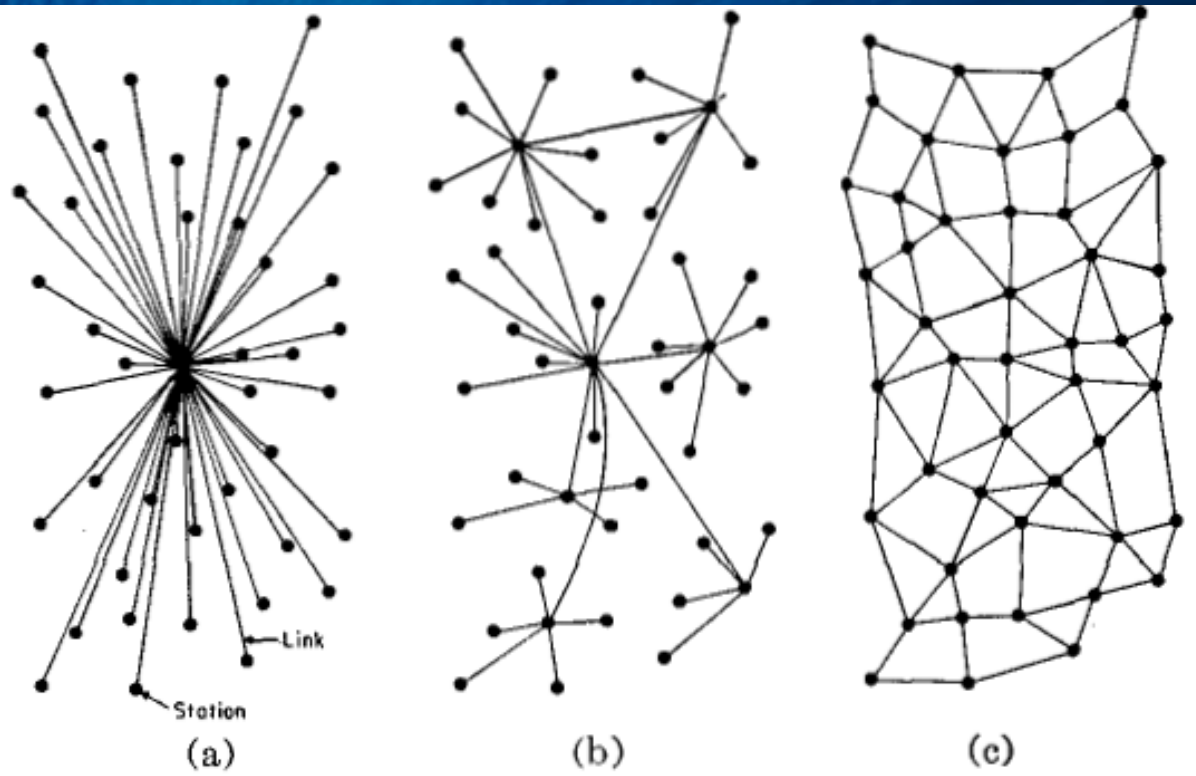
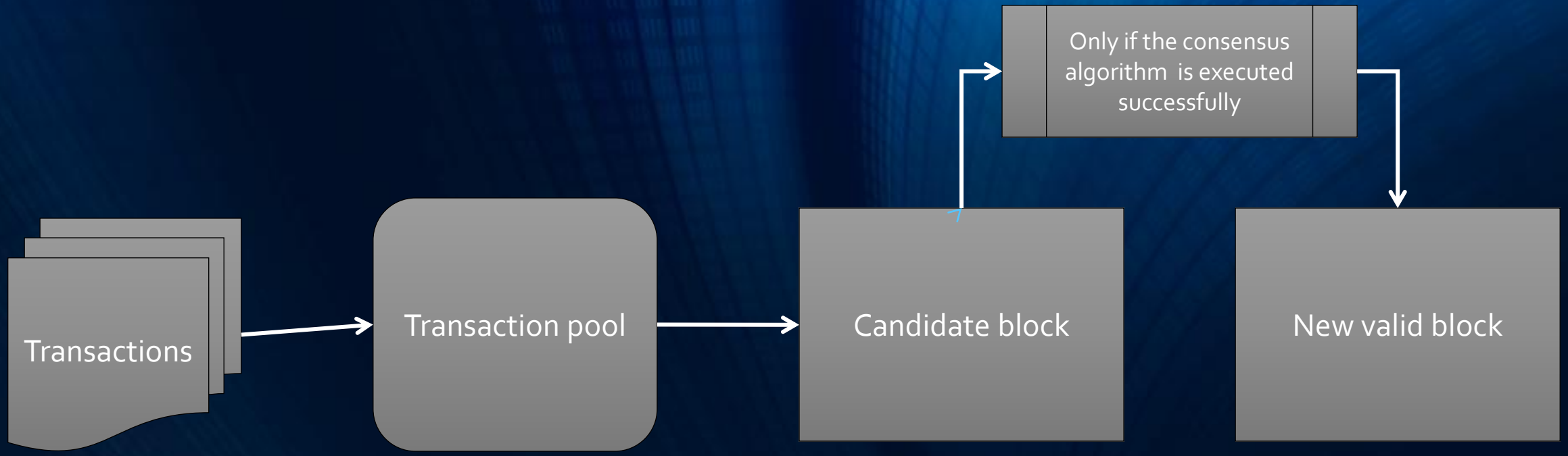


Fig. 1—(a) Centralized. (b) Decentralized. (c) Distributed networks.

Blockchain consensus

- How do blockchain nodes agree which transactions are valid and hence be included in the blockchain?
- The block is the final result of performing ***distributed*** consensus
- Consensus protocols – PoW / PoS. Consensus protocols make the blockchain what it is: store data in an ***open*** but ***immutable*** way



Consensus Protocols

- The consensus protocol does two things:
  - ensures that the **next** block in a blockchain is the one and **only version of the truth**
  - keeps **powerful** adversaries from derailing the system and successfully forking the chain
- **Currently, in Bitcoin, the SHA256 hashing must produce a sequence starting with 18 zeros. The number of zeros is set by the difficulty.**
- **What are the chances to have a hash starting with 0? And 2^n - for every subsequent zero**

Other Protocols:

- **Proof of stake**
  - a 'validator' invests in the coins of the system
  - validators (also called stakeholders, because they hold a stake in the system) are paid strictly in transaction fees.
  - require much less computational power and potentially eliminates the 51% problem
  - other issue: validators with nothing to lose can behave badly, e.g. double sign

- **Proof of activity**
  - a hybrid between PoW and PoS
  - the final block hash will require a signage

- **Proof of burn**
  - By committing your coins to never-never land, you earn a lifetime privilege to mine on the system based on a random selection process.

Consensus Protocols - 2

• Proof of capacity

• you 'pay' with hard drive space. The more hard drive space you have, the better your chance of mining the next block and earning the block reward.
• The more plots you have, the better your chance of finding the next block in the chain.

• Proof of elapsed time (by Intel)

• This system works similarly to proof of work, but consumes far less electricity.
• Instead of having participants solve a cryptographic puzzle, the algorithm uses a trusted execution environment (TEE) – such as Software Guard Extensions (SGX) – to ensure blocks get produced in a random lottery fashion, but without the required work.
• SGX protects selected code and data from disclosure or modification.
• Intel's approach is based on a guaranteed wait time provided through the TEE.
• **the protocol requires investing trust in a third party (Intel). Could be suitable for private blockchains.**

Proof of Service (PoSe)

• not a consensus protocol
• Refers to tier 2 of a crypto ecosystem but works hand in hand with tier 1

Bitcoin & Ethereum blockchain

How do they stack up against each other?

Criteria	Blockhain	
	Bitcoin	Ethereum
Open-Source	Yes	Yes
Currency	bitcoin	ether
Block time	10 minutes	14.5 seconds
Block size	1 MB	No direct limit
Consensus protocol	PoW	PoW + PoS
Protocol function	SHA-256	Ethash
Network	Bitcoin network protocol	ĐΞVp2p
Scripting language	None (Bitcoin Scripting Language)	Solidity (JS+C# libraries)
Storage	Transactions	Transactions + code
Data Structure	Binary hash tree	Modified Radix tree

Bitcoin & Ethereum blockchain - Currency

- Initial distribution: mining vs ICO
- Total supply: 21m vs constant annual rate = 0.3 x ICO
- Ethereum's founders studied Bitcoin economics for a while before coming up with Ethereum

Bitcoin & Ethereum blockchain - Blocks

- Bitcoin: 10 minute blocks, the blockchain can process 3 - 7 transactions per second
- Ethereum: 14.5 sec blocks, the blockchain can process 25 transactions per second
- Compare to Visa's scale

Bitcoin & Ethereum blockchain
- Consensus protocol and function

- Bitcoin: PoW blockchain, hashing SHA-256
- Ethereum: PoW - > PoS blockchain , Ethash-ing:
    - finding a nonce input to the algorithm so that the result is below a certain threshold
    - the consensus function is ASIC resistant

Bitcoin & Ethereum blockchain - Network

- Bitcoin: the Bitcoin Network Protocol, all communications over TCP
- Ethereum: ĐΞVp2p -
    - geared towards building a robust transport
    - well-formed network
    - software interface in order to provide infrastructure which meets the requirements of distributed or decentralized applications such as Ethereum

Bitcoin & Ethereum blockchain - Scripting Language

- Bitcoin: Script
  - Scripting provides the flexibility to change the parameters of what's needed to spend transferred Bitcoins. For example, the scripting system could be used to require two private keys, or a combination of several, or even no keys at all.
  - slightly lower level than C

- Ethereum: Solidity
  - similar to JavaScript, high-level language
  - with Solidity you can create and implement your own “smart contracts” that exist on the Ethereum Blockchain

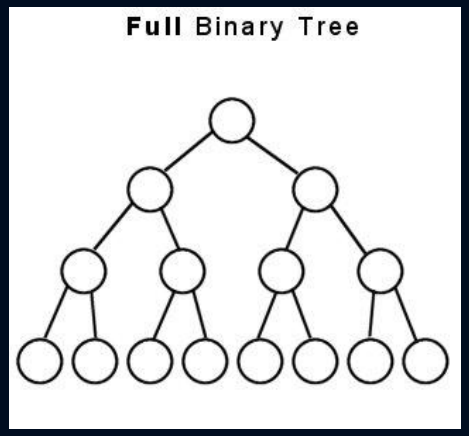
Bitcoin & Ethereum blockchain - Storage

- Bitcoin:
  - transactions
  - some conditions about those transactions

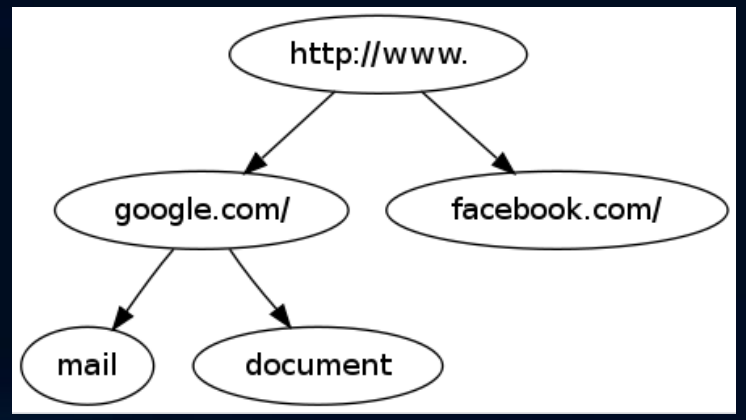
- Ethereum:
  - transactions
  - "smart contracts" – the code that is stored and executed in the blockchain
  - smart contracts essentially make possible blocks on the blockchain to store much more that sending and receiving sums
  - the smart contract contain **state** and **functions, i.e. business logic**
  - can be audited at any time, just like checking if a transaction exists on the blockchain
  - all nodes on the network have a copy
  - smart contracts are executed by the EVM

Bitcoin & Ethereum blockchain - Data Structure

- Bitcoin:



- Ethereum:

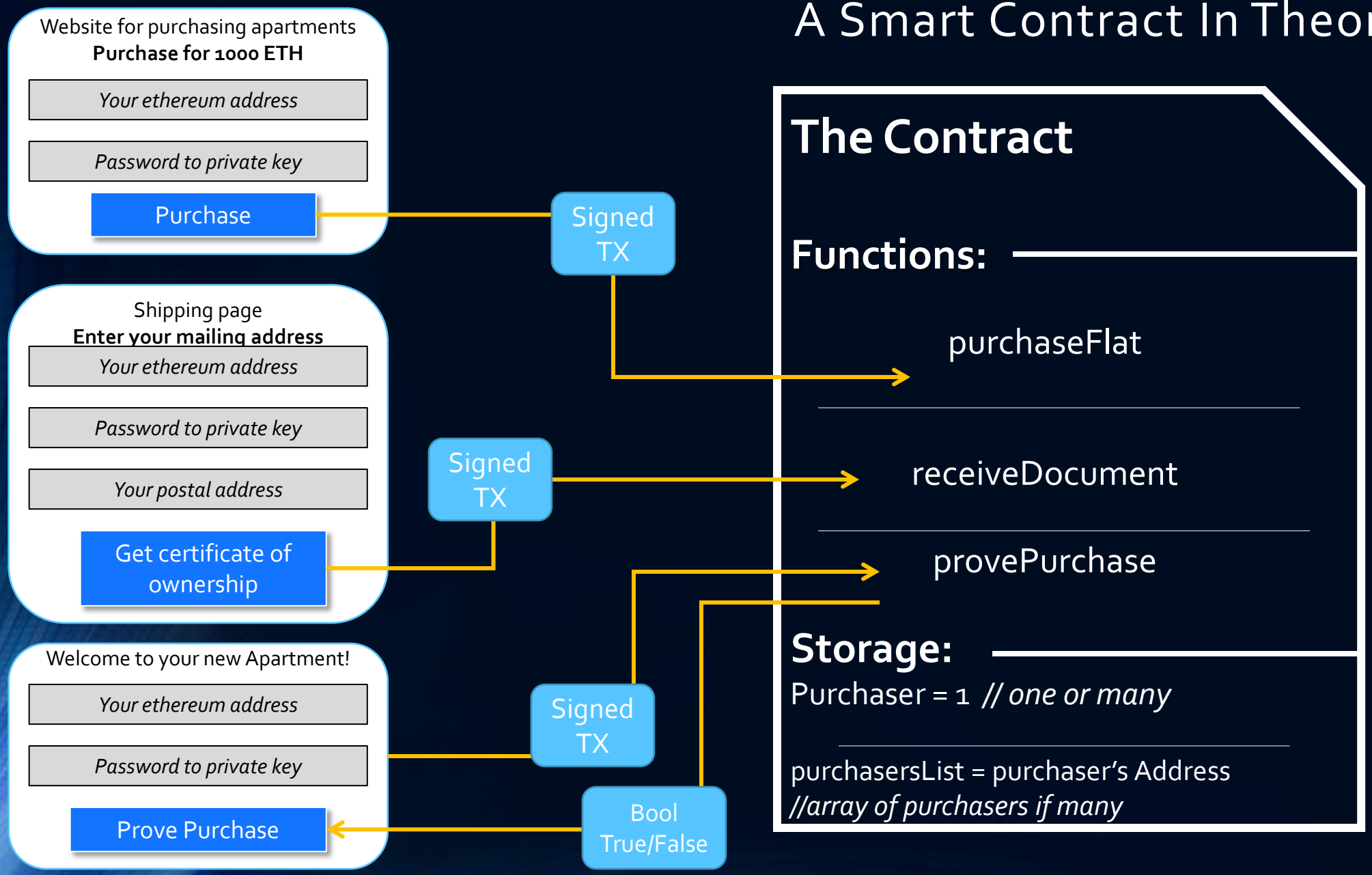


In very, very, very layman's terms -

Ethereum block data structure allows for:

- Much more checks of transactions and states
- Much better light client functioning (remember SPV's?)

A Smart Contract In Theoretical Action



Smart Contracts Essentials

- smart contracts are computer protocols
- they are based on contractual law
- smart contracts **execute the terms of the contract**
- Everything that:
    - can be coded
    - quantified or qualified in one way or another
    - is subject to an agreement

And most importantly, smart contracts are on-the-chain, thus

- everyone can look them up
- they are immutable
- cost some money to be executed, i.e. to call a function in a smart contract

Smart Contracts Essentials - 2

- Contracts are immutable, irrevocable, incorruptible APIs.
- Disclaimer: Solidity does provide a private selfdestruct method.
- Contracts on the public network have 100% uptime.
- Writing to the blockchain will cost you Ether (via gas).
- Writes will take ~15 seconds to be confirmed.
- Reads are free and instantaneous.
- Transactions only begin through external action.

Blockchain Types

Type	Permissioned	Decentralized	Trustless
Public	No	Yes	Yes
Hybrid	Yes	Partially	No
Private i.e. Database	Yes	No	No

- **Public blockchains**
  - no central authority, anyone can read, write and verify(participate in consensus) transactions
- **Hybrid blockchains**
  - no central authority, only selected participants can **write or verify transactions**. Read access can be public or not.
- **Private blockchains**
  - write permissions are centralized to one person/node or group. Read access could be granted to outside people

Blockchain Types By Industry

So a blockchain type would largely depend on the industry in point:

1. FinTech
    - Private blockchains
        - PWC
        - Quorum by JP Morgan
        - KPMG & Microsoft Azure

2. Non-financial
    - Private & public blockchains
        - Maersk & IBM

Blockchain Use cases

Financial	Corporate	Government	Cross-industry	Hospitality
International Payments	Supply chain management	Record management	Accounting	AirBnB
Capital Markets	Healthcare	Identity management	Shareholder's voting	Turo
Trade Finance	Real estate	Voting	Record management	Hotels
Regulatory Compliance & Audit	Media	Taxes	Cyber security	
KYC & anti-money laundering	Energy		Big data	
Insurance			Data storage	
P2P transactions			IoT	

Create Your Own Blockchain Instance

1. **BaaS by Azure**
   - a single-click, cloud-based blockchain developer environment
   - Learn, play & fail fast
   - Create your own private and hybrid blockchains

2. **Rubix by Delloite**
   - Architecture designed to support fully functional private networks

3. **IBM blockchain on Bluemix**
   - Distinguished for its pluggable architecture
   - Writing smart contracts with popular languages

4. **Others**
   - Multichain – target experimenting with blockchain for financial institutions
   - Walletbuilders.com – all in one , Bitcoin-based solution
   - OpenChain – a transaction chain, offering real-time confirmations

What About Adoption?

In terms if technology adoption and maturity:

- Feasible adoption can begin after the technology has matured to a level that can be easily supported
- There is a great chance that developing countries will be “adoption engines”
- Current stage: end of the beginning

If you can have a currency without a government body, then you might also be able to have a system fulfilling government functions, without a government.

Thank you for taking the time & interest!