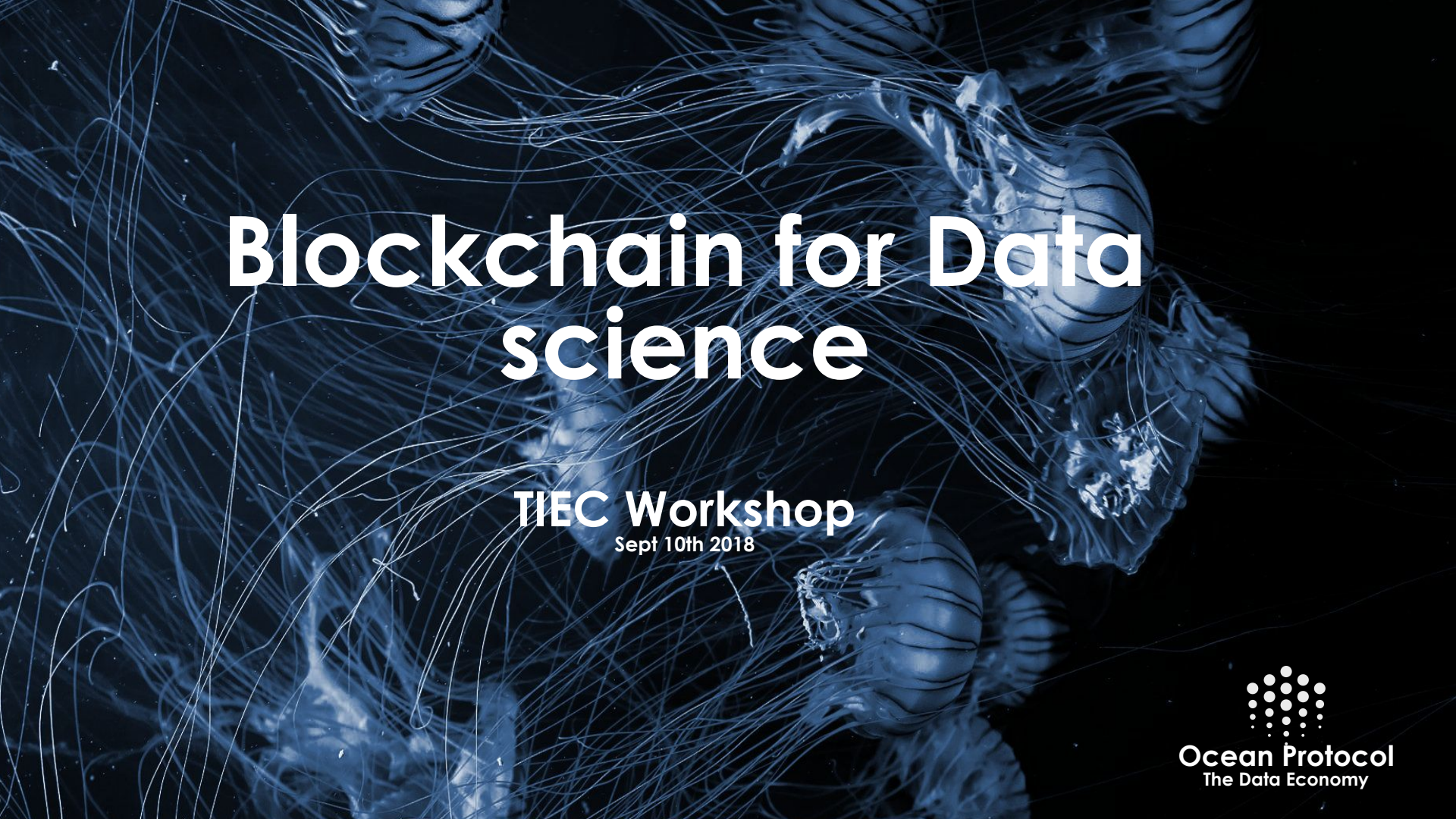
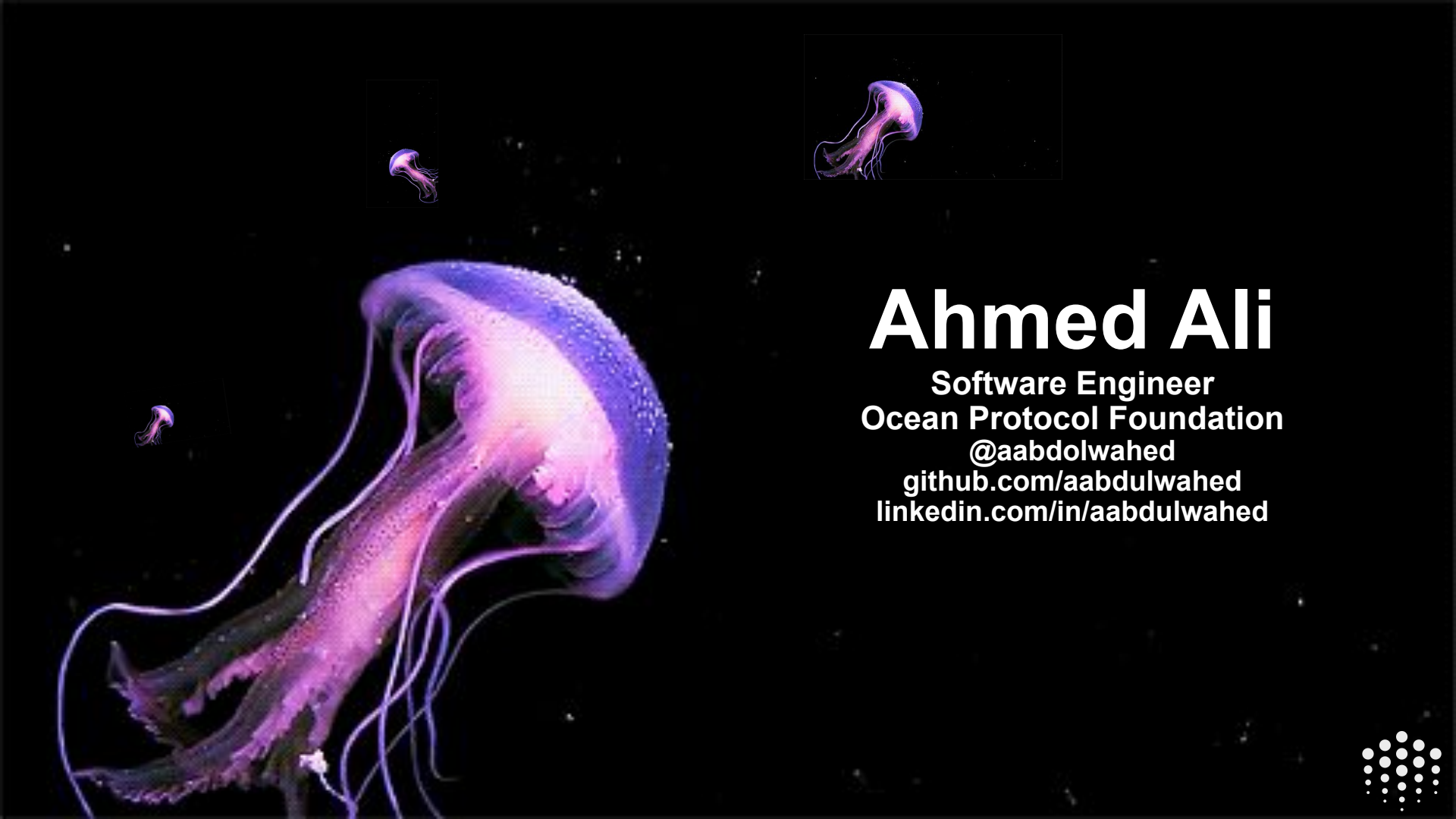

Blockchain for Data science
TIEC Workshop
Sept 10th 2018
Ocean Protocol
The Data Economy



Ahmed Ali

Software Engineer
Ocean Protocol Foundation
@aabdolwahed
github.com/aabdulwahed
linkedin.com/in/aabdulwahed

Objective

Introducing Blockchain mechanics, smart contract essentials and token engineering concepts.

Blockchain For Data-science

Introduction to Blockchain Mechanics
- Introduction
- Core Technologies
- Consensus Protocols

Introduction to Smart Contracts
- What is Smart Contract
- Solidity Essentials
- Lab: Tutorial

Token engineering
- Introduction to TE
- Towards practice of TE

Ocean Protocol
- Tokenize assets with Ocean
- Ocean Roadmap
- Ocean Protocol test network (Plankton)

Open Discussion

Introduction

Financial Crisis

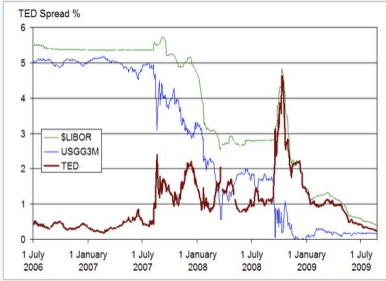
2007–2008

The **financial crisis of 2007–2008**, also known as the **global financial crisis** and the **2008 financial crisis**, is considered by many economists to have been the worst financial crisis since the Great Depression of the 1930s.[1][2][3][4]

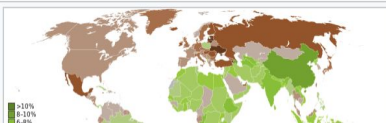
It began in 2007 with a crisis in the subprime mortgage market in the United States, and developed into a full-blown international banking crisis with the collapse of the investment bank Lehman Brothers on September 15, 2008.[5] Excessive risk-taking by banks such as Lehman Brothers helped to magnify the financial impact globally.[6] Massive bail-outs of financial institutions and other palliative monetary and fiscal policies were employed to prevent a possible collapse of the world financial system. The crisis was nonetheless followed by a global economic downturn, the Great Recession. The European debt crisis, a crisis in the banking system of the European countries using the euro, followed later.

In 2010, the Dodd–Frank Wall Street Reform and Consumer Protection Act was enacted in the US following the crisis to "promote the financial stability of the United States".[7] The Basel III capital and liquidity standards were adopted by countries around the world.[8]

Contents [hide]

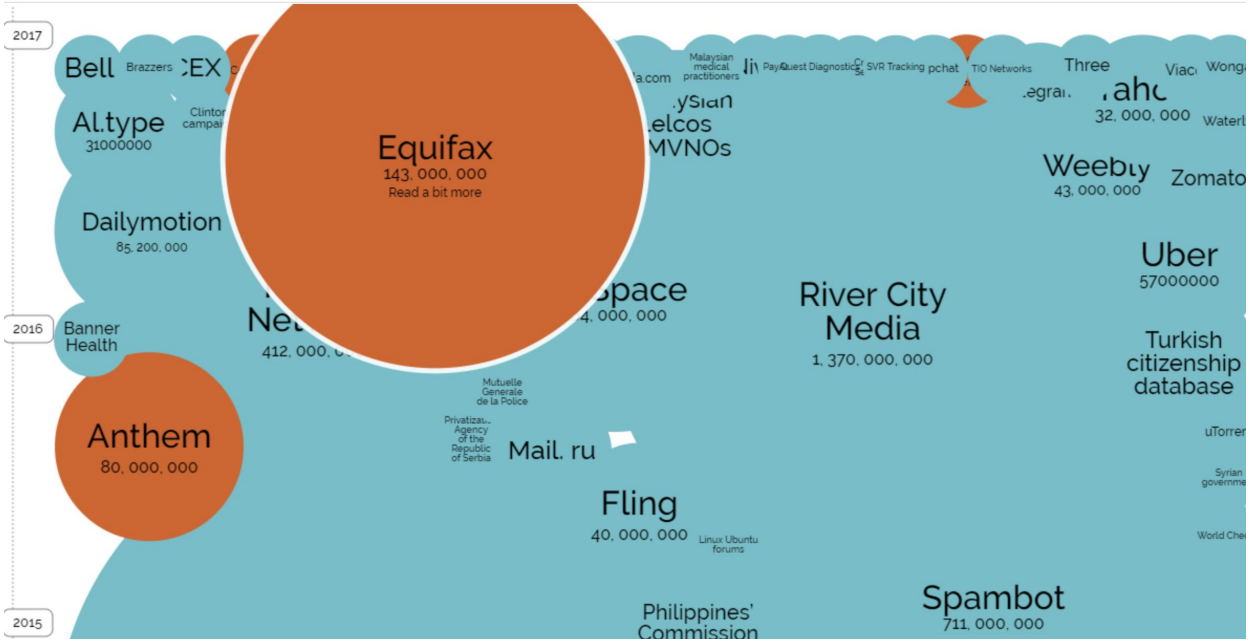


The TED spread (in red) increased significantly during the financial crisis, reflecting an increase in perceived credit risk



Introduction

Data Breaches



http://www.informationisbeautiful.net/visualizations/worlds-biggest-data-breaches-hacks/

Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of

Introduction

Bitcoin

GLOBAL BITCOIN NODES DISTRIBUTION

Reachable nodes as of Sat Sep 08 2018 21:23:28 GMT+0200 (Eastern European Standard Time).

9727 NODES

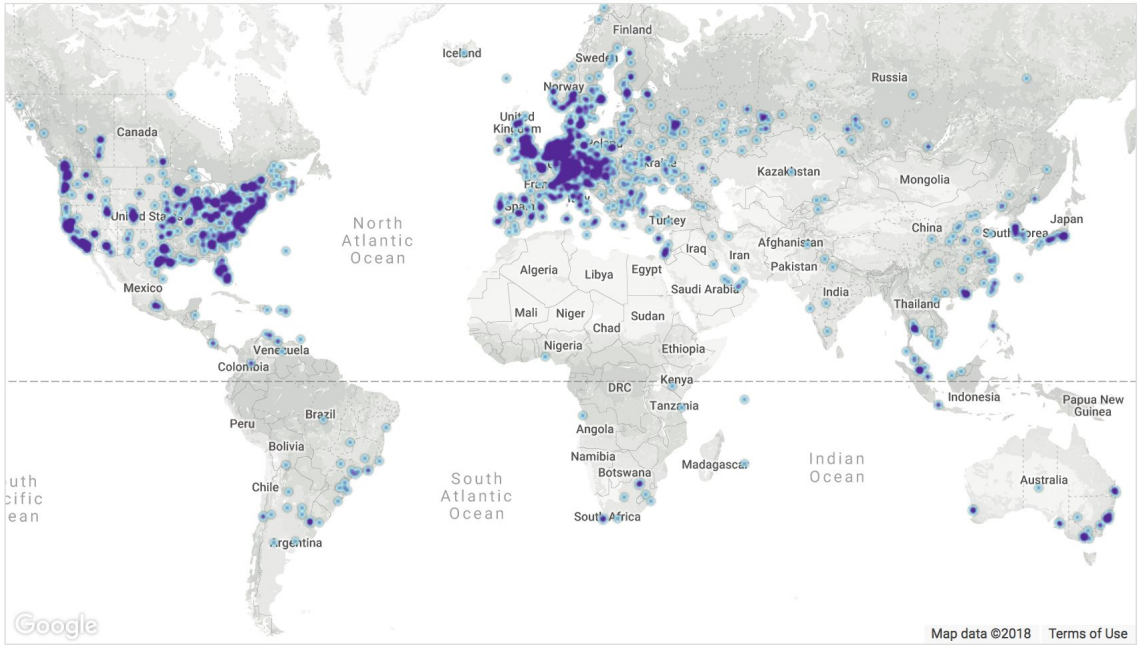
24-hour charts »

Top 10 countries with their respective number of reachable nodes are as follow.

RANK	COUNTRY	NODES
1	United States	2325 (23.90%)
2	Germany	1802 (18.53%)
3	France	669 (6.88%)
4	China	658 (6.76%)
5	Netherlands	487 (5.01%)
6	n/a	449 (4.62%)
7	Canada	362 (3.72%)
8	United Kingdom	288 (2.96%)
9	Russian Federation	276 (2.84%)
10	Singapore	242 (2.49%)

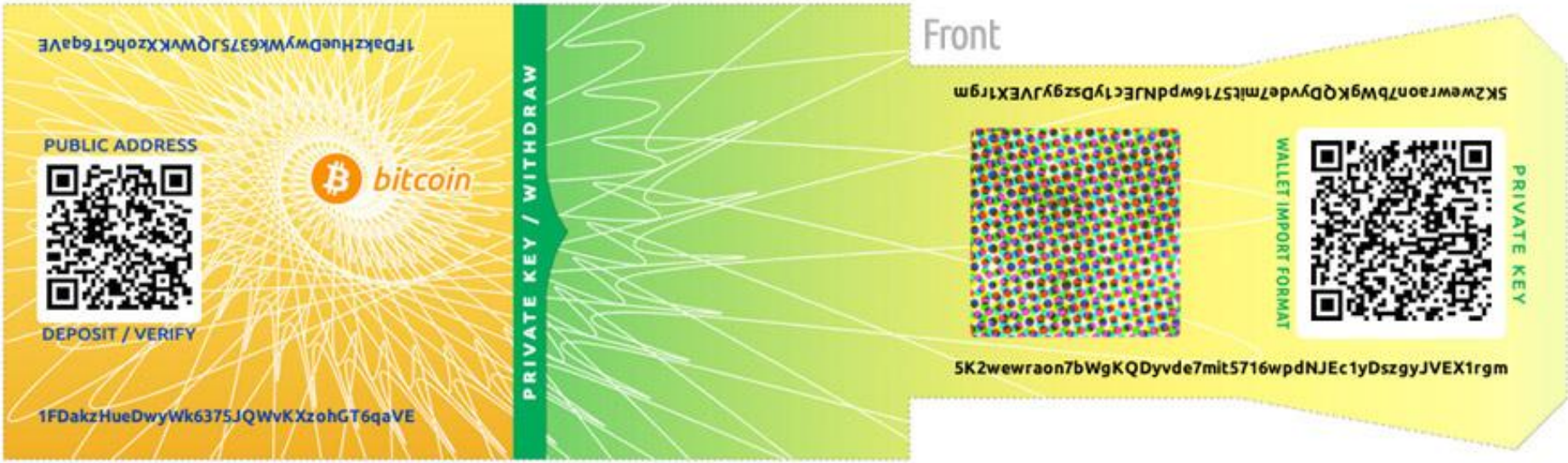
More (100) »

https://bitnodes.earn.com/



Introduction

Paper Wallet



Introduction

Ethereum Project

2013-2014

Crowdfunded **~$20M** in ~ a month Popularized a grand vision of “generalized” cryptocurrency

Applications

Tokens, Lotteries, Cryptocurrency exchanges, Marketplaces, Insurance / hedging, Supply-chain management, Self-sovereign identity, management ,Sharing economy.

https://github.com/ethereum/wiki/wiki/White-Paper#applications

Introduction

Market capital

Top 100 Cryptocurrencies By Market Capitalization

https://coinmarketcap.com/

Cryptocurrencies ▾ Exchanges ▾ Watchlist USD ▾ Next 100 → View All

#	Name	Market Cap	Price	Volume (24h)	Circulating Supply	Change (24h)	Price Graph (7d)
1	Bitcoin	$107,415,185,627	$6,224.54	$3,749,378,889	17,256,737 BTC	-3.68%	
2	Ethereum	$20,548,938,212	$201.78	$1,356,262,018	101,835,998 ETH	-7.52%	
3	XRP	$11,039,153,834	$0.278414	$156,464,580	39,650,153,121 XRP *	-5.63%	
4	Bitcoin Cash	$8,230,879,363	$474.74	$307,474,646	17,337,600 BCH	-4.81%	
5	EOS	$4,303,759,682	$4.75	$504,385,769	906,245,118 EOS *	-6.54%	
6	Stellar	$3,678,902,650	$0.195861	$51,045,700	18,783,271,641 XLM *	-5.70%	

Introduction

What is Blockchain?

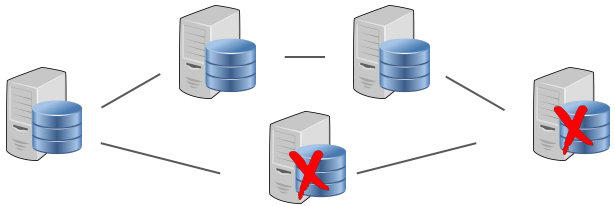
- Irreversible
- Immutable chained blocks
- Each Block holds set of trx
- Byzantine Fault Tolerance

Introduction

Types of Blockchain

Permissioned Blockchains

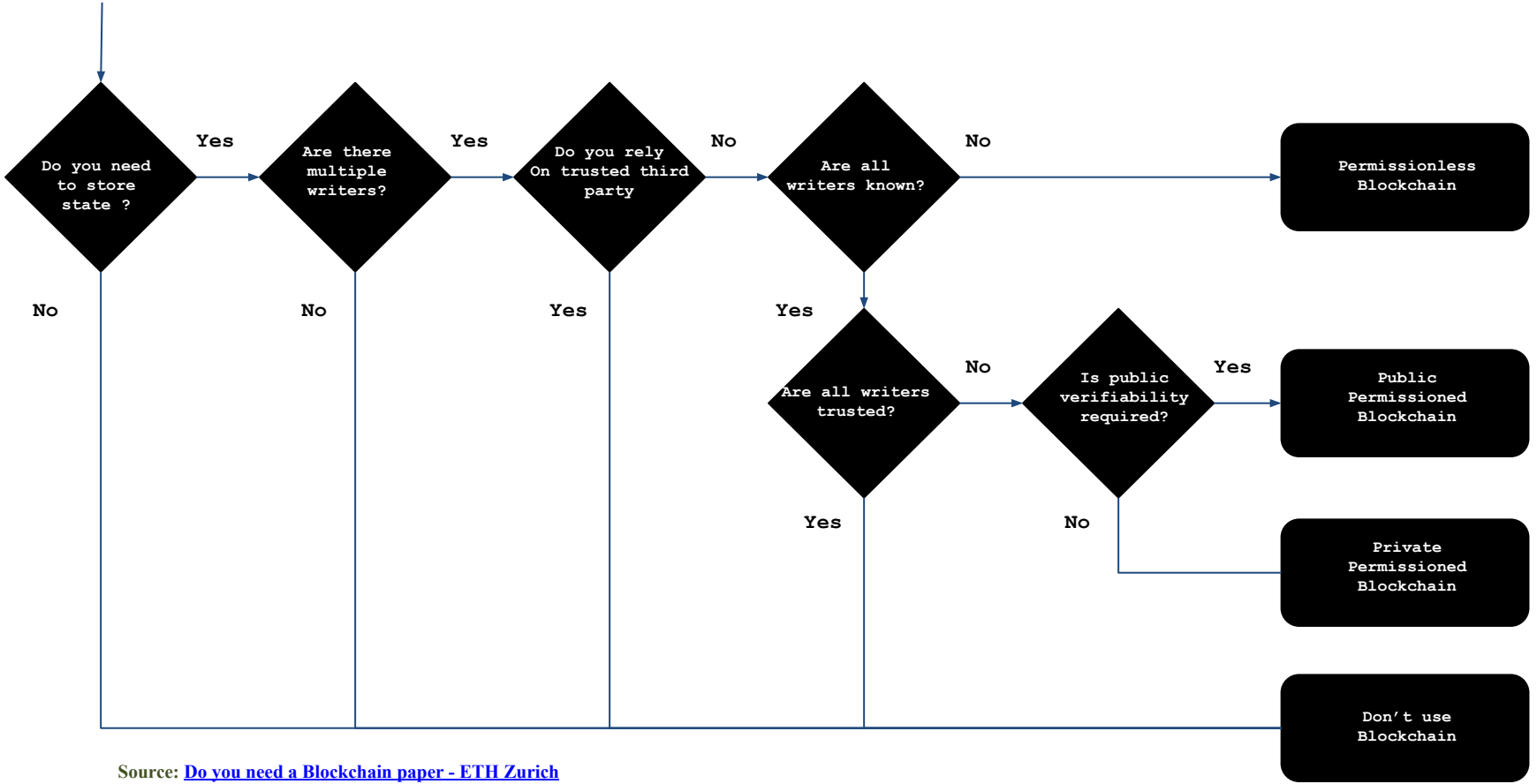
AKA "Consortium blockchain"
Nodes are run by well-known,
mostly trusted entities



Public Blockchains

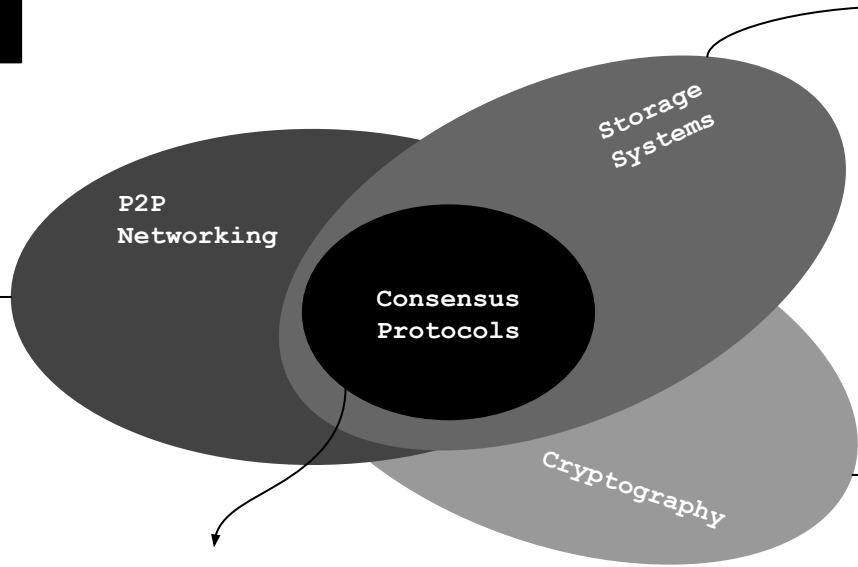
Open for participation by anyone

Introduction to Blockchain Technology



Source: [Do you need a Blockchain paper - ETH Zurich]

Core Technologies



- BigchainDB
- Filesystem
- NomsDB
- RethinkDB
- CouchDB

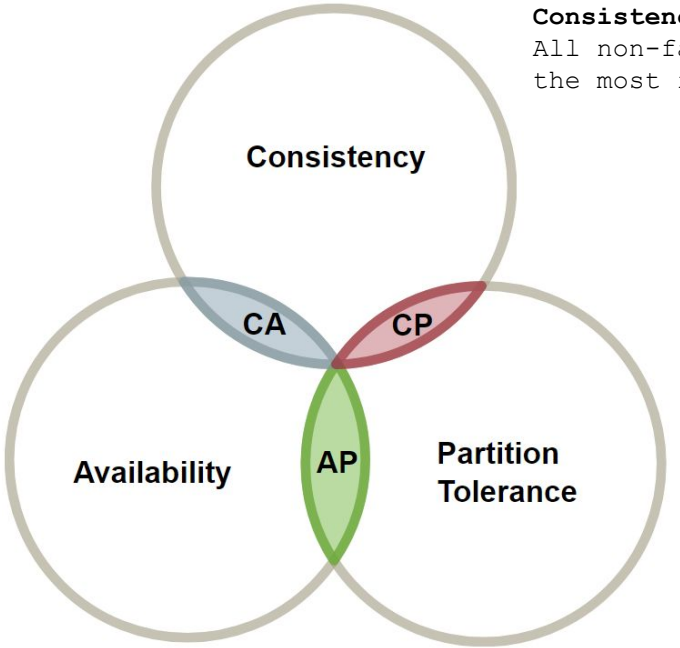
- Gossip protocols
- Bitcore p2p
- Ethereum p2p

- Practical Byzantine Fault Tolerant
- Proof of Work (PoW)
- Proof of Elapsed Time (PoET)
- Proof of Stake (PoS)
- Proof of Authority (PoA)
- Federated Byzantine Agreement
- HoneyBadger BFT
- Thunderla BFT

- ECDSA
- Hashing (SHA3)
- Multi-signature
- Functional Encryption
- Message Integrity
- TLS/SSL and PKI
- Zero-Knowledge Proofs
- Advanced Encryption Standard
- Homomorphic Encryption
- Key derivation function

Consensus Protocols

CAP Theorem



Consistency
All non-faulty nodes provides the most recent state

Availability
Every node has consistent read and write access

Partition Tolerance
The system works despite partitions in the network

Limitation
Any distributed system can only have two of three

Source: http://berb.github.io/diploma-thesis/original/resources/cap.svg

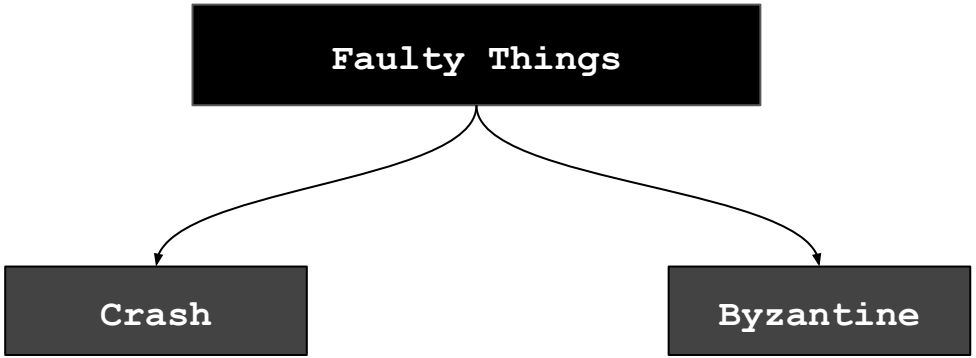
Consensus Protocols

Properties

Safety	Liveness	Fault Tolerant
All faulty nodes produce the same output where state is the same on all nodes.	All Non faulty nodes participating in the consensus eventually produce value (don't stuck)	The network can be neutralized any deciding message during failure

Consensus Protocols

Fault Tolerance



Google
amazon
facebook

- Network Failure
- Power Failure
- Hard drive Failure

- Sybil Attacks
- 51% Attack
- Eclipse Attacks
- Nothing at Stake

AIRBUS
bitcoin
ethereum

Consensus Protocols

Fault Tolerance

Source:
http://container-solutions.com/raft-explained-part-1-the-consenus-problem/

Crash Failure		Usage Patterns								
Project	**Consensus System**	**SR**	**LR**	**SS**	**BO**	**SD**	**GM**	**LE**	**MM**	**Q**
GFS	Chubby			✓				✓	✓	
Borg	Chubby/Paxos	✓				✓		✓		
Kubernetes	etcd						✓		✓	
Megastore	Paxos		✓							
Spanner	Paxos	✓								
Bigtable	Chubby						✓	✓	✓	
Hadoop/HDFS	ZooKeeper	✓						✓		
HBase	ZooKeeper	✓		✓			✓		✓	
Hive	ZooKeeper			✓					✓	
Configerator	Zeus								✓	
Cassandra	ZooKeeper					✓		✓	✓	
Accumulo	ZooKeeper		✓	✓					✓	
BookKeeper	ZooKeeper						✓		✓	
Hedwig	ZooKeeper						✓		✓	
Kafka	ZooKeeper						✓	✓	✓	
Solr	ZooKeeper							✓	✓	✓
Giraph	ZooKeeper		✓		✓				✓	
Hama	ZooKeeper				✓					
Mesos	ZooKeeper							✓		
CoreOS	etcd					✓				
OpenStack	ZooKeeper					✓				
Neo4j	ZooKeeper			✓				✓		

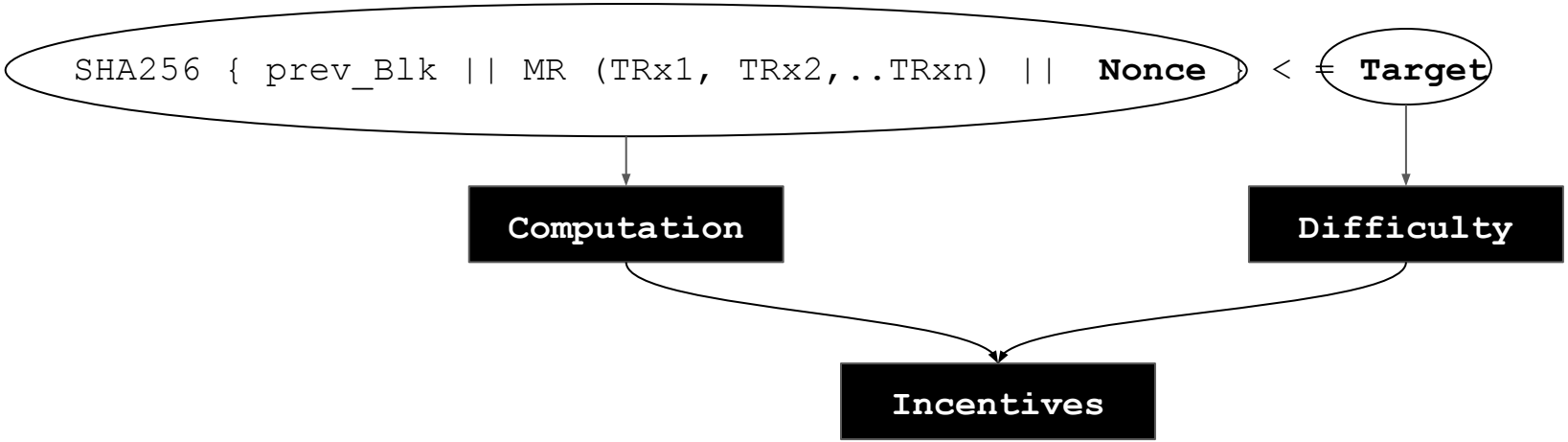
Consensus Protocols

Fault Tolerance

Byzantine Fault Tolerance

Consensus Protocols

Proof of Work (PoW)



Consensus Protocols

Proof of Work (PoW)

Bitcoin Genesis Block

Consensus Protocols

Proof of Work (PoW)

Hash Power rate and Difficulty in Bitcoin network

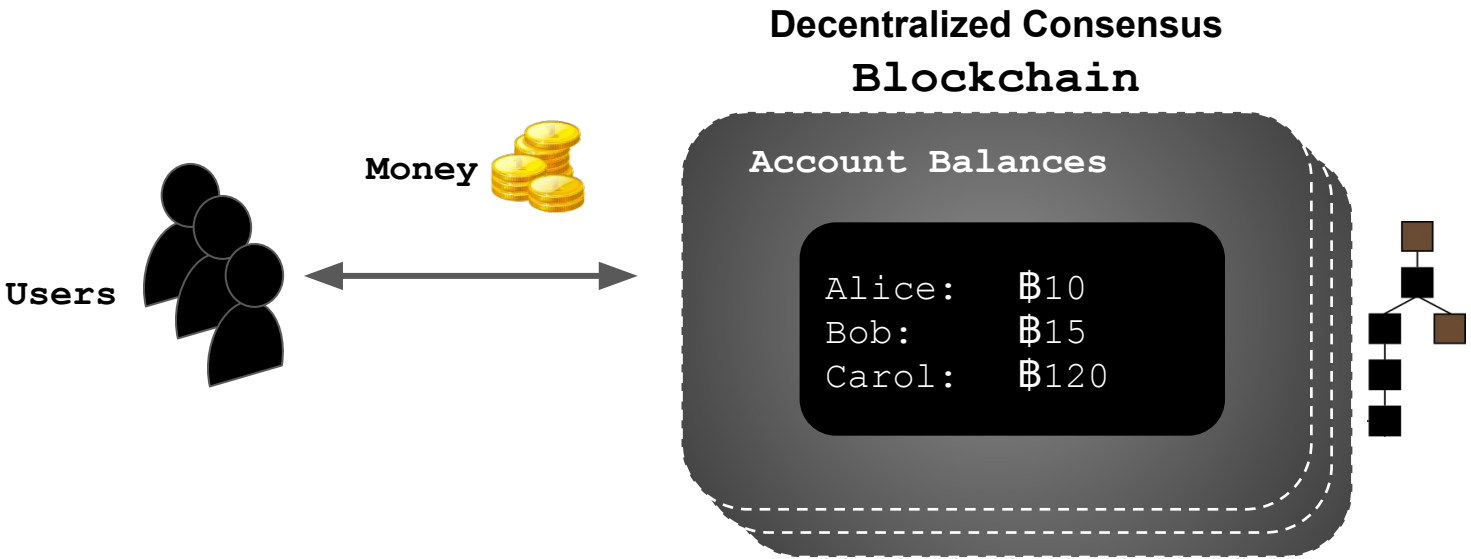
Blockchain For Data-science

- Introduction to Blockchain Mechanics
  - Introduction
  - Core Technologies
  - Consensus Protocols
- Introduction to Smart Contracts
  - What is Smart Contract
  - Solidity Essentials
  - Lab: Tutorial
- Token engineering
  - Introduction to TE
  - Towards practice of TE
- Ocean Protocol
  - Tokenize assets with Ocean
  - Ocean Roadmap
  - Ocean Protocol test network (Plankton)
- Open Discussion

What is Smart Contract

Digital Currency

Digital currency: is just one application on top of a blockchain



What is Smart Contract

Nick Szabo in 1994

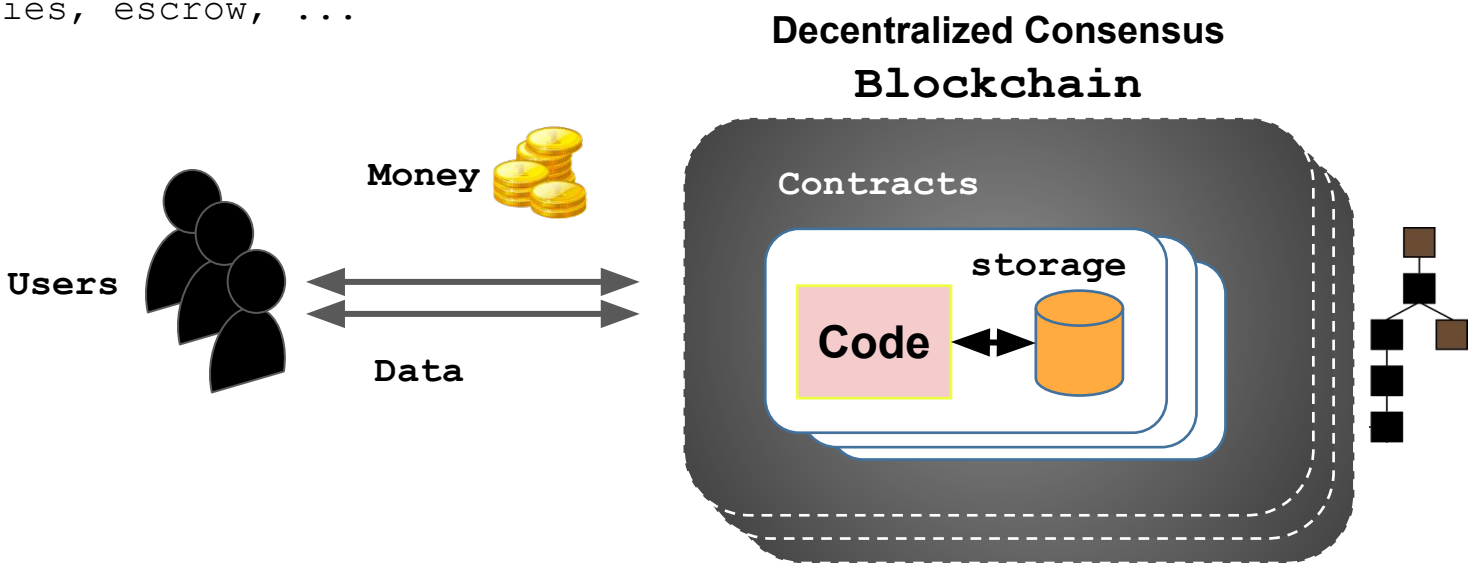
Smart Contracts: is a computerized transaction protocol that executes the terms of a contract. The general objectives are to satisfy common contractual conditions (such as payment terms, liens, confidentiality, and even enforcement), minimize exceptions both malicious and accidental, and minimize the need for trusted intermediaries. Related economic goals include lowering fraud loss, arbitrations and enforcement costs, and other transaction costs.

-Nick Szabo “The Idea of Smart Contracts”

What is Smart Contract

Ethereum

Smart Contracts: user-defined programs running on top of a blockchain where you can define more advanced logic such as auctions, elections, lotteries, escrow, ...



What is Smart Contract

Example

Alice will reveal to Bob a value *x* such that SHA-256(*x*) = 0x2a...

In exchange, Bob will give Alice $10 in cash.

If Alice does not give Bob by July 1, 2018, then she will pay a penalty of US$1 per day that she is late, up to US$100.

Signed: Alice Bob

What is Smart Contract

Traditional vs Smart

	Traditional Contract	Smart Contract
specification	Natural language + “legalese”	Code
identity & consent	Signatures	Digital signatures
dispute resolution	Judges, arbitrators	Proofs
nullification	By judges	Revocation
payment	Carried out by parties separately	built-in
escrow	Trusted third party, settled in $	built-in

What is Smart Contract

Key challenges

Smart Contract provides

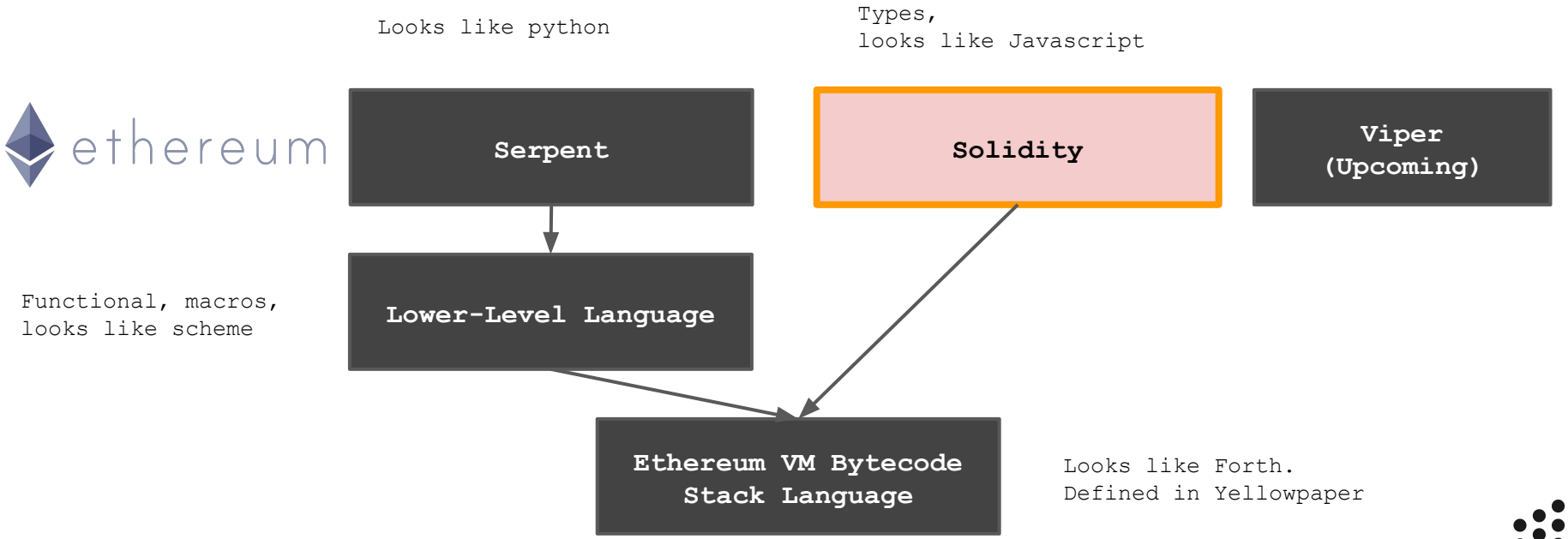
- Correctness
- Availability

But also it has some problems such as

- Privacy
- Expensive
- Scalability issue
- Uncertain delays
- Code is law
- Front Running “tailgating”

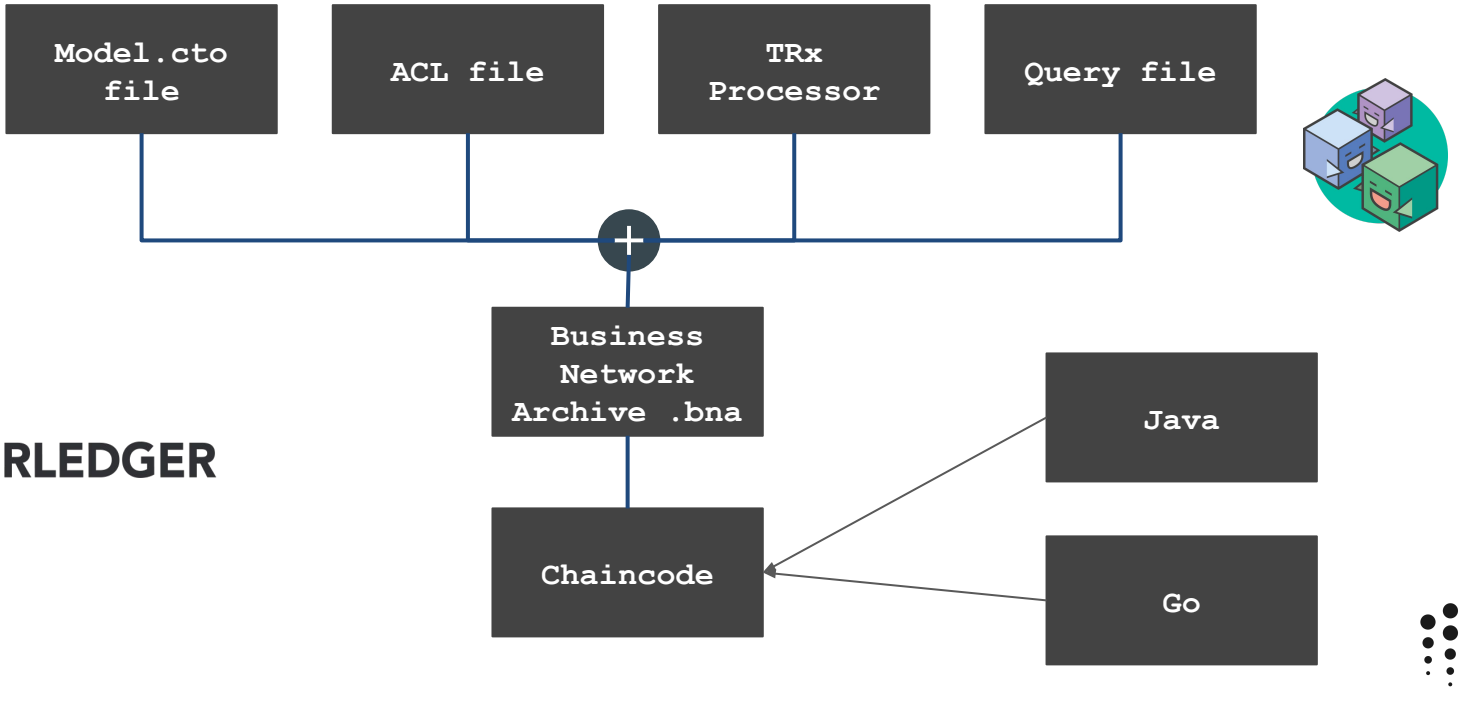
What is Smart Contract

Programming Languages



What is Smart Contract

Programming Languages



Solidity Essentials

Syntax

	Description
Syntax	looks like JavaScript
Contract Structure	follows classes/objects scenario, events, modifiers, struct, enum, state variables, and functions. For moredetails
Static typing	bool, uint8, uint16, uint256, int8, int256 address, string, byte[], mapping(keyType ==> valueType)
Storage and stateful methods	public instance variables, public methods constant, pure,view methods, more details functions.
Control flow	for loops and if statements, documentation

Solidity Essentials

State Variables

https://solidity.readthedocs.io/en/v0.4.21/structure-of-a-contract.html#state-variables

State variables are values
which are permanently
stored in contract storage

```
pragma solidity ^0.4.0;

contract SimpleStorage {
    uint storedData; // State variable
    // ...
}
```

Solidity Essentials

Constructor

https://solidity.readthedocs.io/en/v0.4.21/contracts.html#creating-contracts

Constructor is an optional function with the same name as the contract. Only one constructor is allowed, and this means overloading is not supported.

Solidity Essentials

Functions

https://solidity.readthedocs.io/en/v0.4.21/structure-of-a-contract.html#functions

Functions are the executable units of code within a contract. Functions must be marked **payable to accept funds**

```
pragma solidity ^0.4.0;

contract SimpleAuction {
    function bid() public payable { // Function
        // ...
    }
}
```

Solidity Essentials

Visibility & Getters

https://solidity.readthedocs.io/en/v0.4.24/contracts.html#visibility-and-getters

`external`:
External functions are part of the contract interface, which means they can be called from other contracts and via transactions. An external function `f` cannot be called internally (i.e. `f()` does not work, but `this.f()` works). External functions are sometimes more efficient when they receive large arrays of data.

`public`:
Public functions are part of the contract interface and can be either called internally or via messages. For public state variables, an automatic getter function (see below) is generated.

`internal`:
Those functions and state variables can only be accessed internally (i.e. from within the current contract or contracts deriving from it), without using `this`.

`private`:
Private functions and state variables are only visible for the contract they are defined in and not in derived contracts.

Solidity Essentials

Visibility & Getters

https://solidity.readthedocs.io/en/v0.4.24/contracts.html#visibility-and-getters

The compiler automatically creates getter functions for all **public** state variables.

```
pragma solidity ^0.4.0;

contract C {
    uint public data = 42;
}

contract Caller {
    C c = new C();
    function f() public {
        uint local = c.data();
    }
}
```

Solidity Essentials

Modifiers

https://solidity.readthedocs.io/en/v0.4.24/contracts.html#function-modifiers

Modifiers can be used to easily change the behaviour of functions.

```
modifier onlyOwner {
    require(
        msg.sender == owner,
        "Only owner can call this function."
    );
    _;
}
```

```
function changePrice(uint _price) public onlyOwner {
    price = _price;
}
```

Solidity Essentials

Events

Events allow the convenient usage of the EVM logging facilities, which in turn can be used to "call" JavaScript callbacks in the user interface of a dapp, which listen for these events.

https://solidity.readthedocs.io/en/v0.4.24/contracts.html#events

```
pragma solidity ^0.4.0;

contract ClientReceipt {
    event Deposit(
        address indexed _from,
        bytes32 indexed _id,
        uint _value
    );

    function deposit(bytes32 _id) public payable {
        // Events are emitted using `emit`, followed by
        // the name of the event and the arguments
        // (if any) in parentheses. Any such invocation
        // (even deeply nested) can be detected from
        // the JavaScript API by filtering for `Deposit`.
        emit Deposit(msg.sender, _id, msg.value);
    }
}
```

Solidity Essentials

Events

https://solidity.readthedocs.io/en/v0.4.24/contracts.html#events

Low-Level Interface to Logs: It is also possible to access the low-level interface to the logging mechanism via the functions log0, log1, log2, log3 and log4. logi takes i + 1 parameter of type bytes32.

```
pragma solidity ^0.4.10;

contract C {
    function f() public payable {
        bytes32 _id = 0x420042;
        log3(
            bytes32(msg.value),
            bytes32(0x50cb9fe53daa9737b786ab3646f04d0150dc50ef4e75f59509d83667ad5adb20),
            bytes32(msg.sender),
            _id
        );
    }
}
```

Solidity Essentials

Struct & Enum

Structs are custom defined types that can group several variables

https://solidity.readthedocs.io/en/v0.4.21/types.html#structs

```
pragma solidity ^0.4.0;

contract Ballot {
    struct Voter { // Struct
        uint weight;
        bool voted;
        address delegate;
        uint vote;
    }
}
```

Enums can be used to create custom types with a finite set of values

https://solidity.readthedocs.io/en/v0.4.21/types.html#enums

```
pragma solidity ^0.4.0;

contract Purchase {
    enum State { Created, Locked, Inactive } // Enum
}
```

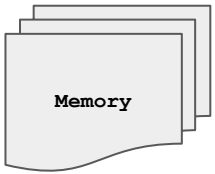
Solidity Essentials

Data Location

Persistent Storage

The default for local variables and state variables

Temporary Storage



The default for function parameters (including return parameters)

https://solidity.readthedocs.io/en/v0.4.21/types.html#data-location

Solidity Essentials

Debugging Strategies

1. Use Log Events
2. Use pure functions
3. Use the Remix debugger
4. Truffle test cases

Solidity Essentials

Ethereum Gas

Gas in Ethereum is a necessary evil

- All miners and full nodes must evaluate all transactions
  - limit **computation** cost
- All miners must store all state
  - limit **storage** use
- Short-cut the halting problem
  - There is an upper **GAS_LIMIT**, so all programs will halt

Source: ECE398SC Solidity Programming

Solidity Essentials

Ethereum Gas

Every EVM operation has a fixed gas cost

	opcodes	gas cost
Basic operations	ADD, MUL, PUSH, JUMP	2-10
Storage read	SLOAD	200
Storage write	SSTORE	5000
Storage write (from zero)	SSTORE	20000
Storage zeroize	SSTORE	-10000
Contract call	CALL, CODECALL, etc.	700
Transaction overhead	n/a	21000
Contract creation	n/a	32000
Contract destruction	SELFDESTRUCT	-19000

For more details about Opcode & gas cost

Source: ECE398SC Solidity Programming

Solidity Essentials

Ethereum Gas

All transactions specify START_GAS, GAS_PRICE

1. If START_GAS × GAS_PRICE > caller.balance, halt
2. Deduct START_GAS × GAS_PRICE from caller.balance
3. Set GAS = START_GAS
4. Run code, deducting from GAS
5. For negative values, add to GAS_REFUND
   a. GAS only decreases
6. After termination, add GAS_REFUND to caller.balance

Source: ECE398SC Solidity Programming

Solidity Essentials

Ethereum Gas

Out-of-gas exceptions are bad news

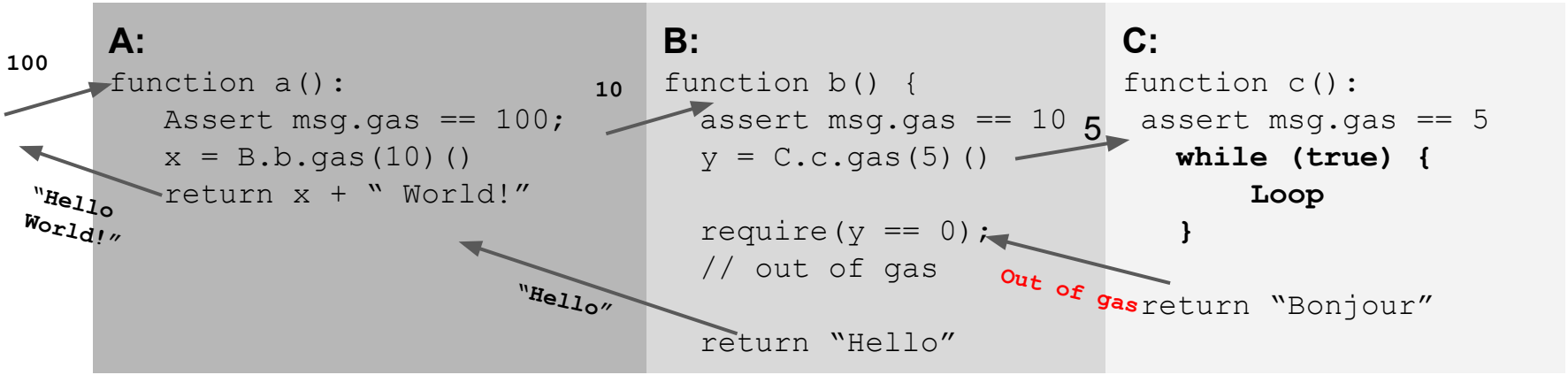
- State reverts to previous value
  - Except that START_GAS * GAS_PRICE is still deducted

Source: ECE398SC Solidity Programming

Solidity Essentials

Ethereum Gas

Callers can choose how much gas to send



(pseudocode: exact syntax used here does not work in Solidity)

Source: ECE398SC Solidity Programming

Solidity Essentials

Example "Namecoin"

Namecoin:

a simplistic **DNS** replacement

- Initially, all names are unregistered.
- Anyone can claim an unregistered name.
- Once it's registered, no one can change it.

Solidity Essentials

Example "Namecoin"

```
contract Namespace {

    struct NameEntry {
        address owner;
        bytes32 value;
    }

    uint32 constant REGISTRATION_COST = 100;
    uint32 constant UPDATE_COST = 10;
    mapping(bytes32 => NameEntry) data;

    function nameNew(bytes32 hash){
        if (msg.value >= REGISTRATION_COST){
            data[hash].owner = msg.sender;
        }
    }

    function nameUpdate(bytes32 name, bytes32 newValue, address newOwner){
        bytes32 hash = sha3(name);
        if (data[hash].owner == msg.sender && msg.value >= UPDATE_COST){
            data[hash].value = newValue;
            if (newOwner != 0){
                data[hash].owner = newOwner;
            }
        }
    }

    function nameLookup(bytes32 name){
        return data[sha3(name)];
    }

}
```

Solidity Essentials

Example MyToken

Create your own CRYPTO-CURRENCY with Ethereum

Lab1

Tutorial

Ethereum Remix IDE and Test Network Tutorial

Blockchain For Data-science

- Introduction to Blockchain Mechanics
  - Introduction
  - Core Technologies
  - Consensus Protocols
- Introduction to Smart Contracts
  - What is Smart Contract
  - Solidity Essentials
  - Lab: Tutorial
- **Token engineering**
  - Introduction to TE
  - Towards practice of TE
- Ocean Protocol
  - Tokenize assets with Ocean
  - Ocean Roadmap
  - Ocean Protocol test network (Plankton)
- Open Discussion

Introduction to TE

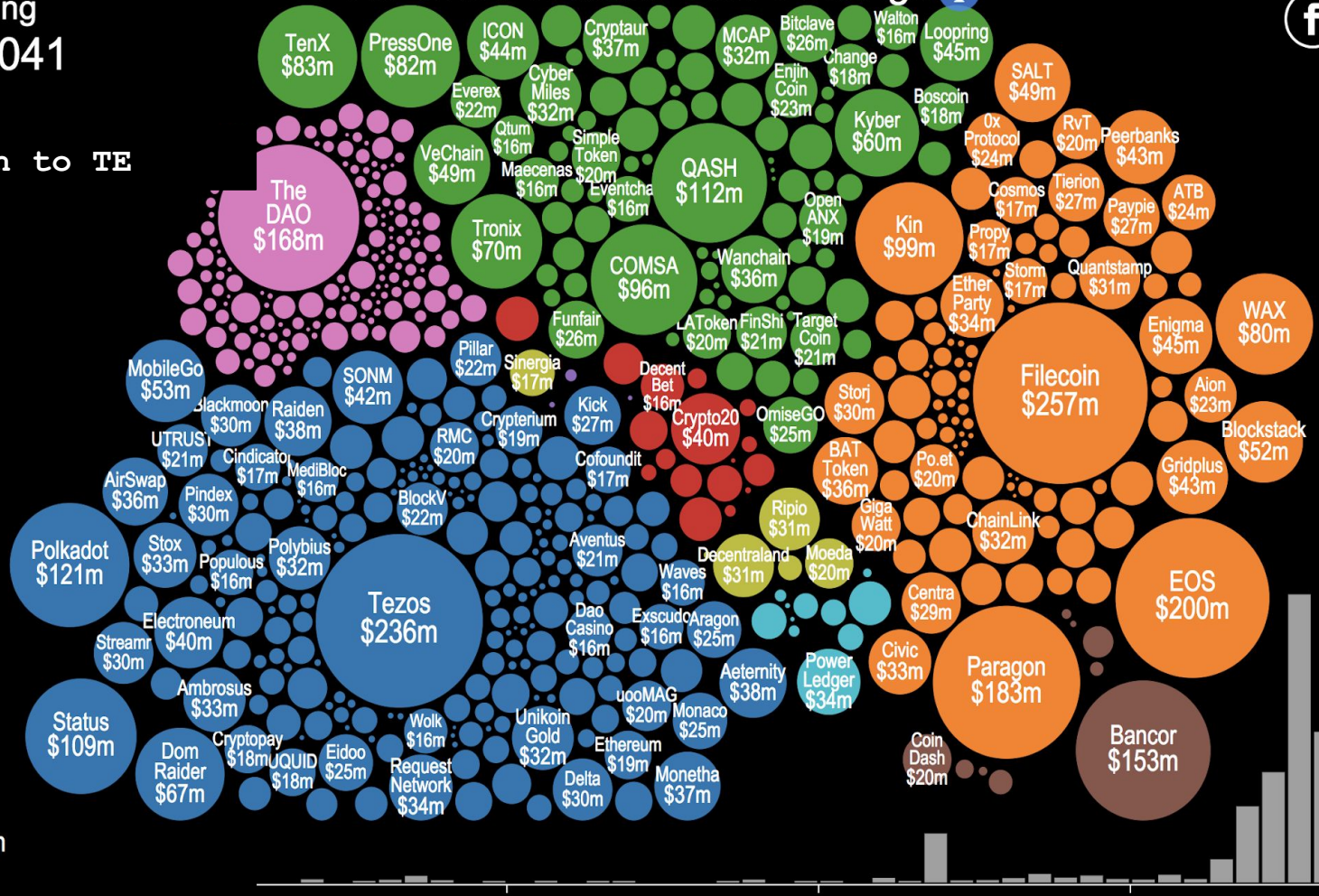
“Show me the incentive
and I will show you the outcome.”

-Charlie Munger

Four Years of Initial Coin Offerings

Total fundraising
$6,391,007,041

Introduction to TE



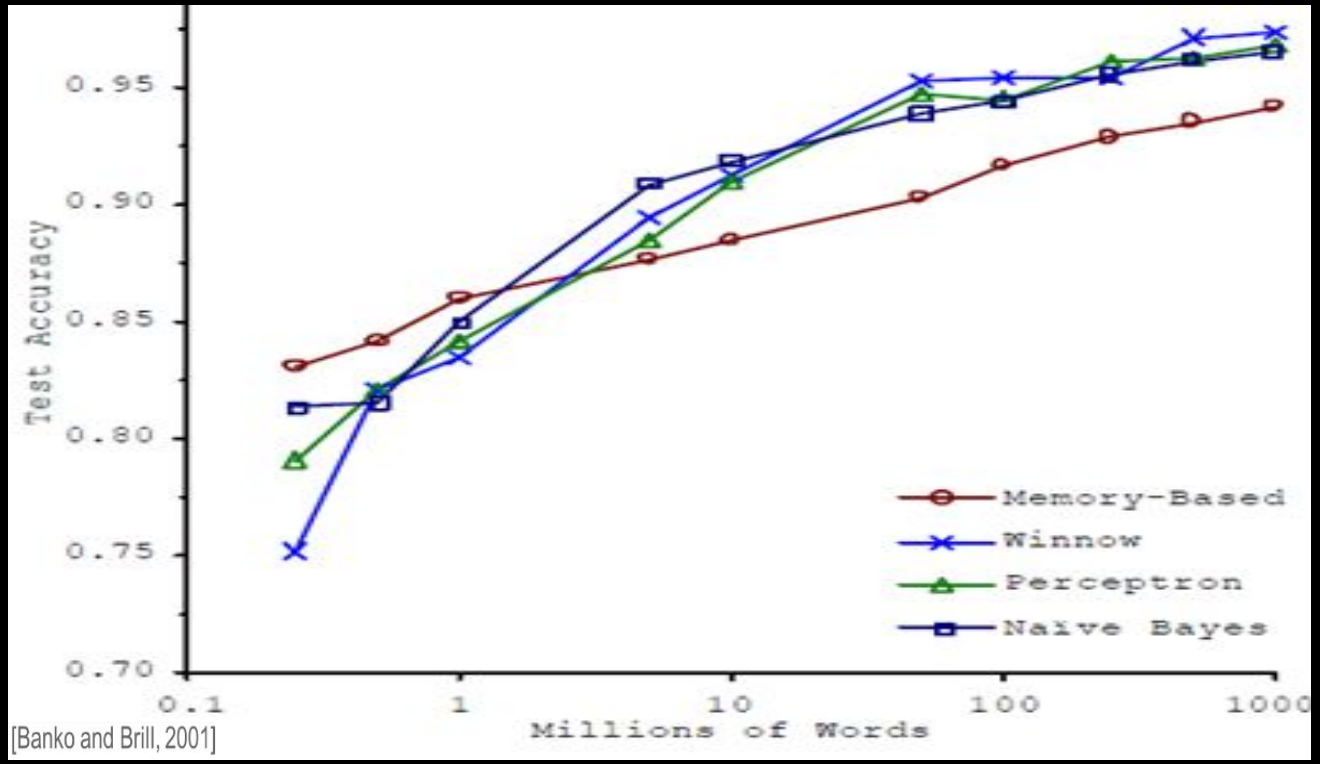
Restart

- Europe
- North America
- Asia
- Caribbean
- South America
- Oceania
- Middle East
- Africa
- Stateless/Unknown

Introduction to TE


“Incentive Machine”
Get people to do stuff
By rewarding with tokens

The Unreasonable Effectiveness of Data



1000% less error!

1000x *more* data

Silo mo' data

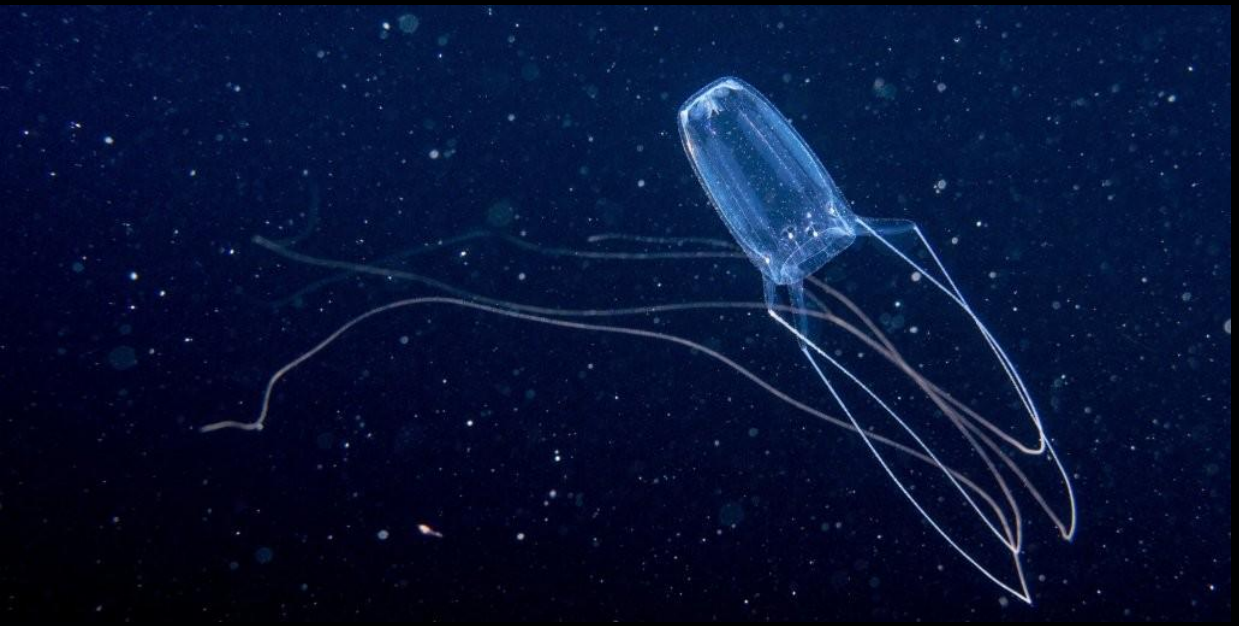
⬇

Mo' accuracy

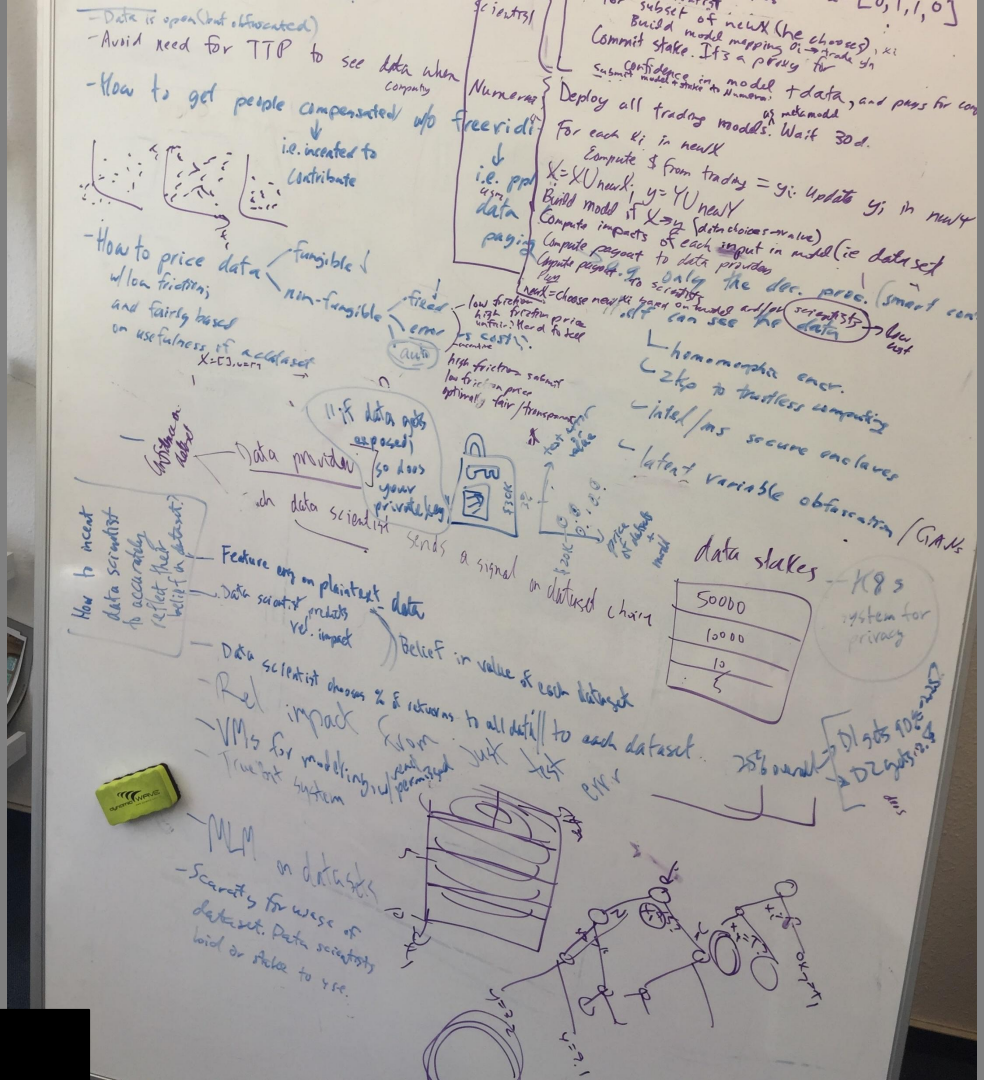
⬇

Mo' $

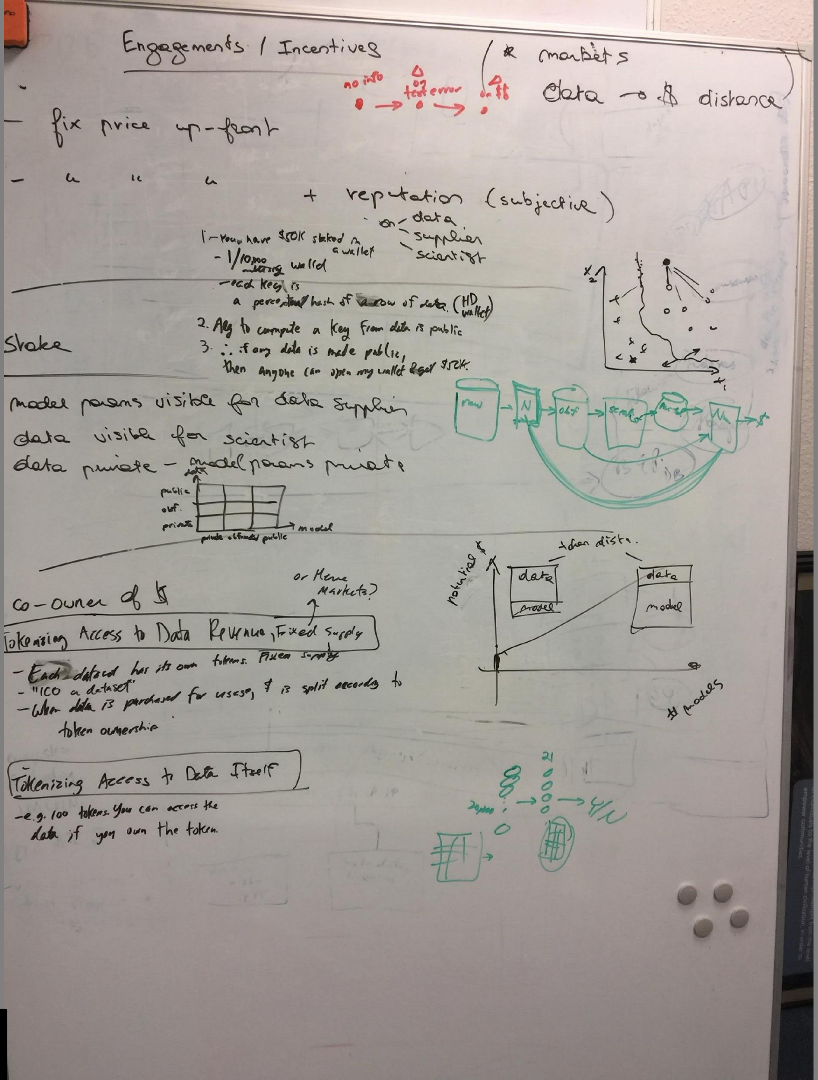
Default incentive: hoard the data



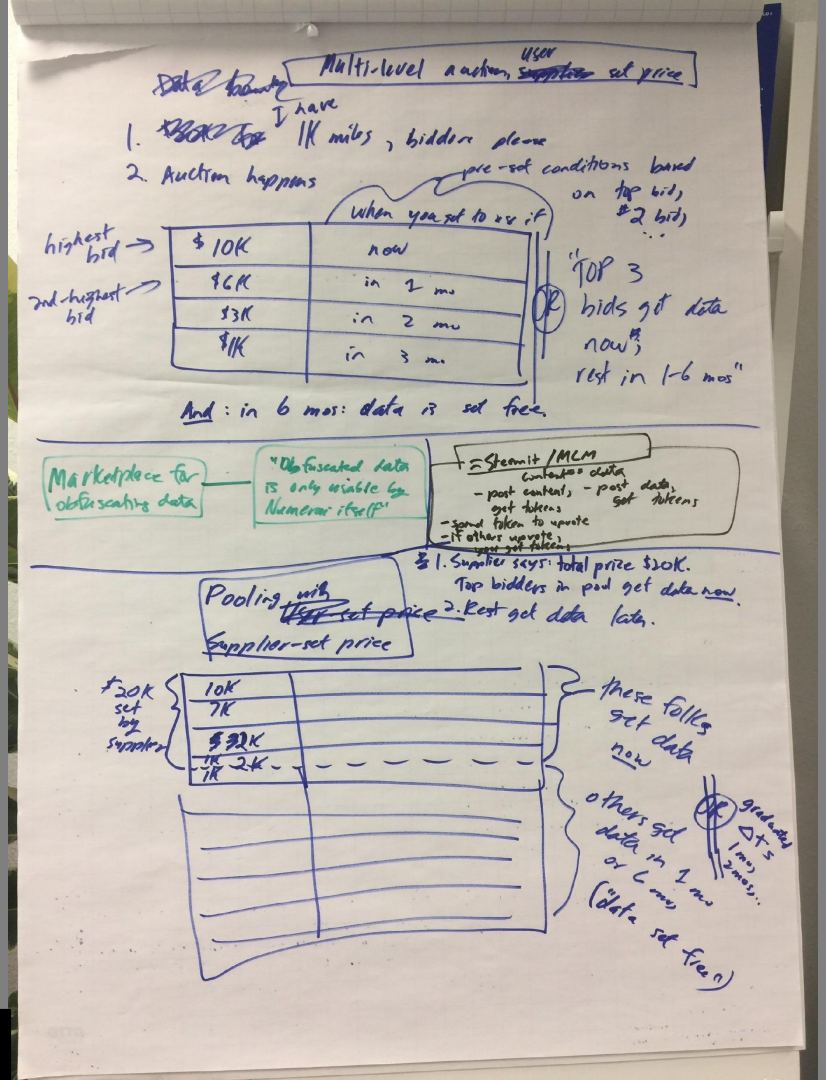
Early iterations

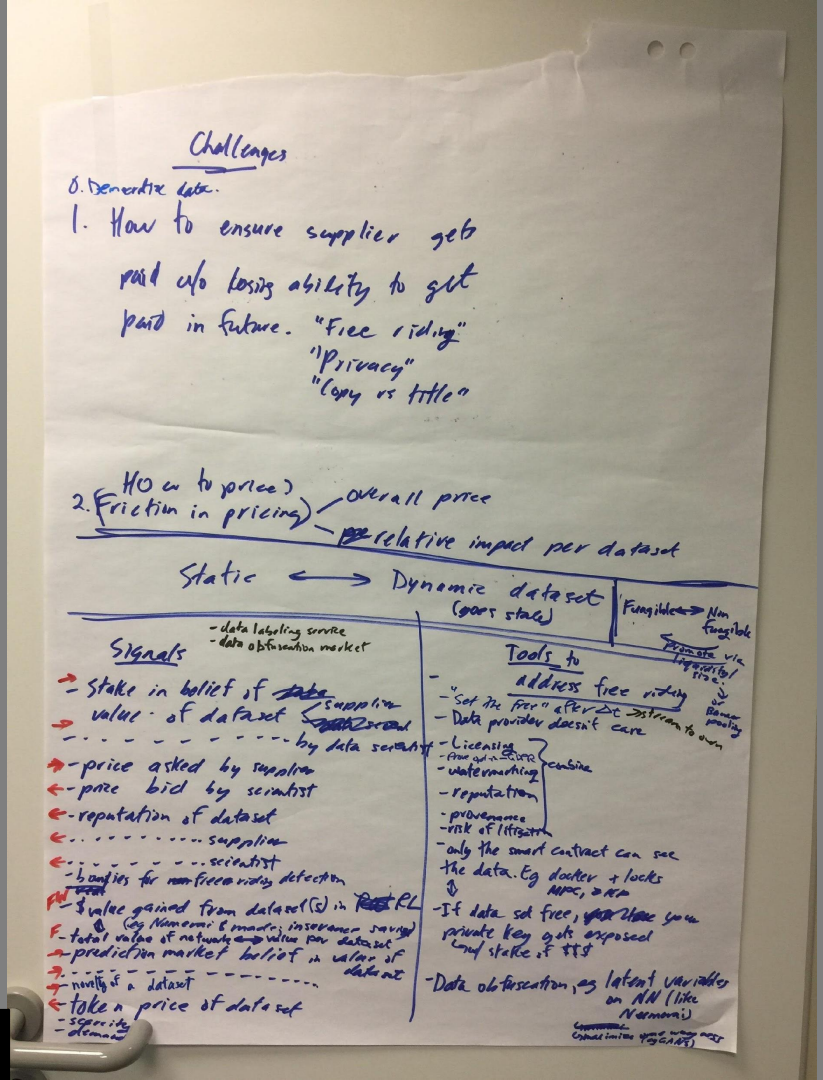


Introduction to TE

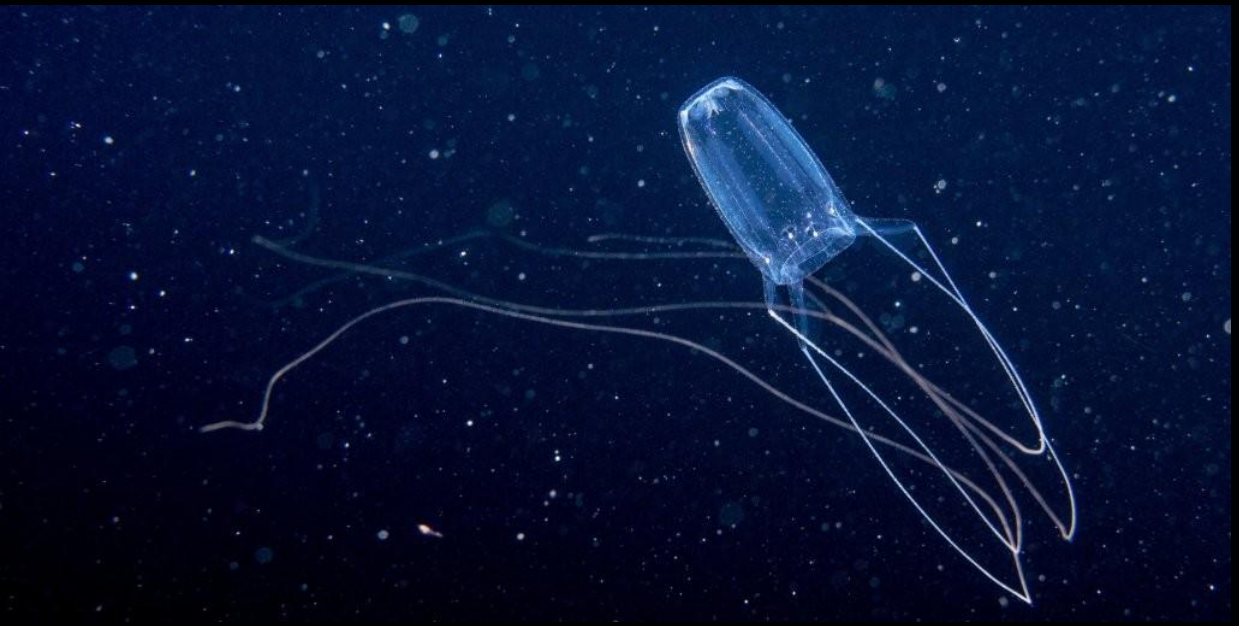


Introduction to TE


Multi-level auction, User set price
I have
1. 1K mibs, bidders please
2. Auction happens
pre-set conditions based on top bid, #2 bid, ...
When you get to vs if
highest bid
2nd-highest bid
$10K now
$6K in 1 mo
$3K in 2 mo
$1K in 3 mo
OR
"TOP 3 bids get data now, rest in 1-6 mos"
And: in 6 mos: data is set free.
Marketplace for obfuscating data
"Obfuscated data is only visible by Numerai itself"
=Stemit/MCM
Contents = data
- post contents, get tokens
- post data, set tokens
- spend token to upvote
- if others upvote, your get tokens
1. Supplier says: total price $20K.
Top bidders in pool get data now.
2. Rest get data later.
Pooling with ~~set price~~
Supplier-set price
$20K set by supplier
10K
7K
$3K
1K 2K
1K
these folks get data now
others get data in 1 mo or 6 mos
(data set free)
OR
graduated Δt's 1 mo, 2 mos, ...



Introduction to TE



Early iterations: *Flailing*

Can we structure this better?

Realizations:

1. Tokenized ecosystems are evolutionary systems.

2. Therefore design ≈ evolutionary algorithm design!

Steps in EA Design

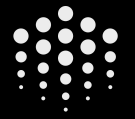
1. **Formulate the problem.** Objectives, constraints, design space.
2. **Try an existing EA solver.** If needed, try different problem formulations or solvers.
3. **Design new solver?**

1. Formulation of optimization problem
Objectives & constraints in a design space

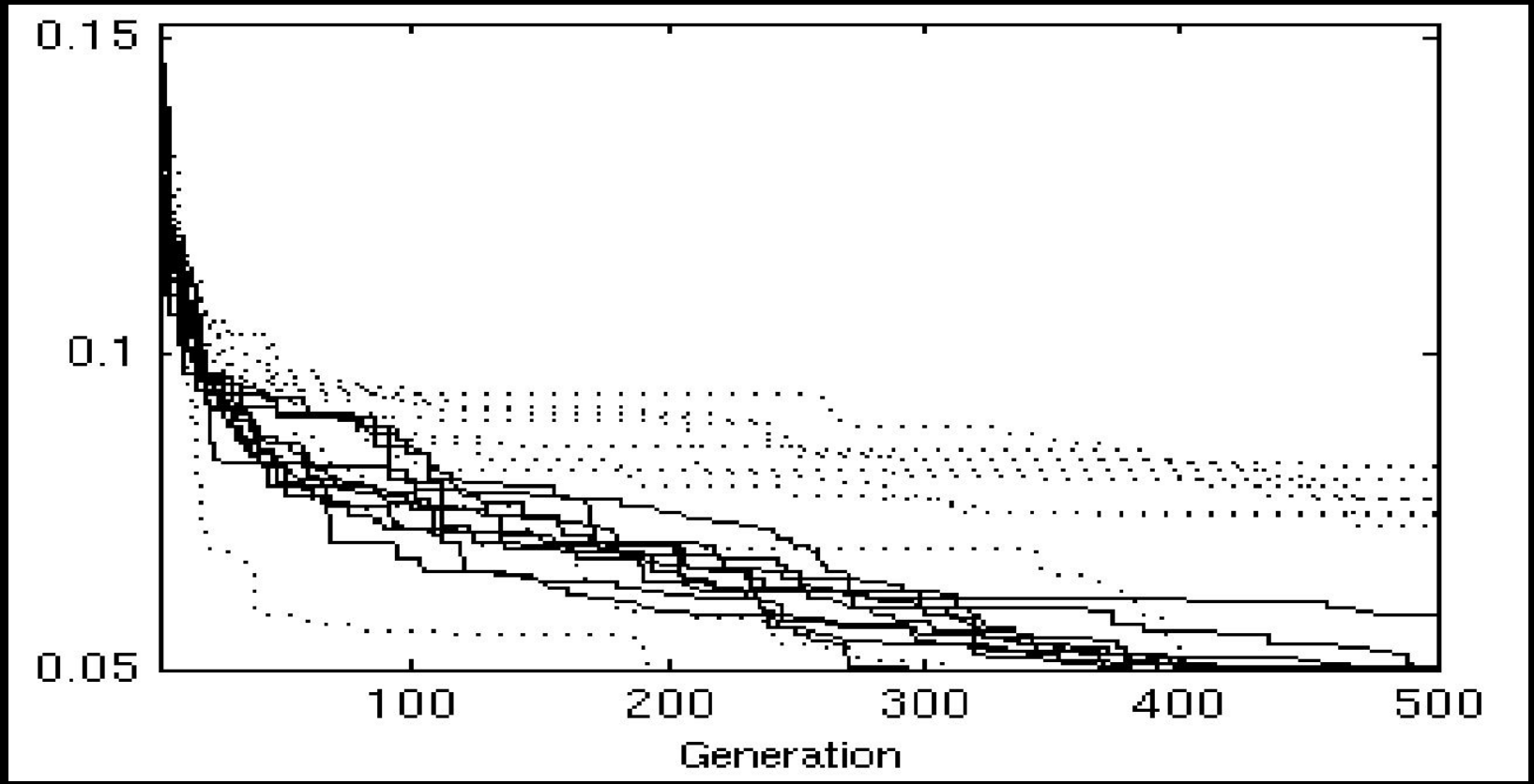
The algorithm's aim is formulated as a constrained multi-objective optimization problem

$$\begin{aligned} \text{minimize} \quad & f_i(\phi) && i = 1 \ldots N_f \\ \text{s.t.} \quad & g_j(\phi) \leq 0 && j = 1 \ldots N_g \\ & h_k(\phi) = 0 && k = 1 \ldots N_h \\ & \phi \in \Phi \end{aligned} \tag{1}$$

where Φ is the "general" space of possible topologies and sizings. The algorithm traverses Φ to return a Pareto-optimal



2. Try an existing EA solver. Does it converge?



3. Design new EA solver

e homo-
motopy
coarsely
ructural
y. Tradi-
ro path,
the zero
several

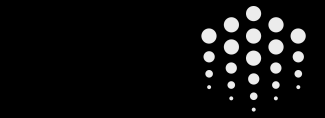
mulated
nalyses,
oint $\boldsymbol{\theta}$}.
nt/other
onnom-
corners
rated in
on (with

TABLE II
PROCEDURE SANGRIAOPTIMIZATION()

Inputs: D, N_a, K, $N_L(k)$
Outputs: $\boldsymbol{d}^*$

1. $N_{gen} = 0$; $P = \emptyset$, $P_{all} = \emptyset$
2. while stop() $\neq True$:
3. if $(N_{gen} \% N_a) = 0$:
4. if $|P| < K$:
5. $P_{|P|+1} = \emptyset$
6. P_0 = SpaceFillIndividuals($N_L(k), N_D, D$)
7. for k = 1 to $|P|$:
8. P_k = SelectParents($P_k, P_{k-1}, N_L(k)$)
9. $P_{k,j}$ = UpdateLocalOptState($P_{k,j}, k$), j = 1 to $|P_k|$
10. P_{all} = unique($P_{all} \cup P$)
11. $P_{|P|} = P_{|P|} \cup$ InnerOptimize(P_{all}, D, k)
12. $\boldsymbol{d}^*$ = $\boldsymbol{d_i}$ in P_{all} with highest Y or Cpk
13. $N_{gen} = N_{gen} + 1$
14. return $\boldsymbol{d}^*$

and all individuals encountered so far in the search, P_{all}.
Lines 2–13 are the generational loop, which repeats until stop



1. Formulate the Problem: [ex. Ocean]

Who are stakeholders? What do they want?

Objectives & constraints

Key stakeholders in Ocean ecosystem

Stakeholder	What value they can provide	What they might get in return
Data/service provider, data custodian, data owner	Data/service (market's supply)	Tokens for making available / providing service
Data/service referrers, curators. Includes exchanges and other application-layer providers.	Data/service (via a provider etc), curation	Tokens for curating
Data/service verifier. Includes resolution of linked proofs on other chains	Data/service (via a provider etc), verification	Tokens for verification
Data/service consumer	Tokens	Data/service (market's demand)
Keepers	Correctly run nodes in network	Tokens for chainkeeping

Obj:

- **Maximize supply of relevant data**

Constraints = checklist:

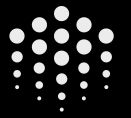
- **For priced data, is there incentive for supplying more? Referring? Spam prevention?**
- **For free data, "" ?**
- **Does the token give higher marginal value to users vs. hodlers?**
- **Are people incentivized to run keepers?**
- **Is it simple? Is onboarding low-friction?**

2. Try Existing Patterns

1. Curation
2. Proofs of human or compute work
3. Identity
4. Reputation
5. Governance / software updates
6. Third-party arbitration
7. …

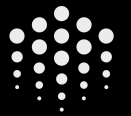
2. Try existing patterns: evaluate on objectives & constraints. [Ex Ocean: None passed…]

Key Question	1	2	3	4
For priced data: incentive for supplying more? Referring?	❌	≈	✔	≈
For priced data: good spam prevention?	≈	✔	✔	✔
For free data: incentive for supplying more? Referring?	❌	≈	❌	✔
For free data: good spam prevention?	≈	✔	≈	✔
Does token give higher marginal value to users of the network, vs external investors? Eg Does return on capital increase as stake increases?	✔	✔	✔	✔
Are people incentivized to run keepers?	≈	≈	✔	✔
It simple? Is onboarding low-friction? Where possible, do we use incentives/crypto rather than legal recourse?	✔	✔	≈	≈



3. Try new patterns: evaluate on objectives & constraints. [Ex Ocean: pass!]

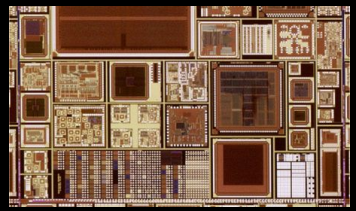
Key Question	1	2	3	4	5	6
For priced data: incentive for supplying more? Referring?	❌	≈	✔	≈	≈	✔
For priced data: good spam prevention?	≈	✔	✔	✔	✔	✔
For free data: incentive for supplying more? Referring?	❌	≈	❌	✔	✔	✔
For free data: good spam prevention?	≈	✔	≈	✔	≈	✔
Does token give higher marginal value to users of the network, vs external investors? Eg Does return on capital increase as stake increases?	✔	✔	✔	✔	✔	✔
Are people incentivized to run keepers?	≈	≈	✔	✔	✔	✔
It simple? Is onboarding low-friction? Where possible, do we use incentives/crypto rather than legal recourse?	✔	✔	≈	≈	✔	✔



On Tools

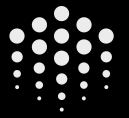
Tools

Q: How do we design circuits?
($50M+ at stake)

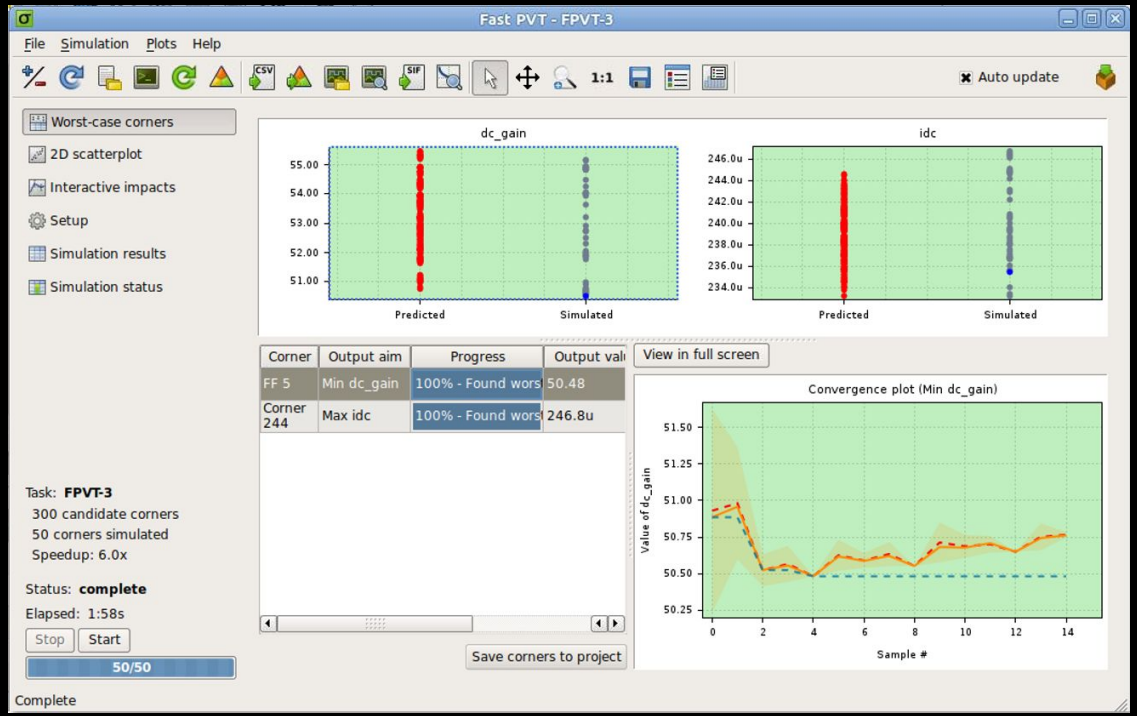


A: **CAD Tools** for

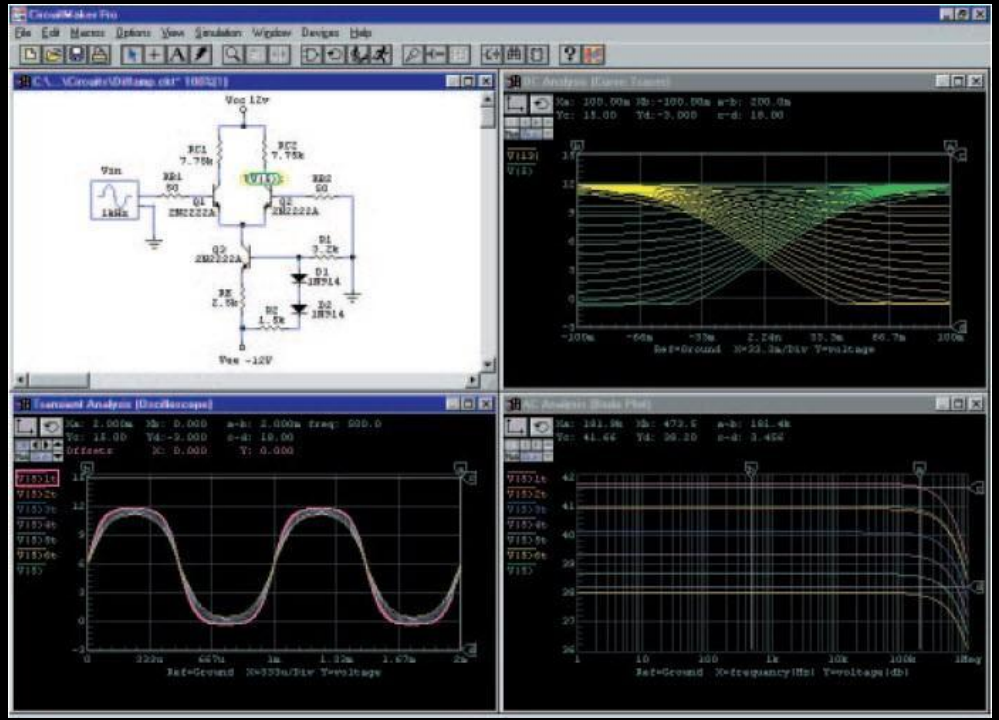
- Simulation,
- Verification, and
- Design



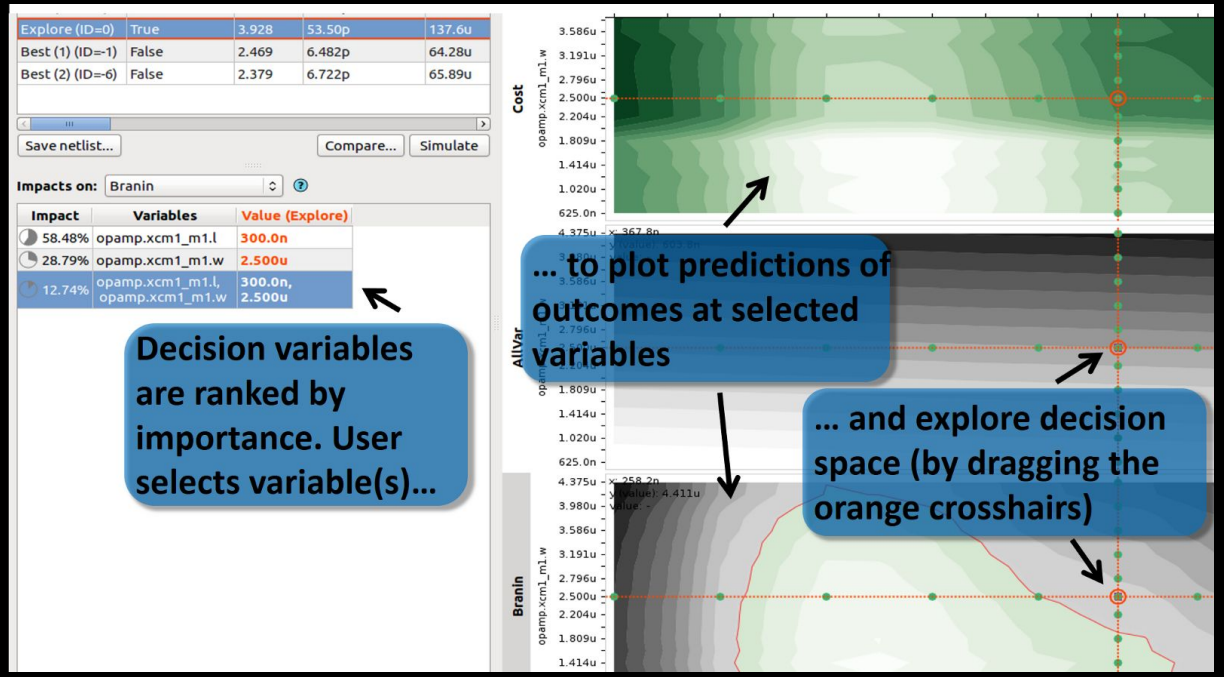
Verification across Worst-Case Conditions



Simulation of Circuit Dynamics



Interactive Design / Exploration



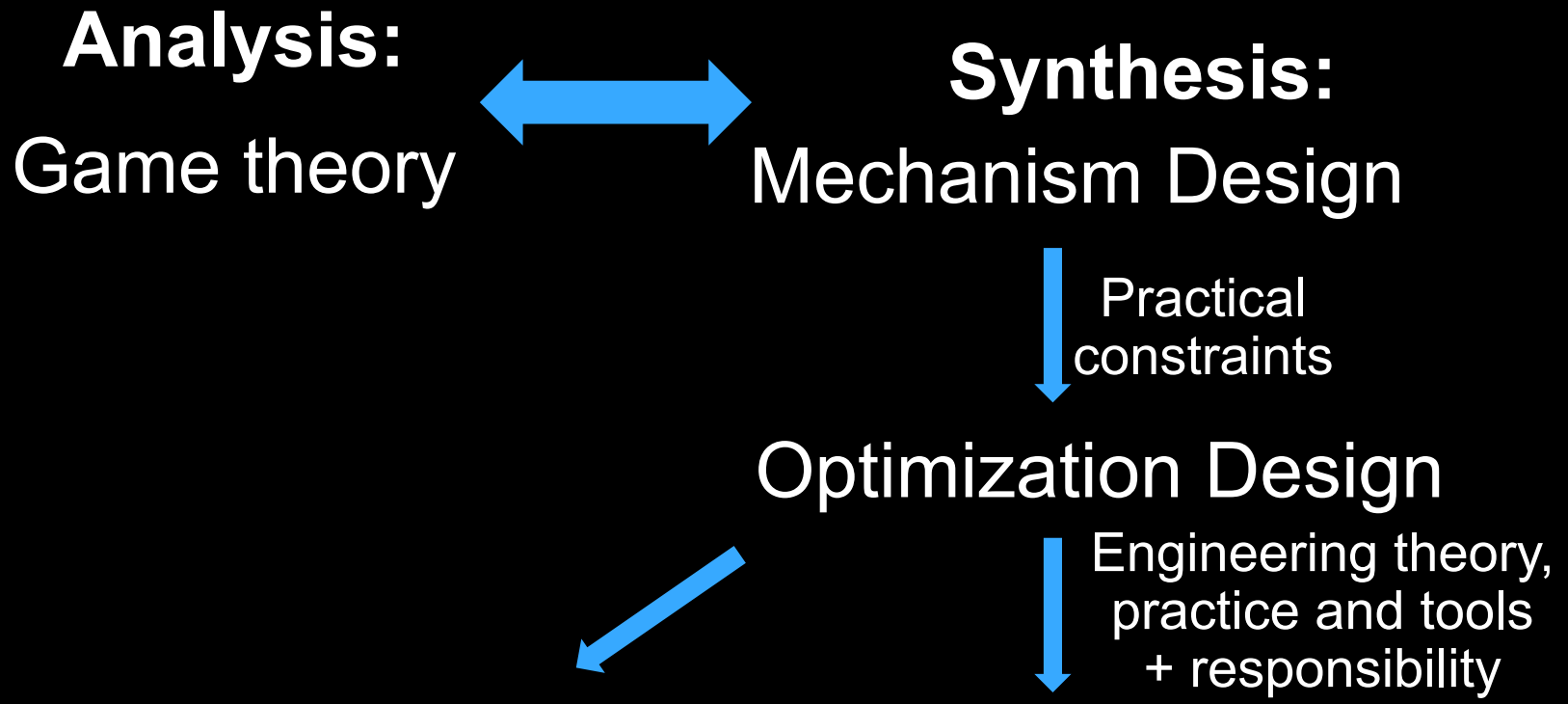
Tools for Tokenized Ecosystems?

- Simulation? _____
- Verification? _____
- Design? ___

- Are we designing tokenized ecosystems … Without tools ?
- Which means we might be getting it all wrong!

Towards Token *Engineering*

Design of Tokenized Ecosystems
From Mechanism Design to *Token Engineering*



Engineering 

is the creative application of science, mathematical methods, and empirical evidence

to the innovation, design, construction, operation and maintenance

of structures, machines, materials, devices, systems, processes, and organizations.

Engineering Responsibility

Engineering has

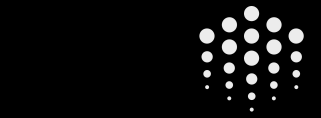
Theory,
Practice,
Tools,
Responsibility

Science ←→ Engineering

- **Engineering** is about building things that work.
- **Science** is about contributing new knowledge.
- They're complimentary.

Therefore **token engineering** is complementary to the science of cryptoeconomics / **token economics**.

Towards a Token Engineering *Community*



TE → TE Community

- A pleasant surprise to me: “Token Engineering” resonated with a *lot* of people
- Many amazing conversations.
- **A collective realization: we need to share knowledge, to learn from each other!**

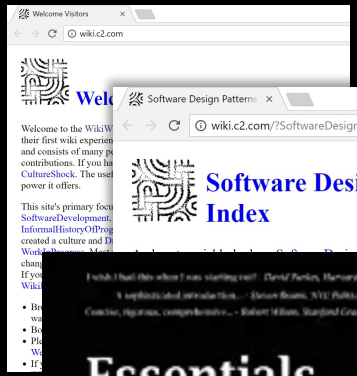
Mission of the TE Community

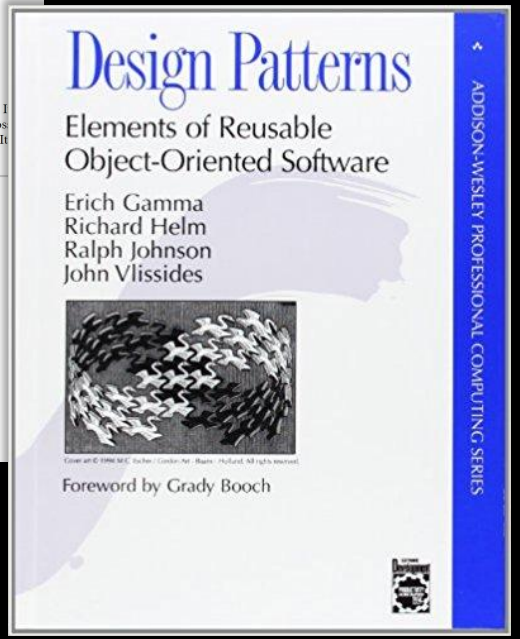
To grow TE into an **engineering discipline**

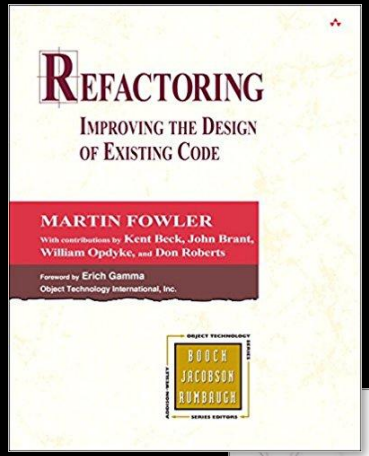
collectively as a community

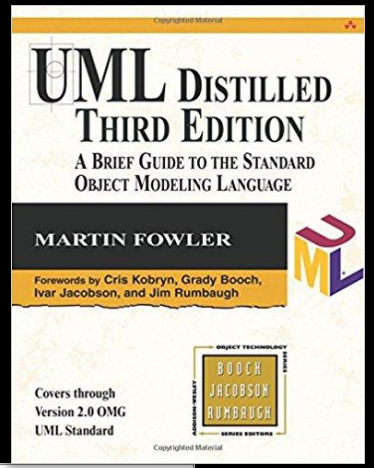
in a decentralized, permissionless, open-source fashion that all can contribute to and all can use.

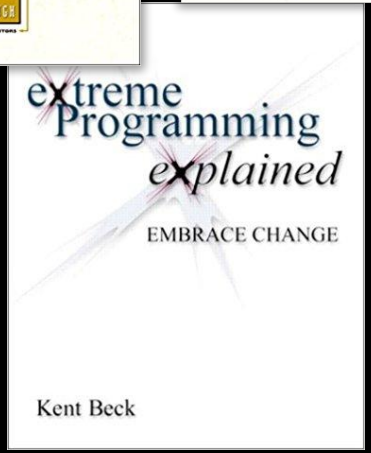
An inspiration: evolution of software engineering. C2 wiki → 💥



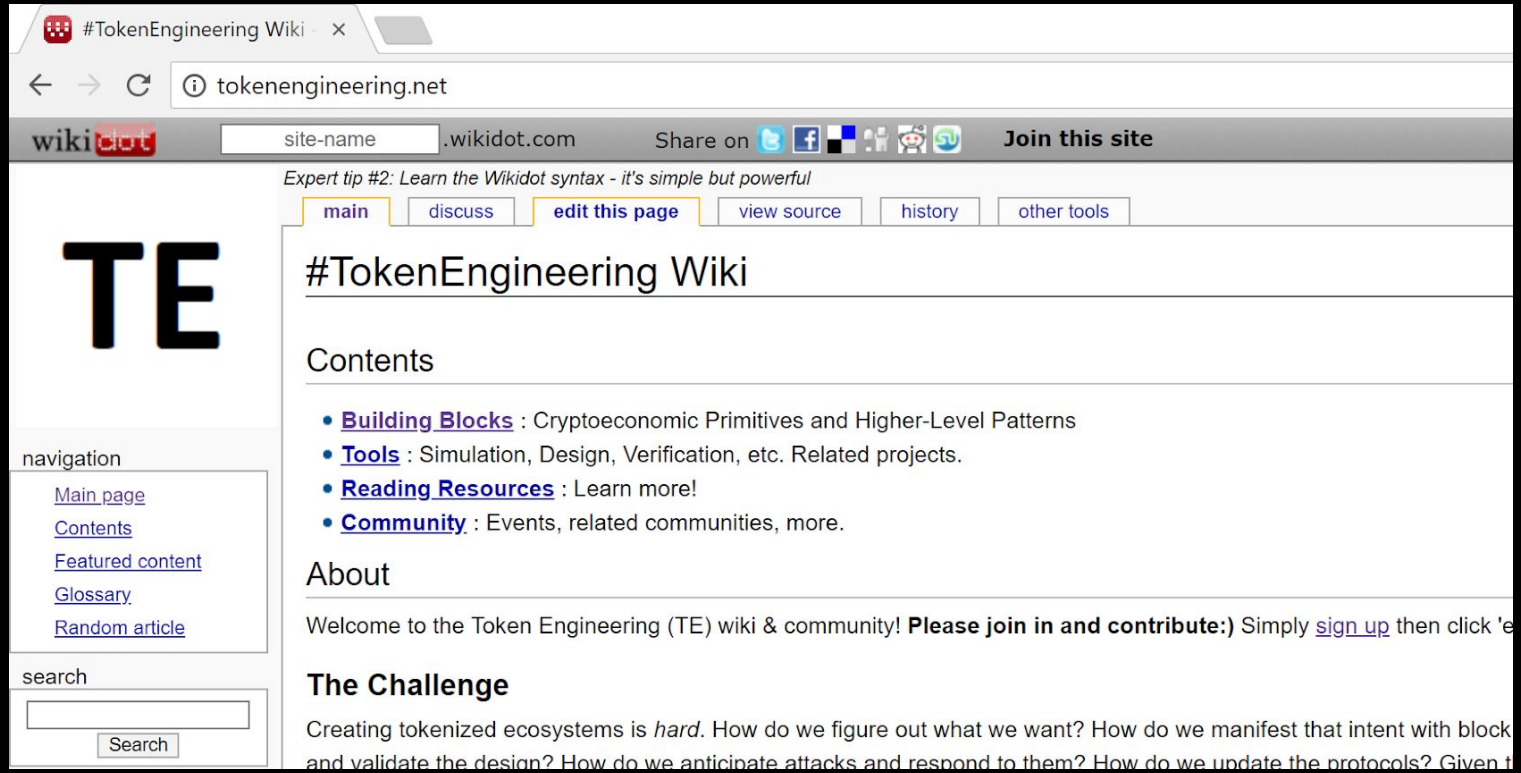


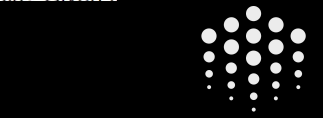


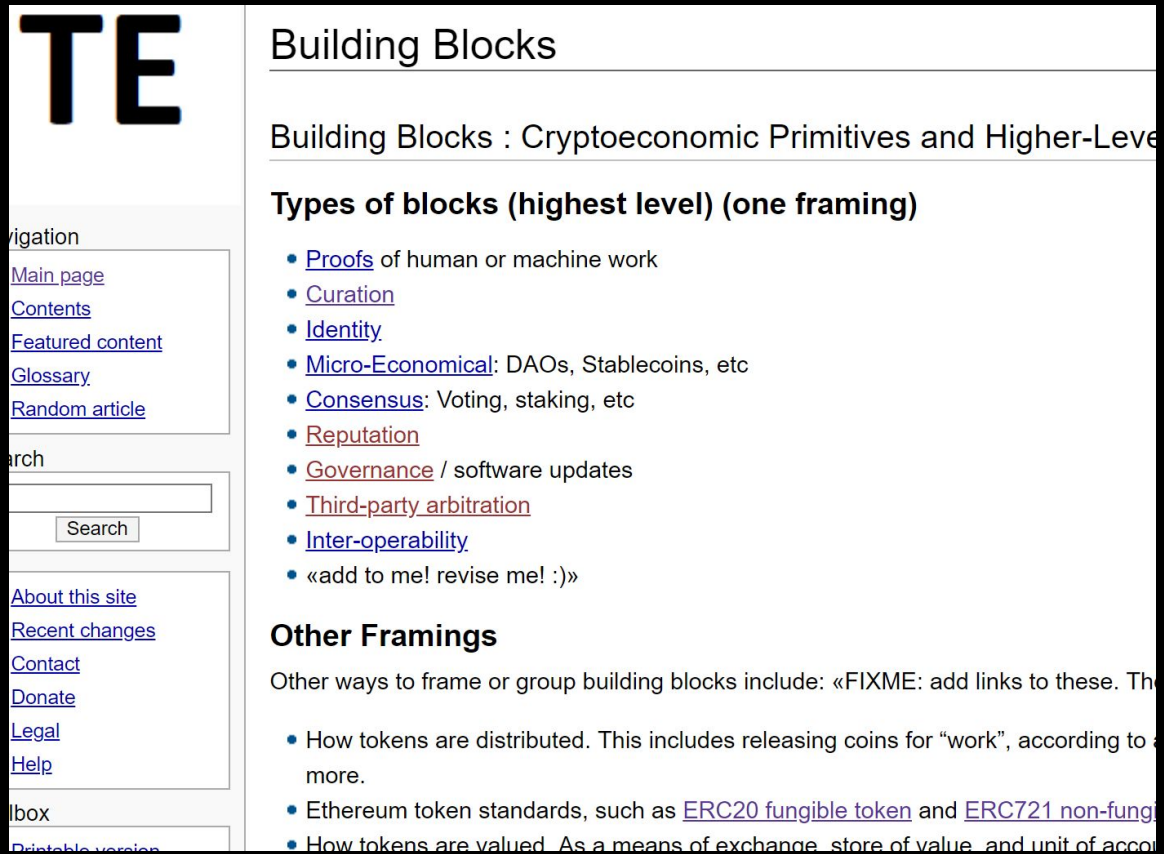


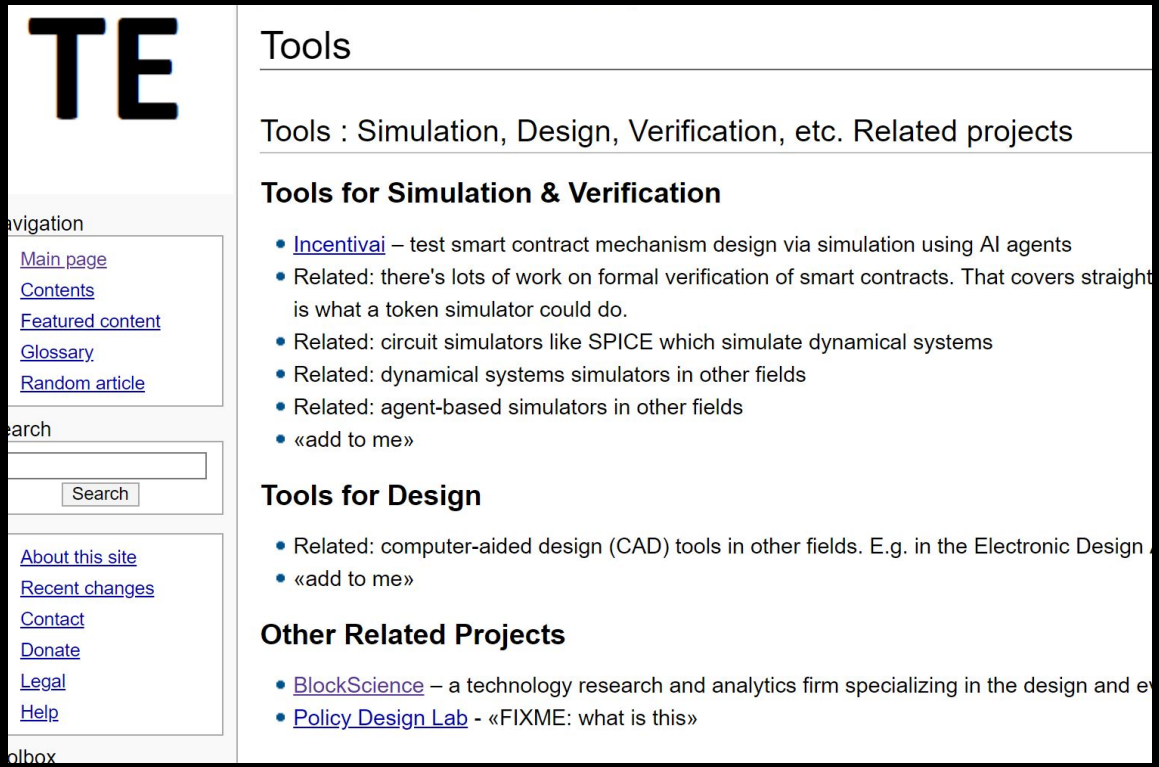


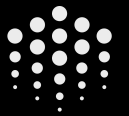
A Wiki for TE: tokenengineering.net






TE
igation
Main page
Contents
Featured content
Glossary
Random article
rch
Search
About this site
Recent changes
Contact
Donate
Legal
Help
lbox
Building Blocks
Building Blocks : Cryptoeconomic Primitives and Higher-Leve
Types of blocks (highest level) (one framing)
Proofs of human or machine work
Curation
Identity
Micro-Economical: DAOs, Stablecoins, etc
Consensus: Voting, staking, etc
Reputation
Governance / software updates
Third-party arbitration
Inter-operability
«add to me! revise me! :)»
Other Framings
Other ways to frame or group building blocks include: «FIXME: add links to these. Th
How tokens are distributed. This includes releasing coins for “work”, according to
more.
Ethereum token standards, such as ERC20 fungible token and ERC721 non-fungi


TE
avigation
Main page
Contents
Featured content
Glossary
Random article
earch
Search
About this site
Recent changes
Contact
Donate
Legal
Help
olbox
Tools
Tools : Simulation, Design, Verification, etc. Related projects
Tools for Simulation & Verification
Incentivai – test smart contract mechanism design via simulation using AI agents
Related: there's lots of work on formal verification of smart contracts. That covers straight
is what a token simulator could do.
Related: circuit simulators like SPICE which simulate dynamical systems
Related: dynamical systems simulators in other fields
Related: agent-based simulators in other fields
«add to me»
Tools for Design
Related: computer-aided design (CAD) tools in other fields. E.g. in the Electronic Design
«add to me»
Other Related Projects
BlockScience – a technology research and analytics firm specializing in the design and e
Policy Design Lab - «FIXME: what is this»



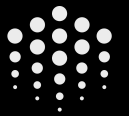
Reading Resources

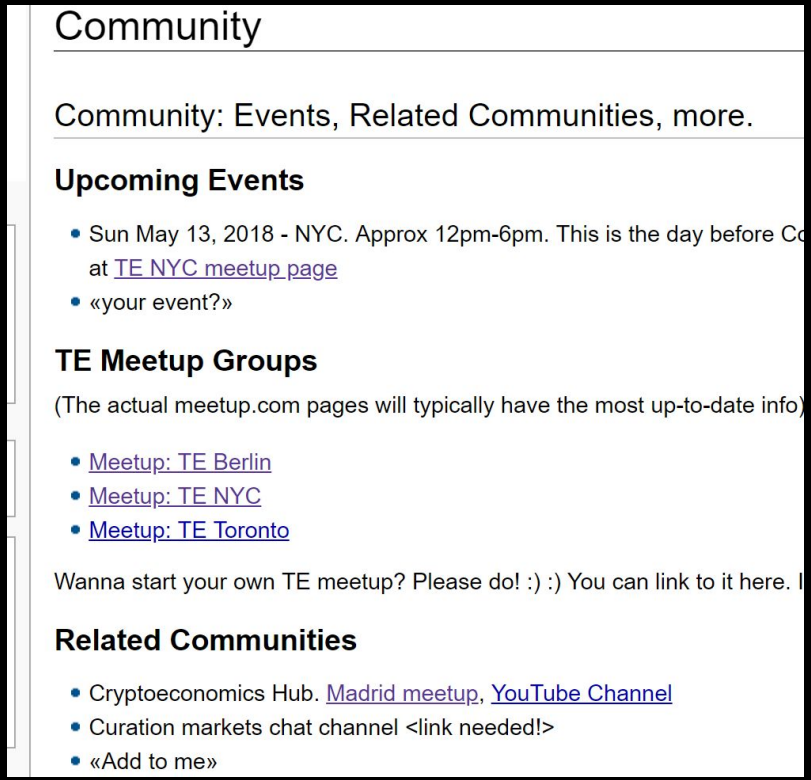
Related Disciplines

- Mechanism Design
- Algorithmic Game Theory
- Economic Systems Design
- Game Mechanics Design
- Systems Engineering
- Publicy Policy Analysis & Design
- Swarm Robotics
- Operations Research

Reading Resources

- Alex Evans, A Crash Course in Mechanism Design for Cryptoeconomic Applications, Oct 2017
- Trent McConaghy, Towards a Practice of Token Engineering Part I, Part II, Part III, Feb 2018
- Michael Zargham, On Engineering Economic Systems Part I, Part II, Mar 2018
- Elad Verbin, Behavioral Crypto-Economics: The Challenge and Promise of Blockchain Incentive Design, Mar 2018
- Jacob Horne, The Emergence of Cryptoeconomic Primitives
- Chris Burniske, The Crypto J-Curve, Aug 2017
- Chris Burniske, Cryptoasset Valuations, Sept 2017
- Adrian Colyer, Designing Secure Ethereum Smart Contracts - a Finite State Machine Approach, Mar 2018
- «add to me» «and start to group these more:) »





Community

Community: Events, Related Communities, more.

Upcoming Events

- Sun May 13, 2018 - NYC. Approx 12pm-6pm. This is the day before Co
  at TE NYC meetup page
- «your event?»

TE Meetup Groups

(The actual meetup.com pages will typically have the most up-to-date info)

- Meetup: TE Berlin
- Meetup: TE NYC
- Meetup: TE Toronto

Wanna start your own TE meetup? Please do! :) :) You can link to it here. I

Related Communities

- Cryptoeconomics Hub. Madrid meetup, YouTube Channel
- Curation markets chat channel <link needed!>
- «Add to me»

Ways to Participate

- Edit this wiki and impart your wisdom! Add blocks, tools, re
- Tweet with #tokenengineering hashtag
- Attend a meetup (see Events). Or: start your own!
- Subscribe to the TE mailing list:

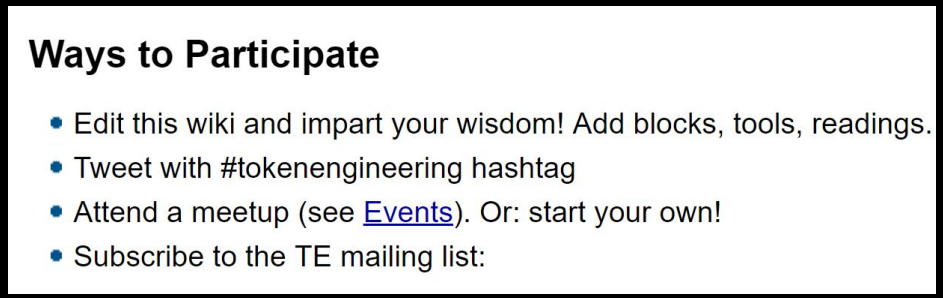
Email Address

Subscribe

;-) dan-coincrunch
;-) achill rudolph
;-) Eden Dhaliwal
;-)
morganmoskalyk
;-) Frederik Bussler
;-) hodlhands
;-)
AngelaKreitenweis
;-) adriankrion
;-) cburniske
 Or Luis Shemtov
 bmann
;-)

Towards a #TokenEngineering Community

- **Token Engineering** = Theory + practice + tools + responsibility in the creation of tokenized ecosystems.
- One framing: like an EA. We'll see other approaches today.
- **TE is a field we can all create together. Now is the time to start:)**



Token Engineering Canvas

Cheat Sheet for #tokenengineering by @nembal & @dimitridejonghe

more at: http://tokenengineering.net

Project Name:

Project Collaborators:

Objective:

answer the question: What does the platform wants to incentivise?

circle the appropriate ↓	Description	1	2	3	4	5	6	7	8	9	10	11	12	13	← Token Patterns ↓	
Ob. / Co.															1	
Ob. / Co.															2	
Ob. / Co.															3	
Ob. / Co.															4	
Ob. / Co.															5	
Ob. / Co.															6	
Ob. / Co.															7	
Ob. / Co.															8	
Ob. / Co.															9	
Ob. / Co.															10	
Ob. / Co.															11	
Ob. / Co.															12	
Ob. / Co.															13	

Step 1	Find the objective	Formulate the goal (problem) of the platform by answering the question of what does this platform wants to incentives.
Step 2	Try already existing patterns	Fix objectives (Ob.), constraints (Co.), utility (Ut.) and attacks (At.) - As many as possible then reduce the number to the most important ones. Keep adding extra ones as some might only come to mind at a later stage"
Step 2.b	Elaborate on the basic patterns	Find out if it fulfils all predetermined objectives and constrains by quickly going through in a mind-game like session. Use ✓/✕ to keep track of ideas

Towards Token Engineering

Blockchain For Data-science

- Introduction to Blockchain Mechanics
  - Introduction
  - Core Technologies
  - Consensus Protocols
- Introduction to Smart Contracts
  - What is Smart Contract
  - Solidity Essentials
  - Lab: Tutorial
- Token engineering
  - Introduction to TE
  - Towards practice of TE
- Ocean Protocol
  - Tokenize assets with Ocean
  - Ocean Roadmap
  - Ocean Protocol testnet (Plankton)
- Open Discussion

Tokenized Assets with Ocean

Economic Incentive for Bitcoin

Objective: Maximize security of network

- **Where "security" = compute power**
- **Therefore, super expensive to roll back changes to the transaction log**

$$E(R_i) \ \alpha \ H_i * T$$

E() = expected value

block rewards

hash power of actor = contribution to "security"

tokens (BTC) dispensed each block

Tokenized Assets with Ocean

Economic Incentive for Ocean

Objective: Maximize supply of relevant data

- **This means: reward curating data + making it available**
- **Where "curating" = betting on data. Reward taste-making.**

$$E(R_{ij}) \alpha \ \log10(S_{ij}) * \log10(D_j) * T * R_i$$

Expected reward for user *i* on dataset *j*

S_{ij} = predicted popularity = user's curation market stake in dataset *j*

D_j = proofed popularity = # times made dataset available

tokens during interval

Ocean Protocol Testnet (Alpha)

ocean

Plankton TestNet

References

1. Bitcoin and Cryptocurrency Technology - Coursera
2. Cryptocurrency Security Fall 2016 - Illinois University
3. Token Engineering
4. Solidity Official Documentation

ocean

ocean

Ocean Protocol Foundation

The Data Economy

aabdulwahed