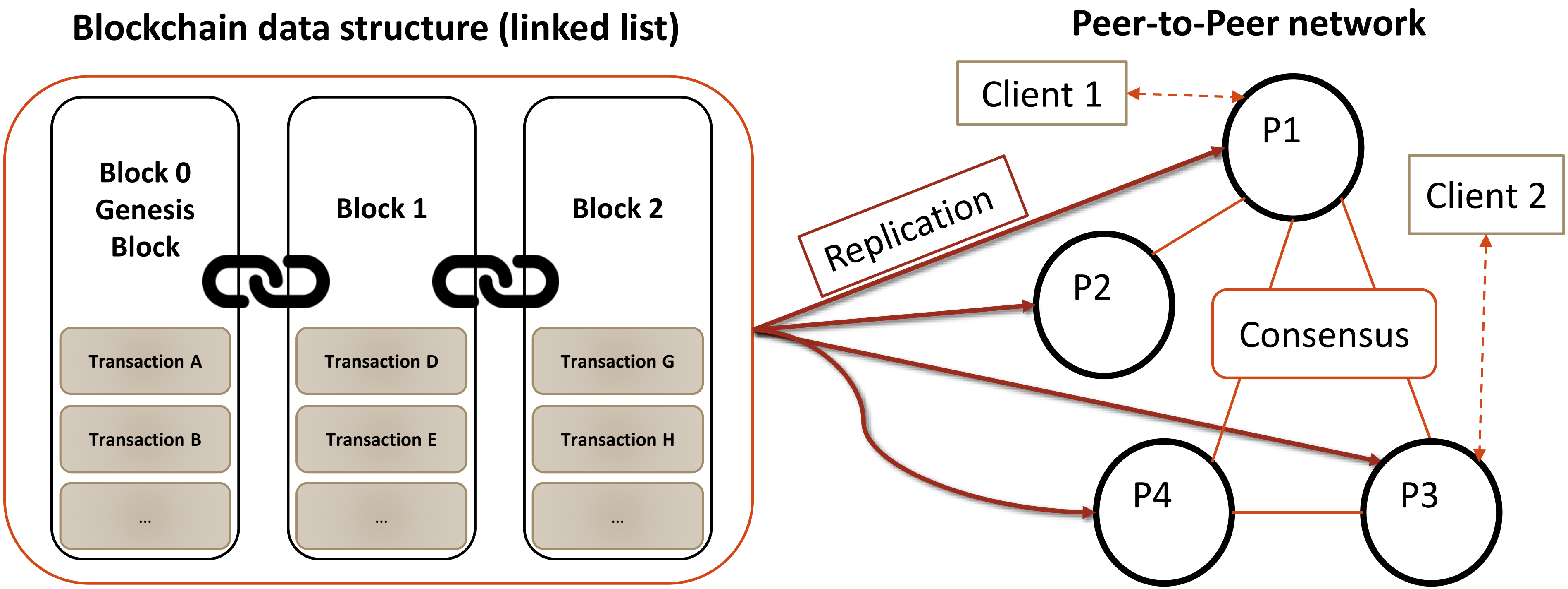
Towards Dependable, Scalable, and Pervasive Distributed Ledgers with Blockchains

BY KAIWEN ZHANG,
HANS-ARNO JACOBSEN

Blockchain 101

Distributed Ledger Technology (DLT)



Blockchain data structure (linked list)

Peer-to-Peer network

Block 0
Genesis
Block

Block 1

Block 2

Transaction A

Transaction B

…

Transaction D

Transaction E

…

Transaction G

Transaction H

…

Replication

Client 1

Client 2

P1

P2

P3

P4

Consensus

Cryptography is used to…

…encrypt data, prevent modification, insert new blocks, execute transactions, and query…

the distributed ledger

Main objectives of the paper

For those new to blockchains…

- Definitions of key terms and concepts
- For a thorough explanation of Bitcoin, Ethereum, Hyperledger, …: “Deconstructing Blockchains” – Tutorial Slides @ https://fuseelab.github.io/#publications

For those looking to get started in blockchain research…

- Literature survey & research directions
- Template for describing potential blockchain use cases

For those already doing research in blockchains…

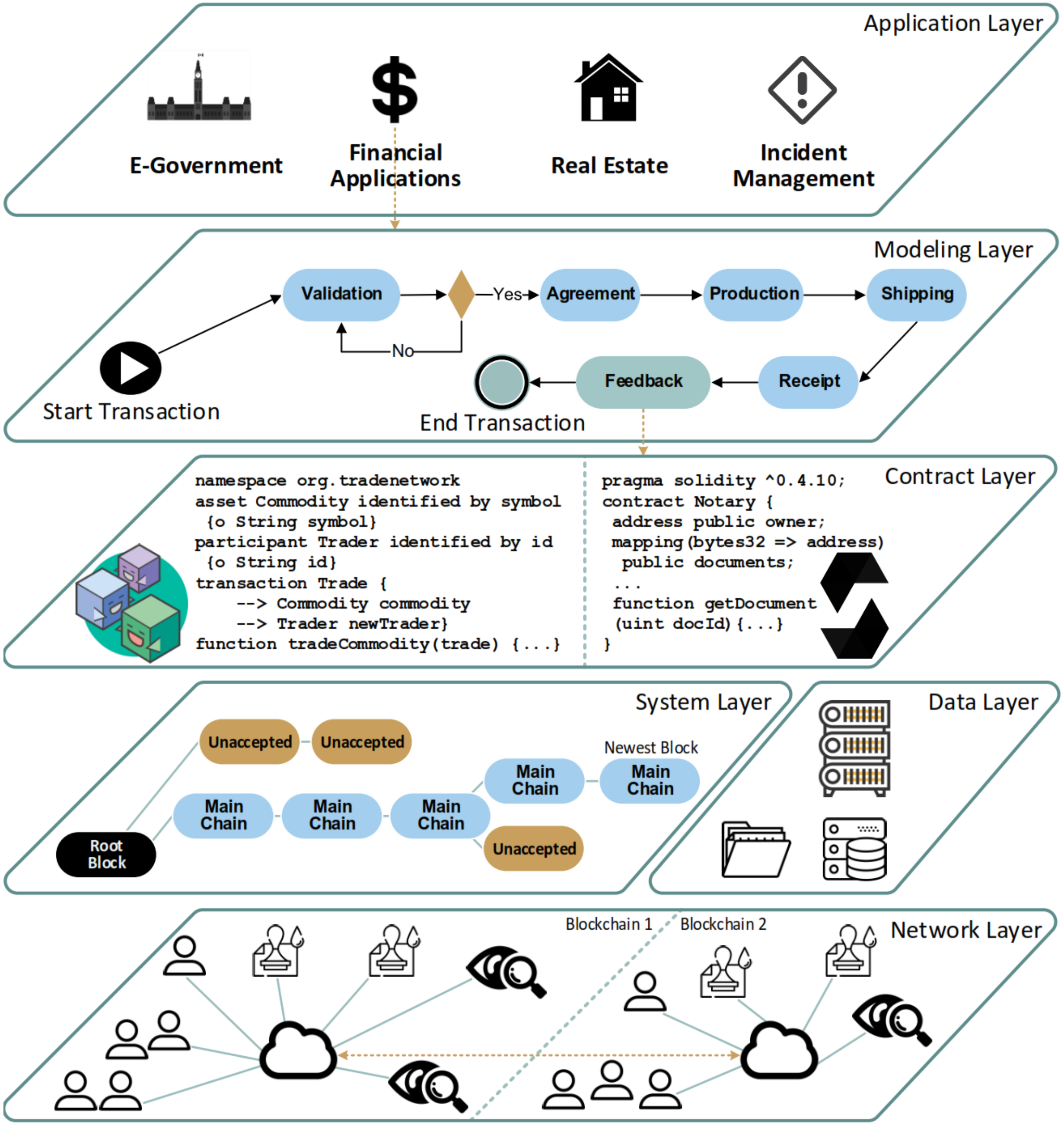
- Three ways to categorize and position your research
- By layer (reference architecture)
- By targeted application (generations)
- By properties impacted (DCS conjecture)

Blockchain Reference Architecture

This vision diagram encompasses all aspects related to blockchain technologies.

Upper layers capture application semantics and their implementation.

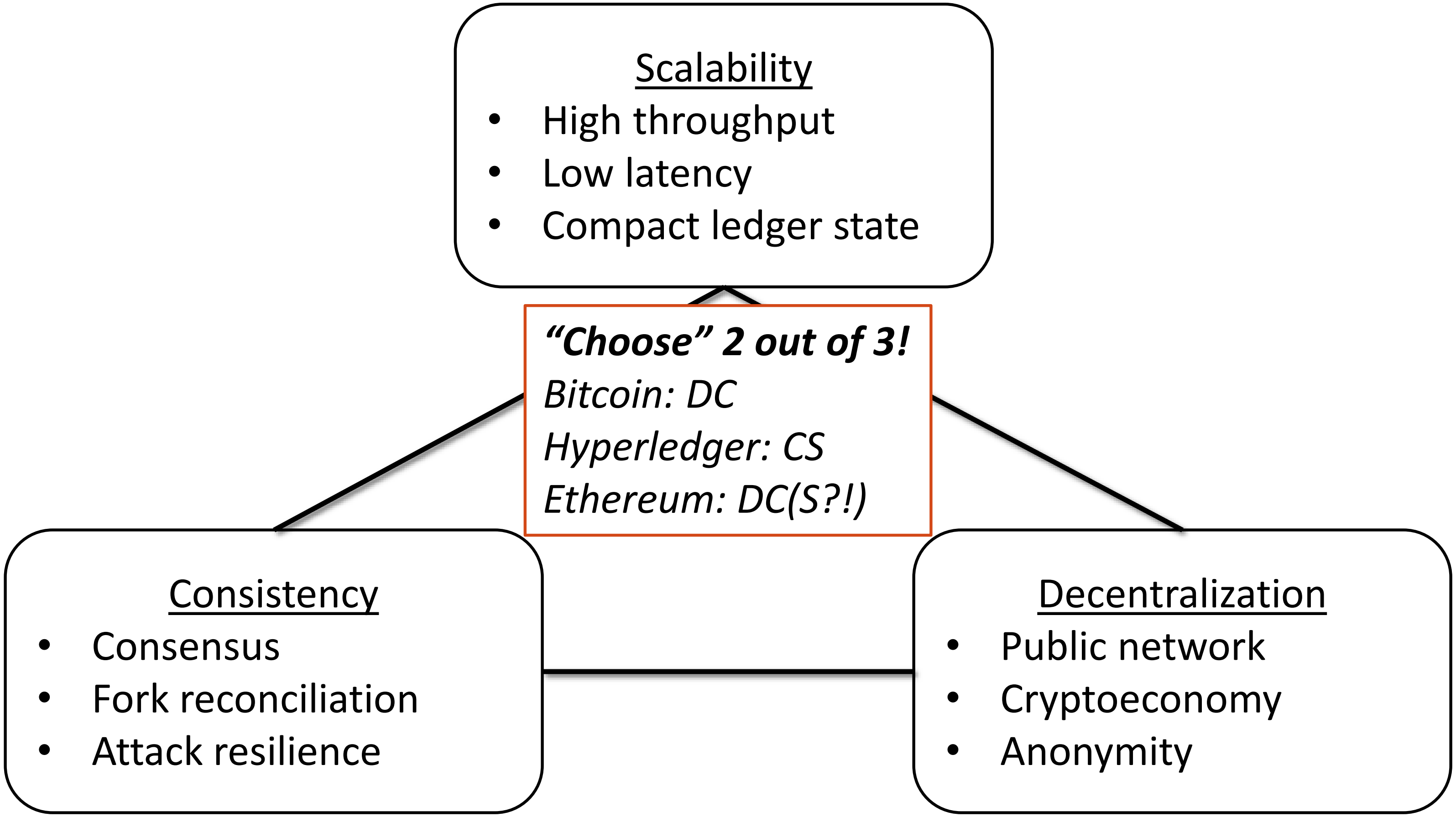
Lower layers are concerned with technical system details.



```
namespace org.tradenetwork
asset Commodity identified by symbol
 {o String symbol}
participant Trader identified by id
 {o String id}
transaction Trade {
    --> Commodity commodity
    --> Trader newTrader}
function tradeCommodity(trade) {...}
```

```
pragma solidity ^0.4.10;
contract Notary {
 address public owner;
 mapping(bytes32 => address)
  public documents;
 ...
 function getDocument
 (uint docId){...}
}
```

“CAP Theorem” for DLTs



Scalability

- High throughput
- Low latency
- Compact ledger state

“Choose” 2 out of 3!

Bitcoin: DC

Hyperledger: CS

Ethereum: DC(S?!)

Consistency

- Consensus
- Fork reconciliation
- Attack resilience

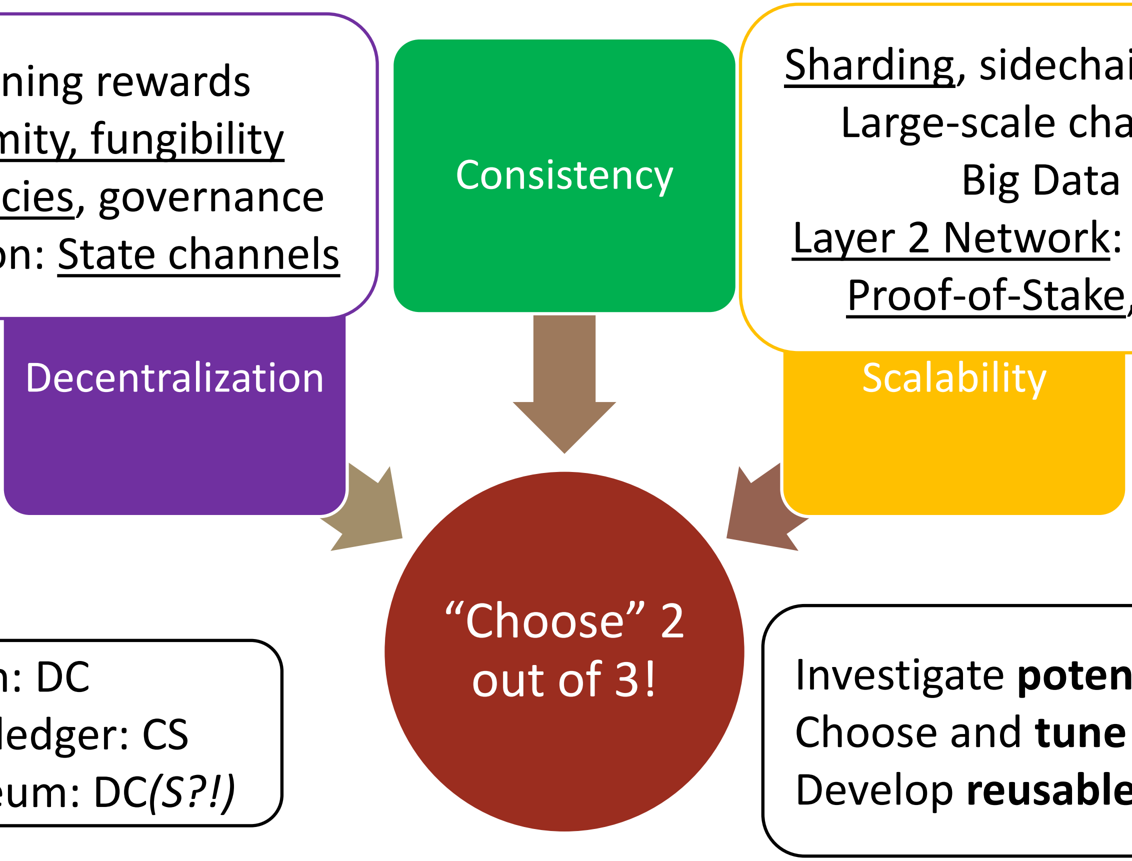
Decentralization

- Public network
- Cryptoeconomy
- Anonymity

The DCS Conjecture

Safe and verifiable smart contracts
Attacker models: <51% attacks
Security of off-chain services (e.g. exchanges)
“Garbage in, garbage out”: IoT barrier

Incentives, mining rewards
Privacy: Anonymity, fungibility
Endorsement policies, governance
Selective replication: State channels



Sharding, sidechains, tree-chains, …
Large-scale chainstate storage
Big Data analytics
Layer 2 Network: Lightning, Raiden
Proof-of-Stake, POET, PBFT, …

Bitcoin: DC
Hyperledger: CS
Ethereum: DC*(S?!)*

Investigate **potential use cases**
Choose and **tune** the right platform
Develop **reusable middleware**

Blockchain 1.0: Currency

Over 13700 public cryptocurrencies available!

Research for 1.0 Apps

Formally analyze the *security* model of Bitcoin

- 51% or less attacks: feather forking, selfish mining, …
- Attacks on: mining pools, currency exchanges, …
- Anonymity and fungibility of cryptocurrency and transactions

Conduct *performance modelling*

- Simulate various Bitcoin scenarios
- Understand impact of network topologies (e.g. partitions)

Develop *scalable* mechanisms with *legacy support* to maintain the *sustainability* of Bitcoin

- SegWit2x
- Bitcoin-NG (NSDI '16)
- Off-chain (Lightning network)
- Algorand (SOSP '17)

Blockchain 2.0: Decentralized Apps

ETHEREUM

Đapps

ĐApps are applications built on blockchain platforms using smart contracts (e.g. Ethereum)

GNOSIS

Forecast market (e.g. betting, insurance)

EtherTweet

Decentralized Microblogging

Token Distribution

Crowdfunding

alice

Charity donation payment

Research for 2.0 Apps

Formal *verify* smart contracts, detect and repair security flaws

- Ethereum Vyper

Develop *scalable consensus* mechanisms which support *smart contracts* in an *public* network (w/ *incentives*)

- Proof-of-Stake (Casper)
- Side-chain (Plasma)
- Sharding (ShardSpace)

Develop *efficient data storage* techniques to store *smart contracts* and the *chainstate*

- AVL+ (Tendermint)
- Merkle Patricia Trees (Ethereum)
- Zero-Knowledge Proofs: zk-SNARK

Blockchain 3.0: Pervasive Apps

Applications involve entire industries, **public sector**, and IoT.

everledger

Diamonds Provenance

FACTOM

Land Registry in Honduras

BlockchainHealth

Electronic Health Records

VOTEWATCHER

Transparent Voting System

Research for 3.0 Apps

Develop *"clean-slate"* scalable distributed ledgers:

- Permissioned ledgers (Hyperledger Fabric)
- Blockless DLTs (IOTA Tangles, R3 Corda Notaries, Hashgraph DAG)

Develop *blockchain modelling tools and middleware*

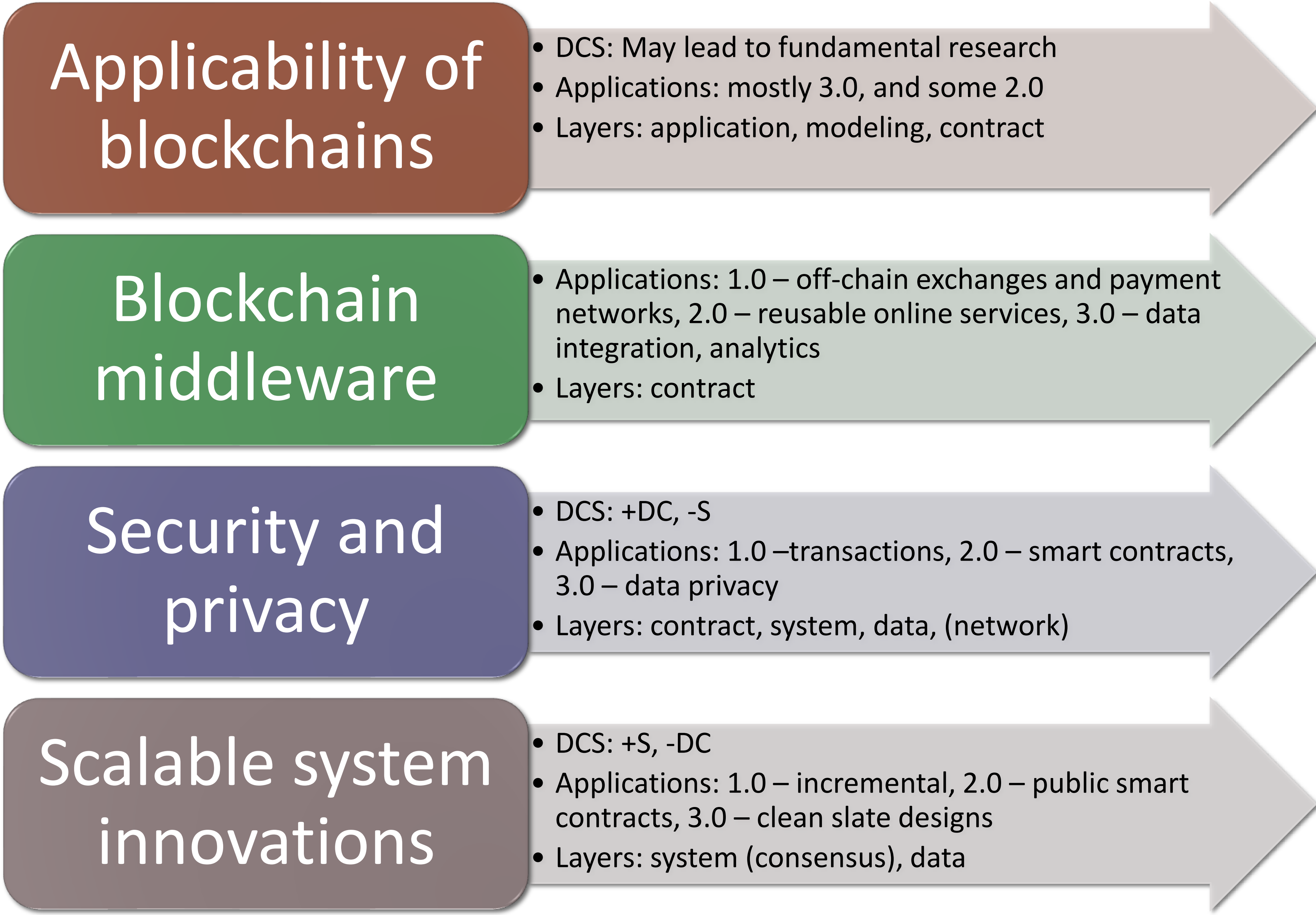
- BPMN, Business Artifacts with Lifecycles, FSM
- Authentication, reputation, auction, voting, etc.

Support strict *governance, security, and privacy* requirements

- State channels
- Endorsement policies

Overcome the *cyber-physical barrier for data entry*:

- "Garbage in, garbage out"
- Object fingerprinting
- Secure hardware sensors



Applicability of blockchains

- DCS: May lead to fundamental research
- Applications: mostly 3.0, and some 2.0
- Layers: application, modeling, contract

Blockchain middleware

- Applications: 1.0 – off-chain exchanges and payment networks, 2.0 – reusable online services, 3.0 – data integration, analytics
- Layers: contract

Security and privacy

- DCS: +DC, -S
- Applications: 1.0 –transactions, 2.0 – smart contracts, 3.0 – data privacy
- Layers: contract, system, data, (network)

Scalable system innovations

- DCS: +S, -DC
- Applications: 1.0 – incremental, 2.0 – public smart contracts, 3.0 – clean slate designs
- Layers: system (consensus), data