



BLOCKCHAIN

Welcome at SfeirSchool
Blockchain 200





Training usages

When can we go on break?



Agenda

1. LEDGER

- Introduction
- Improvements

2. DISTRIBUTION

- Introduction
- Collaboration
- Double spending
- States & Transitions
- Identification
- Security



Agenda

3. OPEN DISTRIBUTION

- Synchronisation (Review)**
- Self-regulating system**

4. USE CASES

- Fund management**

LEDGER

- ❑ Introduction
- ❑ Improvements



1. Ledger: Introduction

What is a ledger ?



Example: “Supervising sales”

Ledger

IDENTITY

Name: Mary KAGHAN

Job: Baker

Company: Au Bon Pain

Employees's Num.: < 3 persons



CONTEXT

Mary's company is a little family firm which is not structured to track sales.

NEED

Mary needs to know the volume of sales, the sold product quantity, and partial sales performance per day.



Example: “Supervising sales”

Ledger

How to effectively monitor sales?



Example: “Supervising sales”

Ledger

BRAINSTORMING

-
- simple solution
 - Need History
 - => keep order
 - which data to track ?
 - Forgery proof



Example: “Supervising sales”

Ledger

- Intuitive solution : “**Ledger**”

Simple solution	✓
Keep order	✓
Elastic data storage	✓
Forgery Proof	✓



Example: “Supervising sales”

Ledger

0.	Product	Time	Price	Q.
1	Pain	06:23	1,20	20
2	Croissant	06:30	0,90	15
3	Pain	06:45	1,20	18
4	Croissant	07:30	0,90	17
5	Croissant	07:55	0,90	13
6	Pain	10:23	1,20	10
7	Croissant	10:45	0,90	7
8	Sandwich	11:12	3,50	12
9	Sandwich	12:05	3,50	23
10	Pain	12:12	1,20	15
11	Pain	13:07	1,20	13
12	Pain	15:34	1,20	5
13	Pain	17:21	1,20	11



Ledger implementation

Ledger

How to implement a ledger ?

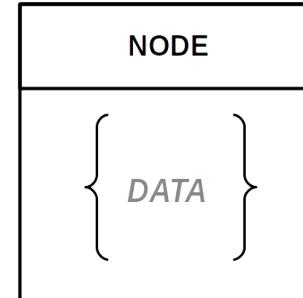


Ledger implementation

Ledger

Rule 1: One node represents a register line

1	Pain	06:23	1,20	20
---	------	-------	------	----

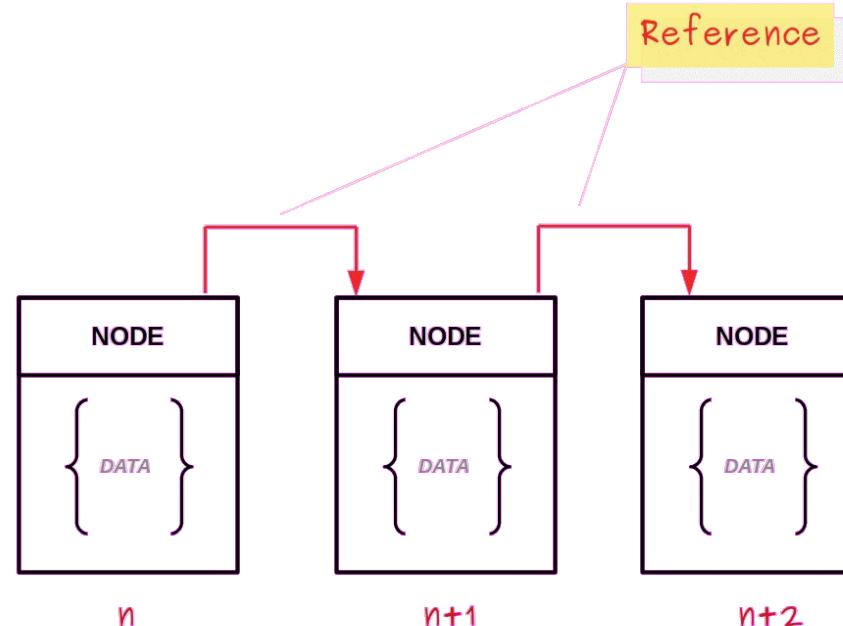




Ledger implementation

Ledger

Rule 2: Each node must be linked to the rest (In order)





Ledger implementation

Ledger

Rule 3: The last inserted node is the most recent





Ledger: Exercises

Statement

```
# git clone https://github.com/ech54/sfeir-schools.git  
# cd 01_introduction_bc  
# docker-compose up --remove-orphan
```



Ledger: Exercises

Statement

File Edit View Insert Cell Kernel Widgets Help Not Trusted | Go ○

Run Cells

Run Cells and Select Below

[sf≡ir] Run Cells and Insert Below

Run All

Run All Above

ucture

Objective : The fir
transactions.

Step 1

Cell Type

Current Outputs

All Output

all golang statements.

Notes :

SfeirSchc

Exercice 1 : (

reate a simple Blockchain structure to register three



Ledger: Exercises - 1

Statement

“Create a ledger”



Ledger implementation

Ledger

Is it safe ?



Ledger: Exercises - 2

Statement

“Unique Identifier”

1. Based on last solution find a better solution to build a unique identifier
2. *(Owner signature)*

NOTE: Think about cryptographic ...



Ledger: Sum-up

- Properties:

- History, Order, Forgery-Proof

- Implementation:

- Chain of blocks (nodes)

- Solution limits:

- In-memory context, mono thread,
self-ownership on your data

CONTEXT

Who is the data master ?

- *I'm the Master of Data*

What is ledger context ?

- *Private*



1. Ledger: Improvement

Ledger in real life ...



Example: “AM Laboratory ...”

Improvement

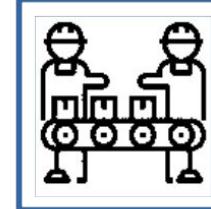
IDENTITY

Name: John WHITEKER

Job: Quality Control Ingeneer

Company: AM Laboratory

Employees's Num.: < 50 persons



CONTEXT

John is working for AM Lab firm. He controls car parts from greatest automotive factories. Automotive growth will increase the number of parts to control per day ...

NEED

Firm comity want to invest on custom Ledger. In few months, 50 persons will input the result of their control into custom Ledger.



Example: “AM Laboratory ...”

Improvement

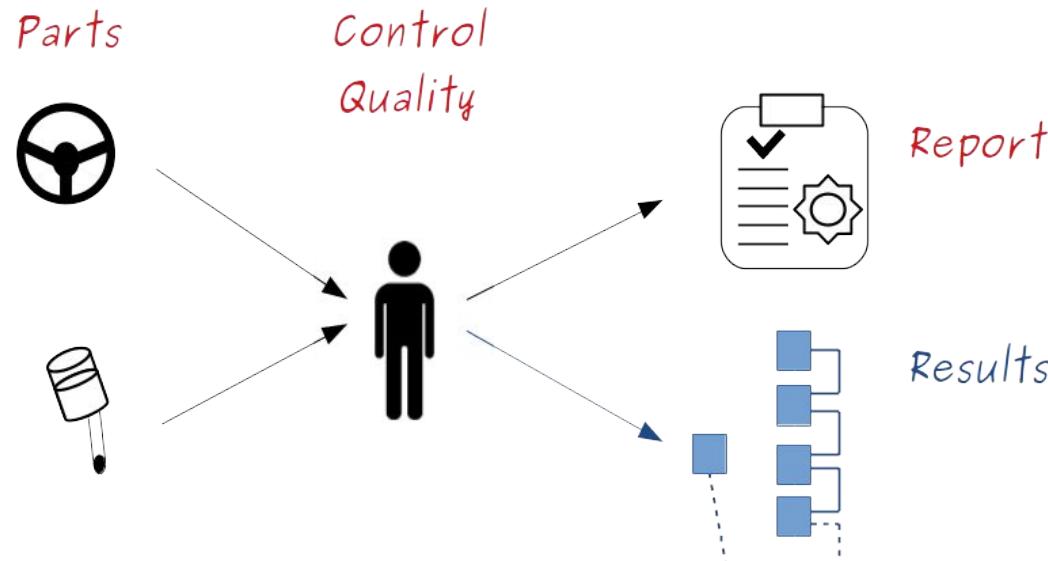
How to use a Ledger in John’s context ?



Example: “AM Laboratory ...”

Improvement

- Expected solution description:





Example: “AM Laboratory ...”

Improvement

Can we reuse our first proposal implementation in the John context?



Example: “AM Laboratory ...”

Improvement

BRAINSTORMING





Example: “AM Laboratory ...”

Improvement

Should we review the first Ledger model ?



Review Model ?

Improvement

- Ledger will contain **quality report result** :
 - Unique tx identifier
 - Last tx identifier
 - Creation date
 - Parts reference
 - Control status [Ok / Nok]
 - Controller reference

NO



Storage Target ?

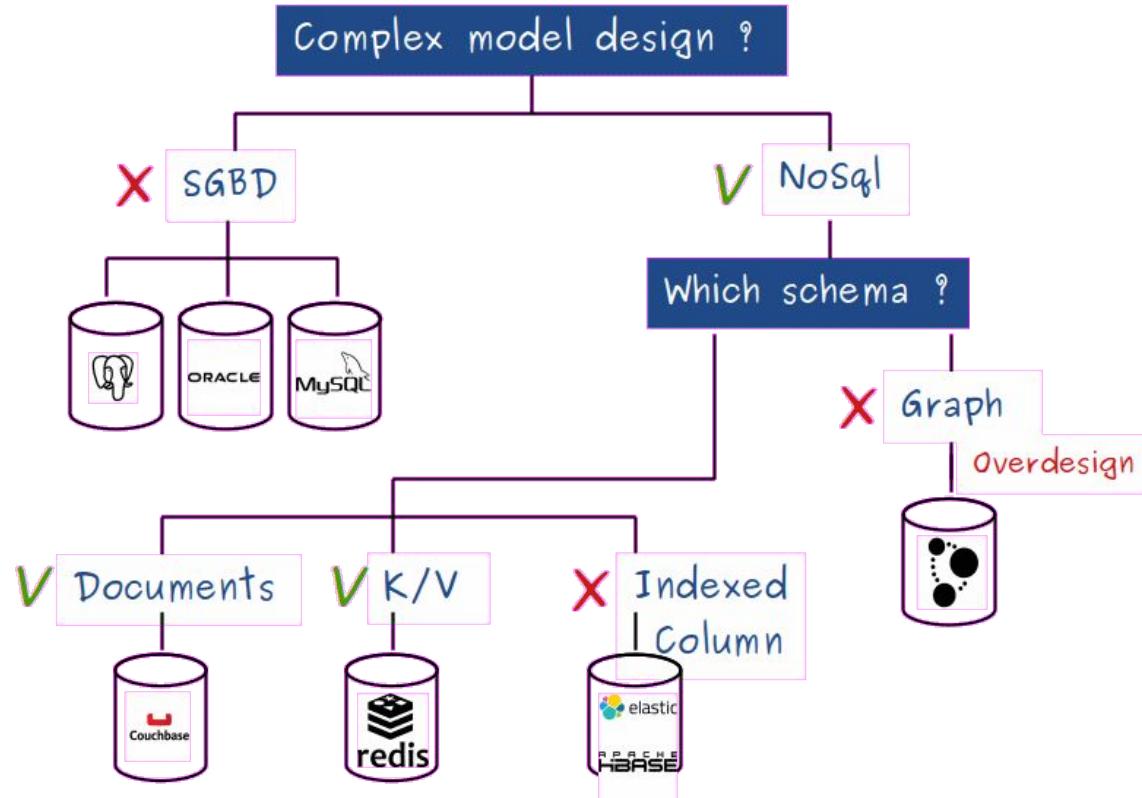
Improvement

Which storage could we used to store Ledger ?



Storage Target ?

Improvement





Number of users ...

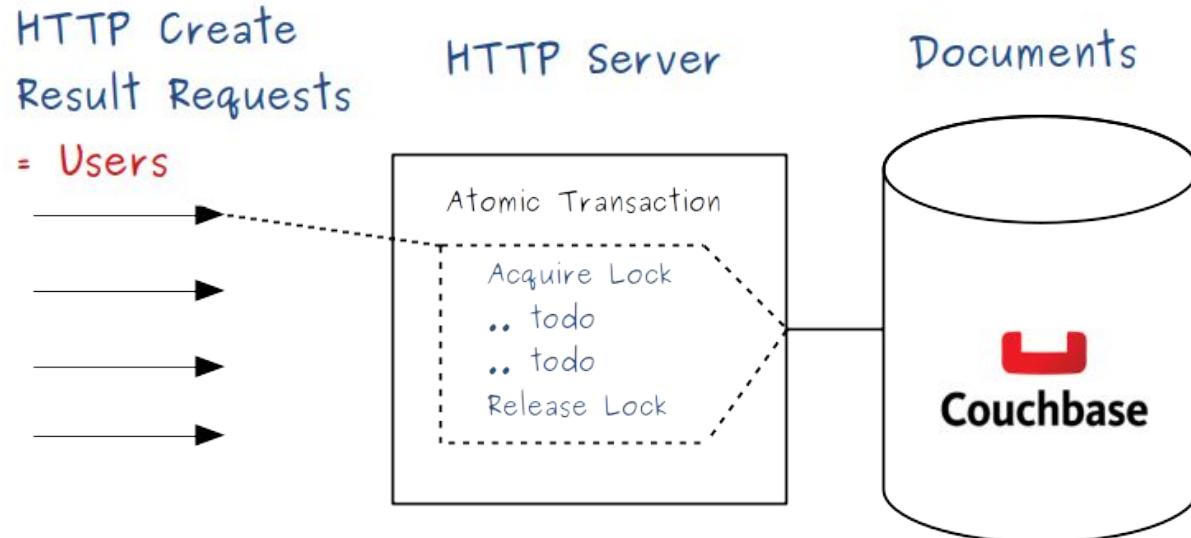
Improvement

How to handle multiple users ?



Number of users ...

Improvement





Number of users ...

Improvement

Which is the limit of solution?



Number of users ...

Improvement

- Lock model has a cost on **performance**
- Lock **synchronizes** all requests



Performance

Improvement

How to improve solution performance ?



Example: “Supervising sales”

Ledger

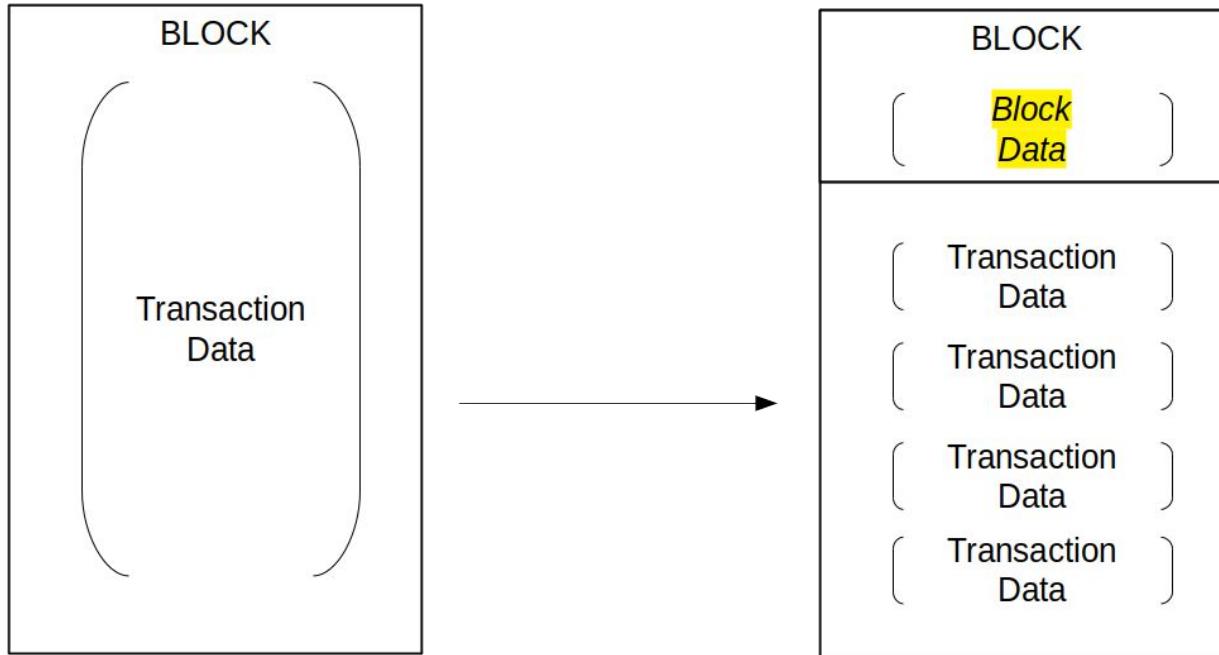
- Intuitive solution : “**Multiplex n transactions per Block**”

Simple solution	<input checked="" type="checkbox"/>
Commit n transactions in storage	<input checked="" type="checkbox"/>



Performance

Improvement





Performance

Improvement

- Block data header:
 - Unique identifier, Last block identifier, timestamp
- Block data body:
 - **n Transactions**
 - Transaction identifier and data



Performance

Improvement

Issues

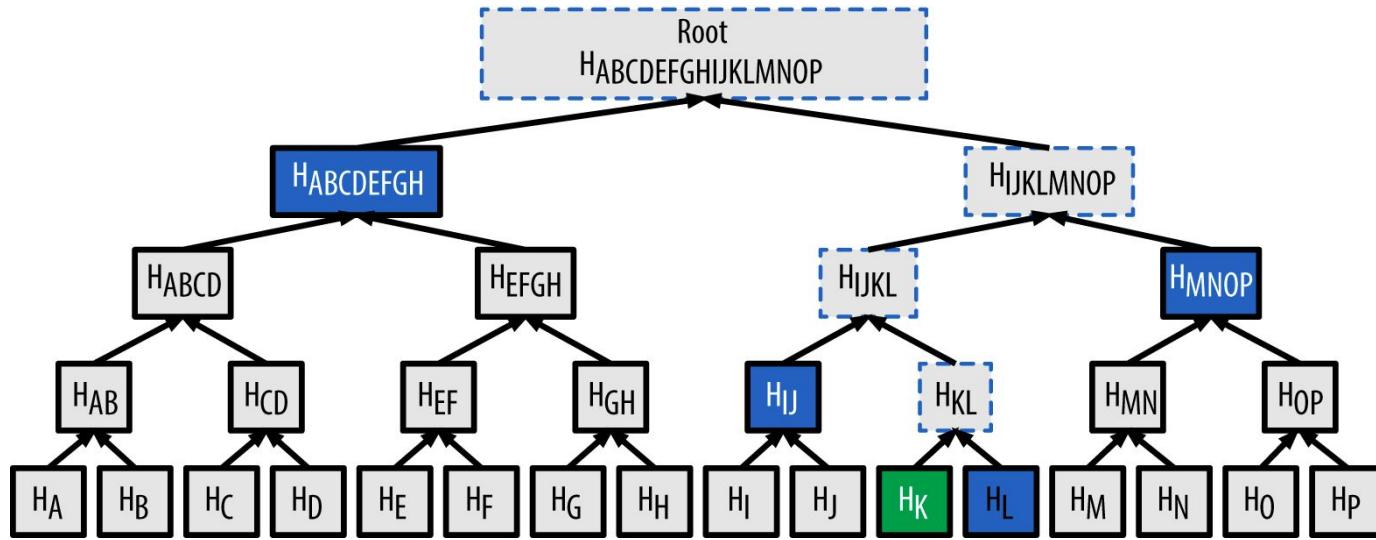
- How to keep order ?
- How to keep forgery-proof property?



Performance

Improvement

- Solution : “**Merkle tree**”





Performance

Improvement

- “**Merkle tree**” principles
 - Each transaction is hashed
 - Each hash and transaction is stored on K/V storage
 - Both hashes are hashes step by step until the root hash



Performance

Improvement

- “**Merkle tree**” properties
 - Order can't be modified
 - Quick search in using the transaction hash
 - Alteration is not possible



Performance: Exercises - 3

Statement

“Review the block implementation to handle multiple transaction”

1. Think about K/V storage for transactions
2. Think about hashing for key generation
3. Hash twice key together until create a root key



Ledger improvements: Sum-up

- Persistence:

Document NoSql solutions (Key / Values)

- Transaction load:

Pessimistic system limitless

Multiplex transactions per block (Merkle tree)

CONTEXT

Who is the data master ?

- *I'm the Master of Data*

What is ledger context ?

- *Private*

DISTRIBUTION

- Introduction**
- Collaboration**
- Double spending**
- States & Transitions**
- Identification**
- Security**



2. Distribution: Introduction

Why distribute a ledger ?



Example: “Outsourced accounting”

Distribution

IDENTITY

Name: Mike MATH

Job: Business accountant

Company: « See my shoes »

Employees's Num.: < 200 persons



CONTEXT

Mike is working for « See my shoes » company. His job is essential to provide driving tools for his job. Accounting is outsourced, Mike used Ledger to return all accounting evidences to an external recipient.

NEED

Mike's need is to improve the exchange between his service and this recipient ...



Example: “Outsourced accounting”

Distribution

How to use accounting Ledger ?

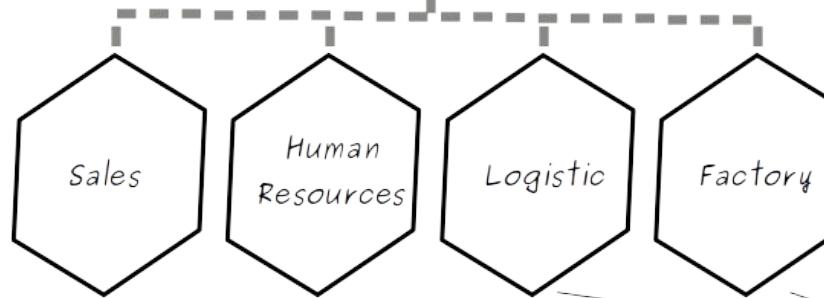
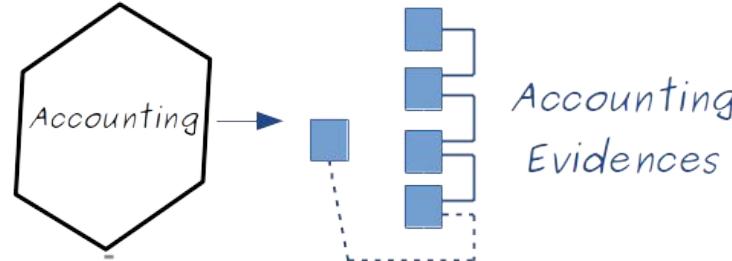


Example: “Outsourced accounting”

Distribution

1 - COLLECT EVIDENCES

Accounting internal process



Provide expense
Furniture
amortization ...



Example: “Outsourced accounting”

Distribution

BRAINSTORMING

Who is
data
master?

Need
READ Only

Distribution
channel

=>

Data governance
Security



Solution ?

CSV file



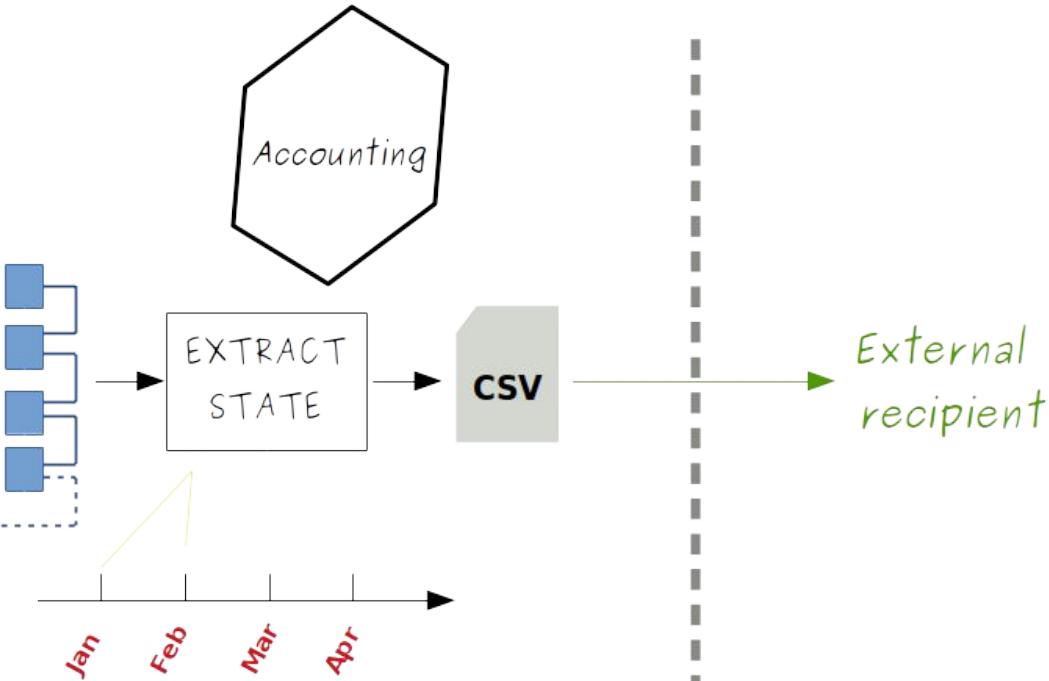
Example: “Outsourced accounting”

Distribution

2-DISTRIBUTE
EVIDENCES

Accounting
Evidences

Accounting external process





2. Distribution

Introduction

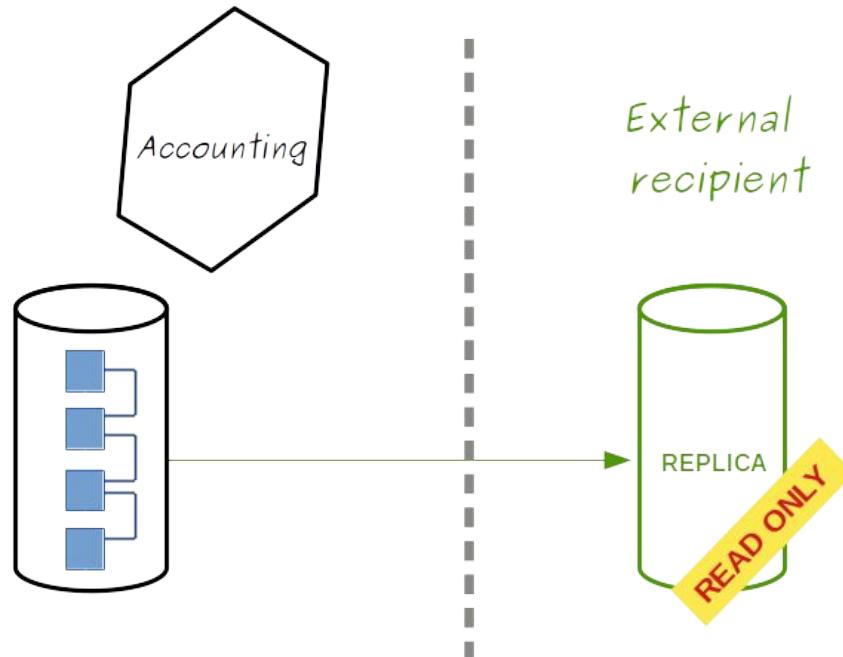
How can we improve distribution
of our Ledger copy ?



2. Distribution

Replication

Accounting external process (review)





2. Distribution

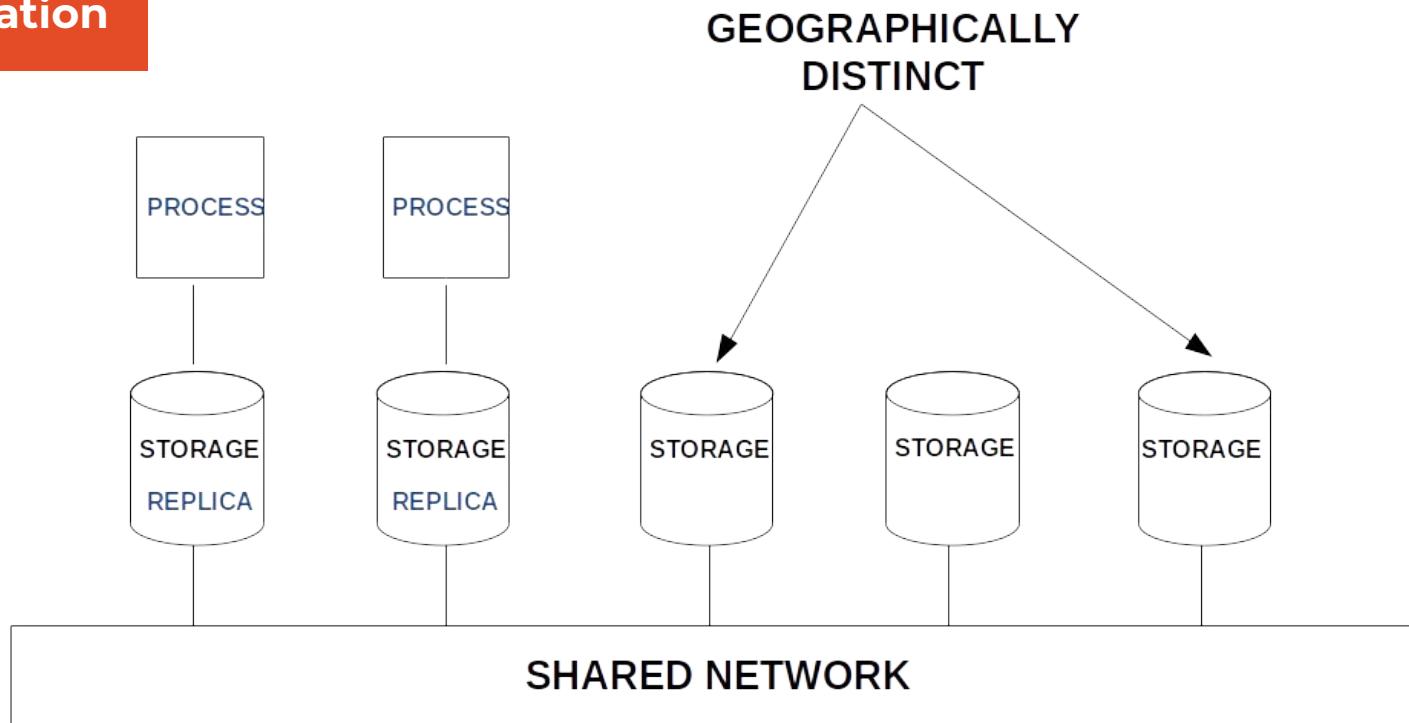
Replication

- Standard “**Replication**” mechanism
 - First step to implement a distributed system or **distributed service**
 - Systems are geographically distincts
 - Network or channel is shared



2. Distribution

Replication





2. Distribution

Replication

Why replication is interesting ?



2. Distribution

Replication

- “Transparency”

- Hide the fact that **process** and **resource** are physically distributed across multiple computers

- “Performance”

- Solution for **performance degradation issues**
- Better availability (Load Balancing)



2. Distribution

Replication

How is it working ?



2. Distribution

Replication

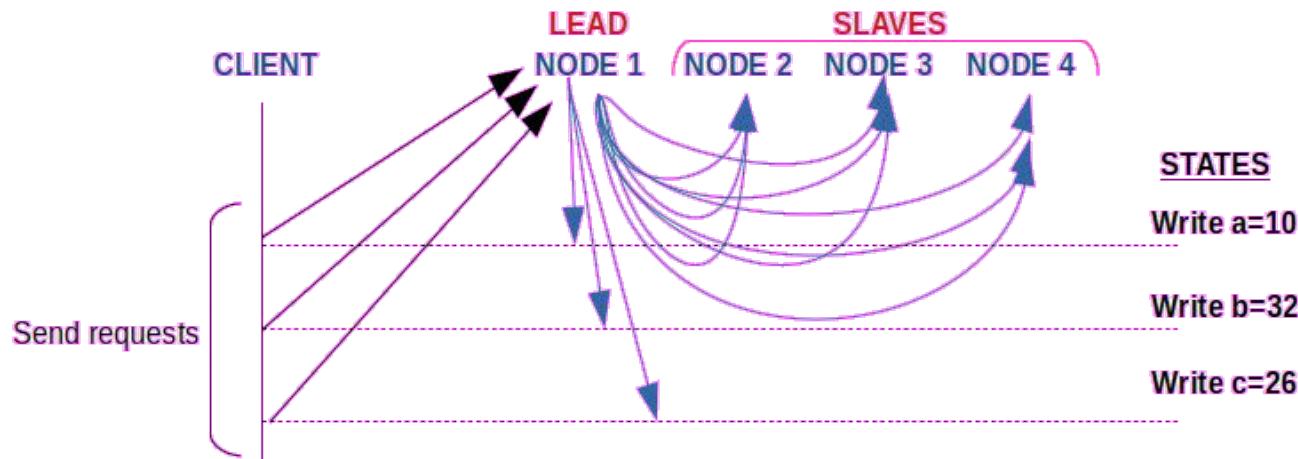
- First mode: “**Passive**” (or primary backup)
 - 1 replica execute all operations issue by a client
 - Periodically replica push states to other replica
 - Other replicas apply update



2. Distribution

Replication

- Example

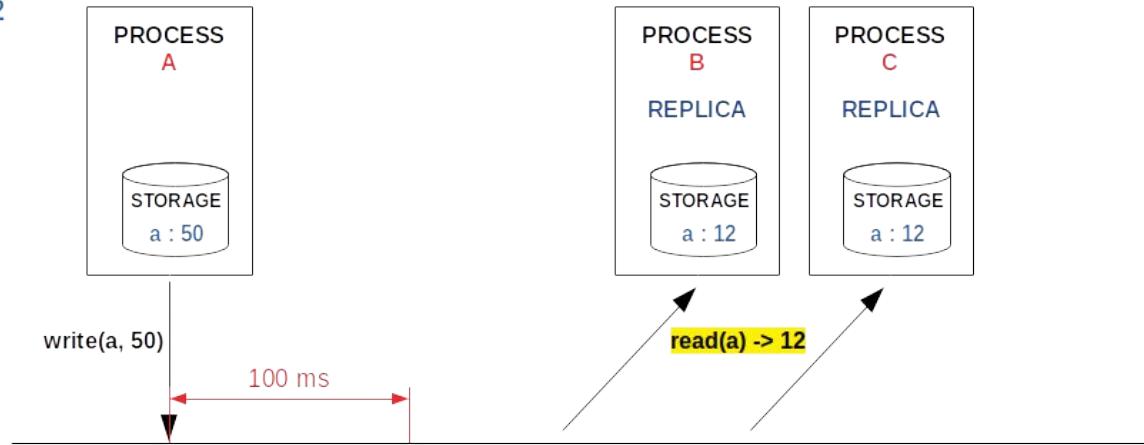




2. Distribution

Replication

Initial value
a : 12



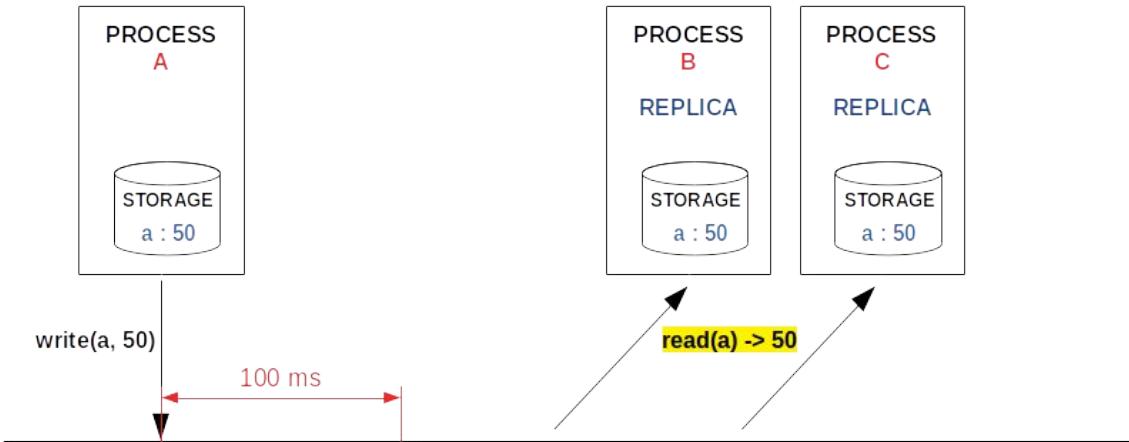
INCONSISTENT REPPLICATION



2. Distribution

Replication

Initial value
 $a : 12$

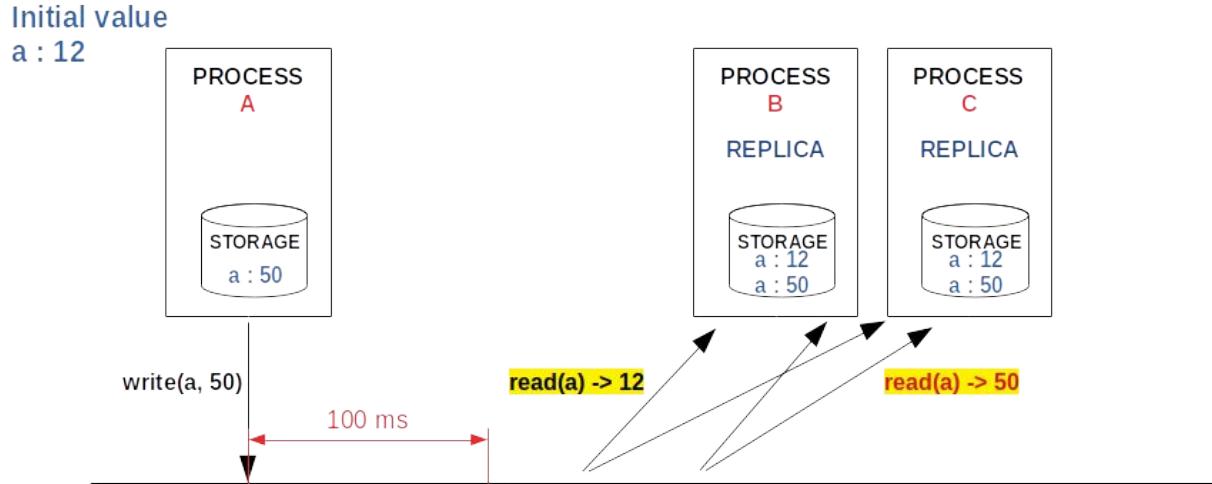


WRONG REPPLICATION



2. Distribution

Replication



WEAK REPPLICATION



2. Distribution

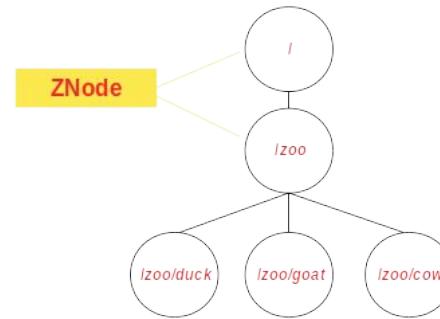
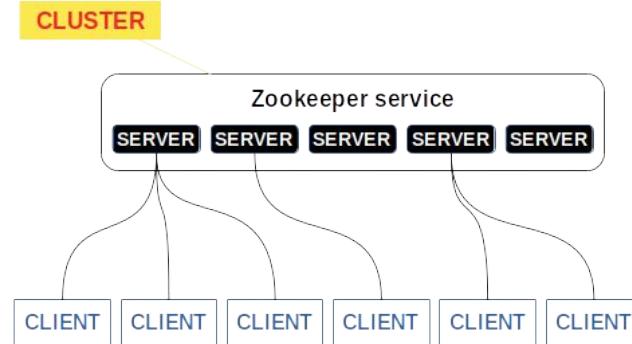
Replication

**Example of distributed service
with “Zookeeper Apache”**



2. Distribution

Replication

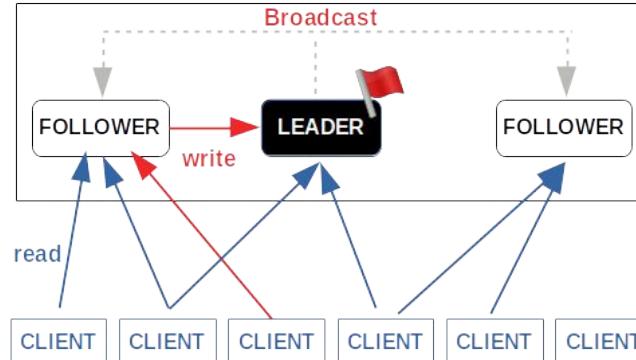


ZK ARCHITECTURE

DATA MODEL

Operation	Description
create	New node in specified path
delete	Remove node from a path
exists	Check for a path if a node is existing
getChildren	Get all children from a node
getData	Get data linked to a node
setData	Save data into node fields
sync	Synchonize client view from Zk

ZK OPERATIONS



PASSIVE REPPLICATION



2. Distribution

Replication
“passive”

- “**Leader election**”
 - Each replica looks for active leader
 - Replica informs new replica that a Leader is existing
 - Each replicat synchronizes his states with Leader
- “**Quorum**” constitutes a majority of replicas
 - Minimum of replicas to store a new state
- “**Atomic broadcast**”

2. Distribution

Replication
“passive”

- “**Consistency**” issues
 - Modifying a copy makes that copy different from the others
- “**Replica**” or follower
 - Look and Inform for existing leader
 - Synchronize his states with Leader orders
 - Participate to Leader election



2. Distribution

Replication
Failures

What are the issues in case of failure ?



Distribution “introduction”: Sum-up

- 1 Data Master :

Only one participant writes, and others participants consume data

Private context

- Replication :

Communication way

Transparency

Consistency, Fault tolerance

CONTEXT

Who is the data master ?

- *I'm the Master of Data*

What is ledger context ?

- *Private, and Ledger is distributed by myself*



2. Distribution

Sharing business



2. Distribution

Collaboration

Who is data master ?



2. Distribution

Collaboration

IDENTITY

Name: John Smith

Job: Software Engineer

Company: Rick Star Game

Employees's Num.: < 200 persons



CONTEXT

John's company is a game software editor, which decided to develop a poker game online for multi-players.

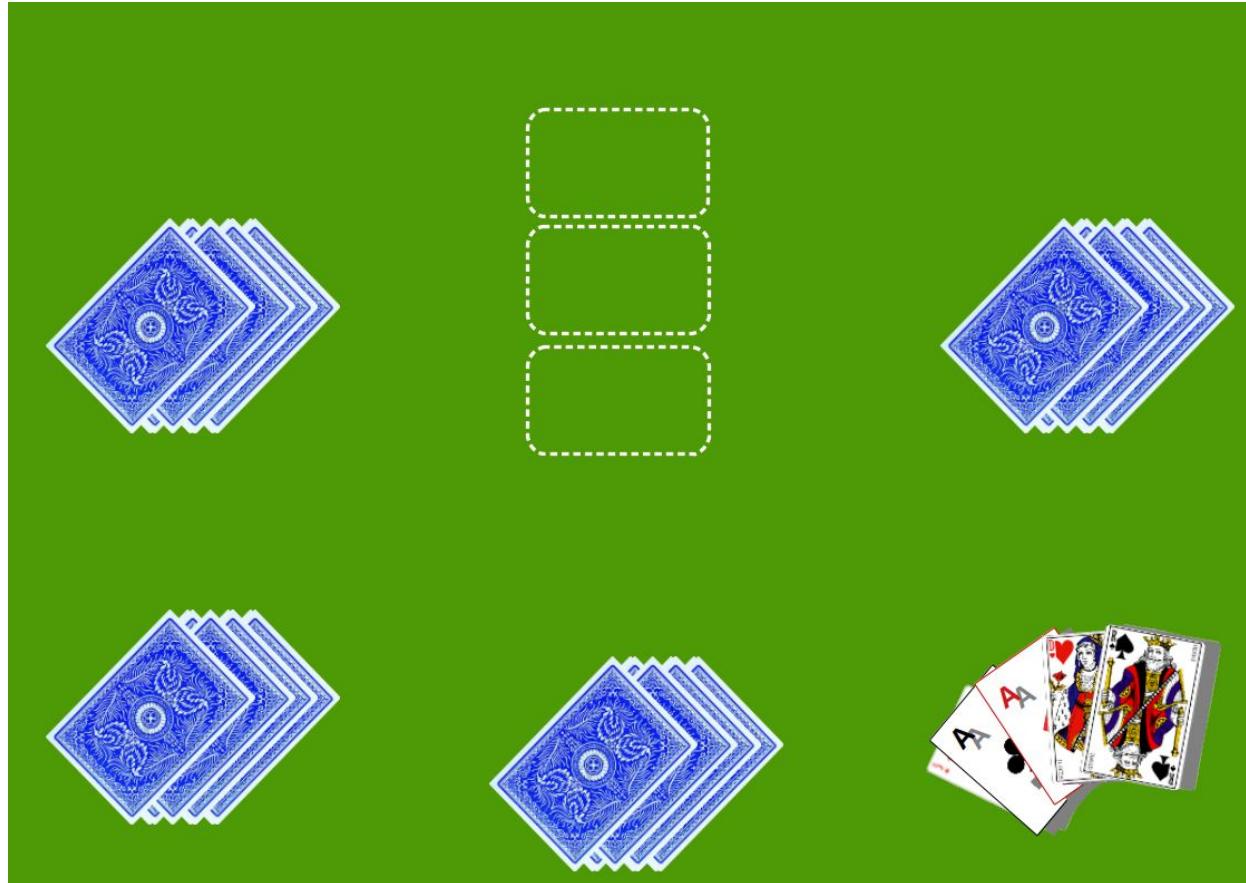
NEED

Each player client will be an autonomous ledger, the need is to synchronize the state of each client.



2. Distribution

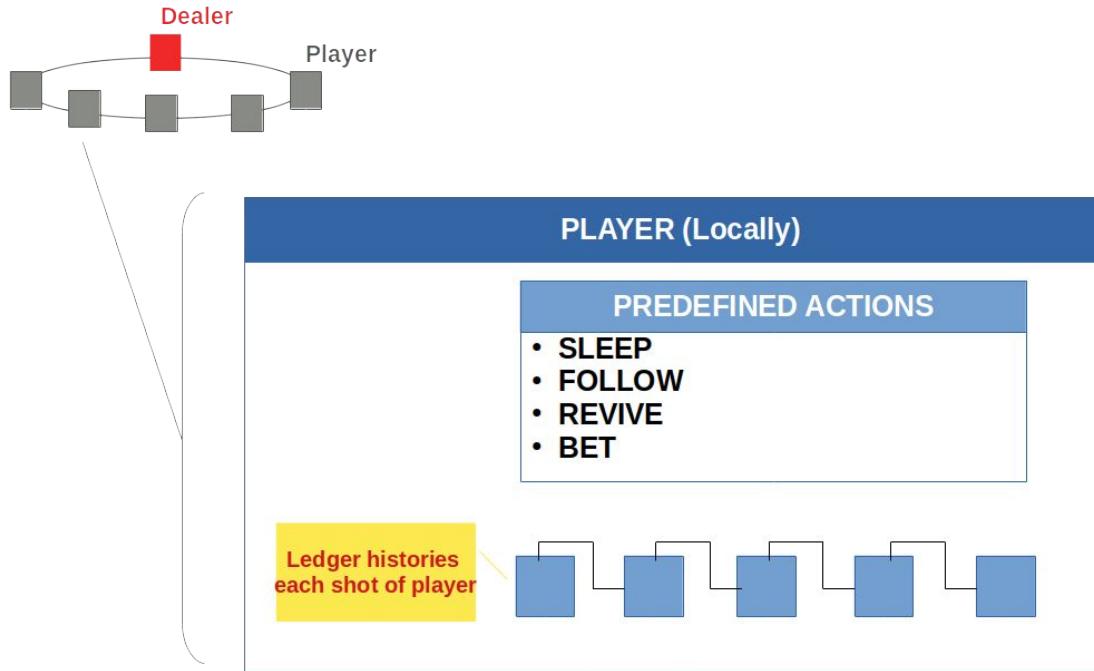
Collaboration





2. Distribution

Collaboration





2. Distribution

Collaboration

1	Distribute	20:00	D>A	5
2	Distribute	20:01	D>B	5
3	Distribute	20:02	D>C	5
4	SLEEP	20:02	C>D	n/a
5	BET	20:03	A>D	2

...

8	PAY	20:05	D>A	6
---	-----	-------	-----	---



2. Distribution

Collaboration

How to synchronize each player ledger ?

2. Distribution

Replication
“active”

- Second mode: “**Active**” (or SMR)
- All commands are sent to all replicas
- Replica need to
 - Share same INITIAL STATE
 - Execute deterministic operations
 - Coordination
 - Replica number or Quorum



2. Distribution

CONSENSUS

Example of coordination: “Practical Byzantine Fault Tolerance”

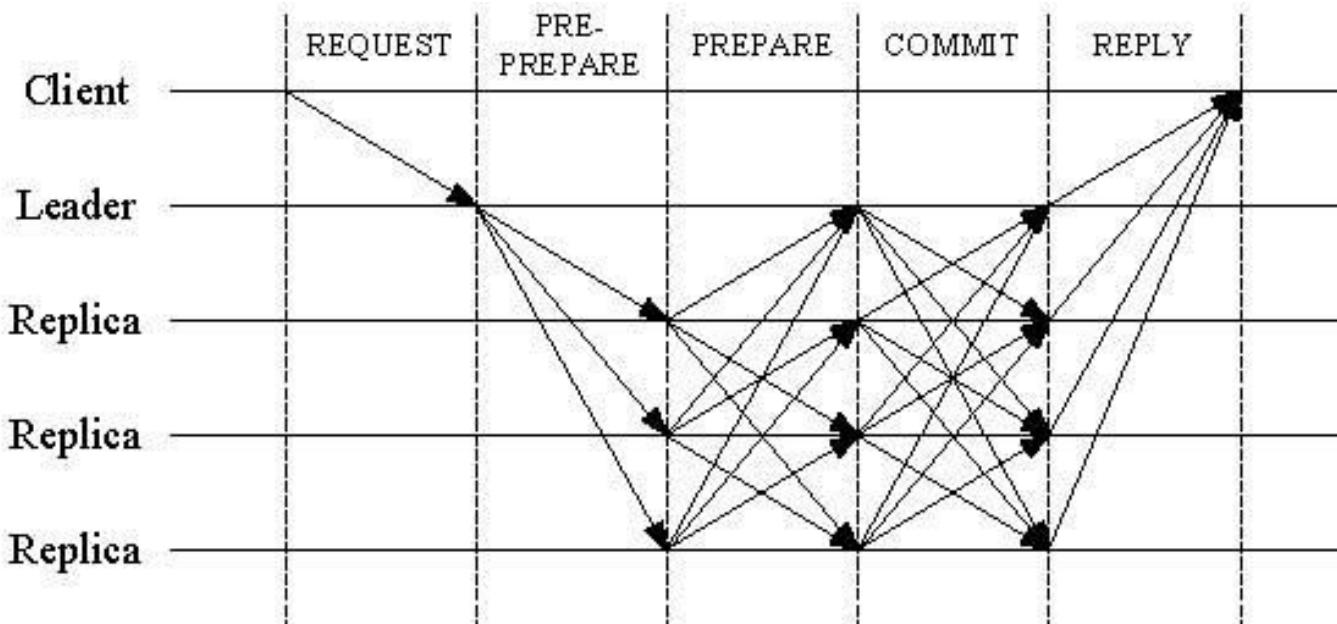
1. **REQUEST:** A client send a request to the Leader
2. **PRE-PREPARE:** Leader delivers the request to other replica
3. **PREPARE:** Each replica “simulates” with local state the request and broadcast results
4. **COMMIT:** A voting allows to accept the request
5. **REPLY:** The accepted result is replied to client



2. Distribution

CONSENSUS

Example of coordination: “Practical Byzantine Fault Tolerance”





Distribution “collaboration”: Sum-up

- n data master :

Each participant can read/write

Private context

- Replication :

Consensus algorithm is depending of usage context

Predefined actions (Game Theory)

ex: PBFT handles failures, request coordination
(decentralized algorithm)



Distribution “collaboration”: Sum-up

- “**ORDER-EXECUTE**” architecture style
 - ORDER transactions
 - BROADCAST (propagate transactions to other peers)
 - EXECUTE transaction sequentially and locally



Distribution “collaboration”: Sum-up

- **“ORDER”**

Based on CONSENSUS (BFT, PoW, Pos)

- **“BROADCAST”**

Based on ACTIVE replication

- **“EXECUTE”**

Handled by a SMR (State Machine Replication)

CONTEXT

Who is the data master ?

- Several Master of Data

What is ledger context ?

- Private, and Ledger is distributed by a network



2. Distribution

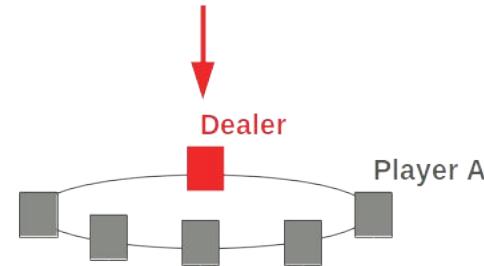
Oups ...

Dealer has paid twice same player !



2. Distribution

Double spending



TRANSACTIONS

Dealer → Player A

Dealer → Player A

LEDGER

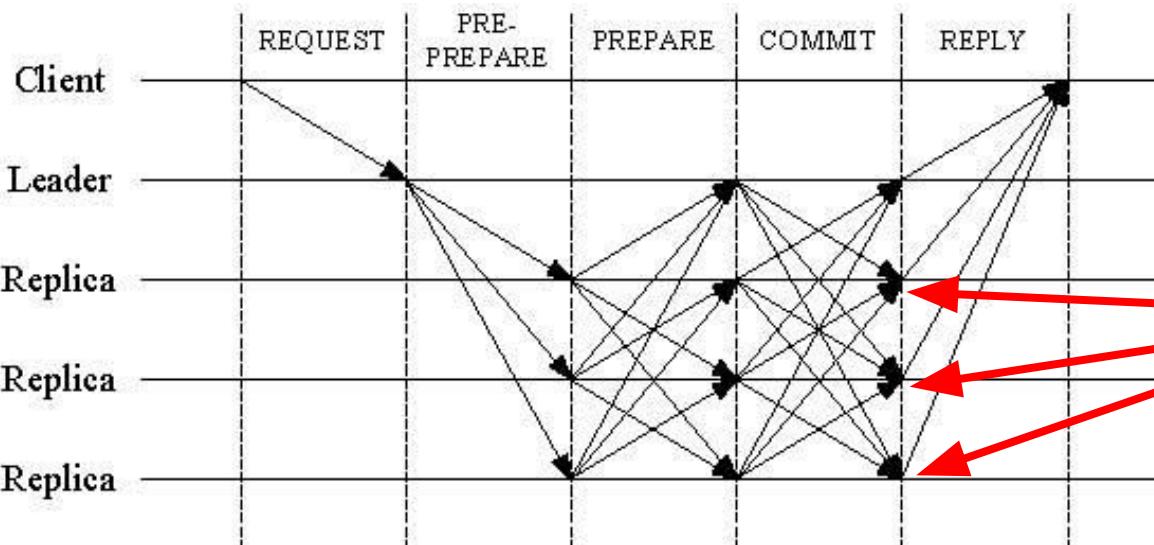
8	PAY	20:05	D>A	6
---	-----	-------	-----	---

9	PAY	20:06	D>A	6
---	-----	-------	-----	---



2. Distribution

Double spending



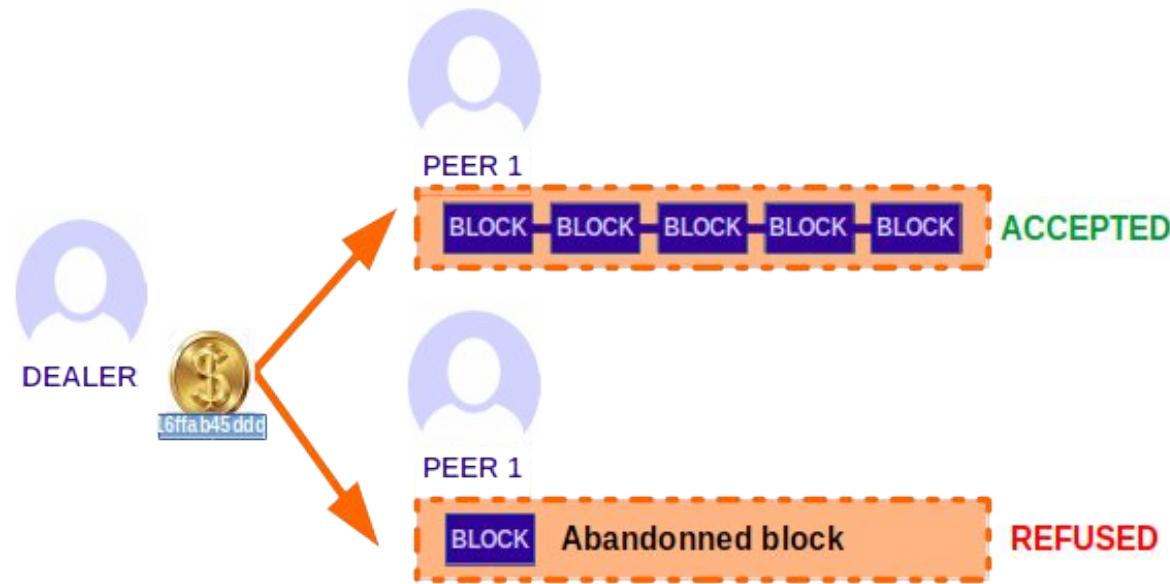
*Wait confirmation
from other
“replica” !!!*



2. Distribution

Double spending

Historic of transaction allows to solve the double spending issue





Distribution “double spending”: Sum-up

- Voting:
Replica inserts transaction in its local ledger based on
“n” replica confirmations
- Data Safe:
Oldest accepted blocks are safe

CONTEXT

Who is the data master ?

- Several Master of Data

What is ledger context ?

- Private, and Ledger is distributed by a network



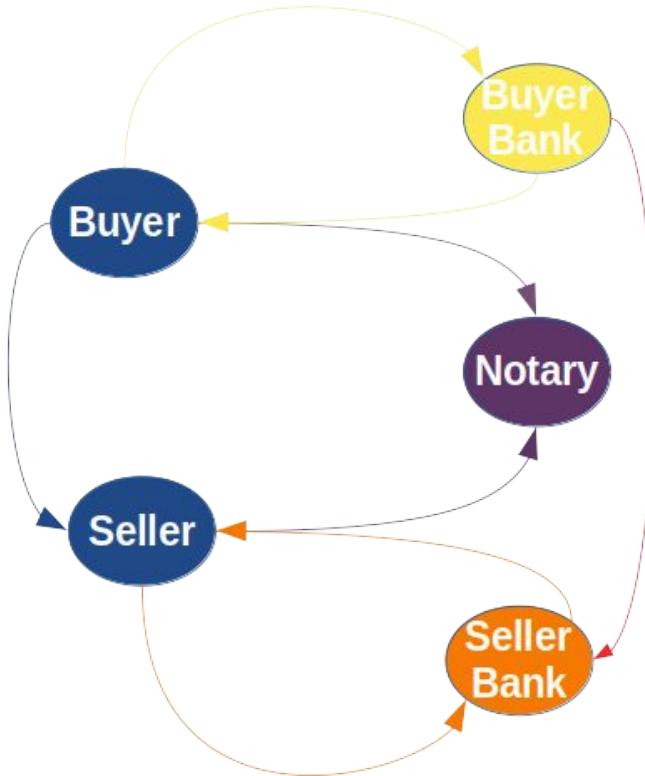
2. Distribution

a distributed state machine ...



2. Distribution

States
Transition





2. Distribution

States
Transition

What's a state machine ?



2. Distribution

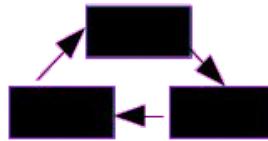
States
Transition



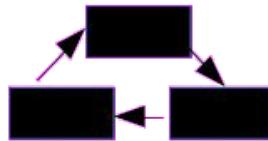


2. Distribution

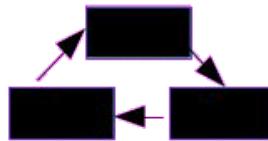
States
Transition



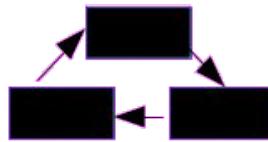
Buyer - Seller



Buyer – Buyer's Bank



Buyer – Seller – Notary



Seller's Bank – Seller



2. Distribution

States
Transition

Ledger, is it a state machine ?



2. Distribution

States
Transition

- Blockchain is a State Machine
- **STATES** are based on **transactions history**
- Some DLT offers extension to change state machine : **SMART CONTRACT**



Distribution “States-Transitions”: Sum-up

- States:
Deduced based on all ledger transactions
- Transitions:
Blockchain State machine
Behavior can be customized = SMART CONTRACT

CONTEXT

Who is the data master ?

- Several Master of Data

What is ledger context ?

- Private, and Ledger is distributed by a network



2. Distribution

I'm not the owner of this transaction !!!



2. Distribution

Identification

How to manage non-repudiation ?



2. Distribution

Transaction

- **FLOW** between several participants
- Transaction is **DIGITALLY SIGNED** by the sender
- Transaction « transports » **DATA**

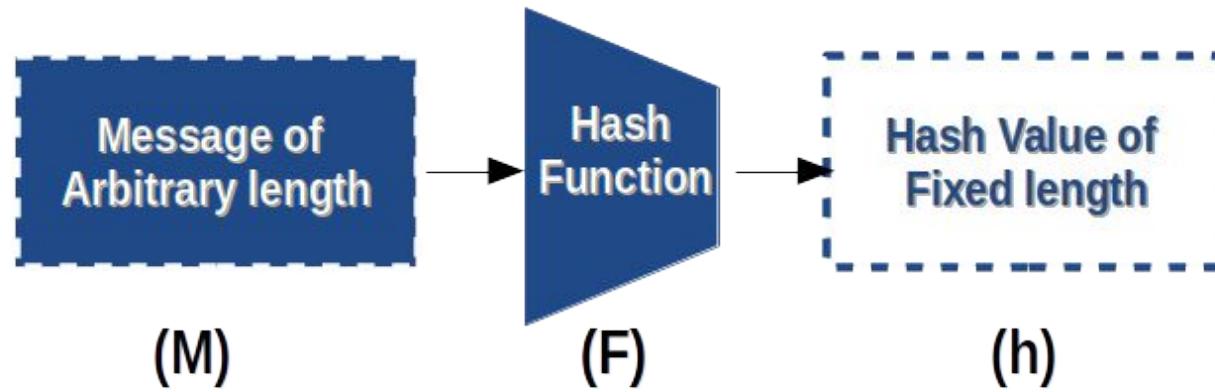




2. Distribution

CRYPTOGRAPHY

HASHING



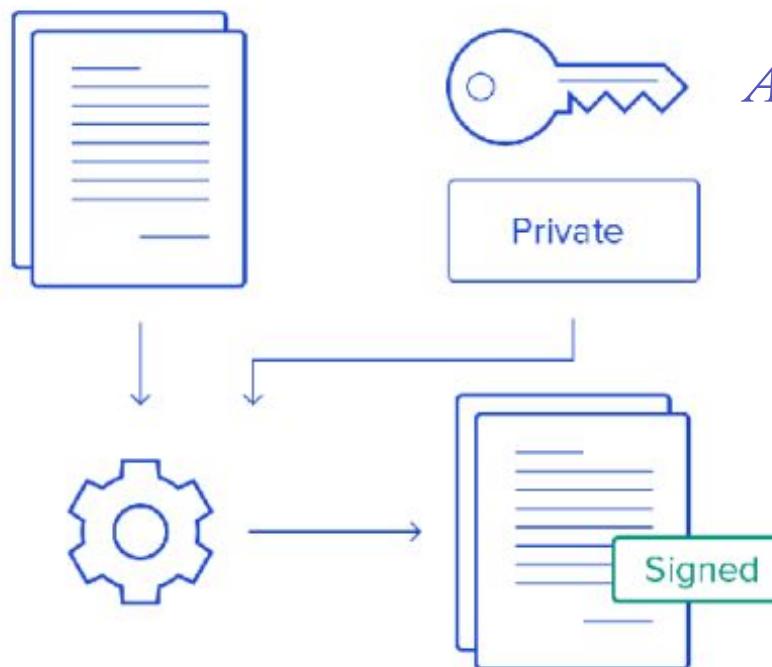


2. Distribution

CRYPTOGRAPHY

Digital Signature

SIGNING



*NON REPUDIATION
&
AUTHENTIFICATION*

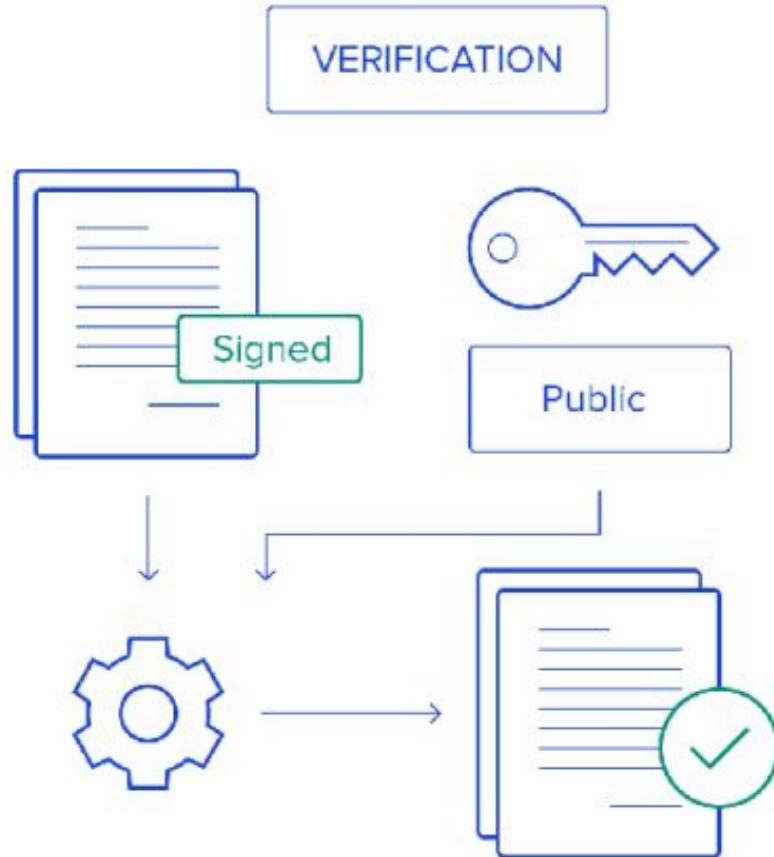


2. Distribution

CRYPTOGRAPHY

VERIFICATION

*CHECK
SIGNATURE*





Distribution “identification”: Sum-up

- Identification:

- 1 Certificate per participant

- Transaction with a Digital signature

- Non Repudiation:

- Transaction can't be revoked by a owner

CONTEXT

Who is the data master ?

- Several Master of Data

What is ledger context ?

- Private, and Ledger is distributed by a network



2. Distribution

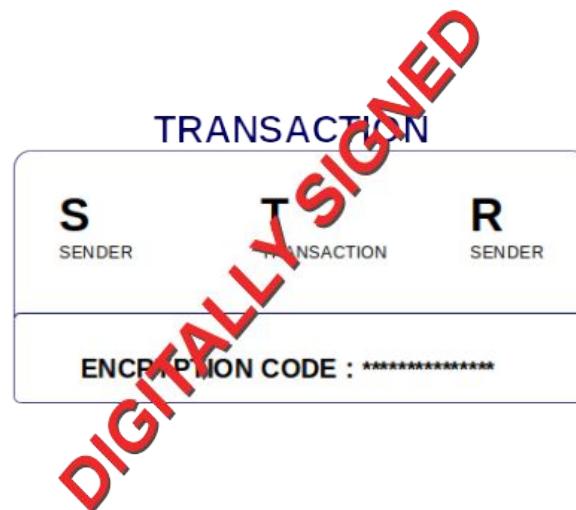
Ledger security



2. Distribution

SECURITY

- Against **FORGERY**
- NON REPUDIATION** of a transaction

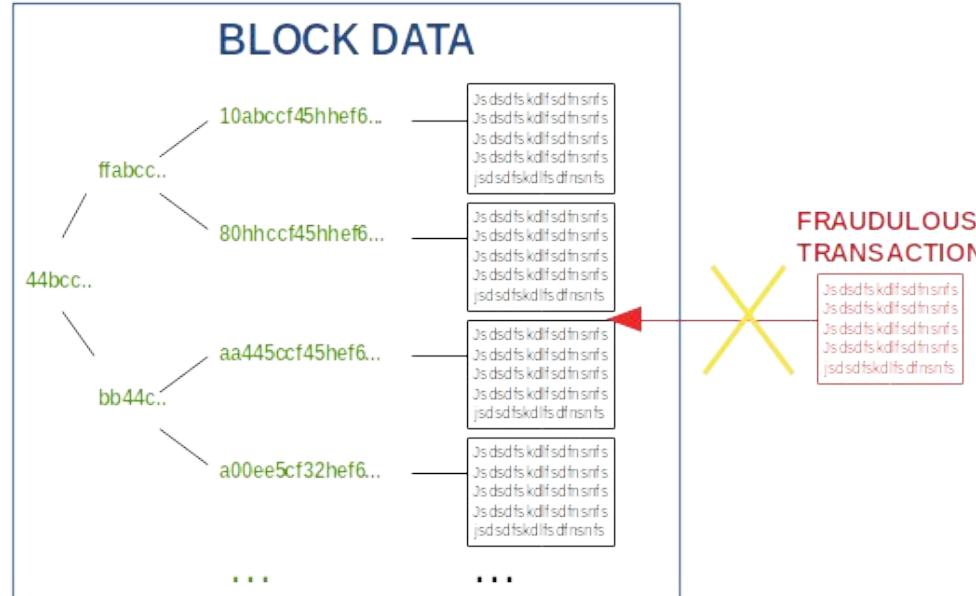




2. Distribution

SECURITY

- Can't insert **FRAUDULENT TRANSACTIONS**
- At **Block level**

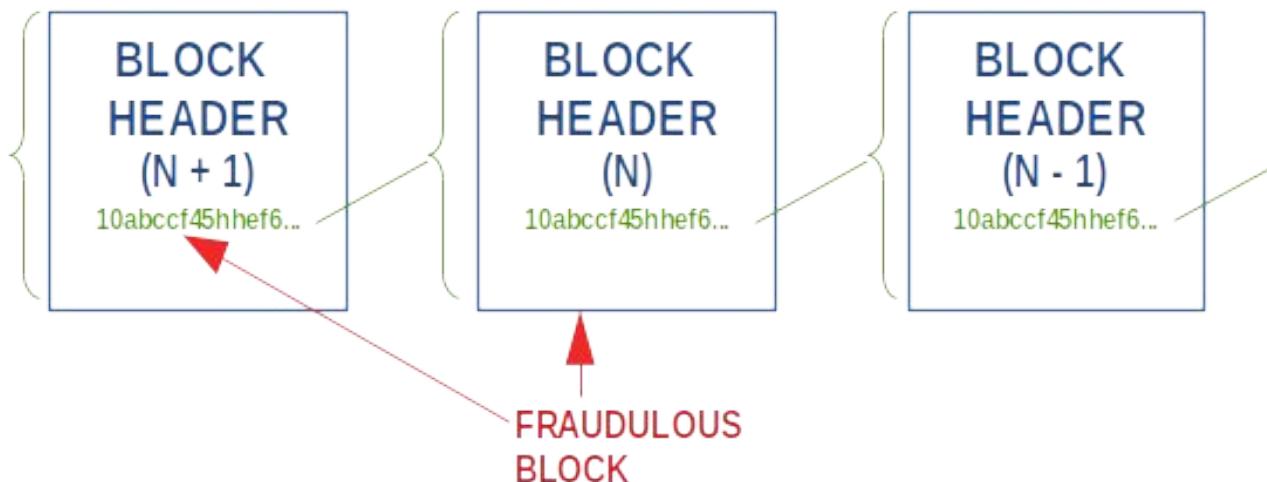




2. Distribution

SECURITY

- ❑ At Ledger level
→ The other block pointers must be recomputed



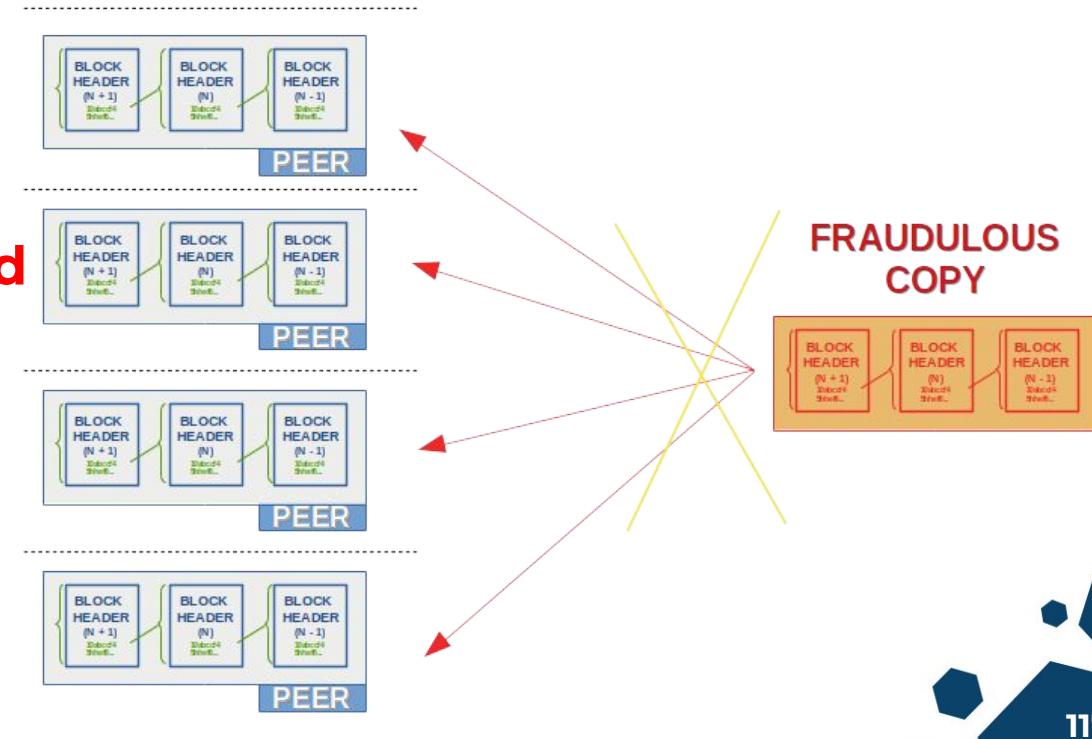


2. Distribution

SECURITY

At **Distributed Ledger level**

→ **The other peer ledger copies must be recomputed**





Distribution “security”: Sum-up

- Different levels:

Transaction

Block

Ledger

CONTEXT

Who is the data master ?

- Several Master of Data

What is ledger context ?

- Private, and Ledger is distributed by a network

OPEN DISTRIBUTION

- ❑ Synchronisation (Review)
- ❑ Self-regulating system



3. Open distribution

Synchronization [review]



3. Open distribution

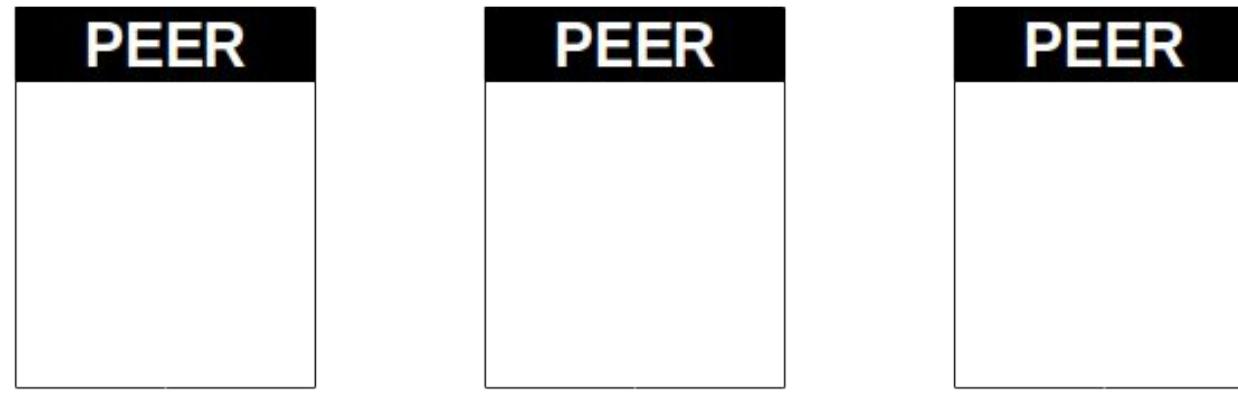
Consensus

- ❑ Example : Proof of work (PoW)
- ❑ Consensus objectives
 - Coordinate block diffusion (new state)
 - Participant can **commit** or **ignore** block
 - Temporize the block creation



3. Open distribution

Consensus



Number of participants

{1}

{1, n}

{n-1}

Role

PROPOSE
NEW
BLOCK

MINER

COMMIT
BLOCK



3. Open distribution

Consensus

❑ PEER roles

Propose new blocks

Mine blocks

Commit blocks



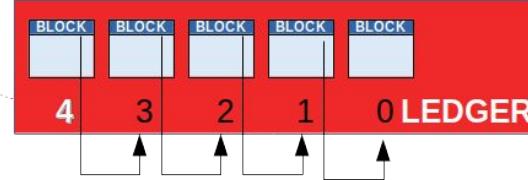
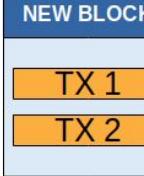
Consensus

PoW

NEW BLOCK
HOLDER

PEER

5



BROADCAST

GAME
+
NEW
BLOCK

PEERS

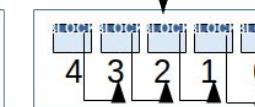
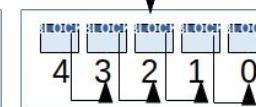
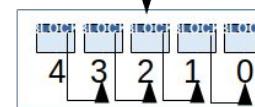
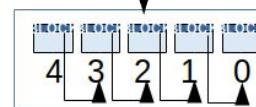
ACCEPT



PROPOSAL
SOLUTION

0b44d8123fab...
+ nonce

COMMIT





Open “consensus”: Sum-up

- ❑ PEER proposes new blocks
 - Link block to last ledger block hash
 - Hash blocks (difficulty)
 - Broadcast blocks and hash
- ❑ PEERs mine blocks
 - Solves the hash to find the nonce
 - Diffuse the matching solution
- ❑ PEERs accepts blocks
 - Verify solutions and blocks
 - Commits the block to local ledger

CONTEXT

Who is the data master ?

➤ Several Master of Data

What is ledger context ?

➤ Public, and Ledger is distributed by a network

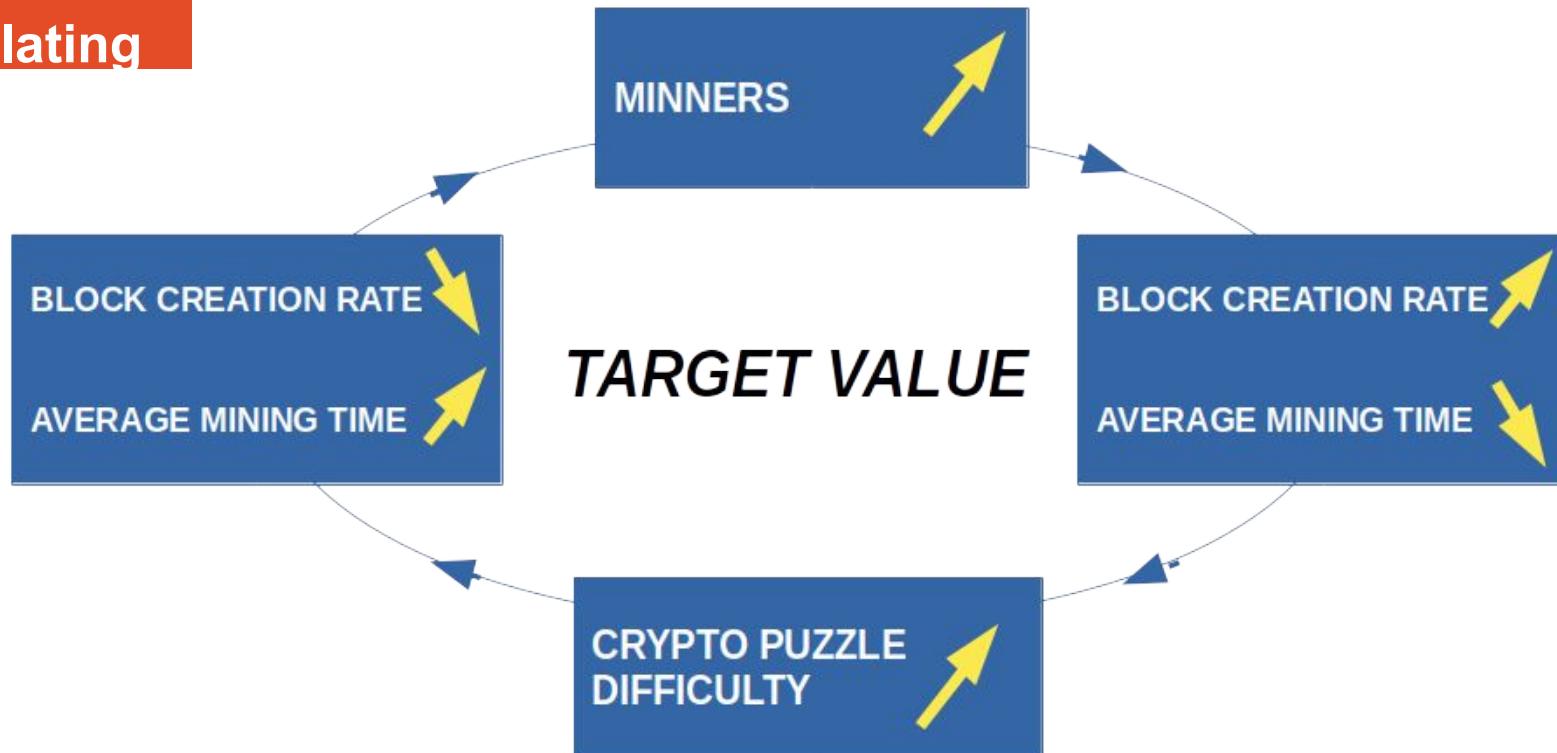


3. Open distribution

Self regulating system

3. Open distribution

Self
regulating



CONTEXT

Who is the data master ?

➤ Several Master of Data

What is ledger context ?

➤ Public, and Ledger is distributed by a network



3. Open distribution

Identification [review]

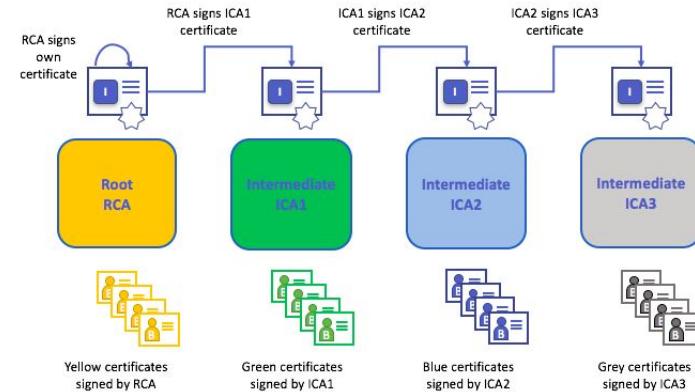
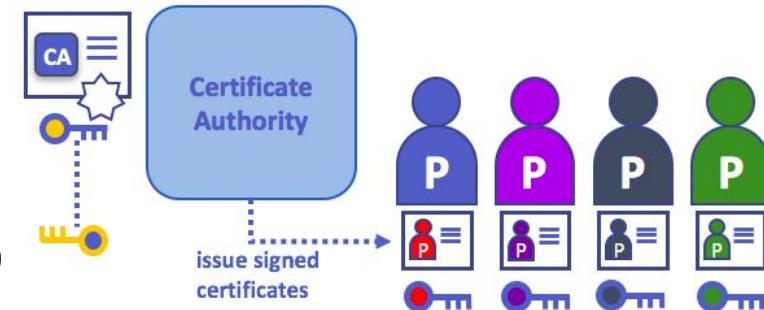
3. Open distribution

Identification

IDENTITY

CA (certificate authorities)

Chain of trusts



DLT SOLUTION REVIEWS



“Solution reviews”

STATE MACHINE	SMART CONTRACT	
DATA ORGANISATION	TRANSACTION	BLOCK LEDGER
DISTRIBUTION	PASSIVE ACTIVE SIMPLE CONSENSUS	ACTIVE COMPLEX CONSENSUS
CONTEXT	PRIVATE	PUBLIC
SECURITY	CERTIFICATE – PRIVATE KEY	

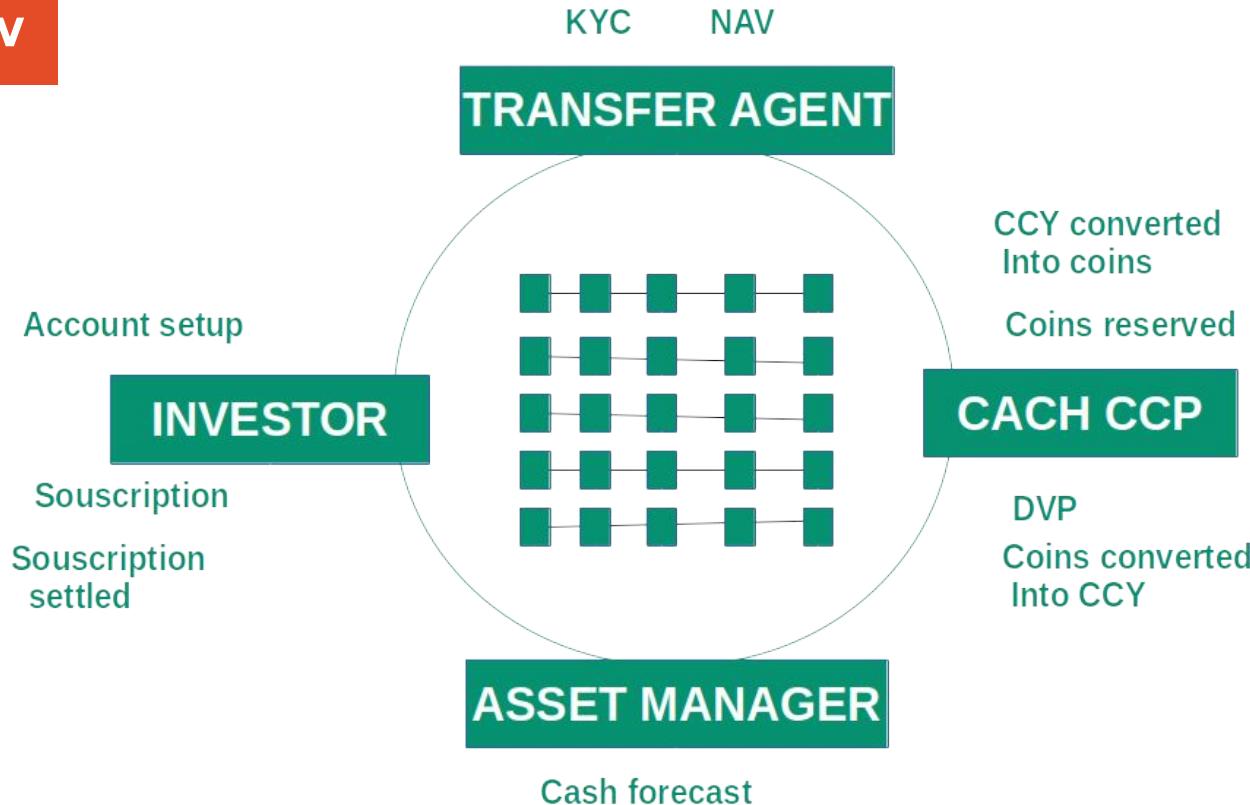
USE CASES

❑ Fund management



Fund Management “FundsDLT”

Overview



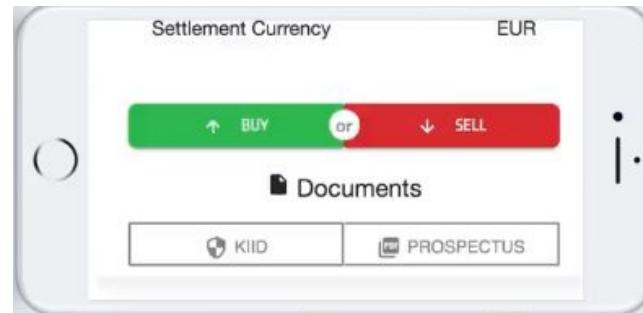
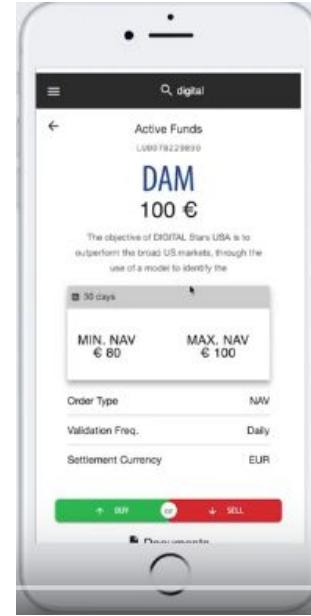


Fund Management “FundsDLT”

Overview

❑ Investor

Account setup
Subscription





Fund Management “FundsDLT”

Overview

Transfer agent

KYC

Nav

The screenshot shows a web-based application for managing KYC (Know Your Customer) data. On the left, there are two filter panels: 'AML/KYC Status' (with 'Validated' selected) and 'Risk level' (with 'All' selected). The main area displays a table of investors with the following columns: Name, ID, Type, Country, Documentation, PEP/AM, Sanction, Documents, Events, Risk Level, and Status.

Name	ID	Type	Country	Documentation	PEP/AM	Sanction	Documents	Events	Risk Level	Status	
Emma MØLLER	EM32499	Direct Investor	Denmark	Validity: 60% Completeness: 50%	New alert	No alert	10	Cloud	Orange	Accepted	<button>Reject</button>
Clarisse NICOLAS	CN38390	Direct Investor	France	Validity: 80% Completeness: 80%	No alert	No alert	10	Cloud	Green	Accepted	<button>Reject</button>
Emma ANDERSEN	EA42314	Direct Investor	Denmark	Validity: 80% Completeness: 80%	New alert	No alert	10	Cloud	Orange	Accepted	<button>Reject</button>
Julia GRAY	JG74384	Direct Investor	United Kingdom	Validity: 90% Completeness: 70%	No alert	New alert	10	Cloud	Orange	Accepted	<button>Reject</button>
Patricia SANTOS	PS37768	Direct Investor	Spain	Validity: 80% Completeness: 100%	No alert	No alert	10	Cloud	Green	Accepted	<button>Reject</button>
Estelle RODRIGUEZ	ER67425	Direct Investor	France	Validity: 80% Completeness: 60%	New alert	No alert	10	Cloud	Orange	Accepted	<button>Reject</button>

In the top right corner of the interface, there is a message: "Blockchain identity: Account unlocked! 0x4b03e166786622be888fb05491a140a3367175b" with a "Delete" button below it.



Fund Management “FundsDLT”

Overview

- ❑ Transfer agent

Thomas Lewis TL18104 Direct investor Luxembourg Validity: 100% Completeness: 70% No alert No alert ID Blocked Accept



Fund Management “FundsDLT”

Overview

❑ CASH CCP

Transaction ID	Entry ID	Order Type	Coin	Number of Fiat CCY	Transaction Date and Time	Status
SW3980212	TL1814	Buy	800	800	2016-12-01 10:47	Executed
SW3980213	HQ2376	Buy	500	500	2016-12-01 09:13	Executed
SW2341890	ZT3847	Buy	12000	12000	2016-12-01 11:34	Executed
SW9098123	FA1629	Buy	3400	3400	2016-11-01 08:37	Executed
SW3980213	UR1982	Buy	3000	3000	2016-12-01 09:13	Executed



Fund Management “FundsDLT”

Overview

❑ CASH CCP

Reserved Cash (-> Transfert Agent is alerted)
Coin conversion

Transaction ID	Entity ID	Order Type	Coins	Number of Fiat CCY	Transaction Date and Time	Status
SW3980212	TL1814	Buy	800	800	2016-12-01 10:47	Executed



Fund Management “FundsDLT”

Overview

□ Asset Manager

The screenshot displays the FundsDLT Asset Manager interface. On the left, a sidebar titled "Filter Criteria" lists "Sub Funds" such as "Active Funds - Convertible" and "GLOBAL FUND - EQUITY USA". The main area features a large image of a woman's face with colorful lines radiating from it, overlaid with a navigation bar: "Digital Assets", "Funds", "Programs", "Activity Log", "Logout". Below this is a "My user profile" section with details: "Arnold Hart", "arnold.hart@adem.com", "Digital Asset Management", "Assigned user role(s): Content consumer", and "Last time you logged in: 30 november 2016 at 5:19pm". A "Log out" link is also present. At the bottom, a table titled "EQUITY SELECTION" shows three rows of data:

Sub Fund Name	Share Class Name	ISIN	Name	Investor ID	Country	Currency	Number of Shares
Active Funds - Digital Stars USA	Active Funds - Digital Stars USA	LU0078229890	Clarisse NICOLAS	CN38390	France	EUR	10,00
Active Funds - Digital Stars USA	Active Funds - Digital Stars USA	LU0078229890	Emma ANDERSEN	EA42314	Denmark	EUR	43,00
Active Funds - Digital Stars USA	Active Funds - Digital Stars USA	LU0078229890	Patricia SANTOS	PS37768	Spain	EUR	275,50



Questions ?

