SMART CONTRACTS AND PENSIONS: CHALLENGES OR OPPORTUNITIES?

Dr. C.A.W. Brewster and Dr. Maarten Everts
OUTLINE

› Definitions
› Failures of Smart Contracts
› Technical Issues
› Philosophical/political Issues
› Features of pension systems
› Uses of Smart Contracts in pension systems
› Platonic ideals vs. Aristotelian reality
DEFINITIONS

‒ Smart contracts
  ‒ are pieces of code
  ‒ codifying agreements and trust relations
  ‒ deployed on a virtual machine …
  ‒ … to be automatically executed by the VM

‒ In the Blockchain + Smart contract world:
  ‒ SCs will control of high value assets:
  ‒ SCs will be:
    ‒ Unchangeable, autonomous and unstoppable
    ‒ Publicly visible and analyzable
    ‒ Run in a public, hostile environment
    ‒ Be written by fallible human beings

“A smart contract is a piece of code which is stored on an blockchain, triggered by blockchain transactions, and which reads and writes data in that blockchain’s database.” – Gideon Greenspan
FAILURES OF SMART CONTRACTS

- There are many high profile cases of Smart Contracts failing:
  - 2016: Most famous: “The DAO”: A smart contract running a virtual company, obtained $150M of funding in ETH, lost $60M due to bug in the code.
  - 2017: Parity multi-sig wallet: bug resulted in loss of $30M
  - 2017: parity multi-sig wallet again: $300M frozen (and lost)
A blockchain platform is a peer-to-peer network of nodes. The nodes collaborate to reach consensus on changes to the database. In Ethereum (for example) state of database is the state of a “world computer” programmable via Smart Contracts. Assumption: a smart contract will execute as specified. Reality: various mechanisms result in different results e.g.: Re-ordering of transaction orders (intentionally) by miners. Unexpected consequences of transactions (e.g. with The DAO allowing re-invocation). Solarity language → compilation produces unexpected results.
Typical examples of SC: An index based agricultural insurance policy, or bank transaction SC:
- The SC has to run on every node in the blockchain
- They all query an oracle (weather service, bank server) and expect to get same data
- BUT there is no guarantee:
  - Oracle may change
  - Oracle may be inaccessible
- A Smart Contract responding to external data is not deterministic (cannot always give the same result)
- Solution here is a “trusted third party” that queries the oracle
Smart contracts exist in multiple layers – from human intention through to CPU instruction.

Each layer needs syntax and **semantics** – semantics to specify the meaning of concepts and the map to the real world.

Real world impinges on the “blockchain world” – the meaning of concepts change, the real world changes.

Conflict of semantics leads to real world failure:
  - Classic example is the Mars Climate Orbiter 1998
  - Hospital kills patients (digitally), Michigan 2003
  - Nuclear attack early warning systems (repeatedly)

Proper semantics means formal vocabularies (ontologies) to systematise descriptions of the world.
(But you still want to use them ….)

Strategies for more robust smart contracts:

- Best practices – risk analysis, security requirements, attack modelling, code audits.
- Design Patterns (Gang of Four – Gamma et al.) – for SC ownership, data provider authentication, transfer of funds
- Static analysis tools – many tools being developed especially for Ethereum’s Solidity language, but also for EVM bytecode analysis
- Formal verification i.e. formal proofs that code is correct
  - This needs to be done at the various layers mentioned
- Use better, more rigorous languages for Smart Contracts e.g. Tezos with the Michelson language (and many others)
- Use languages that do not surprise you
PHILOSOPHICAL/POLITICAL ISSUES

- Philosophically using Smart Contracts implies:
  - We can describe the world, or part of the world perfectly
  - This part of the world will not change
  - We can release the SC onto a blockchain to run forever.
  - There will be no mistakes in the code or the representation of the world
- Logically this denies much that we know about the world:
  - Humans are fallible
  - The world changes
  - We usually like to have democratic/political control of processes
- So we need to be able to revise a SC, change it, adapt it to reality and human needs
FEATURES OF PENSION SYSTEMS

- Most advanced economies (OECD) have pension systems.
- Huge variety in design and structure:
  - Mandatory state provision vs. company provision vs. voluntary savings
  - Also changing continuously in view of political and economic changes.
FEATURES OF PENSION SYSTEMS – THE NETHERLANDS

- Dutch pension system considered one of the best in the world.
- Three pillars: State provision, private (company based) pension, personal investments
- The Dutch pension system is highly unusual in that it covers most of the population, and the ratio of liabilities to assets is very good.
- People change jobs so may (or may not) move pensions – this has been a scenario for smart contracts
- However the real world changes e.g. pension age, ratios of liabilities, and regulatory (legal) changes.
USES OF SMART CONTRACTS IN PENSION SYSTEMS

- The idea is to address problems in the pension system: poor governance, poor performance, hidden costs, poor data management, and fraud.
- “Program everything into smart contracts” will achieve (it is claimed):
  - Better transparency
  - Better governance
- Examples include Auctus (https://auctus.org/) and some people in this room.
- Real motivation is to reduce costs – which in reality means effort and complexity will be transferred to somewhere else in the system – probably the end user.
- Main use of blockchains in the pensions sphere is to provide permanent records --- but what about GDPR?
OPTIMISTISM OR PESSIMISTISM

- Optimists will say: THIS IS AN ENGINEERING PROBLEM
- Look at civil engineering:
  - Many bridges have failed over history
  - Most bridges do not fail now
- Problem is really robust software systems are relatively few.
- Most modern computer science has worked in an “agile” manner, a “move fast and break things” philosophy
- This may not work with pension funds …

- If we can roll back, adapt, change functionalities, intervene THEN blockchains with smart contracts may be useful.
PLATONIC IDEALS VS. ARISTOTELIAN REALITY

- Smart contracts are a Platonic ideal – they indicate a belief in perfect idealised reality.
  - We imagine pensions as abstract ideals with perfect features and performance.
- Aristotle believed in observation of reality, rather than inferring from abstract principles.
  - In reality, pensions are complex with lots of variations and exceptions.
- We can systematise and automate processes – this seems to be a passion of our culture – but to avert disaster we need to keep humans in the loop.
THANK YOU FOR YOUR ATTENTION

Take a look: Blockchain.TNO.NL

TNO’s whitepaper on Smart Contracts: