Practical Challenges for Public Blockchains

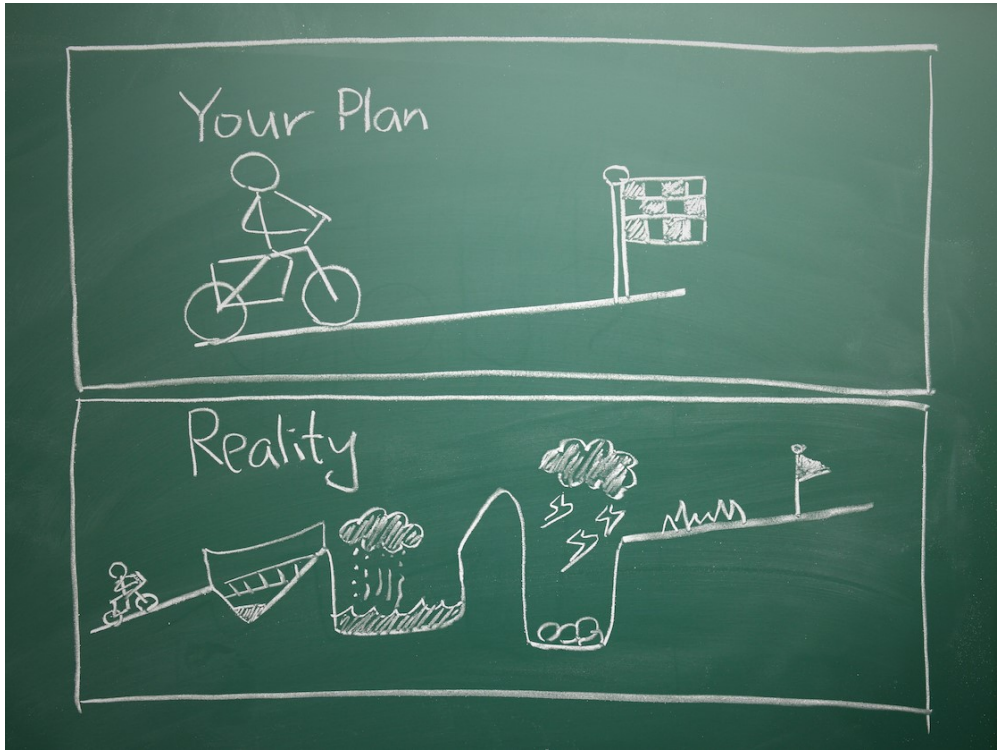
Cork Blockchain Meetup #5, 2018-09-12

Johannes Ahlmann

source: https://i.ytimg.com/vi/-hu2u-kuD8A/maxresdefault.jpg

Practical Challenges

- Storage
  - cost, governance, data availability
- Smart Contract Development
  - limitations, security, verification
- On-Chain vs. Off-Chain World
  - Oracles
- Governance
  - how to negotiate changes & address issues
- Confidentiality
  - how to keep others from seeing my data
- Authentication
  - is a user who they say they are
  - identity is a bearer instrument



http://simplecrm.com.sg/blog/wp-content/uploads/2015/12/Plan-vs-Reality-small.jpeg

Storage

- This may well be the biggest challenge for current generation public blockchains
- Can store hash of document (proof of existence)
- Eternal storage vs. time-bound
- Elective storage vs. guaranteed
- Who has ownership/ control over the data. Smart contract can't "own" off-chain data
- On-chain
  - Cost of on-chain storage (1 MB ~ between $1,000 and $10,000)
- Off-Chain:
  - IPFS, Ethereum Swarm, BigchainDB, FileCoin
- EOS application stack
  - storage, query services, resource limits, governance

http://www.dreams.metroeve.com/wp-content/uploads/2016/11/dreams.metroeve_storage.jpg

Smart Contracts

- "A smart contract is a piece of code that is stored on an blockchain, triggered by blockchain transactions and which reads and writes data in that blockchain's database."
- Data kept, code executed on every node
- Smart Contracts are immutable and need to foresee any possible future eventuality. Assumption prove to be wrong, objective changes, bug/error is discovered, how to communicate successor contract to users, how to transfer funds to successor contract
- Need conflict resolution mechanism
- Will not replace legal contracts any time soon;
  more like the "contract" of money getting from your bank account to a merchant's bank account

https://blockgeeks.com/wp-content/uploads/2016/10/smart-contracts.jpg

Smart Contract Aspects

- Very few people on the planet can write correct contracts with high confidence/ likelihood (citation needed)
- Programming languages/ tooling not very mature
- (Solidity) Language Limitations, no standard library
- Same Virtual Machine, single execution pointer,
  but your code may interact directly with other people's code
- Verification, Formal proofs?
- Smart contracts are immutable after deployment,
  any eventuality needs to be foreseen
- Life-cycle, ending a contract, funds disbursement
- Ownership, Control, Versioning, who is liable?
- Basically no data confidentiality
- Finding the Greedy, Prodigal, and Suicidal Contracts at Scale
- 3 Smart Contract Misconceptions

http://emenda.com/wp-content/uploads/2016/04/Cert_Vert.png

Smart Contract Failures 1/2

- Reentrancy - contract is called again (re-entered) during execution
  - “In simple words, it’s like a bank teller that doesn’t change your balance until she has given you all the money you requested."
    "Can I withdraw $500? Wait, before you update your ledger, can I withdraw another $500?”
- DAO Contract lost 615,391 ethers (~ $50 million at the time)
- Parity multi-signature wallet bug allowed attackers to steal more than 150,000 ethers (~ $30 million dollars at the time) and block another 513,774 ethers
- Smart Contract Best Practices - Known Attacks
- A Survey of Attacks on Ethereum Smart Contracts

Attacks

Level	Cause of vulnerability	Attacks
Solidity	Call to the unknown	4.1
	Gasless send	4.2
	Exception disorders	4.2, 4.5
	Type casts	—
	Reentrancy	4.1
	Keeping secrets	4.3
EVM	Immutable bugs	4.4, 4.5
	Ether lost in trasfer	—
	Stack size limit	4.5
Blockchain	Unpredictable state	4.5, 4.6
	Generating randomness	—
	Time constraints	4.5

Table 1. Taxonomy of vulnerabilities in Ethereum smart contracts.

Source: "A Survey of Attacks on Ethereum Smart Contracts", Atzei N., et al, 2017

Contract security - reentrancy attack

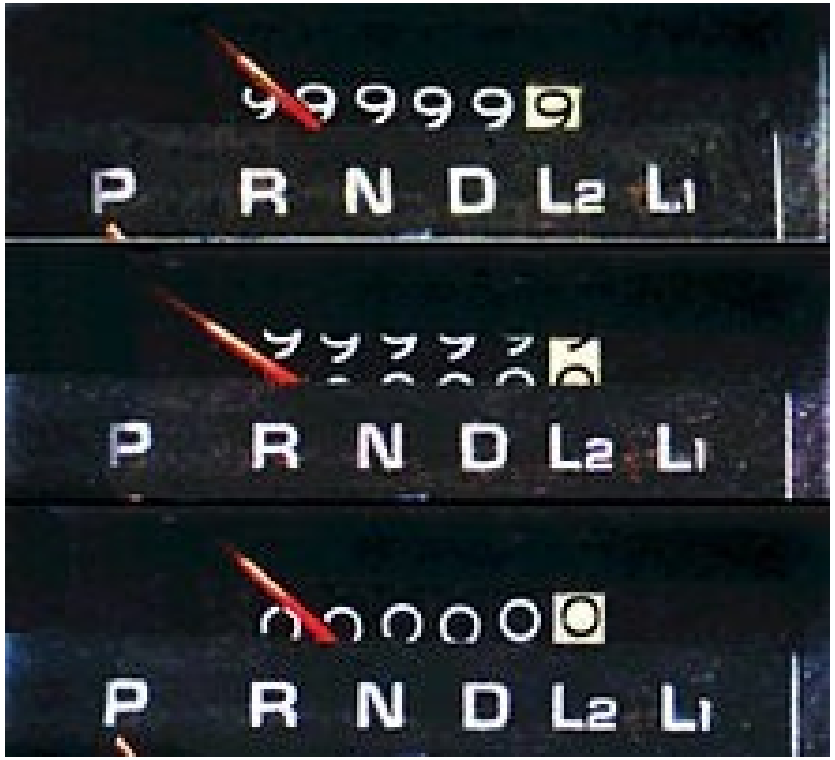
```
// INSECURE
mapping (address => uint) private userBalances;

function withdrawBalance() public {
    uint amountToWithdraw = userBalances[msg.sender];
    require(msg.sender.call.value(amountToWithdraw)());
// At this point, the caller's code is executed, and can call withdrawBalance again
    userBalances[msg.sender] = 0;
}
```

https://consensys.github.io/smart-contract-best-practices/

Smart Contract Failures 2/2

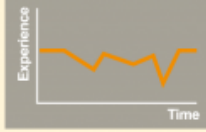
- Overflows and underflows
- Public functions
  - can be called by anyone (by functions from inside the contract, by functions from inherited contracts or by outside users)
- delegatecall
  - "contract can dynamically load code from a different address at runtime. Storage, current address and balance still refer to the calling contract, only the code is taken from the called address."



https://cdn-images-1.medium.com/max/800/0*TqqC7Raq_6sW7QdG.jpg

Smart Contract Challenges

1



The adoption curve

Businesses must overcome many different obstacles to transform the current transactional environment that has a regular, high-volume stream of records entering the system.

2

The learning curve

The transformation of many business roles is necessary. Lawyers must learn how to write computable code, and judges must learn how to interpret it, or rely on expert witnesses to testify to valid interpretations.

3

The reality of the legal and regulatory environment

Regulations are among the least automated elements of the business ecosystem. Because of its nature, they are inflexible too. Smart contracts may be the trigger to tackle this reality.

4

The complexity of the business ecosystem

The installed base of technologies, processes, and procedures reflects many assumptions businesses need to revisit when considering how to build smart contract capabilities into specific business processes.

5

A long way to standardisation

What exactly are the best practices in smart contracts, and how will standards emerge?

6

Competition

Smart contracts are just now becoming viable. Many other ways to codify agreements already exist and are used regularly. Similarly, peer-to-peer lending through software-as-a-service marketplace vendors, for example, is more mature and growing rapidly.

7

Governance-related issues

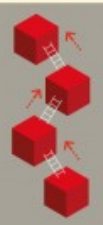
How should governments regulate such contracts? How would governments tax these smart contract transactions?

8

Data Privacy

A blockchain stores information everywhere and forever. Then, how can it guarantee data privacy, especially under the GDPR? Researchers are working on new protocols to make available "secret contracts" instead of "smart contracts", with nodes on the blockchain able to compute data without ever "seeing" it.
It's still a project, however.

9



Operational issues and execution speed

Having a dependency chain guarantees smart contract's reliability. In Ethereum, an open software platform based on blockchain, smart contracts can only be executed in chronological order— a rule o computation must execute after another rule completes. Because parallel processing isn't supported, the process renders the blockchain slow, with the risks of having some nodes (computers) unable to finish computations on time. The transaction, then, isn't validated and the system becomes worthless.

10

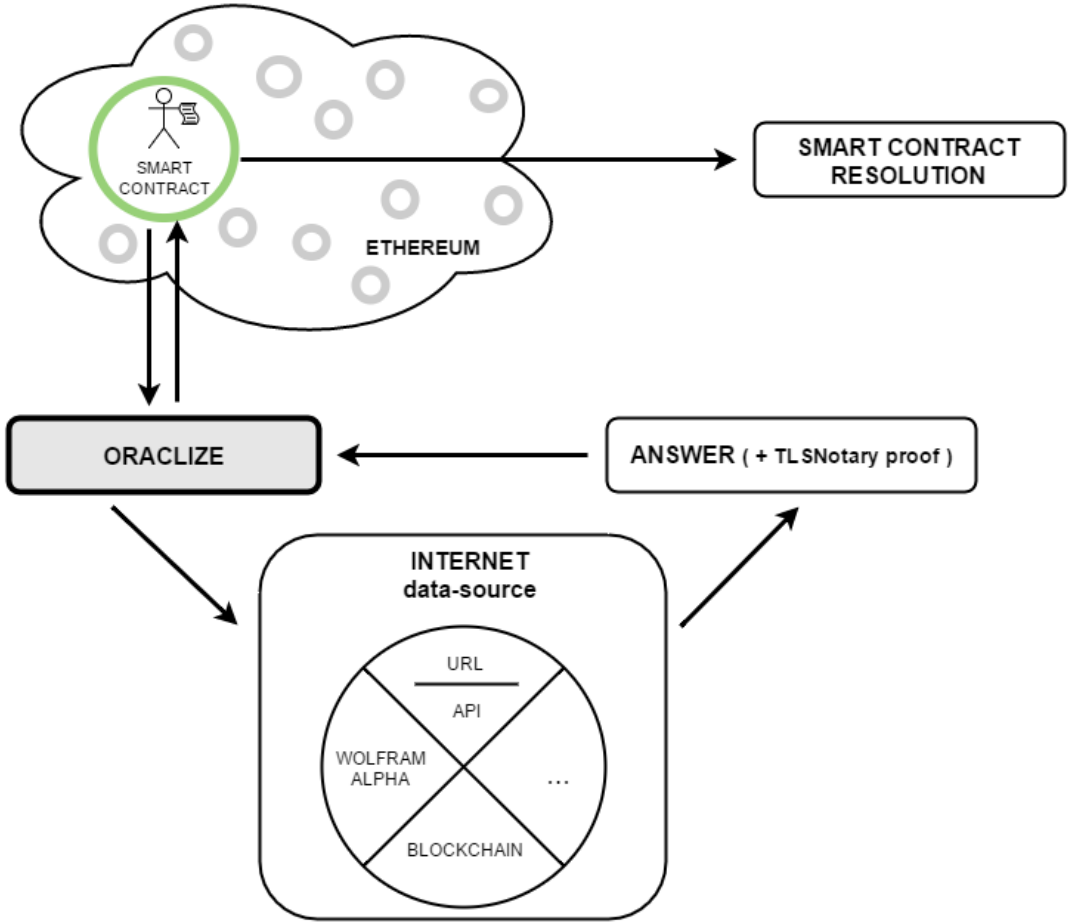
People's expectations

The hype about blockchain, cryptocurrencies and smart contracts have led enthusiasts, the public and even entrepreneurs to imagine smart contracts applications that aren't viable. We all dream with autonomous independent software, but it's necessary to convey the right message and educate people about the possibilities and limitations of new technologies.

https://blog.pwc.lu/wp-content/uploads/2018/06/Smart-contracts_Infographic_10-Challenges-768x1253.png

Oracle

- Allows contracts to react to/ take into account off-chain events
- Service that makes off-chain information available on-chain
- Downside: Inherently centralized, Single Point of Failure
- Ideally: Use multiple sources, voting
- What if Oracle changes its mind after the fact (doping scandal, insider trading, etc.)
- How to deal with unforeseen circumstances (match was cancelled, public company is taken private, country disappears, currency disappears)



https://cdn-images-1.medium.com/max/1200/0*nEUpj7fK5HaHAOsK.png