collective intelligence labs

How to unite all blockchain using standards

To unlock a Collaborative Mind: Uniting human wisdom with AI brain for better action.

what is Collaborative Mind?

Collaborative Mind is a symbiosis of collective intelligence and AI computational power.

Collaborative Mind uses AI to analyze human experience and input - creativity, sociality and intellect - to find patterns and reveal hidden opportunities.

With Collaborative Mind we can focus our efforts on the most promising and impactful projects across diverse fields, working towards a better society.

how Collaborative Mind works

Projections

Our vision of the future

Tokenized IP

X

Reputation

Impact from our actions

Verifiable identity

X

AI

Our brain to learn

Infrastructure

Our memory

Omnichain smart-contracts

projections / vision

Vision is our ability to see the future and act to make it real.

We express vision as projections of desired future and actions towards it, such as ideas, opinions and funding.

reputation / impact

We can learn from previous actions and outcomes to come up with new ideas and avoid mistakes.

Reputation shows how successful one is in projecting and implementing the vision, and how positive change they do.

AI / brain

AI augments our intelligence and helps us to make impactful and fruitful decisions.

It learns on our previous experience to find insights and recommend optimal action.

Infrastructure / memory

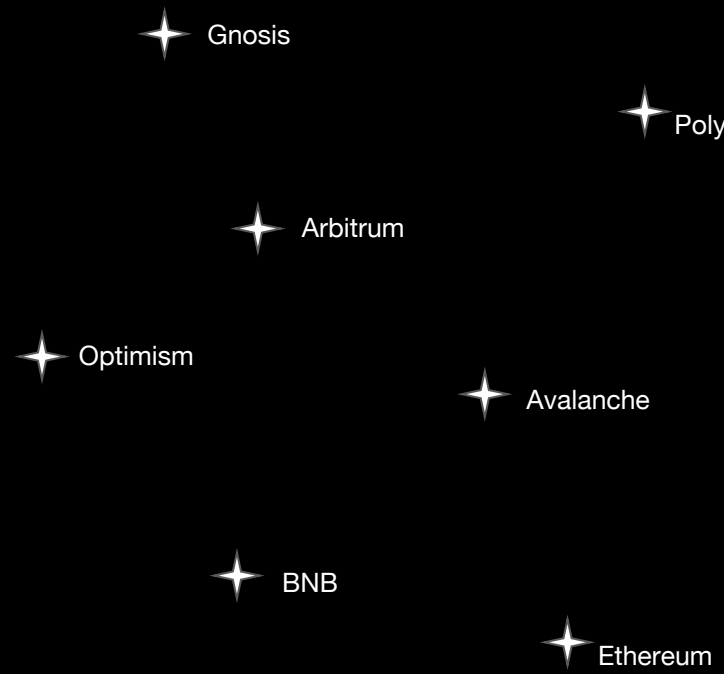
Omnichain smart-contracts

Work on all blockchains and use their combined security and scalability.

1. **Deploy once** and forget about specific chains.
2. **Efficient and secure** thanks to superb architecture.
3. **Fail proof:** automatic conflict resolution and easy rollbacks.

why

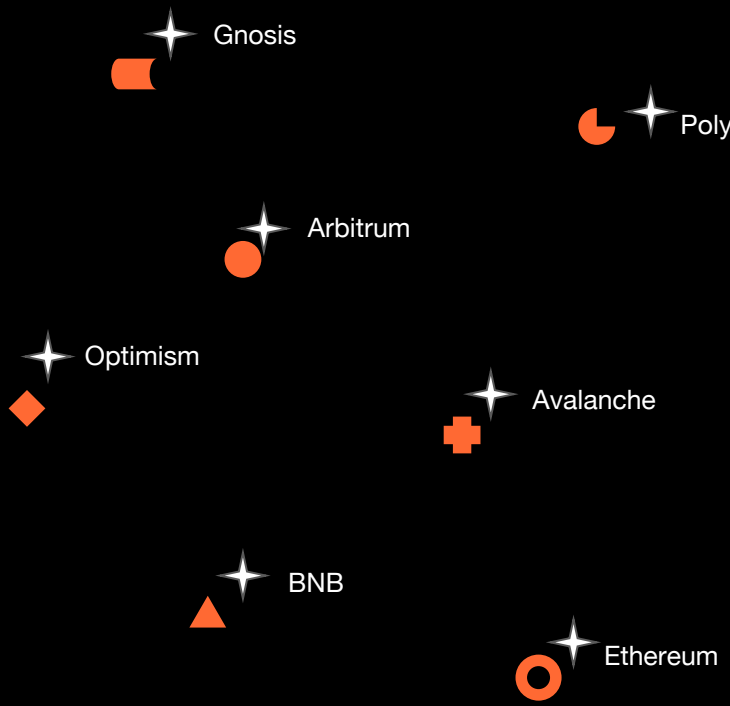
Web3 is fragmented into dozens of stand-alone, isolated blockchains, with a few limited options for cross-chain interaction.



In 2022 users lost $2 billion worth of crypto in bridge hacks, which makes 69% of all stolen funds that year.

problem

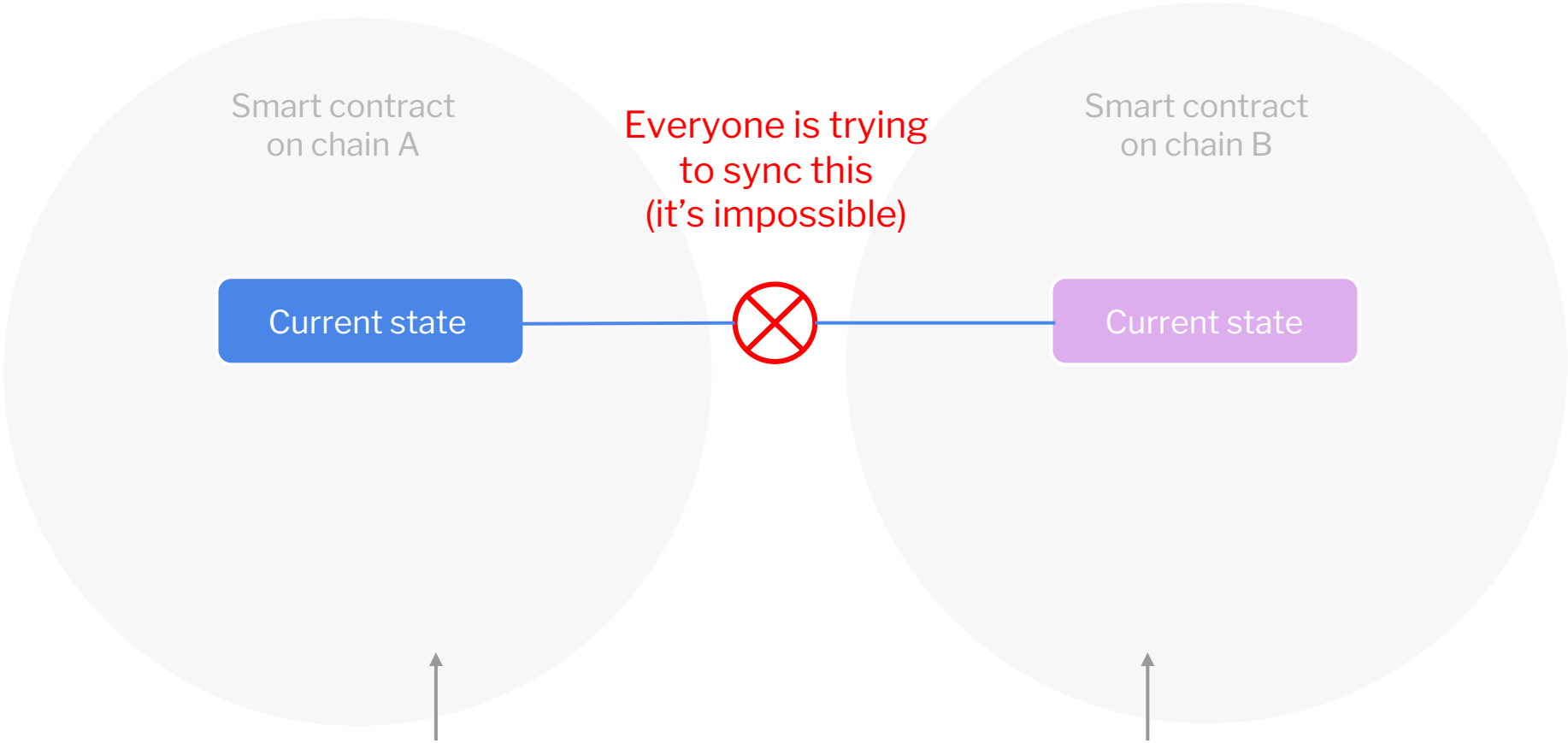
Smart-contract, or dApp, can not simultaneously exist on even two chains.



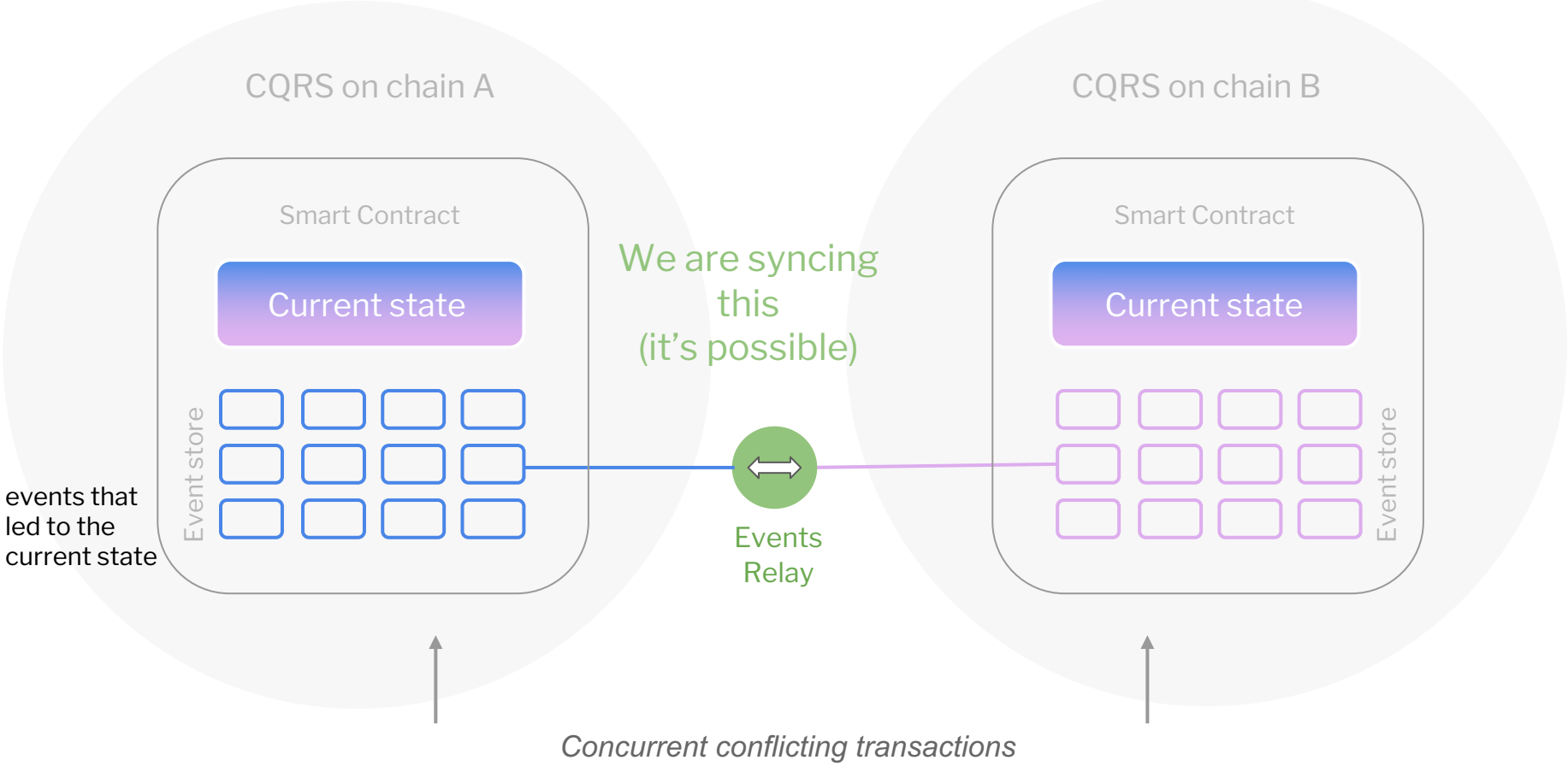
we solved the web3 biggest challenge,
that no one has solved so far

We achieved full data consistency of smart-contracts deployed on multiple chains.

the challenge



how we solved it



architecture

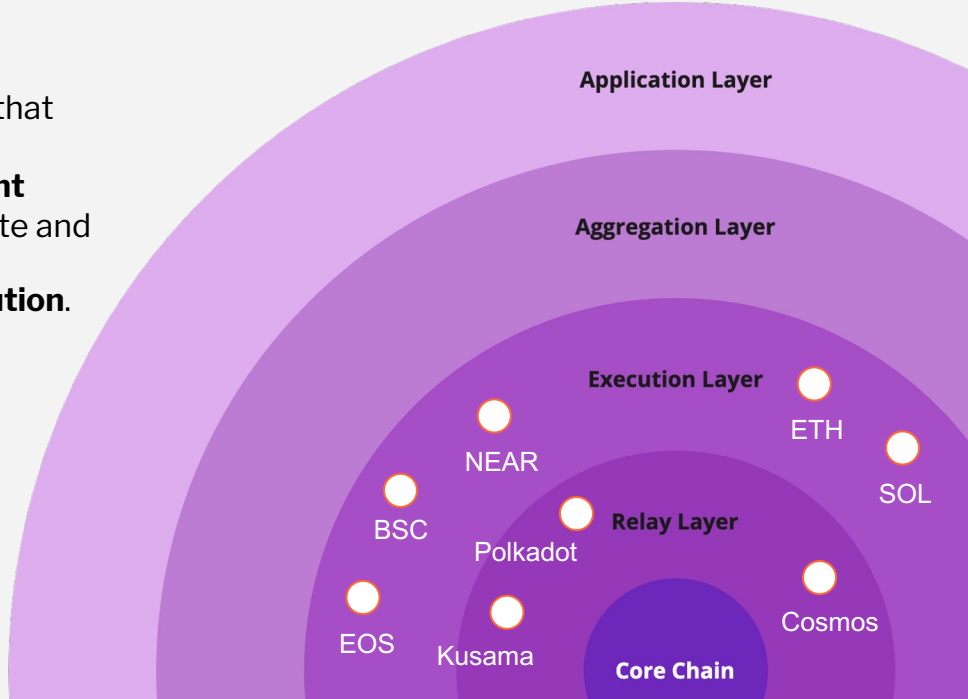
Layered, event-centric architecture to achieve complete interoperability.

Command Query Responsibility Segregation (CQRS)

- **Improves smart contract performance** by separating write and read requests.
- **Faster and cheaper transactions** and consistent view of data across all chains.

Event Sourcing

- **Records all events** that affected data.
- **Maintains consistent state**, that is accurate and valid on all chains.
- **Easy conflict resolution**.
- **Easy to recover**.



We have a Proof of Concept

that can synchronize events history and restore valid state of smart-contacts on multiple chains.

What it does:

1. showcases an infrastructure sufficient to deploy omni-chain smart contracts (event-centric execution environment, cross-platform data format, router, read model and relay)

2. showcases how the smart-contracts
a) get in sync
b) resolve conflicts

roadmap

Proof of concept
(already working)

Testnet release
(with a limited number of pre-set smart contracts)

Public release
(with any custom smart contracts available)

Q2 2023 → Q3 2023 → Q4 2023 → Q1 2024

Phase 1: Foundation

EVM chains CQRS + ES execution environment;
Singleton aggregate smart-contract implementation;
MongoDB support for aggregation layer;
Centralized relay implementation;
Kafka support for messaging from relay to aggregation layer;

Phase 2: Scaling

Any number execution chains synchronization;
Complete transactive fee coverage algorithm (including events transmission fees);
Relay-chain implementation;
Multi-relay implementation;
Optimistic validity proofs implementation;
Dynamically adding new aggregate types;

Phase 3: Optimization

Router aggregate pre-execution;
Centralized core instance implementation;
Snapshots implementation;
Aggregates dynamic rebalancing + Dynamically adding new execution chains;
Cross-aggregates transactions.

Phase 4: Security

ZK validity proofs implementation;
Implementing ZK-based domain-specific language for aggregate;
Core chain implementation;
Governance DAO implementation;

mind exercise / use case

AI-powered decentralized VC.

team

Alex Shkor
CEO and Architect

Tech entrepreneur and distributed system architect for last 14 years, ex-CEO at DEIP, ex-CTO at Paralect.

LinkedIn

Alexey Kulik
Head of R&D

14 years software engineer and former R&D head in venture studio.

LinkedIn

Yahor Tsaryk
Engineer

Software Engineer and Tech Lead with 10+ years experience in IT industry, ex-CTO at DEIP Creator Economy Protocol.

LinkedIn

research

2015: Collective Intelligence Labs (CIL) starts research of applications of blockchain for storing and securing creative and intellectual data, or IP.

2017: R&D of **decentralized multidimensional reputation system** - DAS *(link to white paper)*. TRL 4 (validated in lab).

2019: Open Research and Investment Platform aka ORIP - **platform for multidimensional evaluation of research projects by experts and automated funding decisions**. TRL 5 (validated in relevant environment).

2020: Proof of Share protocol *(link to white paper)*, IP Ledger - **registration of IP on blockchain and verifiable proofs of sharing, access and usage**. TRL 7 (demonstrated in operational environment).

2023: Omnichain *(link to white paper)* - **seamless, high performance, and secure true multi-chain infrastructure**. TRL 4 (validated in lab).

advisors

Leonid Lozner

Co-founder of
EPAM Systems
(NYSE: 18B)

Gregory Young

Inventor of CQRS,
Founder of
EventStore

Sergei Barisiuk

CTO, Co-founder of
PandaDoc,
a legal tech unicorn

Violetta Chekan

VC investor, founder of
VisCaeli Startup Lab

architecture: application layer

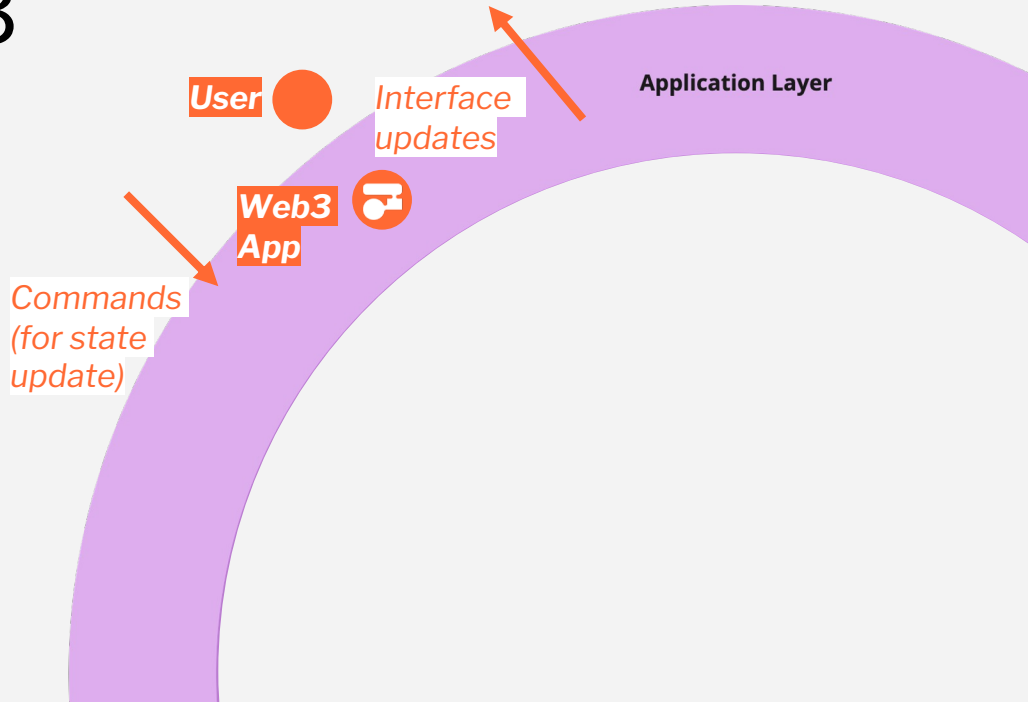
Provides an interface to interact with Web3

How it works:

Application uses a cross-platform data format (ProtoBuff) to form an operation to web3 based on user input.

All apps are registered in the protocol and sign an operation with their private keys with a use of account abstraction model to merge account on all the blockchains into single one.

```
Command
{
                    AggregateID: #ID# //smart-
contract ID
                    UserID: #ID#
                    Payload: CommandPayload:
                    UserSignature: signature
}
```



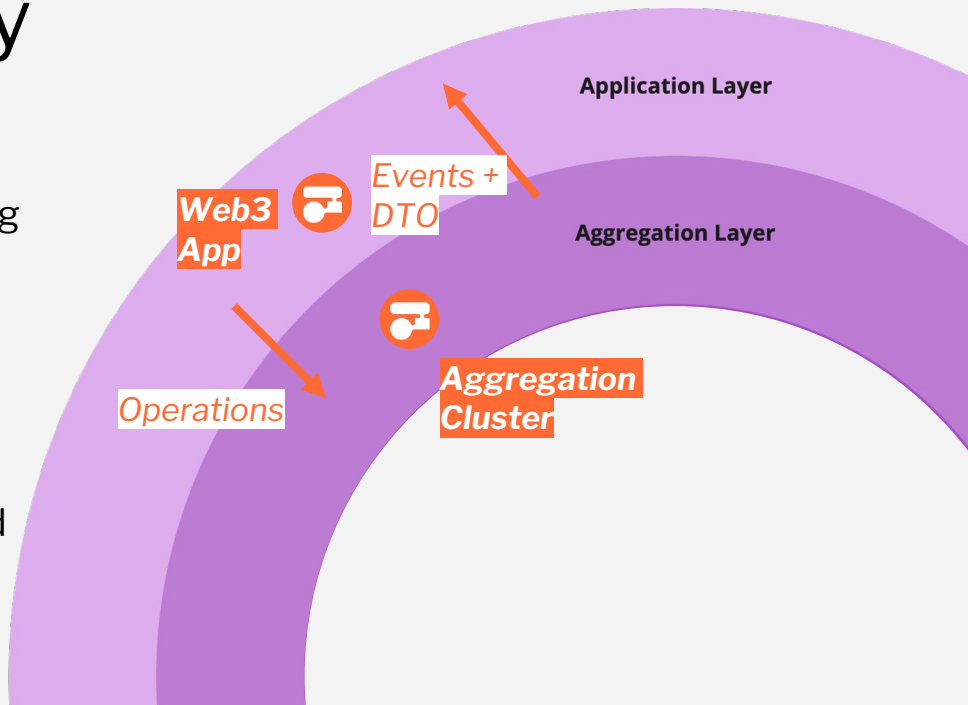
architecture: aggregation layer

Read&write to all chains in the most optimal way

How it works:

It calculates the optimal execution path by preplaying aggregate executions, taking into account factors such as price, speed, and security. It then routes operations accordingly.
It also continuously listens to the event stream from multiple connected execution layer blockchains in order to build a read model of web3 that is optimized for read requests from applications.

```
Operation
{
                    AggregationClusterID:
#ID#;
                    Commands:
commands;
                    AppID: #ID#;
                    AppSignature:
signature
}
```



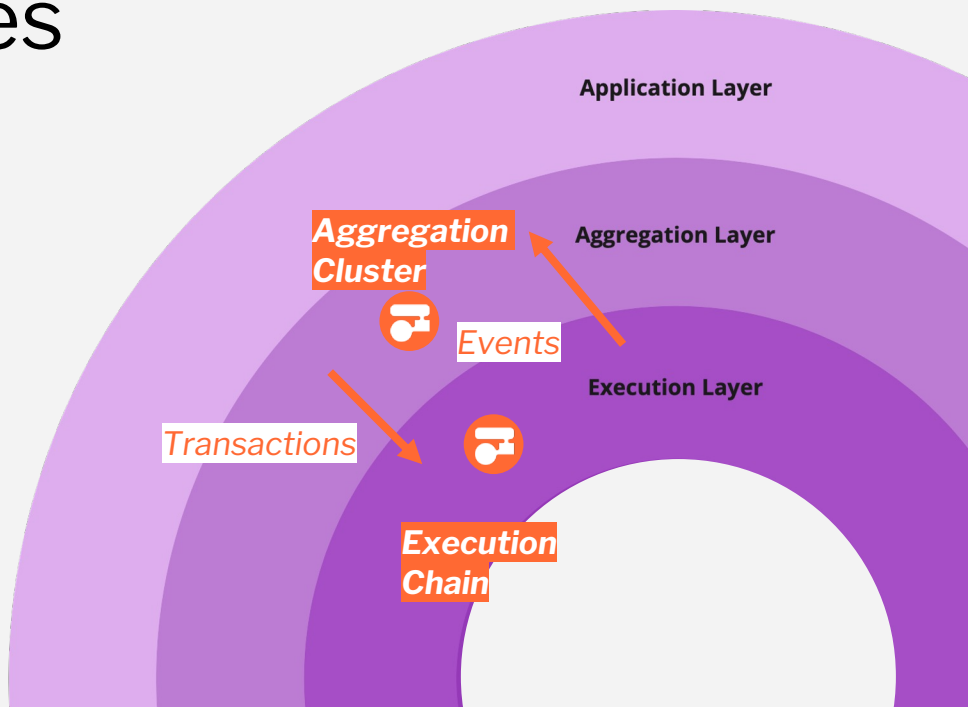
architecture: execution layer

Validates and executes commands, and updates the state of web3

How it works:

On every blockchain a special interpreter of DSL receives and deserializes operations from aggregation cluster and executes them validating the logic.
It then publishes the state updates in form of events to an event store and to all the subscribed aggregation clusters and relay chains.

```
Transaction
{
        ExecutionChainID: #ID#;
        Operations: operations[];
                ExecutionParameters: params // chain
specific parameters like gas price, etc;
                AggregationClusterID: #ID#;
                AggregationClusterSignature: signature;
}
```



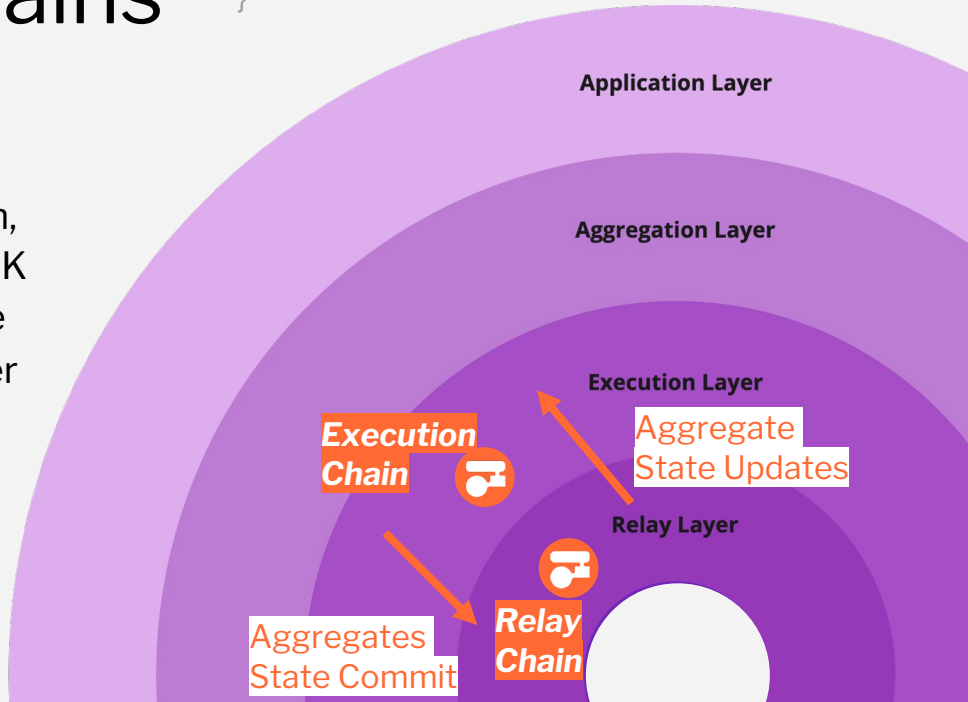
architecture: relay layer

Syncing smart-contracts data among all blockchains

```
AggregateUpdate
{
        RelayChainID: #ID#;
                AggregateRootHash: hash;
                AggregateValidationParameters: // state transition validation parameters (ZK validity proofs or optimistic validation parameters);
        Events: events[];
                ExecutionChainID: #ID#;
                ExecutionChainSignature: signature;
}
```

How it works:

At each block generated by the execution layer chain, the stream of aggregate state updates, events and ZK validity proofs, is committed to the relay chain where it undergoes validation and is broadcasted to all other execution layer chains that are subject to state transitions of these specific aggregates.



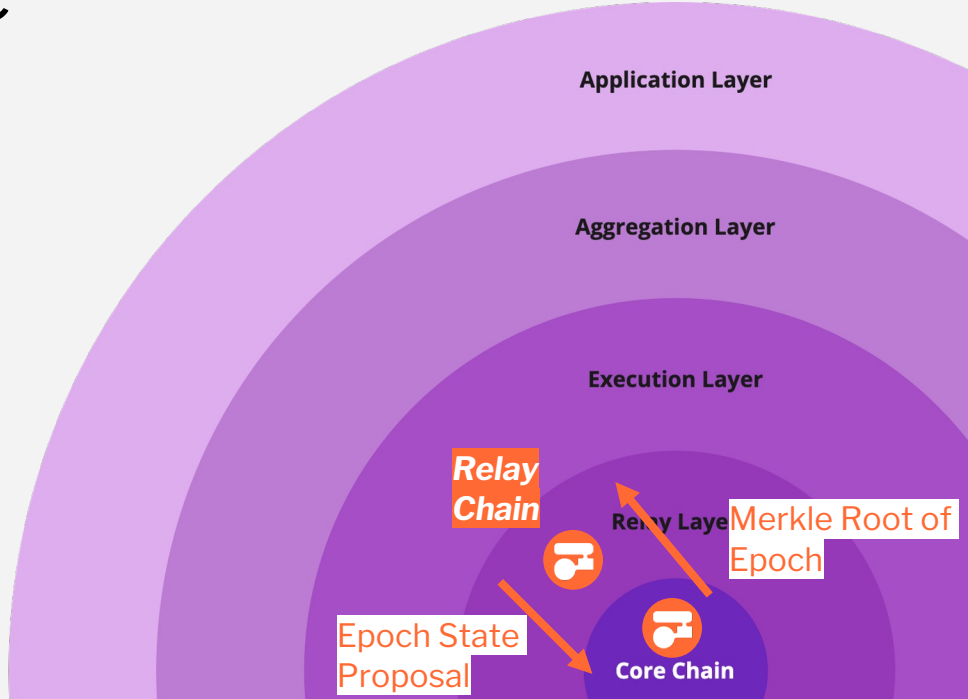
architecture: core layer

Synchronizes all the relay chains and finalizes the state of web3

How it works:

On every block relay chain submits the current merkle tree of aggregates state to the core layer to synchronize the state across all the relay chains. Once the core layer finalizes the epoch, it broadcasts the actual final state of all aggregates to relay, execution, aggregation and application layers.

```
EpochProposal
{
        CoreChainID: #ID#;
                AggregatesMerkleRoot: merkleRoot;
        EpochNumber: uint;
                EpochTimestamp: timestamp;
                RelayChainID: #ID#;
                RelayChainSignature: signature;
}
```



contact

Alex Shkor, CEO and Architect

alex@cilabs.ai
Telegram