BLOCKCHAIN AND DISTRIBUTED LEDGER TECHNOLOGIES

Lecture 1 - Overview

Volker Skwarek
Hochschule für Angewandte Wissenschaften Hamburg

HWI HAMBURG

HAW HAMBURG

LEARNING OUTCOMES

HWI HAMBURG HAW HAMBURG

After this lecture studens shall

- **know**
  - the origins of Blockchain technologies
  - why they have been introduced for digital currencies
  - what Bitcoin is
- **have got a deeper understanding of**
  - the basic protocol of Bitcoin transactions and the role of participating nodes
  - evolution steps and major types of BC/DLT systems
- **derive and transfer**
  - supporting use cases for BC/DLT systems

ORIGINS OF BLOCKCHAINS AND DISTRIBUTED LEDGER TECHNOLOGIES

HWI HAMBURG HAW HAMBURG

blockchain and distributed ledger technologies (BC/DLT) base on **2 security pricinples from 1990es, combining the to 1 protocol**

- **application**: unalterable storage of trusted information on trustless servers
- **assumptions**:
  - no system is that secure, that it cannot be manipulated
  - every system can be deleted or destroyed
  - extensive communication between systems is possible

- **target**: even if manipulation at time t is possible, manipulation of data before t must not stay unrecognized

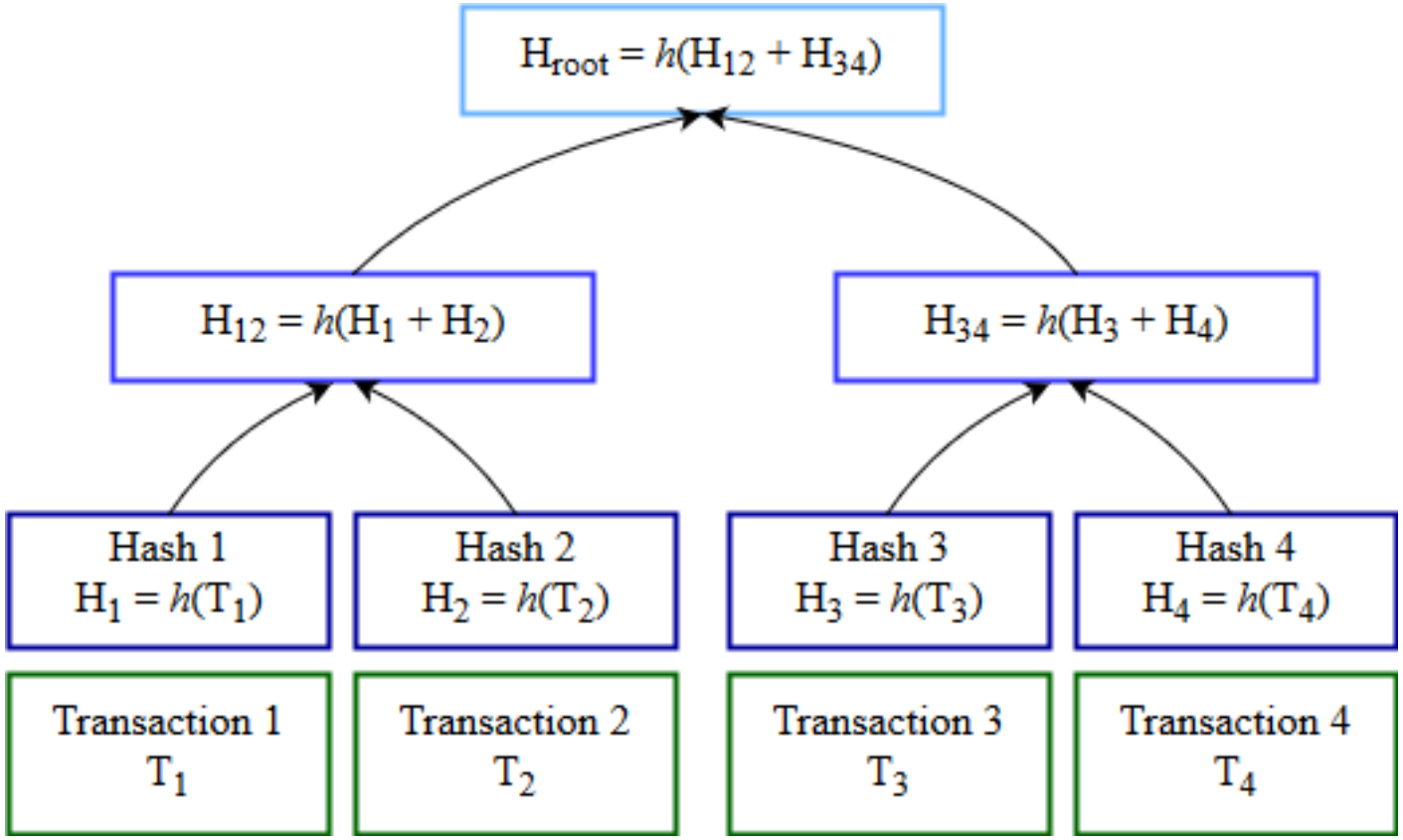
Sources/ Further reading:

Haber, S., Stornetta, W.: How to time-stamp a digital document. Advances in Cryptology-CRYPT0'90. 437–455 (1991).
Schneider, B., Kelsey, J.: Cryptographic Support for Secure Logs on Untrusted Machines. San Antonio, Texas, USA (1997).

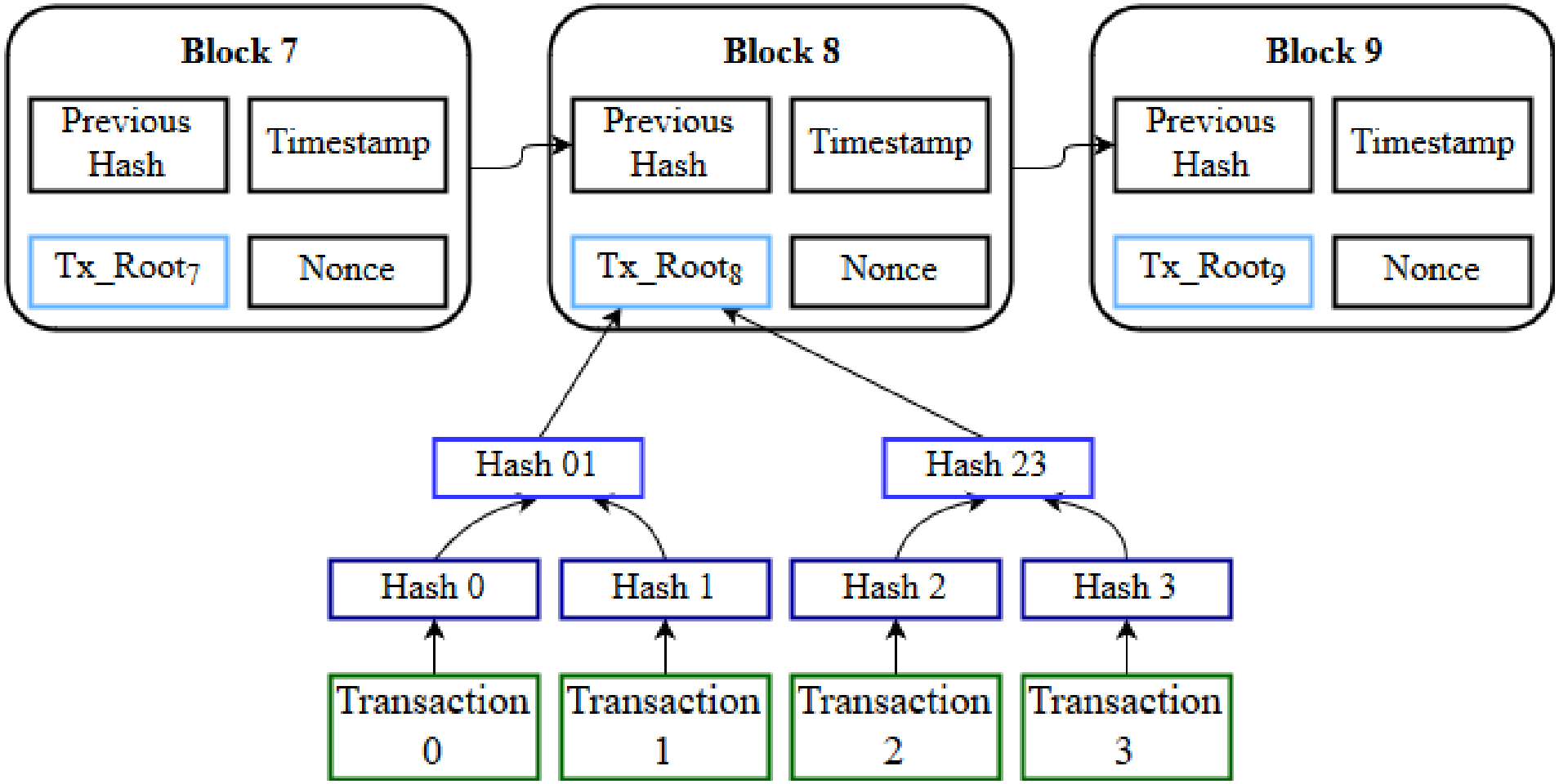
UNDERLYING PRINCIPLES (1/4)
COMPRESSING TRANSACTIONS BY HASHING

HWI HAMBURG HAW HAMBURG

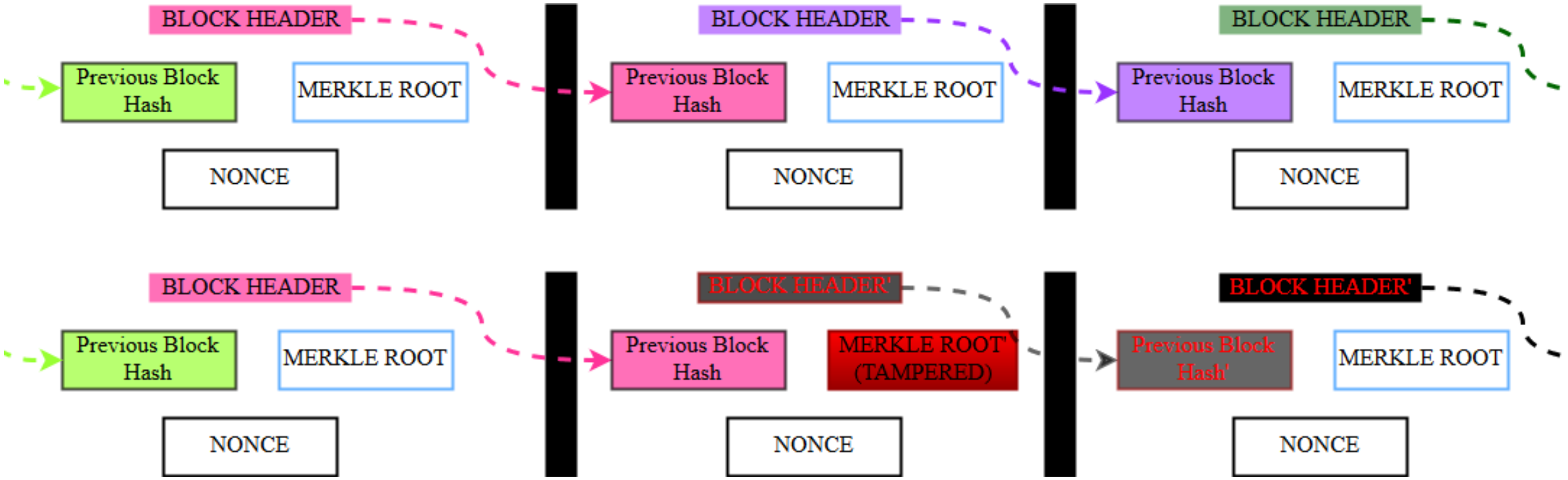
E.G. Merkle Tree



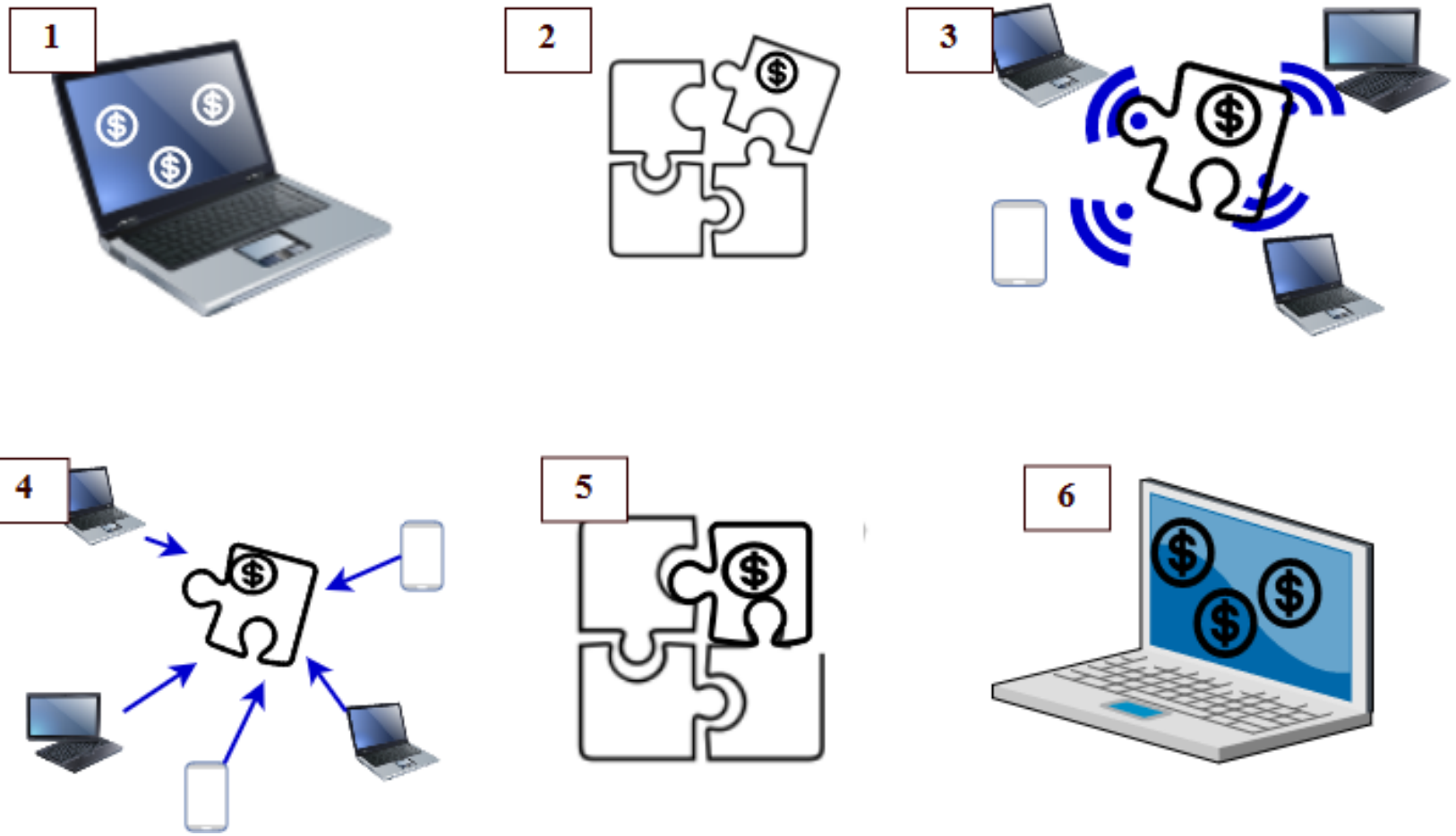
UNDERLYING PRINCIPLES (2/4) (BLOCK)CHAINING



UNDERLYING PRINCIPLES (3/4)
HOW THE CHAIN PROTECTS ITSELF



UNDERLYING PRINCIPLES (4/4)
DISTRIBUTING AND LEDGERING



- **chaining:** a manipulation of a single transaction needs to (but cannot) alter the whole information chain
- **distributing:** full system information is distributed to each participant of the system and requires a >50% attack to be altered
- **hashing:** transactions are stored in the system and blockwise compressed (hashed) to a single value
- **trusted:** all authorisations for transactions are cryptographically confirmed and traceable by private/public key principles
- **consensus based:** transaction is reused only if parties agree on ist correctness

ORIGIN OF BLOCKCHAIN/DLT
DIGITAL CURRENCY FOUNDED BY BITCOIN IN 2009

HWI HAMBURG HAW HAMBURG

A group or person with the pseudonyme Satoshi Nakamoto published this application first for the use in Bitcoin in 2008

- peer-to-peer version of electronic cash,
- allowing online payments to be sent directly between different parties,
- without the need of intermediaries such as financial institutes,
- prevent double-spending
- network timestamps transactions by hashing them into an ongoing chain
- hash-based proof-of-work, forming a record that cannot be altered without redoing the proof-of-work
- security assumption: majority of CPU power is controlled by nodes not cooperating to attack the network

Source/ Further reading:
Pseudonymous grouped named Satoshi Nakamoto published in 2008: Bitcoin: A Peer-to-Peer Electronic Cash System

BLOCKCHAIN AND DLT
- MORE THAN DIGITAL CURRENCY

HWI HAMBURG HAW HAMBURG

Non-Financial Use Cases			
Marketplace	Blockchain in IoT	Smart Contracts	Reviews/ Endorsement
Providing premium rights & brand based coins	Filament, ken Code - ePlug, Chimera-inc.io	Otonomos, Mirror, New system Technologies, Symbiont	The World Table, Asimov, TRST.im
Real Estate	Diamonds	App Development	Digital Identity
Factom	Everledger	Proof of wondership for modules in app development: Assembly	Trustatom, Uniquid, Onename, Sho Card
Authentication & Authorization	Gold & Silver	Network Infrastructure & APIs	Storage & Delivery, Digital Content
BlockVerify, Degree of Trust, Everpass, The Real McCoy	Bit Reserve, Real Asset Co., BitShares, DigitalTangible (Serica)	Ethereum, NXT, Codius, Chromaway, BlockCypher, Hello Block, Corona, Mastercoin	BitProof, Ascribe, Blockcai Stam (Alexar

Financial Use Cases			
Data Storage	Gaming	Ride Sharing	Trading Platforms
Storj.io, Peernova	Play, Deckbound, PlayCoin	La'zooz	BitShares, Coins-e, Spritzle, Secure Assets, Kraken, MUNA, equityBits
Currency Exchange & Remittance		P2P Transfers	
CryptoSigma, Stellar, Kraken, Fundrs.org, Billion, Ripple, Coinbase, BitPesa		BitnPlay, DeBuNe, BItBond, BTC Jam, Codius	

USE CASE EXAMPLE: LAND REGISTRY

Honduras to build land title registry using bitcoin technology

By Gertrude Chavez-Dreyfuss
Reuters 15 May 2015

A bitcoin sticker is seen in the window of the 'Vape Lab' ca
possible to both use and purchase the bitcoin currency, in
2015. REUTERS/Peter Nicholls/Files

By Gertrude Chavez-Dreyfuss

NEW YORK (Reuters) - Honduras, one of the poorest countrie
agreed to use a Texas-based company to build a permanent a
system using the underlying technology behind bitcoin, a com
Thursday.

Schweden nutzt jetzt offiziell die Blockchain für Grundbucheintragungen

7. Juli 2017 | Sven Wagenknecht

Schon seit 2016 ist bekannt, dass Schweden an einer Blockchain-Lösung diesbezüglich forscht. Ende Mai wurde dann die letzte Testphase erfolgreich abgeschlossen. Trotz der fortschrittlichen Digitalisierung des Grundbuchamtes, soll die Blockchain zu deutlichen Effizienzsteigerungen führen.

Konkret sollen so um die 100 Millionen Euro eingespart werden können, die für Bürokratie und Betrugsfälle jedes Jahr fällig werden. Das es tatsächlich zu so hohen Einsparungen kommt, bleibt jedoch zu bezweifeln. Darüber hinaus haben aber auch die Banken Interesse an dem Projekt, da sich so in der Zukunft auch Hypothekengeschäfte über eine Blockchain darstellen lassen. Entsprechend wundert es auch nicht, dass zwei schwedische Banken bei dem Projekt involviert sind.

Georgia: Authorities Use Blockchain Technology for Developing Land Registry

in Innovation

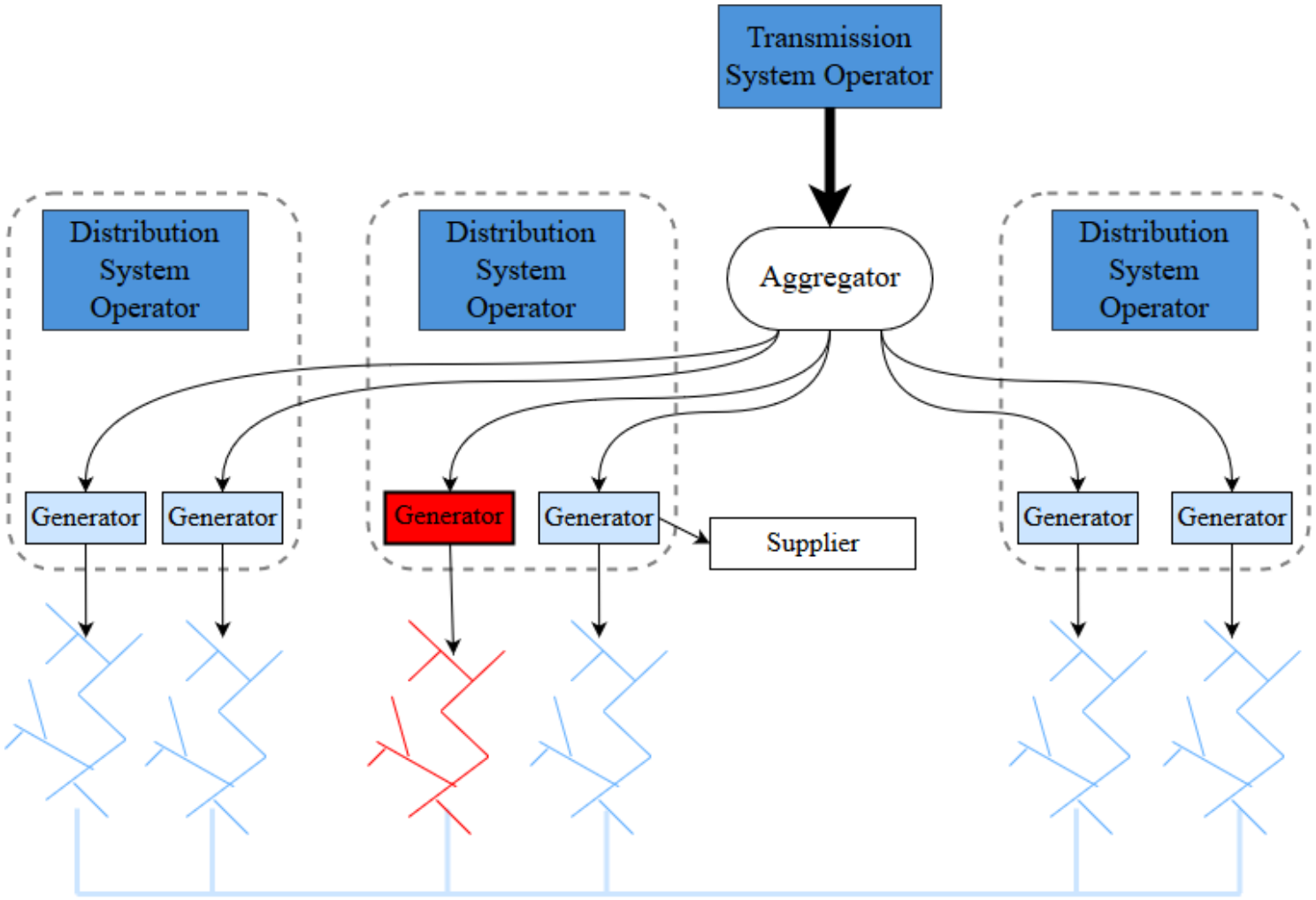
neering
erty

tcoin.

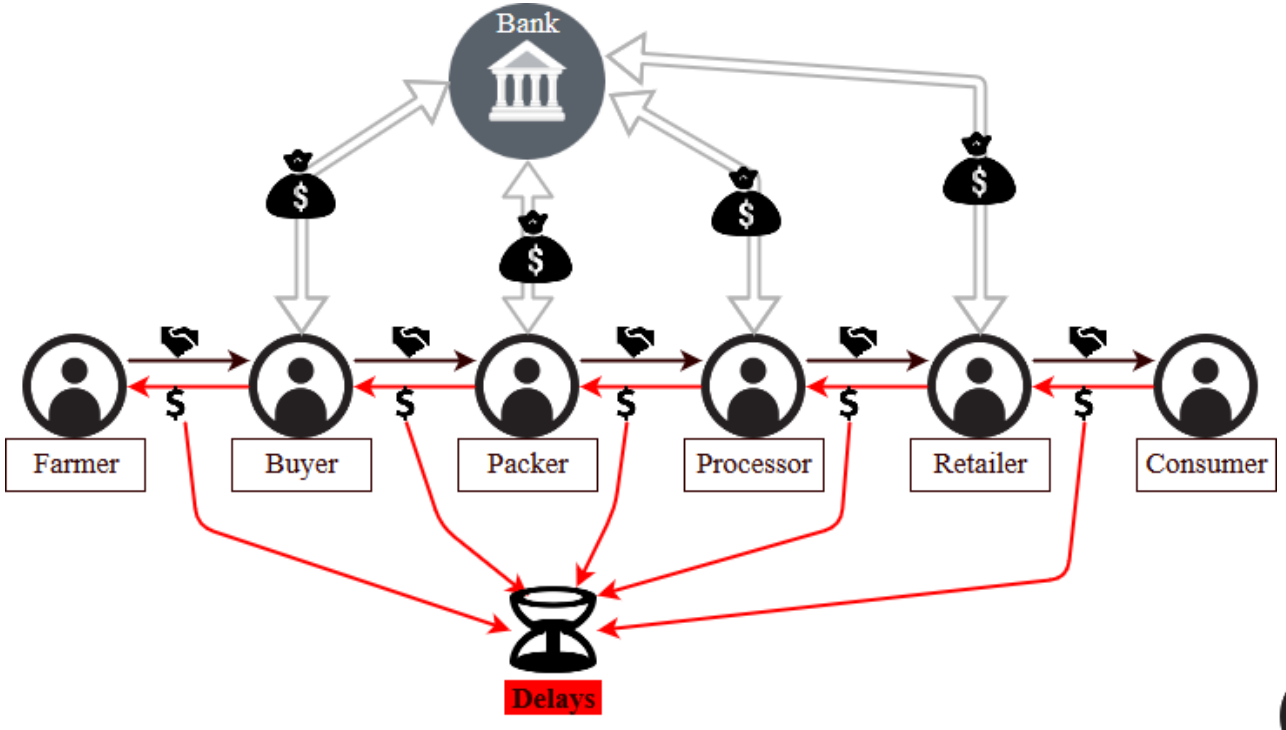
oping
an a
land
n
am
es,

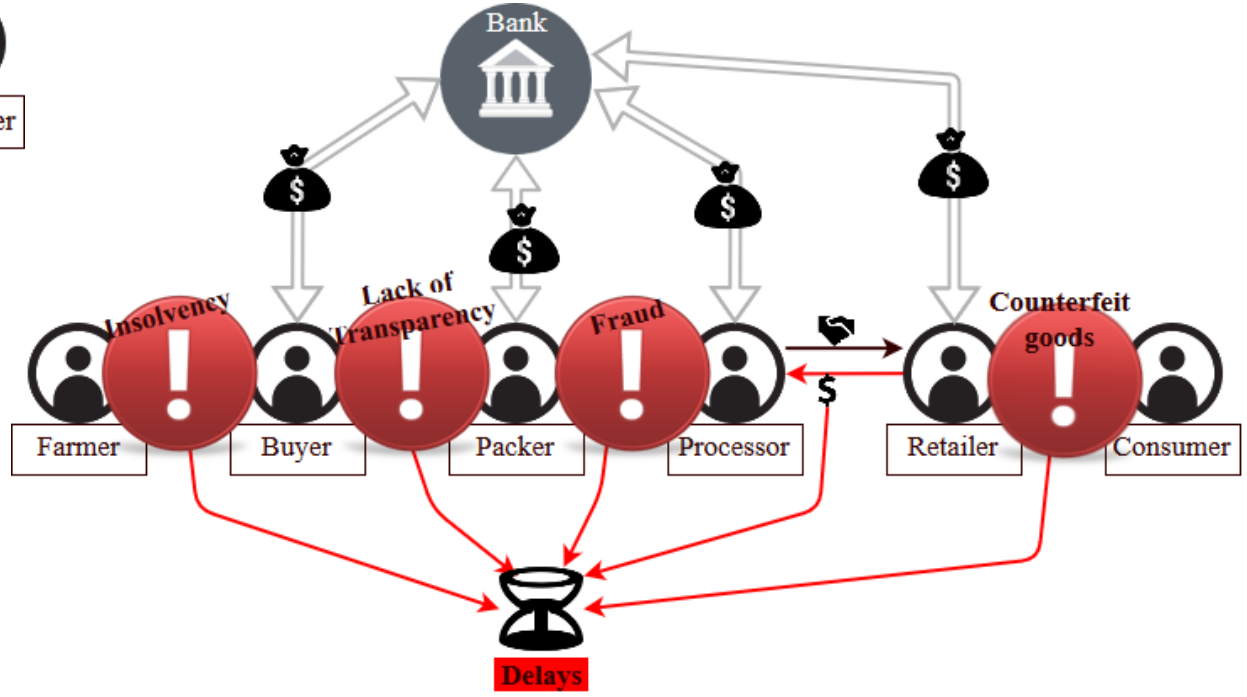
The chairman of Georgia's National Agency of Public Registry, Mr. Papuna Ugrekhelidze, signs a new memorandum of understanding with the CEO of the BitFury Group, Mr. Valery

USE CASE EXAMPLE:
ENERGY TRADE AND GRID STABILITY

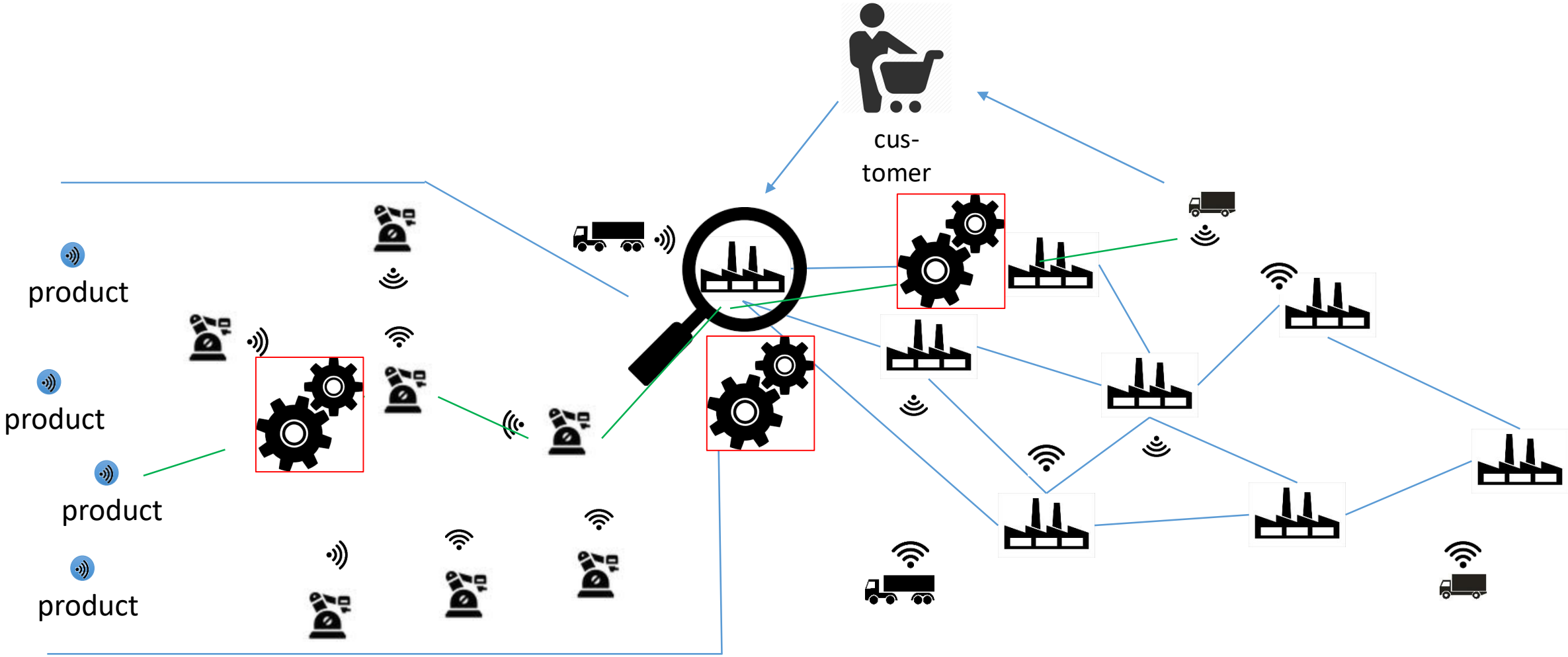


USE CASE EXAMPLE: TRADE FACILITATION





USE CASE EXAMPLE: ORDER-ENTRY-MANAGEMENT PRODUCTION WITH A SERIES OF SCS

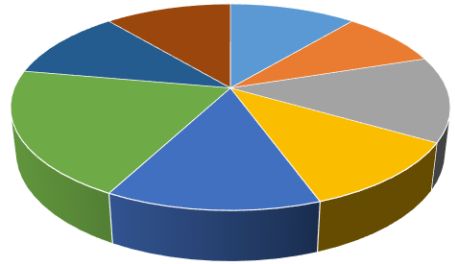


WHAT MAKES A USE CASE A GOOD USE CASE?

Use Cases should leverage at least one or more of the basic propertiers of BC/DLT

- trust
- distribution – temporally or spatially
- communication
- (reduction of) interfaces
- asynchronity

classification of use cases for smart contracts (45 evaluated)



- supply chain management
- license management
- machine-machine-automation
- energy trading/management
- automated regular contractual transactions
- registry services
- tracking and quality control

USE CASES ADAPTED FROM INDUSTRIE 4.0 - SCENARIOS

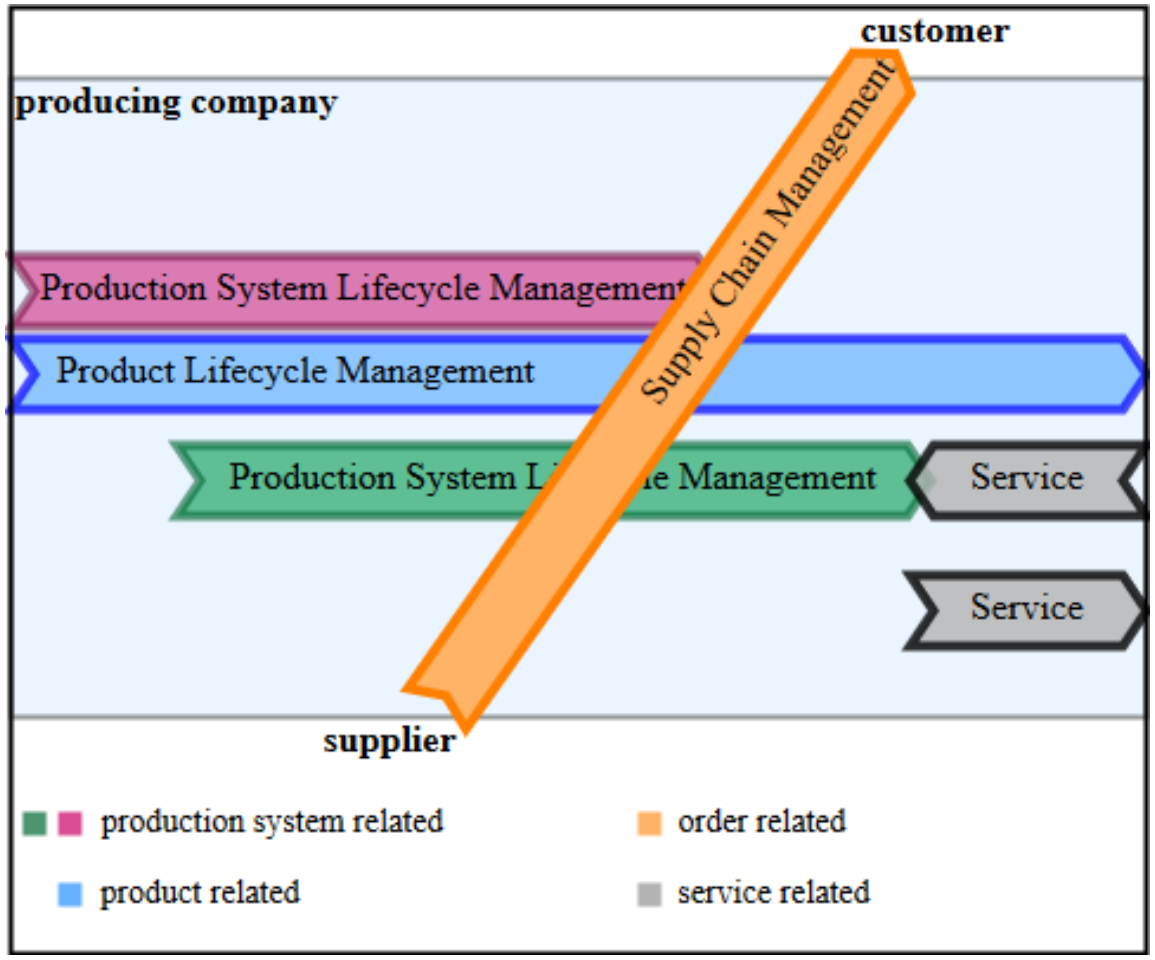
HWI HAMBURG HAW HAMBURG

Possible Use Cases:

- order entry management production (AGP)
- changeable factory (WFF)
- selforganized adaptive logistics (SAL)
- value based services (VBS)
- man-machine-interaction in the production (MTI)
- smart product development from smart production (SP2)
- innovative product development (IPE)
- seamless and dynamic engineering of production units (DDA)
- recycling management (KRW)

Source/ Further reading:
Bundesministerium für Wirtschaft und Energie ed., Weiterentwicklung des Interaktionsmodells für Industrie 4.0-Komponenten, (2016)

USE CASES ADAPTED FROM INDUSTRIE 4.0 - SCENARIOS



DECISION CRITERIA FOR BLOCKCHAIN USE CASES

Or: Let‘s use BC/DLT – applications where we don‘t have better solutions without them!

- ☑ BC/DLT are able to secure transactions without a trusted central instance
- ☒ BC/DLT require a lot of memory capacity as the do not forget (in their pure sense)
- ☒ BC/DLT consume a lot of bandwidth for communication
- ☒ BC/DLT (may) consume a lot of energy depending on their mining and consensus process

➡ It‘s crucial for success and acceptance of BC/DLT to find a good use case

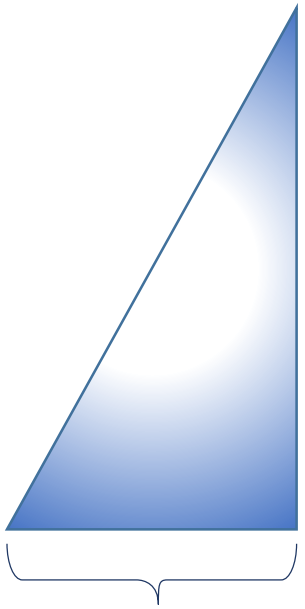
EVOLUTIONARY STEPS OF BLOCKCHAINS

Generation 1: presentation, transaction (e.g. Bitcoin)

Generation 2: interaction by smart contracts (e.g. Ethereum, Hyperledger)

Generation 3: interaction between systems (e.g. sidechain)

Generation 4: blockchain operating systems



degree of autonomous interaction

Source/Further reading:
Fuchs, C. et. al..: Theoretical Foundations of the Web: Cognition, Communication, and Co-Operation. Towards an Understanding of Web 1.0, 2.0, 3.0. Future Internet. 2, 41–59 (2010)

SMART CONTRACT – A SHORT DEFINITION

Usual quotation from Nick Szabo, who gave the name to Smart Contracts (Idea of Smart Contracts, (1997), paragraph 2, center):

> „A canonical real-life example, which we might consider to be the primitive ancestor of smart contracts, is the humble vending machine. Within a limited amount of potential loss (the amount in the till should be less than the cost of breaching the mechanism), the machine takes in coins, and via a simple mechanism, which makes a freshman computer science problem in design with finite automata, dispense charge and product according to the displayed price.
> […]
> **Smart contracts go beyond the vending machine in proposing to embed contracts in all sorts of property that is valuable and controlled by digital means.**“

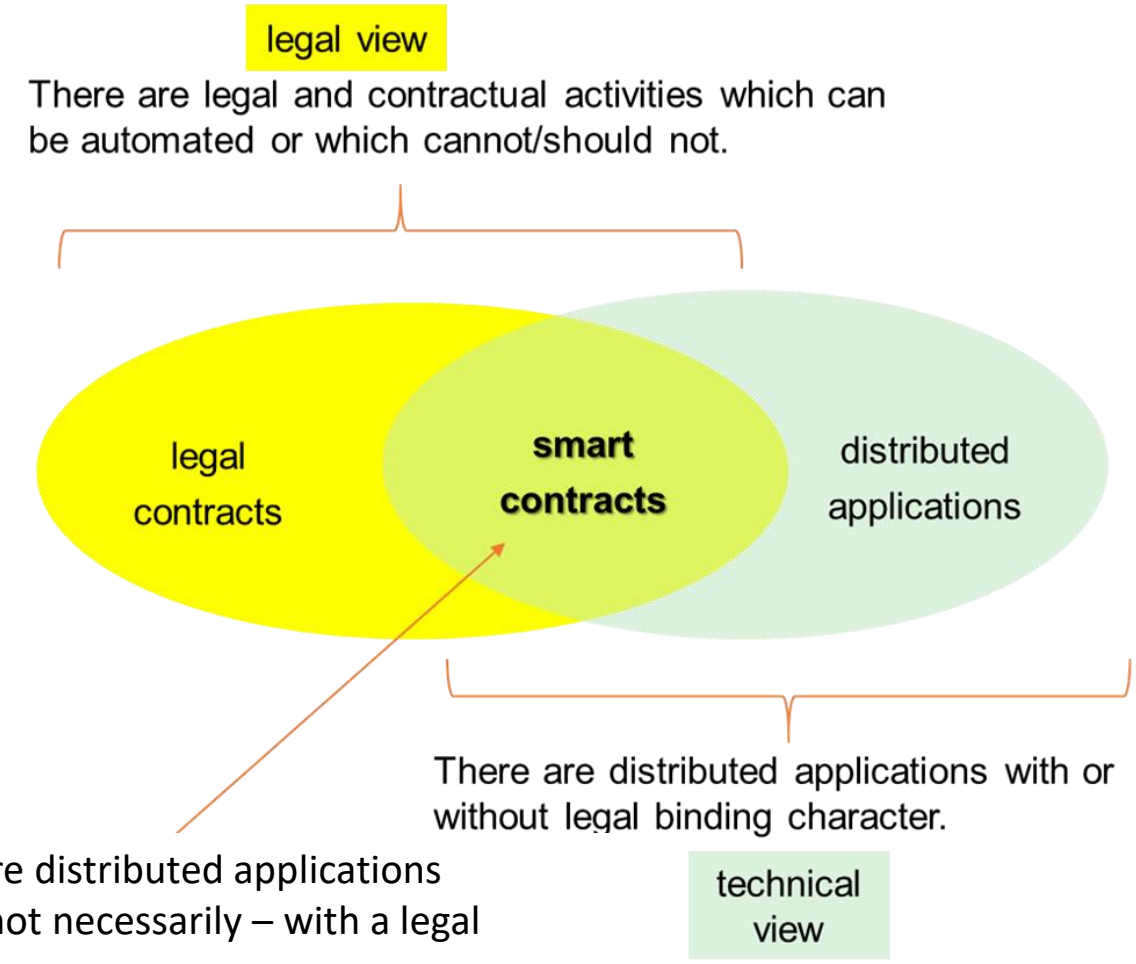
SMART CONTRACT BY NICK SZABO

Further from Nick Szabo (Idea of Smart Contracts, (1997), paragraph 2, center):

> „Many kinds of contractual clauses (such as collateral, bonding, delineation of property rights, etc.) can be embedded in the hardware and software […]“

Important addition: Nick Szabo talks about contractual clauses = contractual elements, not necessarily only about contracts

ANOTHER PERSPECTIVE ON SMART CONTRACTS



smart contracts are distributed applications potentially – but not necessarily – with a legal binding intention

WHAT DOES SUCH A SMART CONTRACT IN ETHEREUM LOOK LIKE?

Ethereum Vision: unstoppable censorchip-resistand self-sustaining decentralised world computer	
Solidity, Serpent, LLL	Smart Contract programming languages
pyethapp, geth, eth	main Ethereum software, written in different languages
ETH	inbuilt native cryptocurrency of Ethereum used for paying for Smart Contracts to run
Whisper, Swarm, Ethereum Virual Machine	decentralized file storage, computation, communication protocols
Serenity, Metropolis, Frontier, Homestead	different software releases

WHAT SMART CONTRACTS IN ETHEREUM REALLY ARE

HWI HAMBURG HAW HAMBURG

Externally Owned Accounts (EOAs)

- have an ether balance,
- can send transactions (ether transfer or trigger contract code),
- are controlle by private keys,
- have no associated code.

Contract Accounts

- have an ether balance,
- have associated code,
- code execution is triggered by transactions or messages (calls) received from other contracts.
- when executed
  - perform operations of arbitrary complexity (Turing completeness)
  - Manipulate its own persistant storage, i.e. can have ist own permanent state
  - can call other contracts

WHAT DOES A SMART CONTRACT LOOK LIKE…

HWI HAMBURG HAW HAMBURG

```
pragma solidity ^0.4.20;

contract Hello  {                                   // A string variable
string internal greeting;

event GreetingChanged(string _greeting);     // Events that gets logged on the blockchain

function Hello(string _greeting) public {                  // The function with the same name as the class is a constructor
      greeting = _greeting;
   }

function setGreeting(string _greeting) external {                  // Change the greeting message
      greeting = _greeting;
      // Log an event that the greeting message has been updated
      GreetingChanged(_greeting);
   }

function greet() external view returns (string) {                  // Get the greeting message
      return greeting;
   }
}
```

Virtual Machine/Compiler Version

SC-name

„event“ creates a BC log-entry

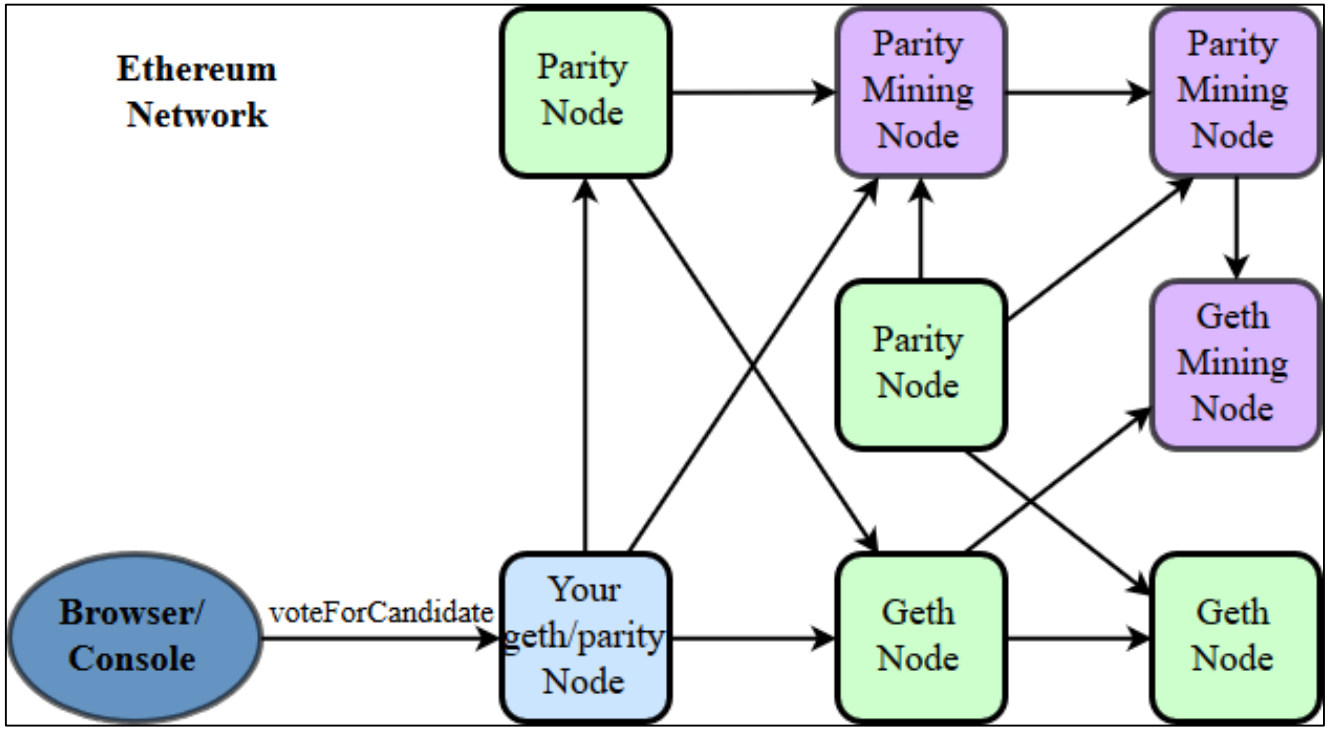
function name, externally callable with parameters

function name, externally callable without parameters but visible return

LIFECYCLE OF A SMART CONTRACT IN ETHEREUM

```
Voting.deployed().then
    (function(instance) { instance.voteForCandidate('Nick', {gas: 140000, from:
web3.eth.accounts[0]}).then
        (function(r){ console.log("Voted successfully!")
        })
    })
```

0. compile binary
1. deploy
2. instantiate
3. call function



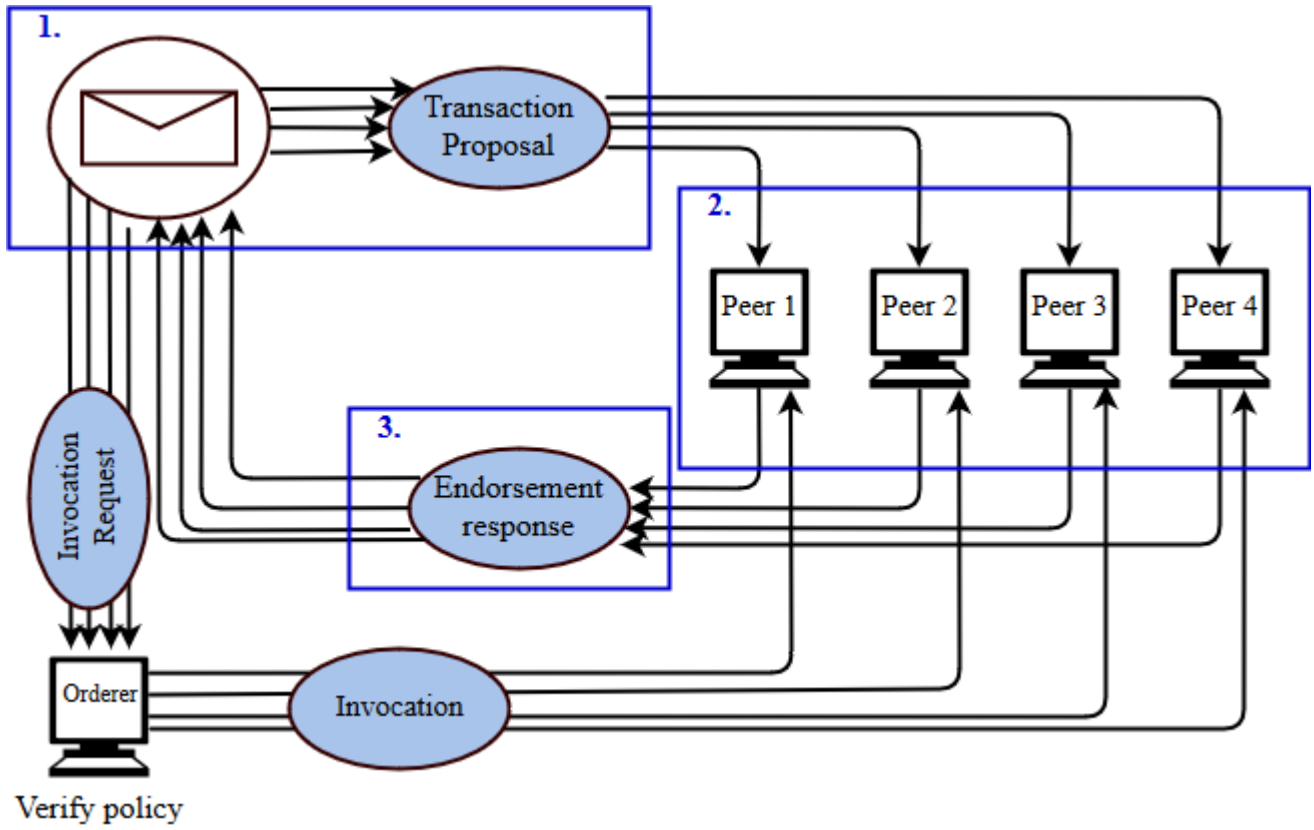
ALL BLOCKCHAINS ARE THE SAME?
NEVER - HERE: HYPERLEDGER FABRIC

HWI HAMBURG HAW HAMBURG

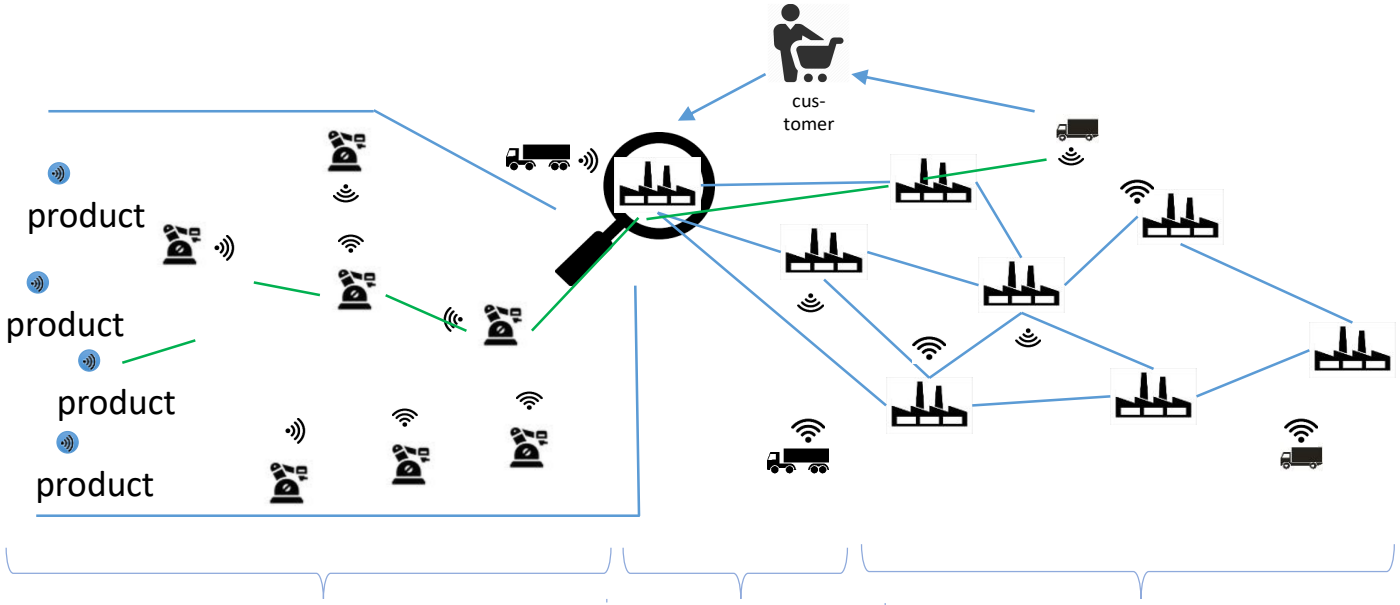
- Hyperledger Fabric keeps data offchain on every single node
- 3-step concept: execute → order → validate
- endorsement policy defines, which node is allowed/enabled to execute chaincode
  - explicit peers must all endorse transactions of type T
  - majority of peers must endorse transactions of type U
  - at least 3 peers must endorse transactions of type V

ALL BLOCKCHAINS ARE THE SAME? NEVER! HERE: HYPERLEDGER FABRIC AS DLT ONLY

1. Fabric starts with a transaction proposal sent to some peers for endorsement.
2. Endorsing peer executes the chaincode, which (if it succeeds) yields an actual transaction for the ledger.
3. Endorsing peer then signs the transaction and returns it to the proposer. This is the Execute step in execute-order-validate.
4. Creator of the proposal receives enough signatures to satisfy the endorsement policy, it can submit the transaction to be added to the ledger. (Order step)

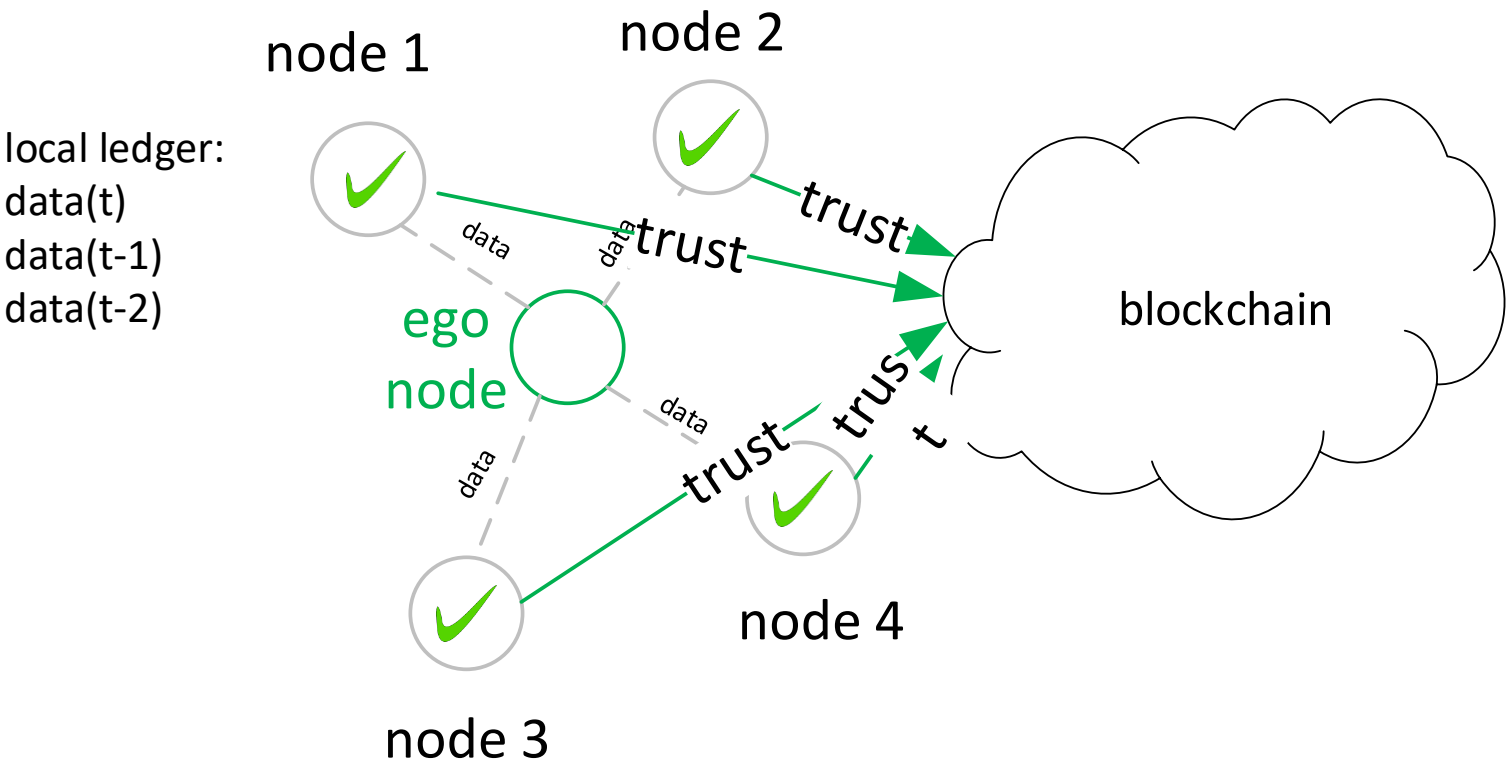


HOW TO ATTACH EXTERNAL INFORMATION TO SCS



scenario A: classic	standard sensor- and productionnetwork	oracle e.g. oraclize	Cloudchain: e.g. Hyperledge, Ripple, Ethereum
scenario B: modern	Sensorchain: e.g. HAW-Sensorchain, IOTA	border gateway	cloudchain
scenario C: innovative	Sensorchain	metachain overledger	cloudchain

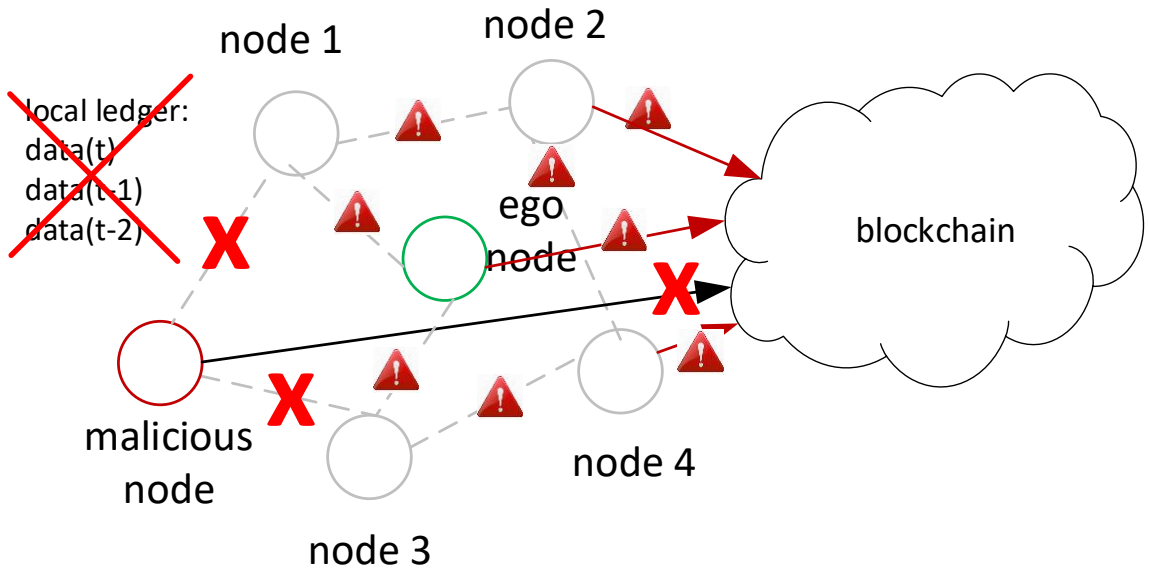
SOLUTION: DISTRIBUTED SENSOR LEDGER



METHOD OF INTRUSION DETECTION

Attacking sensor in an established network can be detected by using the concept of practical Byzanthene Fault Tolerance:

- compare ist information with existing ledgers
- communicate doubts about trustworthiness among sensor network
- get doubts confirmed by other sensor nodes
- inform blockchain about doubts and suppress malicious information

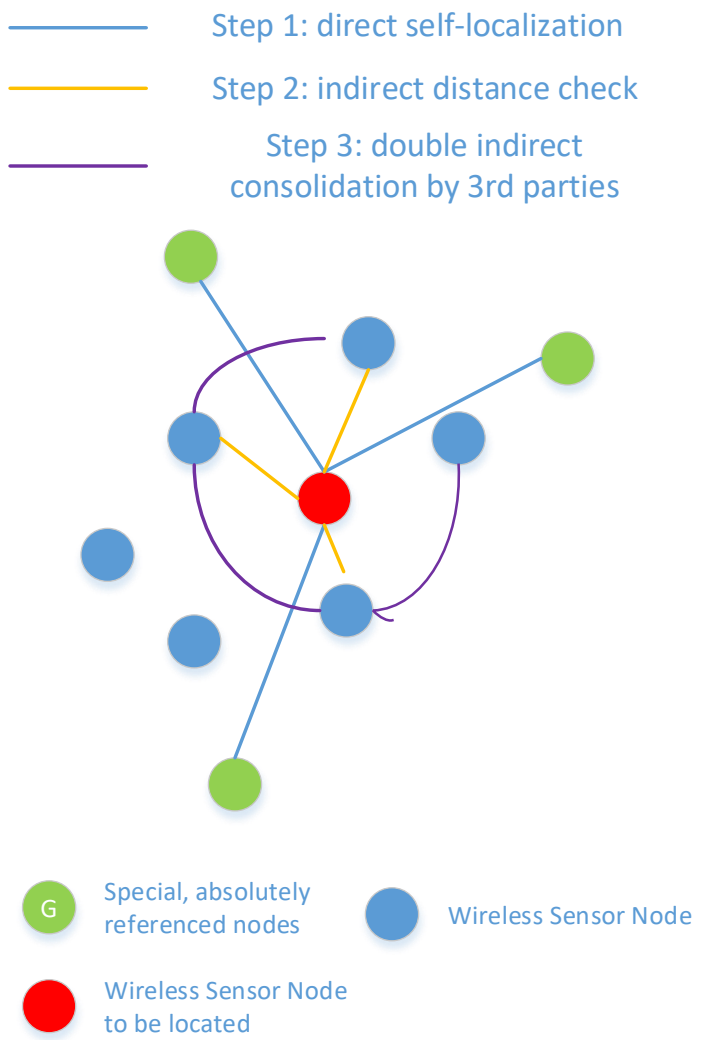


IDENTITY MANAGEMENT

- identity of sensor node = combination of unique ID- number and location

- location is determined in 3 steps and 2 different ways
  1. self-determination of ego-location by every sensor node
  2. verification of ego-location by strength-of-field based distance measurement
  3. consolidation of self- and verified location by comparison with information from other network nodes

- Within a network with BFT-number of nodes, a fake of the location is hardly possible without being detected



THE „PRESENT“ FUTURE – CHAINS OF BLOCKCHAINS

HWI HAMBURG HAW HAMBURG

	Purpose	Interoperability	Scalability	Fault Tolerance
Interledger	Payments across different payments system based on DLT	1-C-1	Ledgers allow connectors to run nodes	Depends on notaries or institutions that validates transactions
Virtualchain	Ability to migrate from one DLT to another for fault tolerance	1-C-1	Ledgers allow to write metadata	Depends on the two blockchains involved in the migration
Cosmos	Overcome Blockchain limits and transfer assets	N-C-N	Should implement IBF to talk with The Hub	Confined in the zones (User responsiblity on where they move coins)
Overledger	Build a messaging layer for multi-ledgers applications	N-N	Ledger's readability and/or writeability	Protocol based
Sidechain	Add new innovating features to the main crypto currencies	1-1	Ledger's compliance with two-way peg	Security faults on sidechain are confined in the sidechain itself.
Aion	Solve Blockchain isolation problem	N-C-N	Aion-compatible	Bompatible blockchain Aion-1 follow rules and consensus of Aion-1
Polkadot	Transfer assets and data (smart contract)	N-C- N	Should implement the polka dot security consensus	Para chains follow rules and consensus of Polkadot