Decentralized System: Introduction and Possible Future Research

Dr. Bambang Purnomosidi D. P.
@bpdp (twitter), bambangpdp (telegram, instagram, fb)

STMIK AKAKOM - May 18, 2018

Agenda

1. Cryptocurrency
2. Blockchain
3. Decentralized Application (DApp)
4. DApp Development
5. Future Research Directions

Cryptocurrency

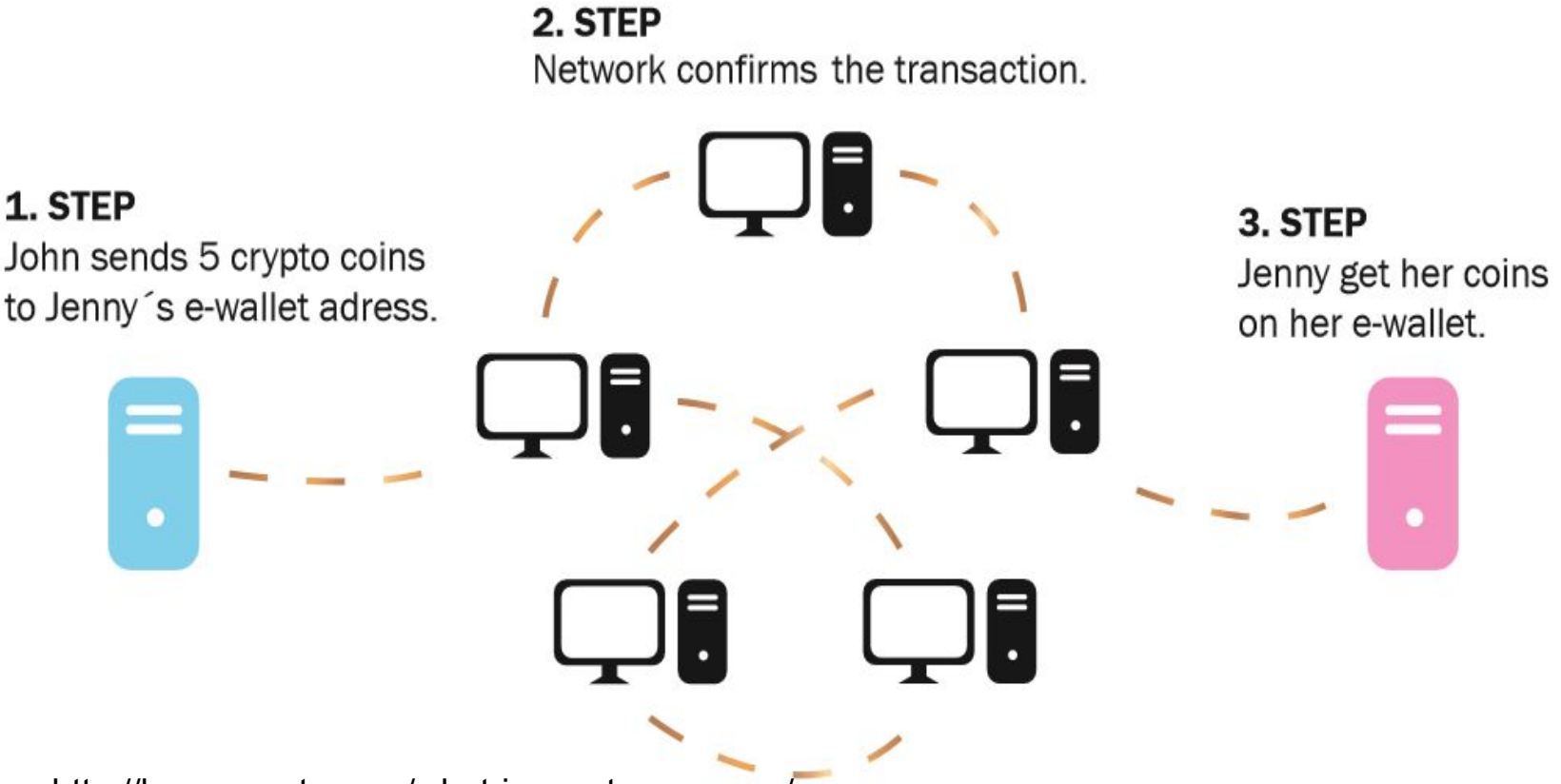
Agenda for Cryptocurrency

1. What is cryptocurrency?
2. How does cryptocurrency work?
3. List of cryptocurrencies
4. Why are there so many cryptocurrecies?
5. Cryptocurrency mining

What is Cryptocurrency?

- A digital asset, a type of digital currency.
- Transferred from peer-to-peer, no middlemen (no bank, no financial institution).
- Just as currency (money), its main function is as medium of exchange.
- Uses cryptography to:
  - secure transaction
  - control the creation of additional units
  - verify the transfer of assets
- Decentralized, as opposed to central banking system, made possible by blockchain
- First cryptocurrency: bitcoin (2009). Other than bitcoin, they are called **altcoin**.

How Does Cryptocurrency Work?



source: http://learn-crypto.com/what-is-crypto-currency/

List of Cryptocurrencies

1. First cryptocurrency: **Bitcoin** (2009)
2. **Altcoin**: coin other than Botcoin: litecoin, ether, ada, etc
3. Since Bitcoin, many cryptocurrencies emerge. As of May 17, 2018, there are Litecoin, Ether, and more (1592 cryptocirrencies). See: https://coinmarketcap.com/all/views/all/

Why Are There So Many Cryptocurrencies?

- Because that is possible.
- One may use already available blockchain system (such as Ethereum) to create cryptocurrency.

Cryptocurrency Mining

- Mining: a validation of transactions
- It’s a process of adding transaction records to cryptocurrency public ledger of past transactions.
- See https://www.buybitcoinworldwide.com/mining/

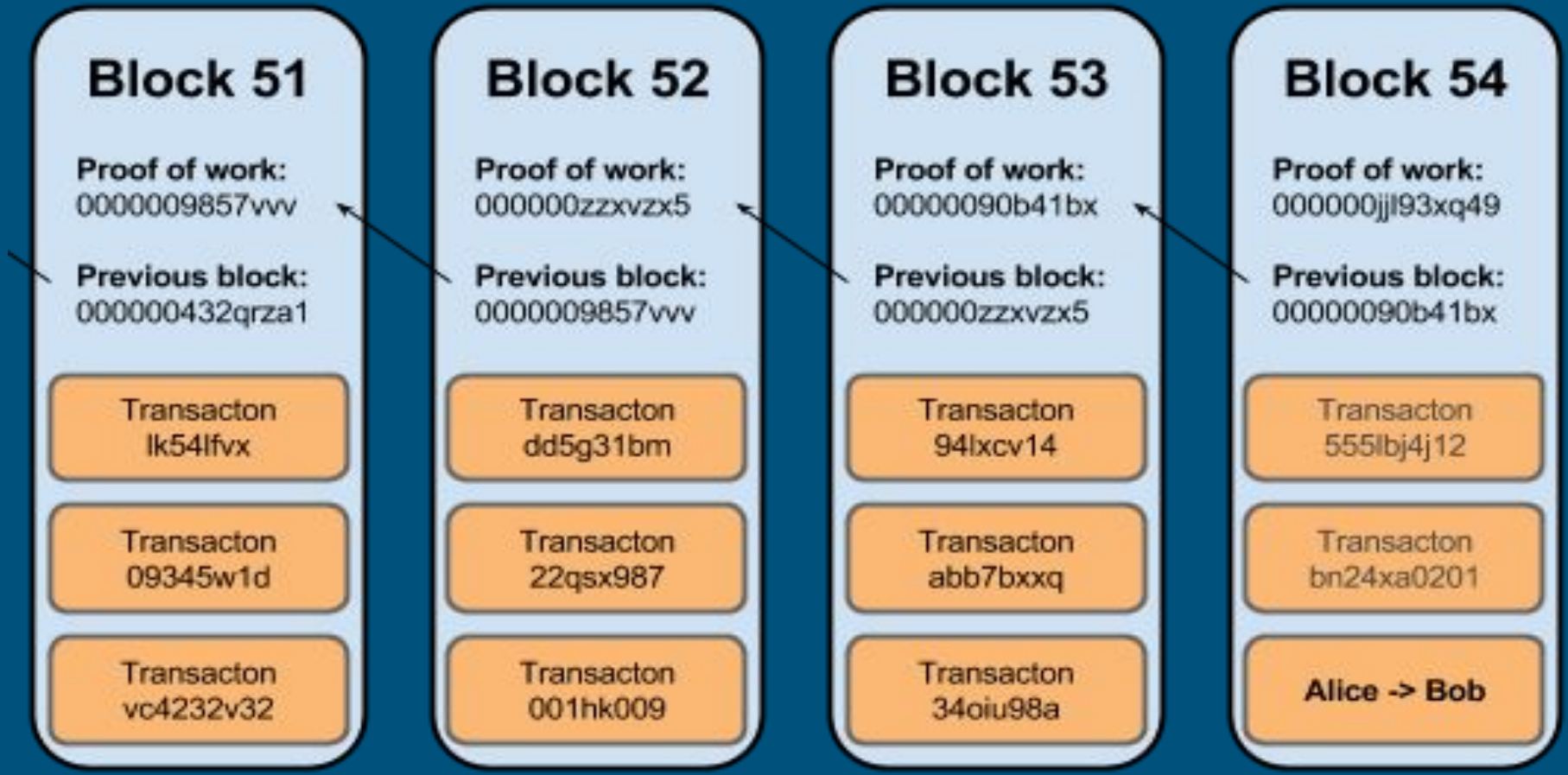
Blockchain

Agenda for Blockchain

1. What is blockchain?
2. How does blockchain work?
3. Blockchain structures
4. Blockchain explorer
5. Types of blockchain
6. Blockchain usage
7. Public blockchain: Ethereum
8. Private blockchain: Multichain
9. Consortium blockchain: Corda

What Is Blockchain?

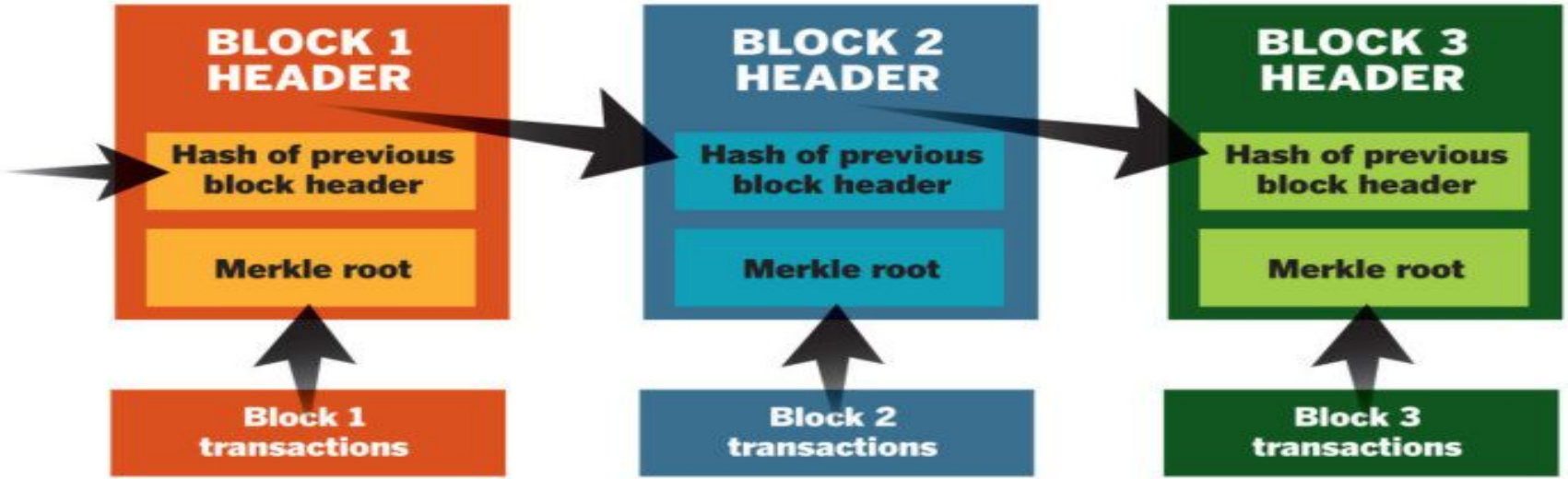
- “We can define the blockchain as a system that allows a group of connected computers to maintain a single updated and secure ledger.” (Michele D’Aliessi, 2016).
- “A blockchain is a decentralized and distributed digital ledger that is used to record transactions across many computers so that the record cannot be altered retroactively without the alteration of all subsequent blocks and the collusion of the network”. (Stephen Armstrong, 2018).
- Preliminary research: Cryptographically secured chain of blocks (Stuart Haber dan W. Scott Stornetta, 1991), Ross J. Anderson (1996), Michael Doyle (1997), Bruce Schneier and John Kelsey (1998), Nick Szabo (1998, bit gold), Stefan Konst (2000).
- Satoshi Nakamoto (2008): Bitcoin



Source: http://dataconomy.com/2015/10/wtf-is-the-blockchain-a-guide-for-total-beginners

How does blockchain work?

With blockchain technology, each page in a ledger of transactions forms a block. That block has an impact on the next block or page through cryptographic hashing. In other words, when a block is completed, it creates a unique secure code, which ties into the next page or block, creating a chain of blocks, or blockchain.



source: https://www.cio.com/article/3055847/security/what-is-blockchain-and-how-does-it-work.html

SIMPLIFIED BITCOIN BLOCK CHAIN

Blockchain structures

1. **Blocks**. Blocks hold batches of valid transactions that are hashed and encoded into a Merkle tree. Each block includes the cryptographic hash of the prior block in the blockchain, linking the two. The linked blocks form a chain. A **hash tree** or **Merkle tree** is a tree in which every leaf node is labelled with the hash of a data block and every non-leaf node is labelled with the cryptographic hash of the labels of its child nodes
2. **Decentralization**:Every node in a decentralized system has a copy of the blockchain. Data quality is maintained by massive database replication and computational trust. No centralized "official" copy exists and no user is "trusted" more than any other.
3. **Openness and accessibility**
4. **Immutability and irreversibility**
5. **Openness and accessibility**
6. **Anonymity**

Blockchain explorer

- Online block chain browser which displays the contents of individual blockchain blocks and transactions and the transaction histories and balances of addresses.
- Example: **Hyperledger Explorer**

Types of Blockchain

1. **Private blockchains**: a fully private blockchain is a blockchain where write permissions are kept centralized to one organization.
2. **Consortium blockchains**: a consortium blockchain is a blockchain where the consensus process is controlled by a pre-selected set of nodes; for example, one might imagine a consortium of 15 financial institutions, each of which operates a node and of which 10 must sign every block in order for the block to be valid.
3. **Public blockchains**: a public blockchain is a blockchain that anyone in the world can read, anyone in the world can send transactions to and expect to see them included if they are valid, and anyone in the world can participate in the consensus process.

Blockchain usage

1. Distributed ledger for cryptocurrency (FinTech)
2. Land / Property registeration (Sweden and Georgia)
3. Decentralized Library (Alexandria project)
4. Govenment and National Currency: e-Dinar (Tunisia) and eCFA (Senegal)
5. Digital assets tracking
6. Identity
7. Digital Voting
8. Distributed Storage
9. … and many more.

Public blockchain: Ethereum

1. https://ethereum.org
2. Ethereum is a decentralized platform that runs smart contracts: applications that run exactly as programmed without any possibility of downtime, censorship, fraud or third party interference.
3. It has **ether** as its cryptocurrency.
4. Provides **smart contracts** for DApp development
5. Provides **Ethereum Virtual Machine**. EVM is an environment where arbitrary code of smart contracts and other operations can be executed.
6. There are some high level language for DApp development:
    a. Solidity
    b. Serpent
    c. LLL

Ethereum Client

- Ethereum client refers to any node able to parse and verify the blockchain, its smart contracts and everything related. It also allows you/provides interfaces to create transactions and mine blocks which is the key for any blockchain interaction.
- Official Reference Implementation:
  - Go: go-ethereum (geth):: https://geth.ethereum.org/
  - C++: cpp-ethereum (eth): http://www.ethdocs.org/en/latest/ethereum-clients/cpp-ethereum/
  - Python (pyethapp): https://github.com/ethereum/pyethapp
- 3rd party:
  - Rust: parity: https://parity.io
  - Java: https://github.com/ethereum/ethereumj
  - JS: https://github.com/ethereumjs/ethereumjs-vm
  - Ruby: https://github.com/janx/ruby-ethereum

Private blockchain: Multichain

- https://www.multichain.com
- An open source platform for private blockchains, which offers a rich set of features including extensive configurability, rapid deployment, permissions management, native assets and data streams. Although it is designed to enable private blockchains, MultiChain provides maximal compatibility with the bitcoin ecosystem, including the peer-to-peer protocol, transaction/block formats and Bitcoin Core APIs/runtime parameters.

Consortium Blockchain: Corda

- https://corda.net
- Corda is an open source blockchain project, designed for business from the start. Only Corda allows you to build interoperable blockchain networks that transact in strict privacy. Corda’s smart contract technology allows businesses to transact directly, with value.

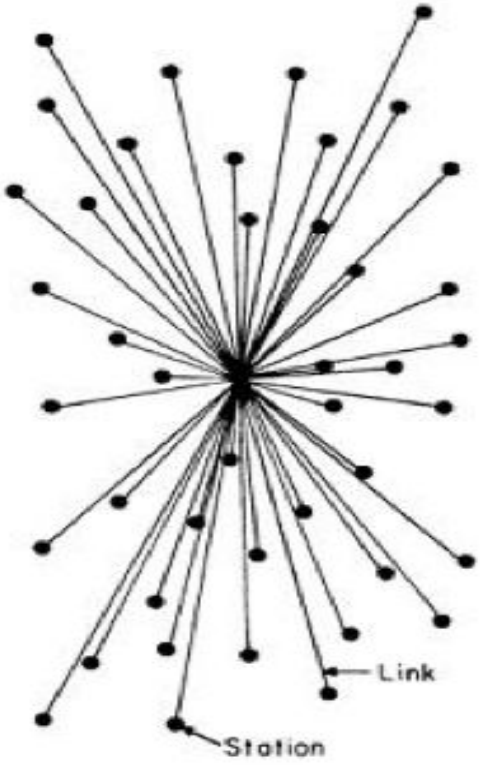
Decentralized Application (DApp)

Agenda for Decentralized Application (DApp)

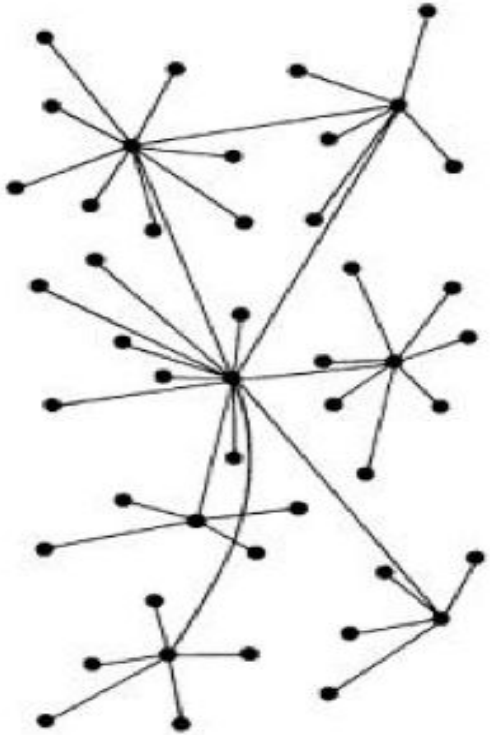
1. What is DApp?
2. DApp architecture
3. DApp taxonomy
4. Ecosystem for DApp development

What is DApp?

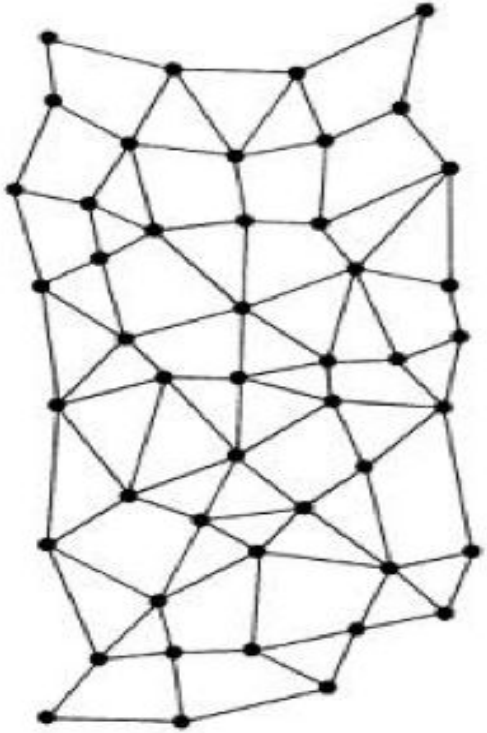
- DApp is an abbreviated form for decentralized application. A DApp has its backend code running on a decentralized peer-to-peer network. Contrast this with an app where the backend code is running on centralized servers.



CENTRALIZED
(A)



DECENTRALIZED
(B)



DISTRIBUTED
(C)

source: http://www.peacebiennale.info/blog/paul-baran-centralized-decentralized-and-distributed-networks-1964/

DApp Architecture

- Just as other common pattern in software development, we also need to separate some components in DApp.
- There is front-end (could be anything: Web, mobile, etc.
- There is also back-end code which access immutable database / distributed ledger.

DApp Taxonomy

1. **Type I**: These types of DApps have their own blockchain (like Bitcoin). Other altcoins also fall under this category as well.
2. **Type II**: decentralized applications use the block chain of a type I decentralized application. Type II decentralized applications are protocols and have tokens that are necessary for their function. The Omni Protocol is an example of a type II decentralized application.
3. **Type III:** These types of DApps use the protocol of a Type II DApp., For example, the SAFE network uses the Omni Protocol for issuing SafeCoins that are then used to build distributed file storage..

DApp Development

Agenda for DApp Development

1. Ecosystem for DApp Development
2. DApp Development for public blockchain (Ethereum)
3. DApp Development for private blockchain (Multichain)
4. DApp Development for consortium blockchain (Corda)

Ecosystem for DApp Development

- Blockchain Platform:
  - Ethereum
  - Tezos
  - Corda
  - Exonum
  - Hyperledger Project
- Development Tools
- See https://www.stateofthedapps.com/ for inspiration.

DApp Development for Public Blockchain (Ethereum)

- The development process of a ĐApp often entails a Whitepaper and a working prototype, a token sale, an initial coin offering (ICO), its implementation and launch.
- From the workflow, developer tasks are:
  - Create token
  - Create application
  - Deploy application to EVM

Ethereum DApp Development Without Framework

- Never use public blockchain for test.
- What you have to learn:
  - Smart contract and how to use Solidity programming language. Get Solidity compiler (https://github.com/ethereum/solidity/releases). Some Linux distros provide official package (ex: Arch Linux pacman: **community/solidity 0.4.18-1**)
  - How to deploy your smart contract to EVM
  - Documentation for Solidity: http://solidity.readthedocs.io/en/latest/
- Development environment: Atom, Emacs, Vim, Intellij, VSCode, See Awesome Solidity (https://github.com/bkrem/awesome-solidity)
- Step:
  - Build everything in local computer. Use **testrpc** to mimick blockchain (https://github.com/ethereumjs/testrpc).
  - Test on the staging blockchain (Ether and any other resource fees are fake).: Ropsten. See https://github.com/ethereum/ropsten and https://ropsten.etherscan.io/ . Also *--morden* when run Ethereum clients.
  - Real Ethereum deployment. Mainnet - Homestead.

DApp Development With Framework in Ethereum

- Truffle Framework (http://truffleframework.com/) - A Swiss army knife for smart contract development and deployment.
- DApp (https://dapp.readthedocs.io/en/latest/) - simple command line tool for smart contract development for package management, source code building, unit testing, simple contract deployments.

Deployment in Ethereum

- In deployment, understanding **Gas** is important.
- Transactions on the Ethereum network require fees in the form of **gas**. The amount of gas depends on the amount of computation required to complete the transaction.
- Estimating transaction costs: http://ethdocs.org/en/latest/contracts-and-transactions/account-types-gas-and-transactions.html

DApp Development for Private Blockchain (Multichain)

- https://www.multichain.com
- Source code: https://github.com/MultiChain/multichain
- Steps:
  - Download **multichain** from https://www.multichain.com/download-install/
  - There are 4 *binary executables*: multichain-cli, multichaind, multichaind-cold, multichain-util
  - Create blockchain using multichain-util, initializing blockchain, also can be used to connect to existing blockchain with multichaind, manipulate blockchain using multichain-cli
  - Explore blockchain using multichain explorer: https://github.com/MultiChain/multichain-explorer (Python)
  - See https://github.com/MultiChain/multichain-web-demo if you need an example of multichain blockchain with PHP and all front-end Web

DApp Development for Consortium Blockchain (Corda)

- https://corda.net
- Source code: https://github.com/corda/corda
- Build with Kotlin (https://kotlinlang.org)
- Needs JDK (also Kotlin if you want to use Kotlin), JDK 8 is the only supported JDK at the moment.
- Also need to understand how Gradle works. See https://gradle.org.
- See example at https://github.com/corda/cordapp-example

- A Corda network is made up of nodes running Corda and CorDapps
- The network is permissioned, with access controlled by a doorman
- Communication between nodes is point-to-point, instead of relying on global broadcasts
- CorDapps (Corda Distributed Applications) are distributed applications that run on the Corda platform. The goal of a CorDapp is to allow nodes to reach agreement on updates to the ledger. They achieve this goal by defining flows that Corda node owners can invoke through RPC calls

Future Research Directions

- The role of economics in distributed system and vice versa
- Limits of decentralization
- Coordination costs
- Modeling behavior of participants
- Complex game-theoretic interactions
- Fault attribution
- Common patterns in cryptoeconomic mechanism desig
- Consensus Algorithm
- Sharding
- Protocol Economics
- Virtual Machine Upgrade
- Hardforking
- Random Number Generation
- Privacy
- Decentralized Exchange

The End