Ethereum VM illustrated

exploring some mental models and implementations

Takenobu T.

WIP

Rev. 0.00.0

NOTE

- Please refer to the official documents in detail.
- This information is current as of Mar, 2018.

Contents

1. Introduction
  - Blockchain
  - World state
  - Account and contract
  - Transaction
  - Message
  - Decentraized database

2. Virtual machine
  - Ethereum virtual machine (EVM)
  - Instruction set
  - Message call
  - Code generation
  - Gas
  - Atomicity and ordering
  - WASM

3. References

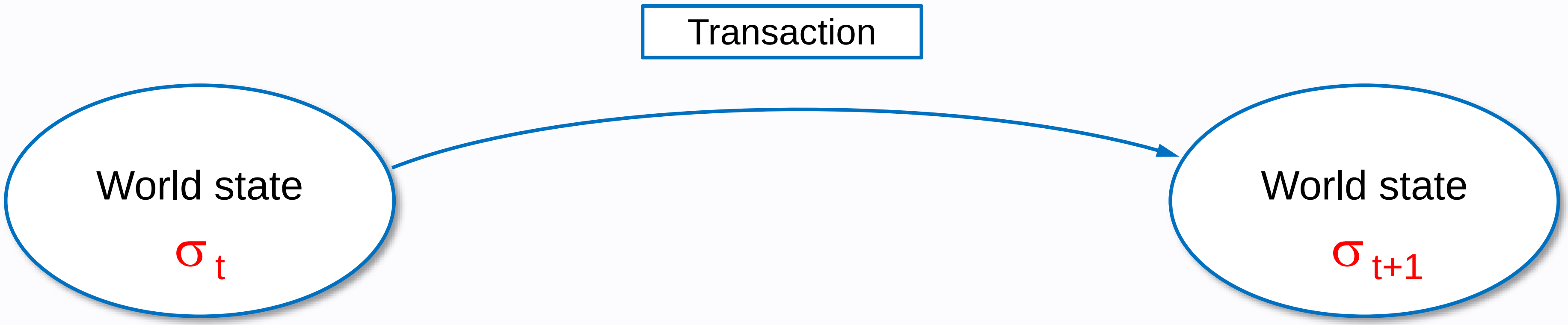
Appendix A
  - Web3 API
  - Geth, Remix, Metamask, Truffle
  - Consensus

1. Introduction

1. Introduction

Blockchain

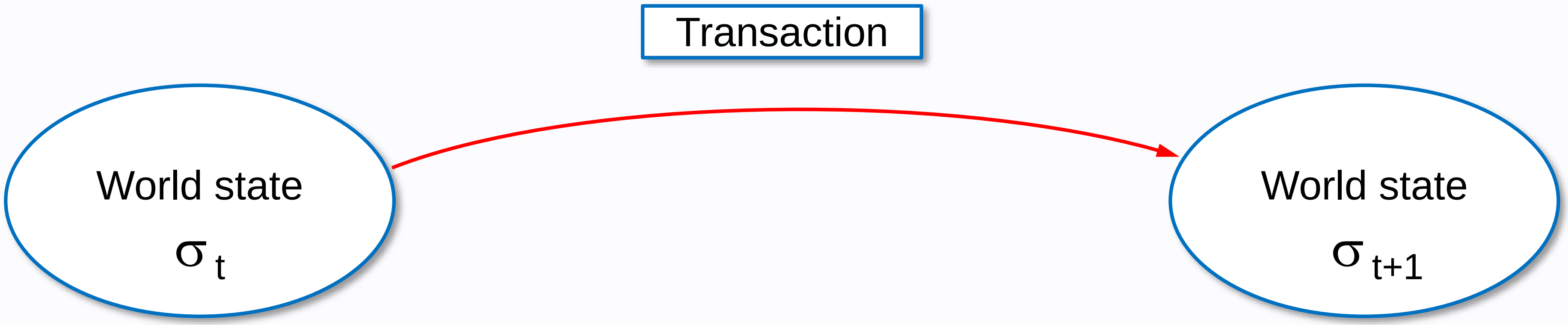
A transaction-based state machine



Ethereum can be viewed as a transaction-based state machine.

References : [E1] Ch.2

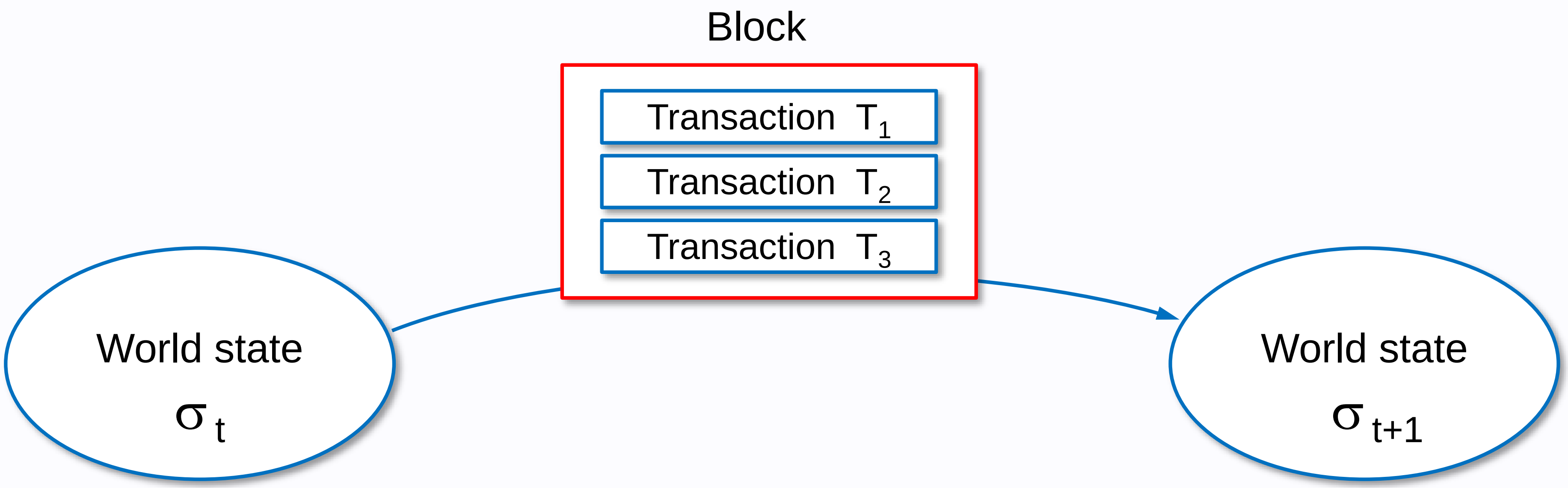
A transaction-based state machine



A transaction represents a valid arc between two states.

References : [E1] Ch.2

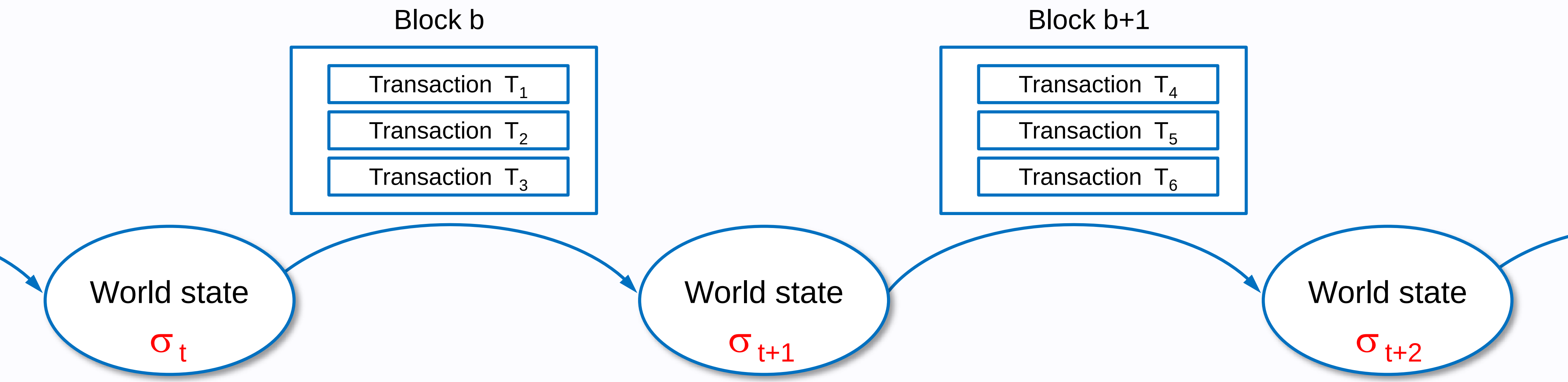
Block and transactions



Transactions are collated into blocks.
A block is a package of data.

References : [E1] Ch.2, Ch.4, [E2]

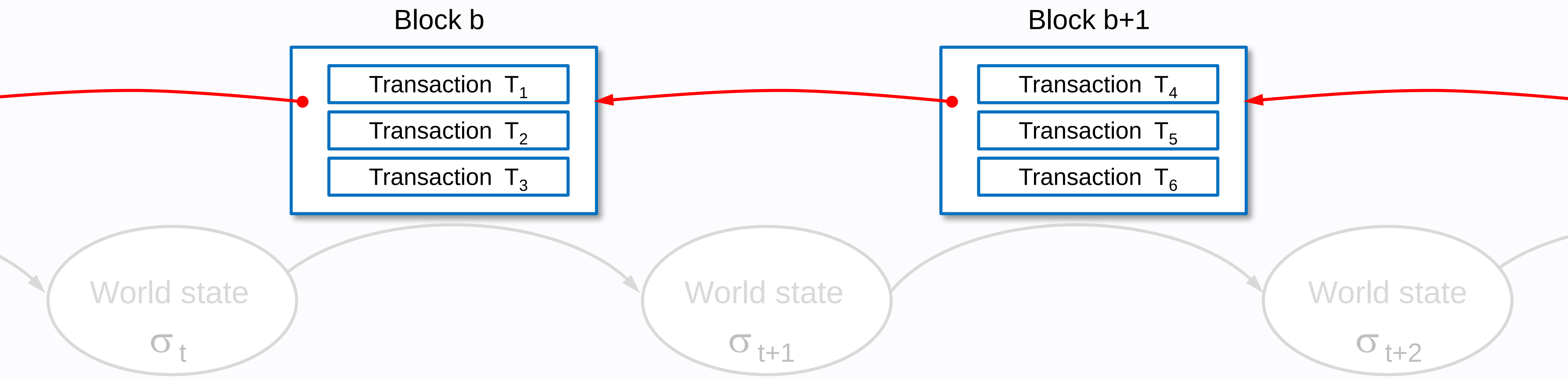
Chain of states



As states view,
Ethereum can be viewed as a chain of states.

References : [E1] Ch.2, Ch.4

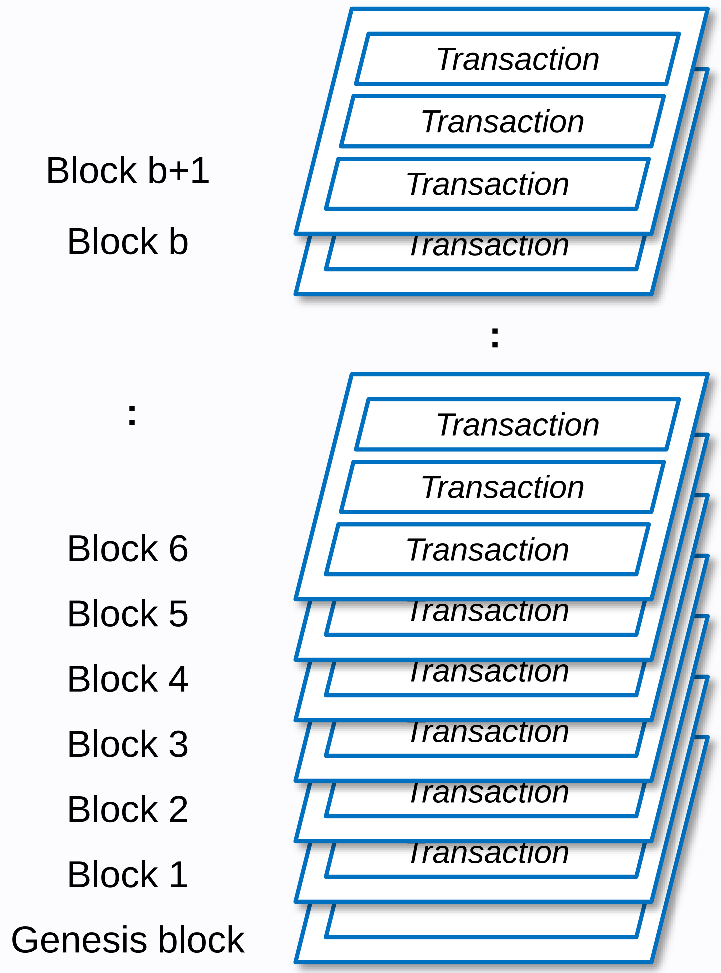
Chain of blocks: Blockchain



As implementation view,
Ethereum can be also viewed as a chain of blocks, thus `BLOCKCHAIN`.

References : [E1] Ch.2, Ch.4

Stack of transactions : Ledger



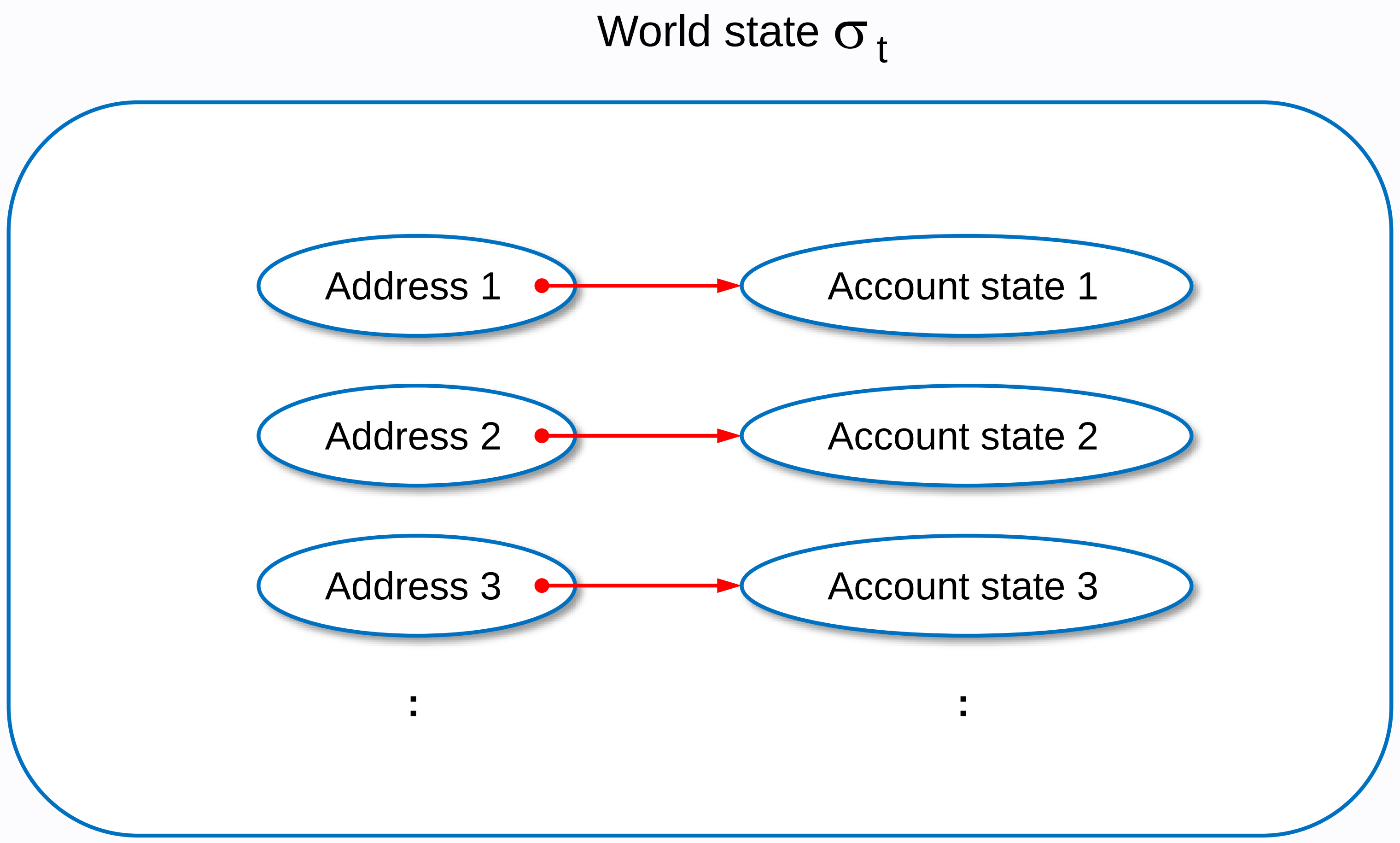
As ledger view,
Ethereum can be also viewed as a transaction stack, thus `LEDGER`.

References : [E1]

1. Introduction

World state

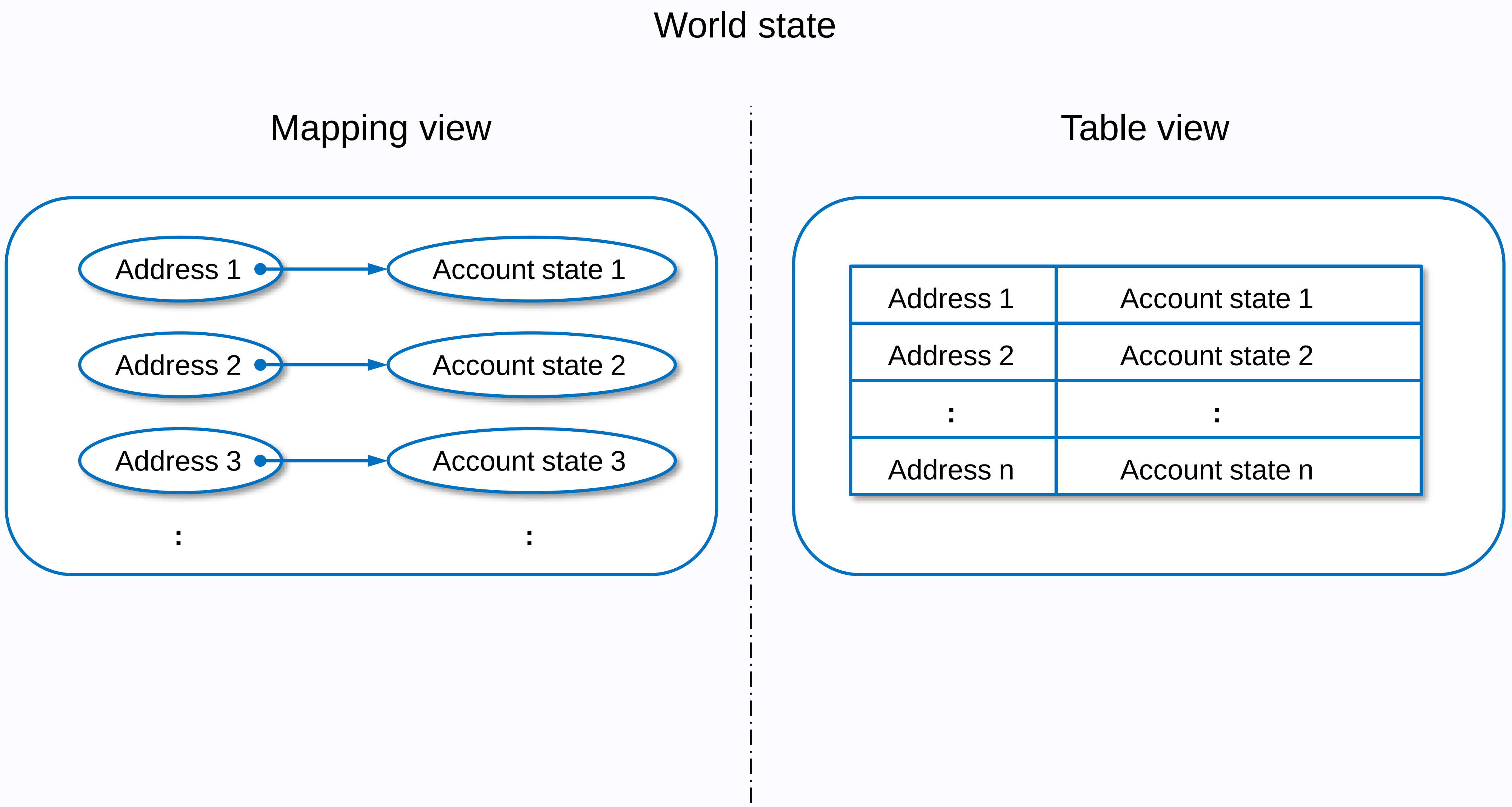
World state



The world state is a mapping between address and account state.

References : [E1] Ch.4

World state



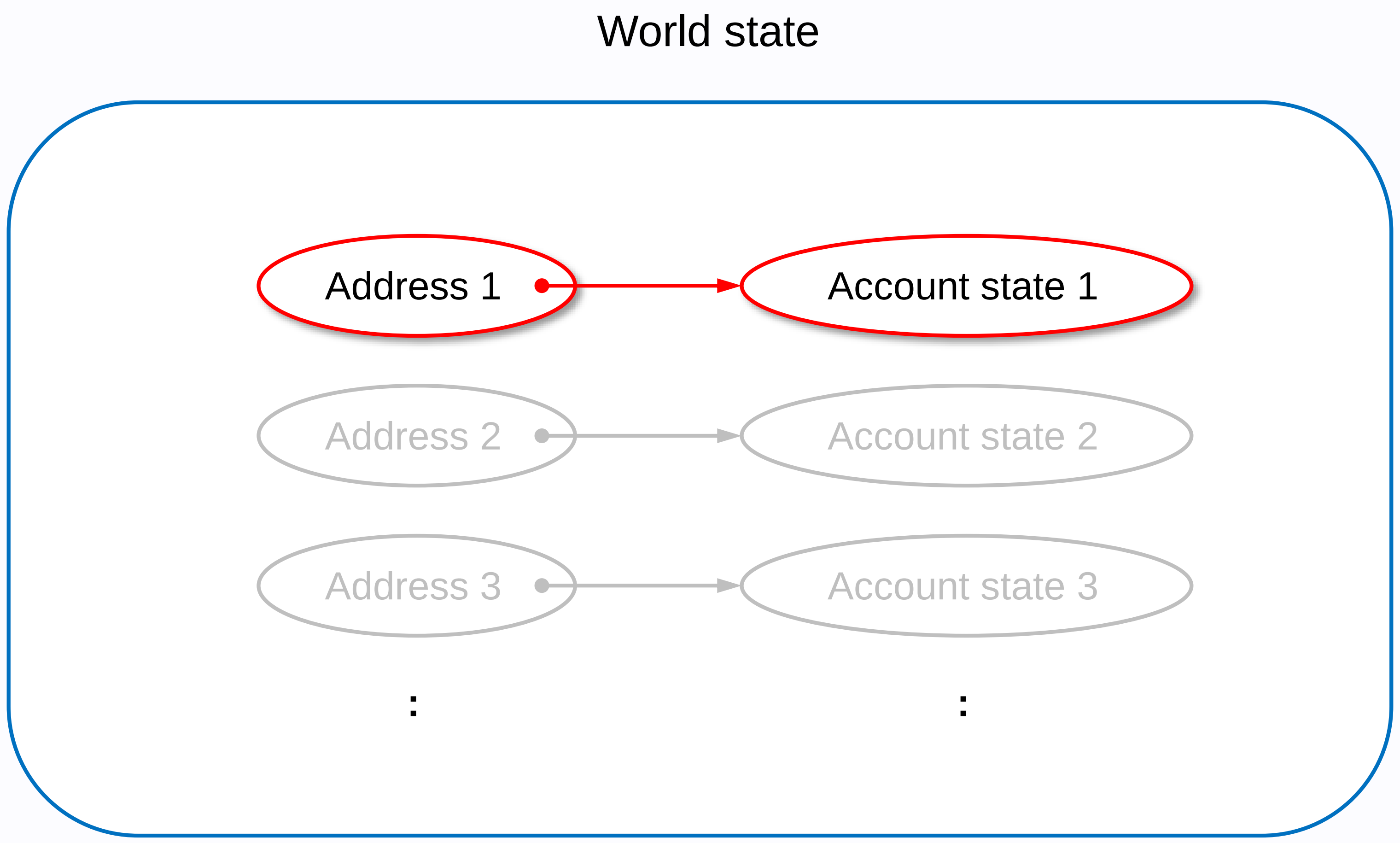
Address 1	Account state 1
Address 2	Account state 2
:	:
Address n	Account state n

References : [E1] Ch.4

1. Introduction

Account and contract

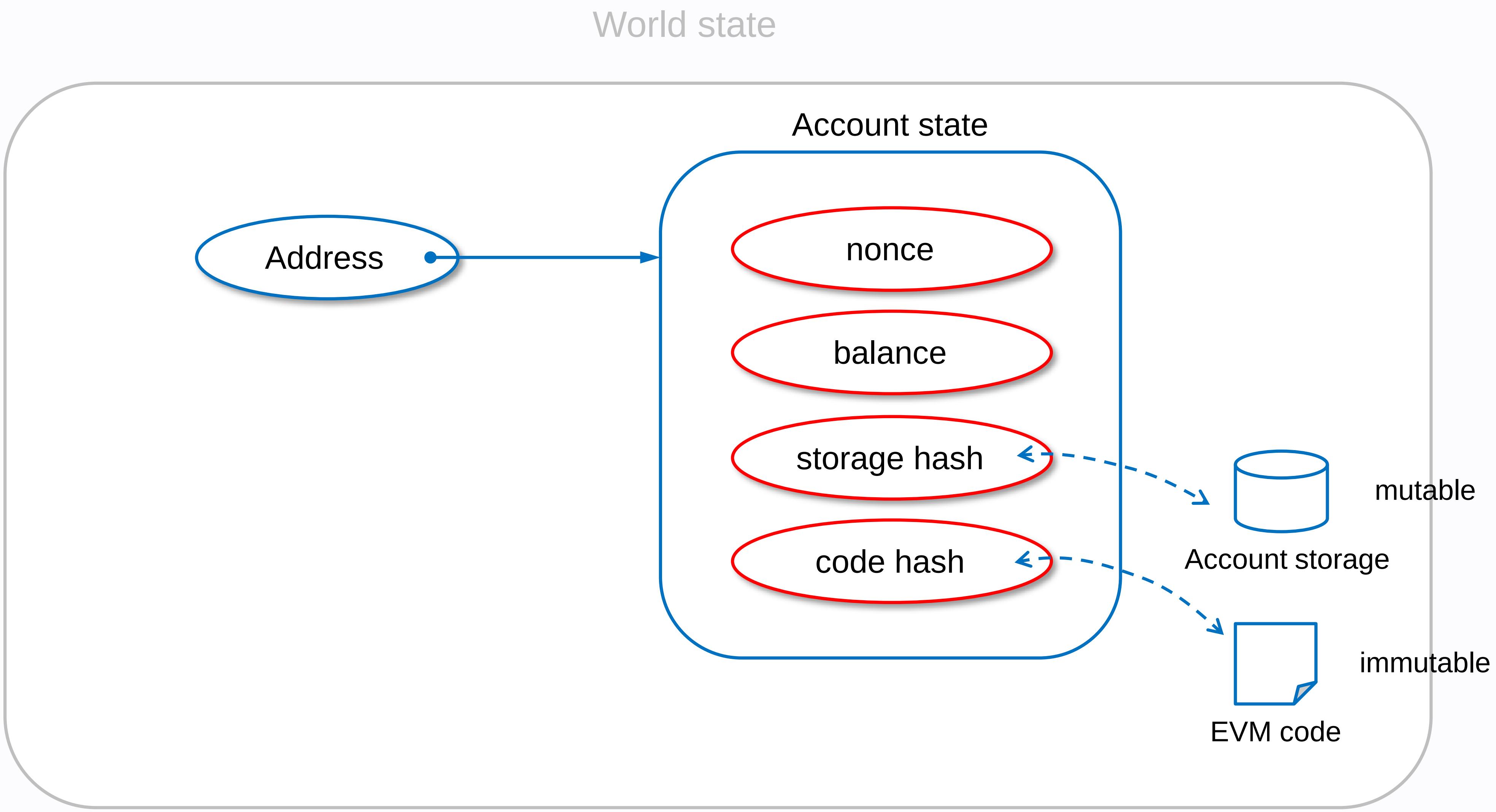
Account



An account is an object in the world state.

References : [E2]

Account state



An account state could contain EVM code and storage.

Serialized with RLP
Maintain with modified Markle tree (trie)

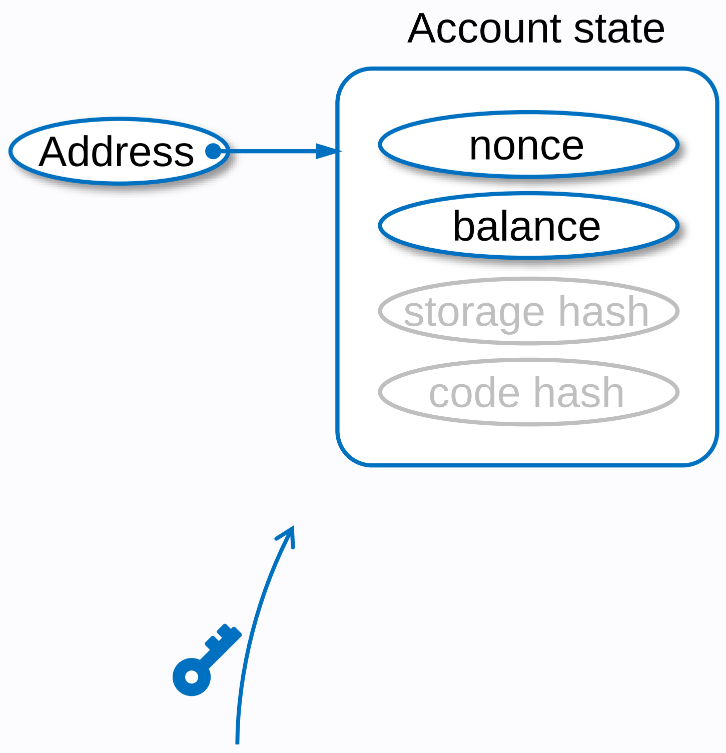
References : [E1] Ch.4

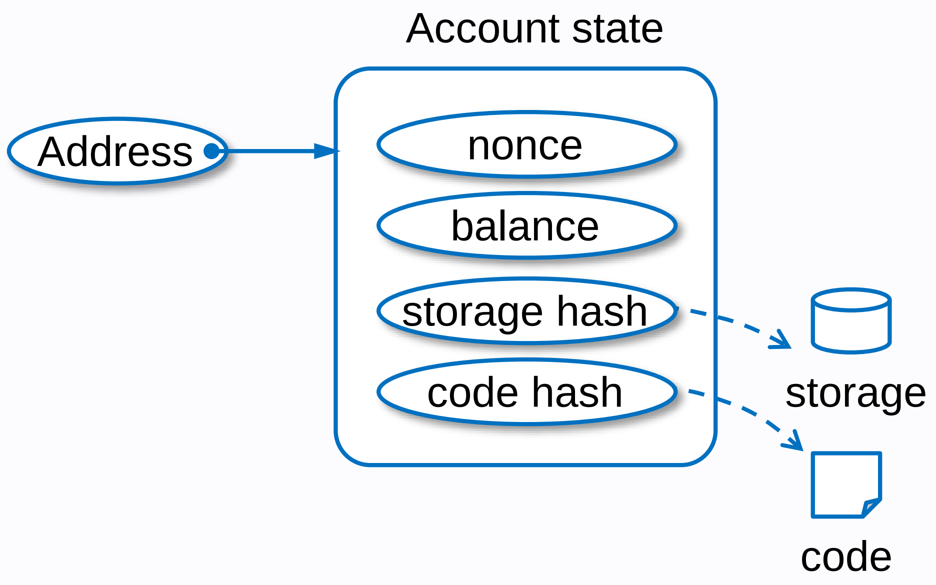
two type of account

Externally owned account (EOA)

(a simple account, non-contract account)

Contract account



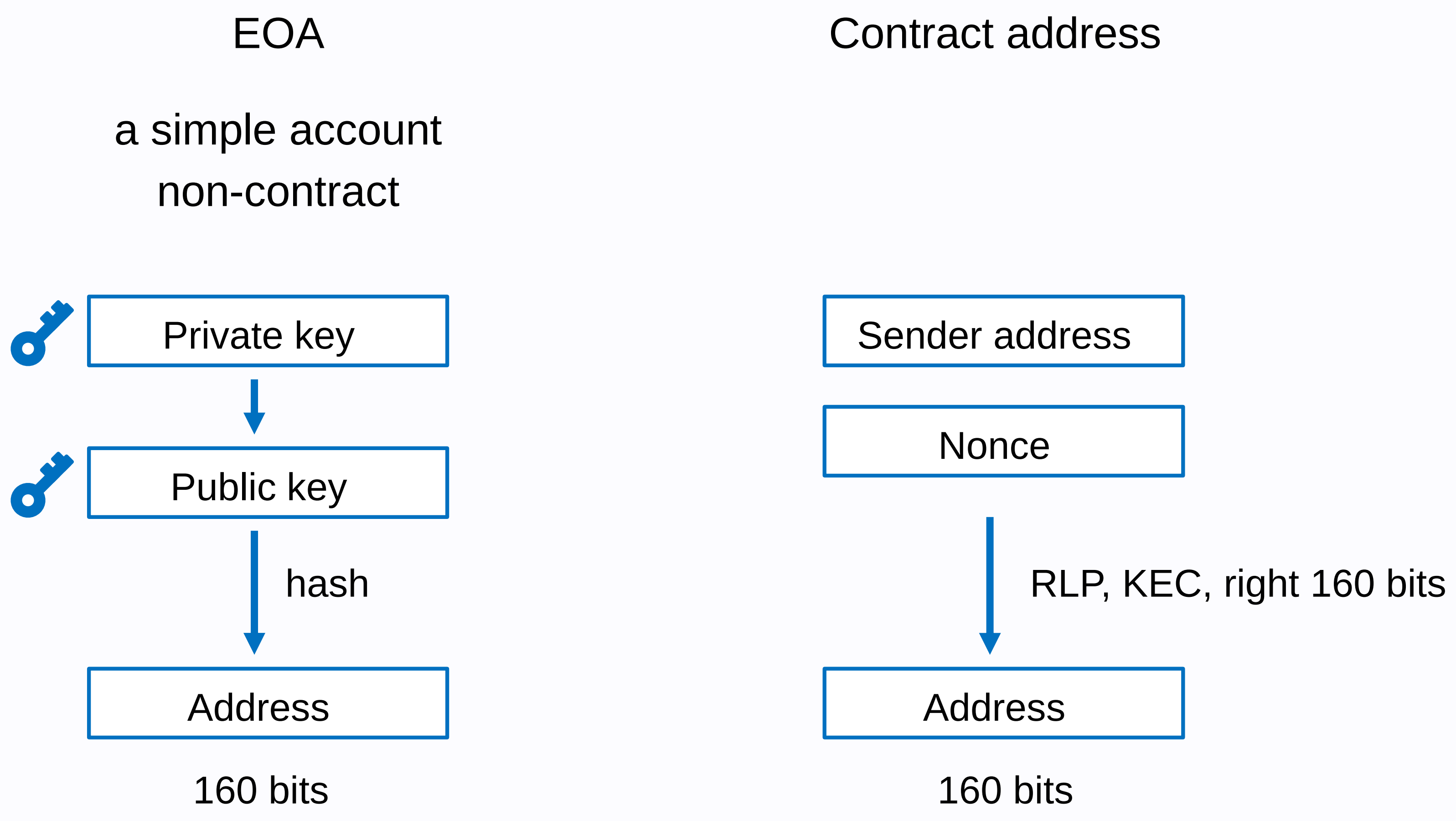


EOA is controlled by a private key.
EOA cannot contain EVM code.

Contract contains EVM code.
Contract is controlled by EVM code.

References : [E1] Ch.4, [E2]

Address of account



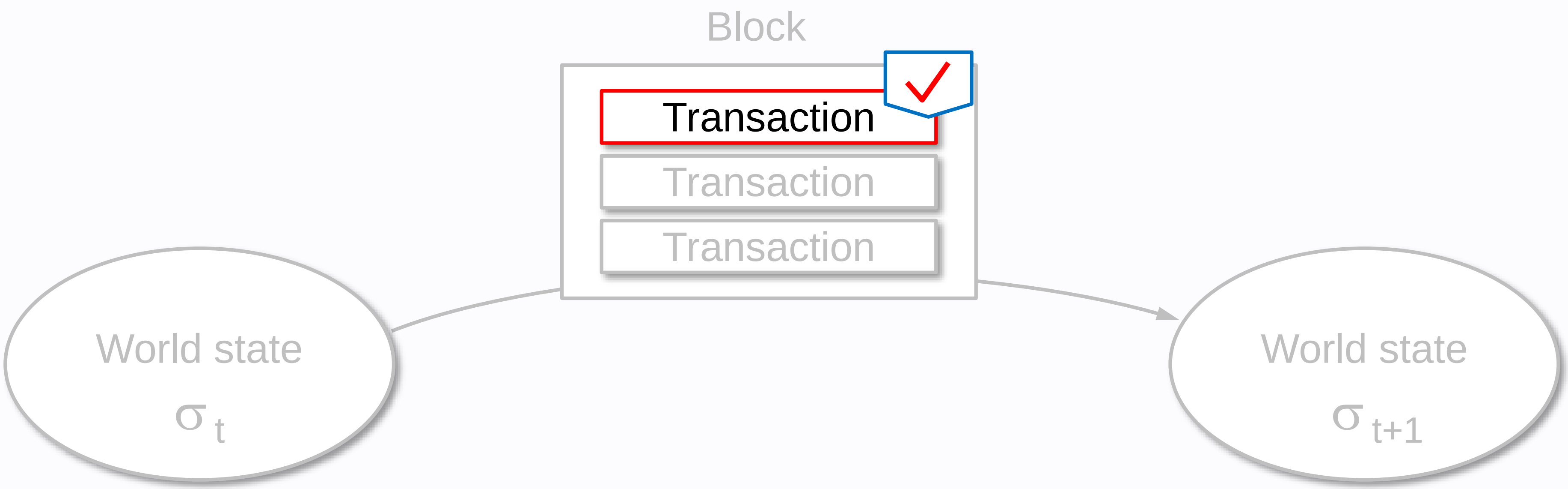
An address is essentially the representation of a public key.

References : [E1] Ch.4, Ch.7, [E2]

1. Introduction

Transaction

A transaction

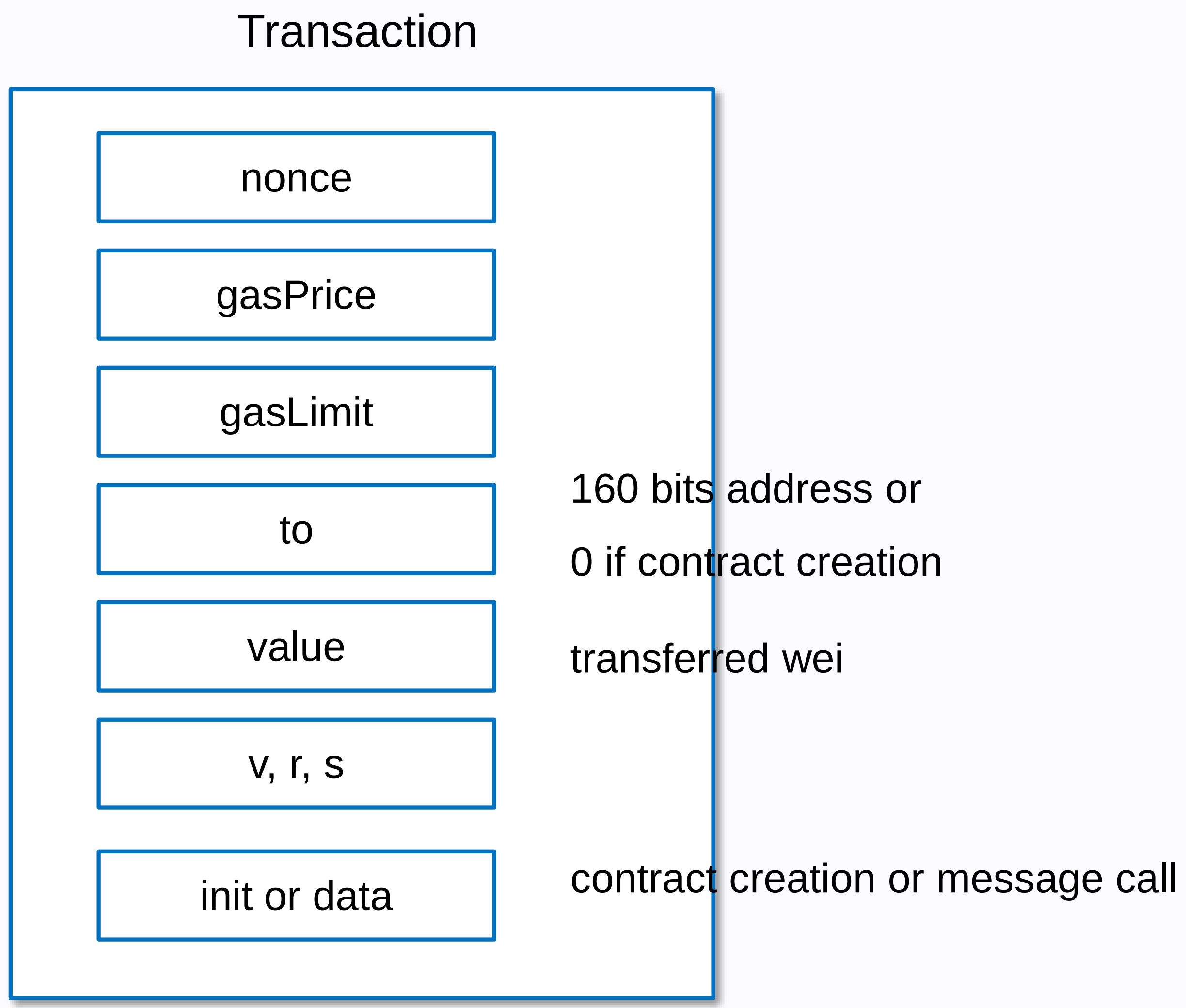


A transaction is a single cryptographically – signed instruction.

(A transaction is a digitally signed message. [E2])

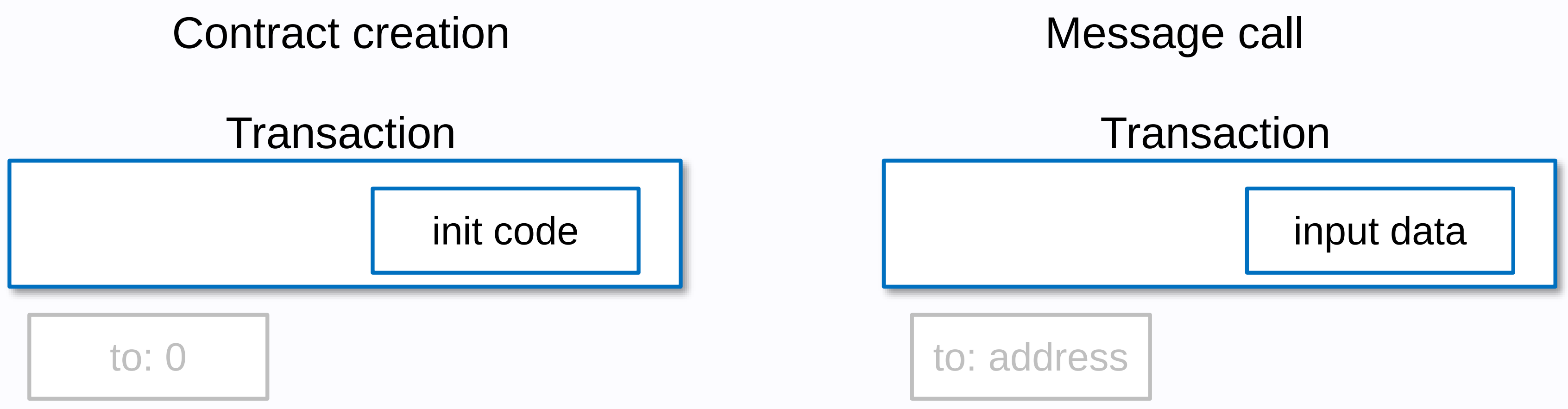
References : [E1] Ch.2, Ch.4, [E2]

Field of a transaction



References : [E1] Ch.4

Two types of transactions



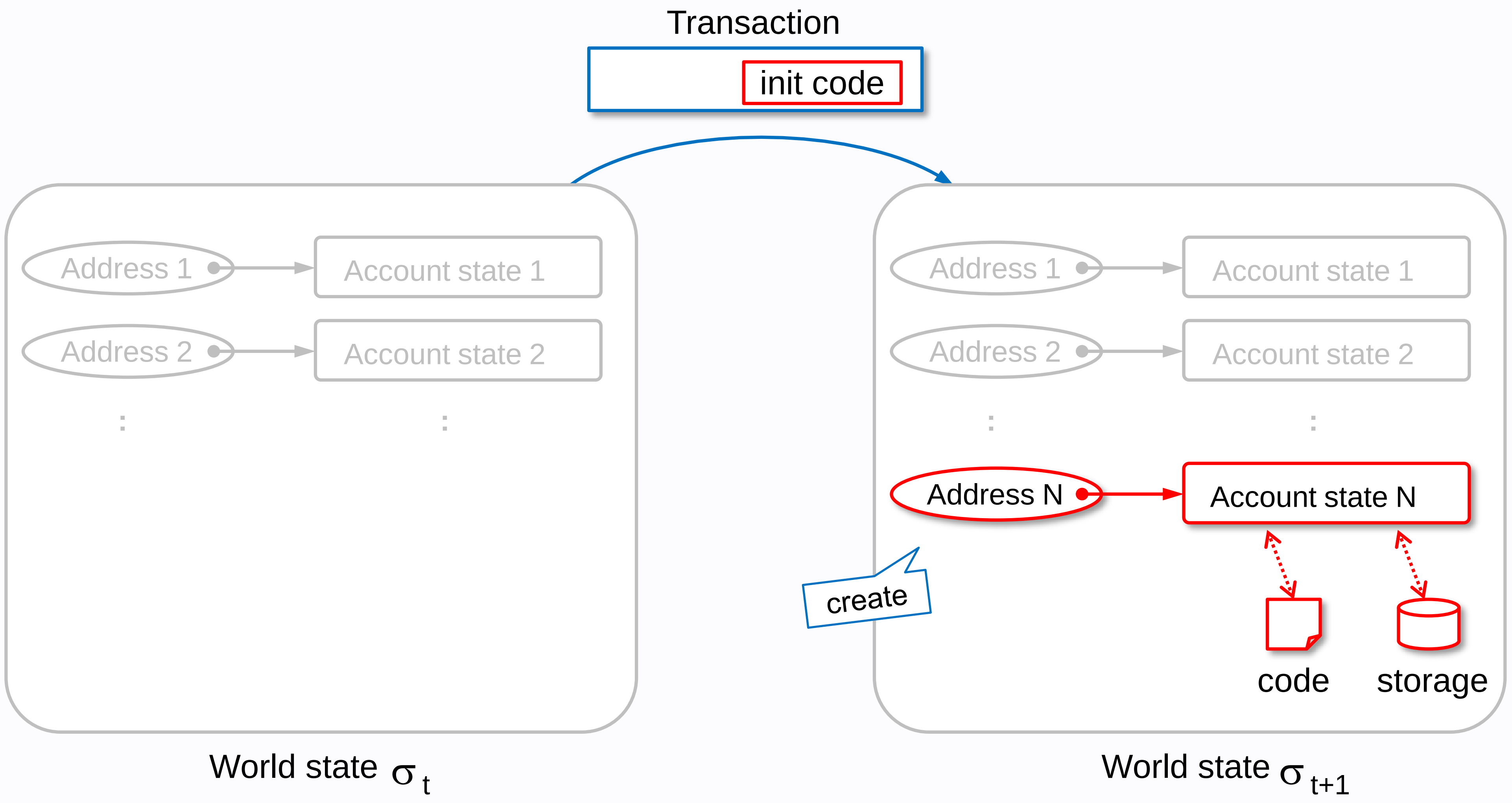
There are two types of transactions.

result in message calls

result in the creation of new accounts

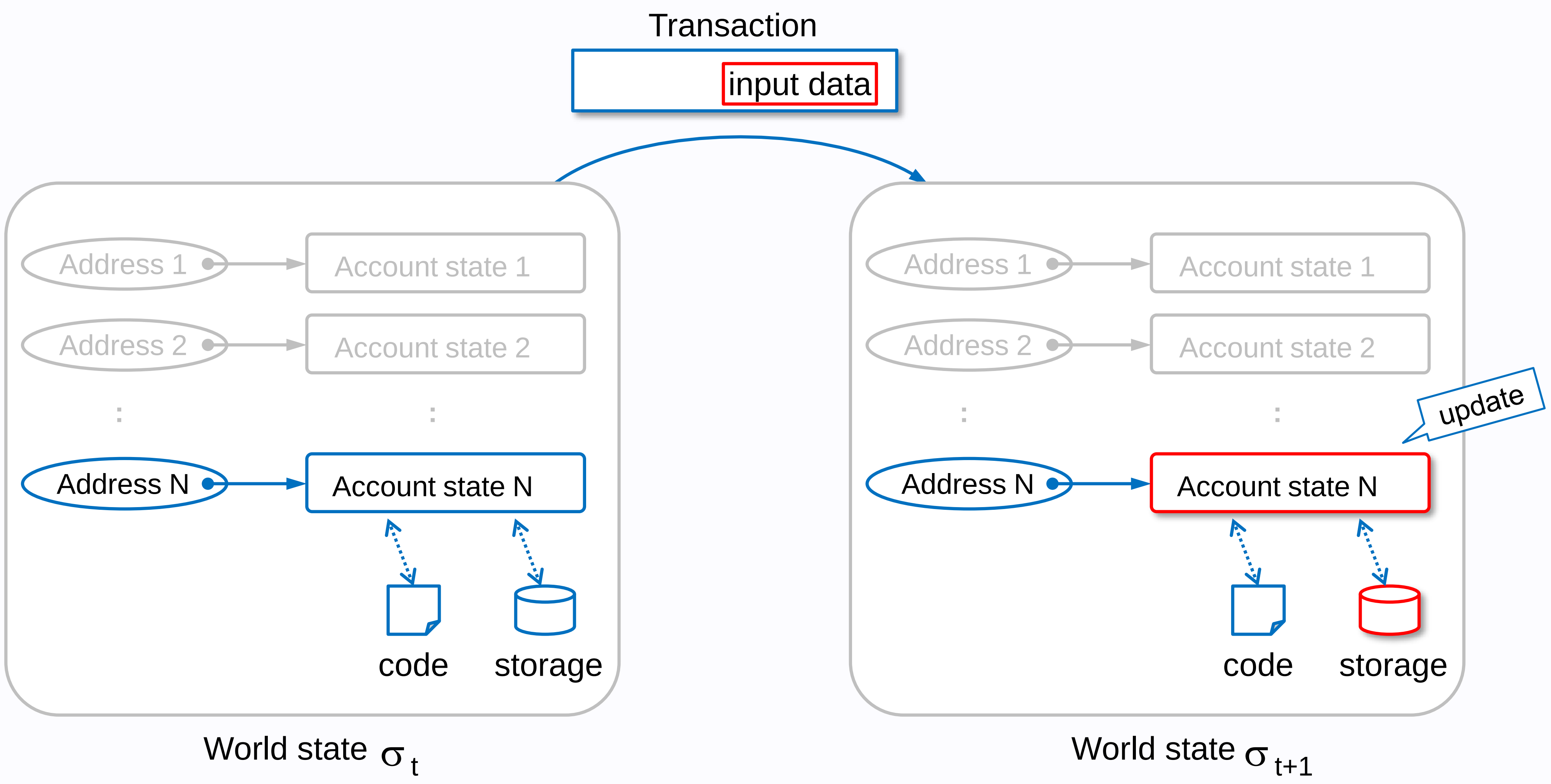
References : [E1] Ch.4

Contract creation



References : [E1] Ch.4

Message call



References : [E1] Ch.4

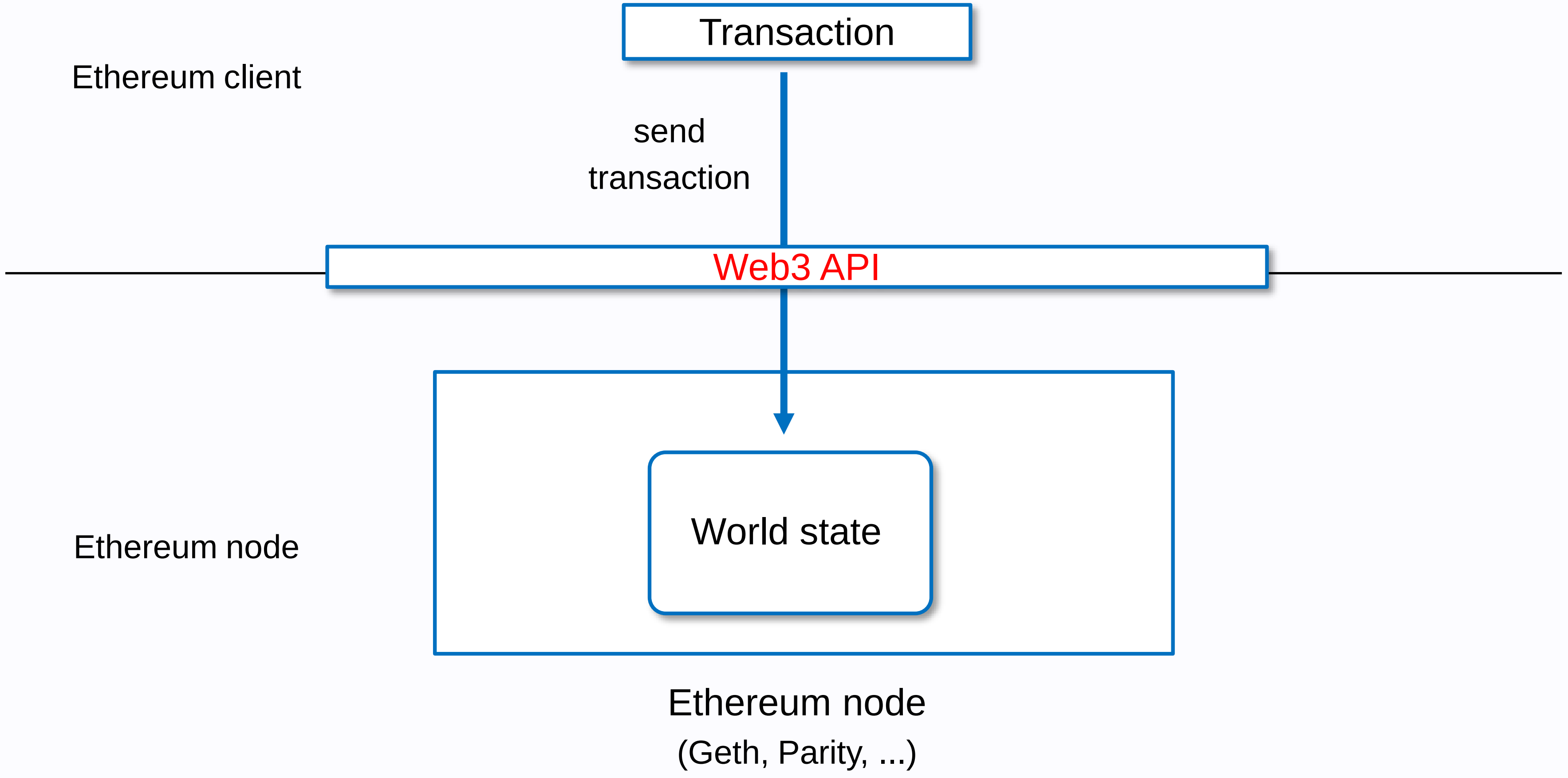
Atomic

Transaction

A transaction is atomic operation.
That is, All (complete done) or Nothing (zero effect).

References : [E1] Ch.2, Ch.4

A transaction to a node

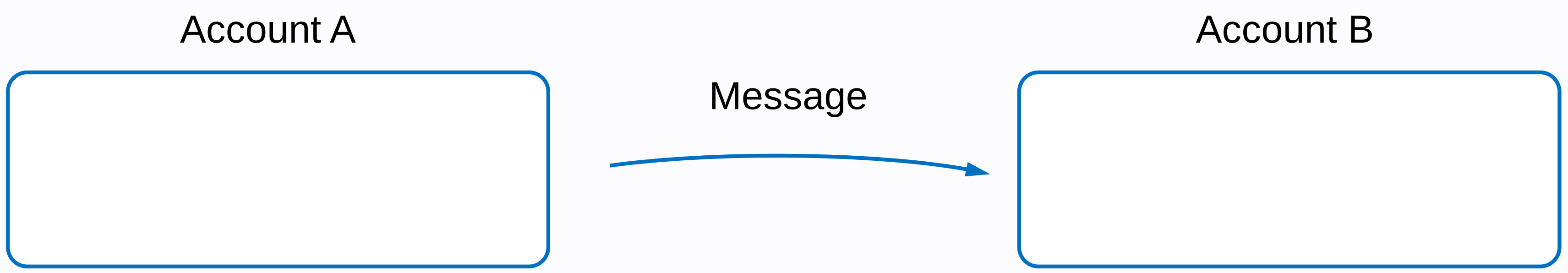


References : [E1] Ch.2, Ch.4

1. Introduction

Message

Message



“Transaction” and “messages” in Ethereum are different.

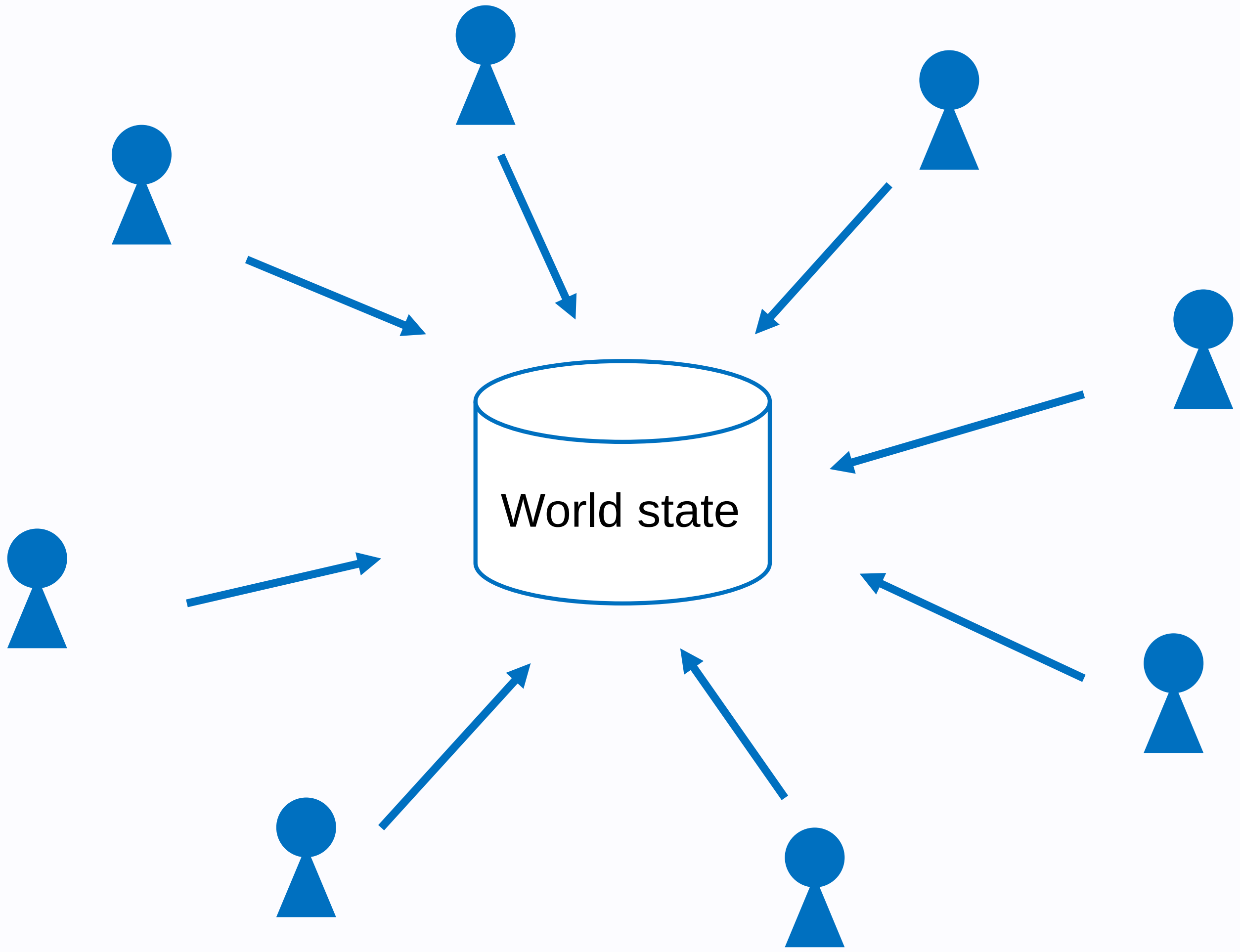
A sort of “virtual transaction” sent by EVM code from one account to another.

Every transaction triggers an associated message.
Messages can also be sent by EVM code.

References : [E2]

Decentralised database

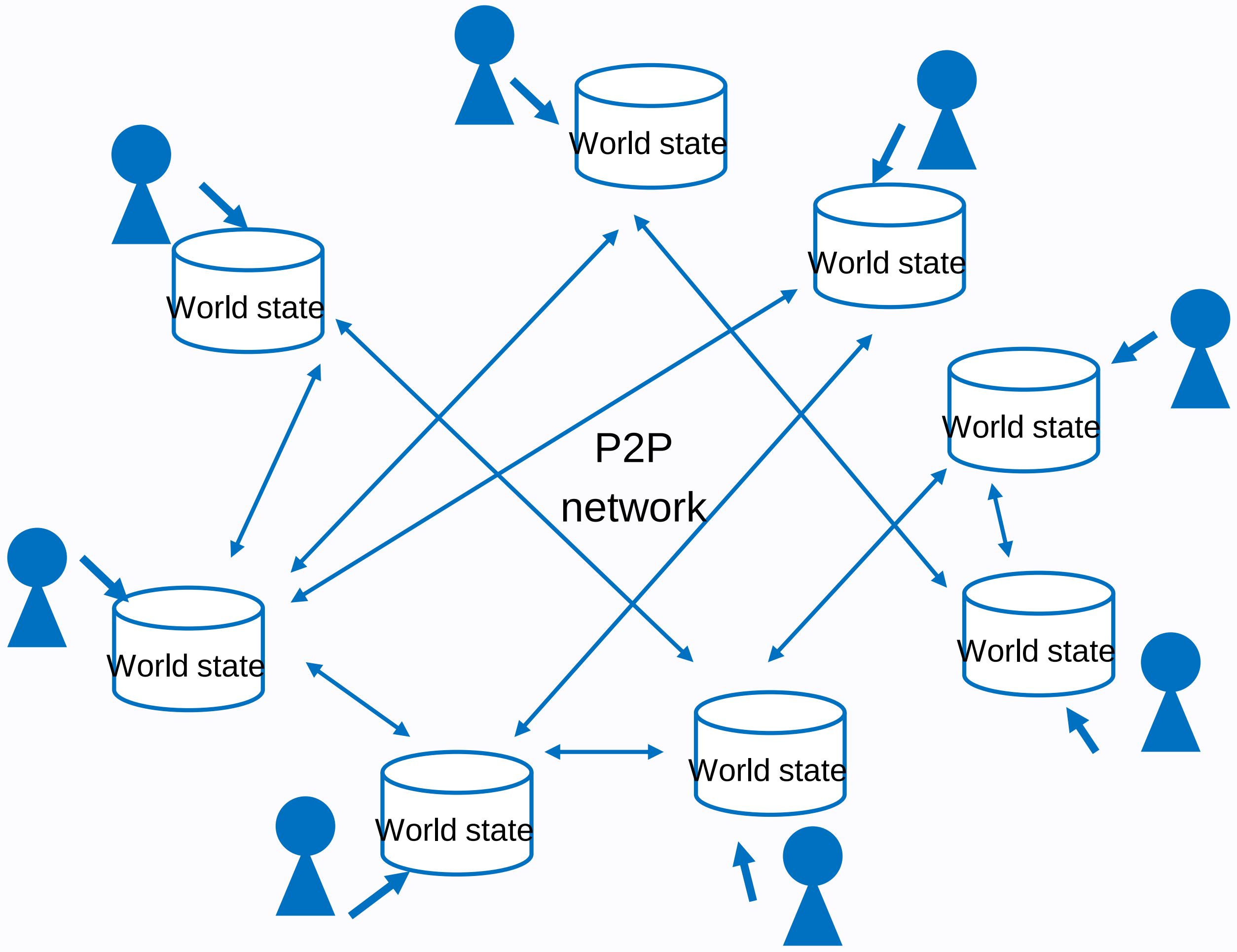
Globally shared, transactional database



A blockchain is a globally shared, transactional database.

References : [E7] Ch.7

Decentralised database



A blockchain is a globally shared, decentralised, transactional database.

copy, p2p

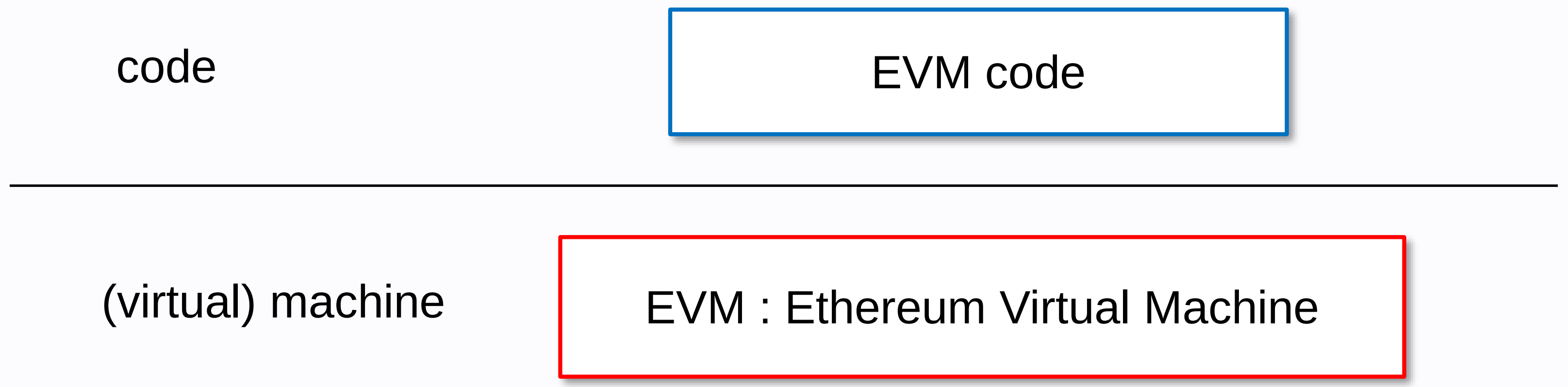
References : [E7] Ch.7

2. Virtual machine

2. Virtual machine

Ethereum virtual machine (EVM)

Ethereum virtual machine

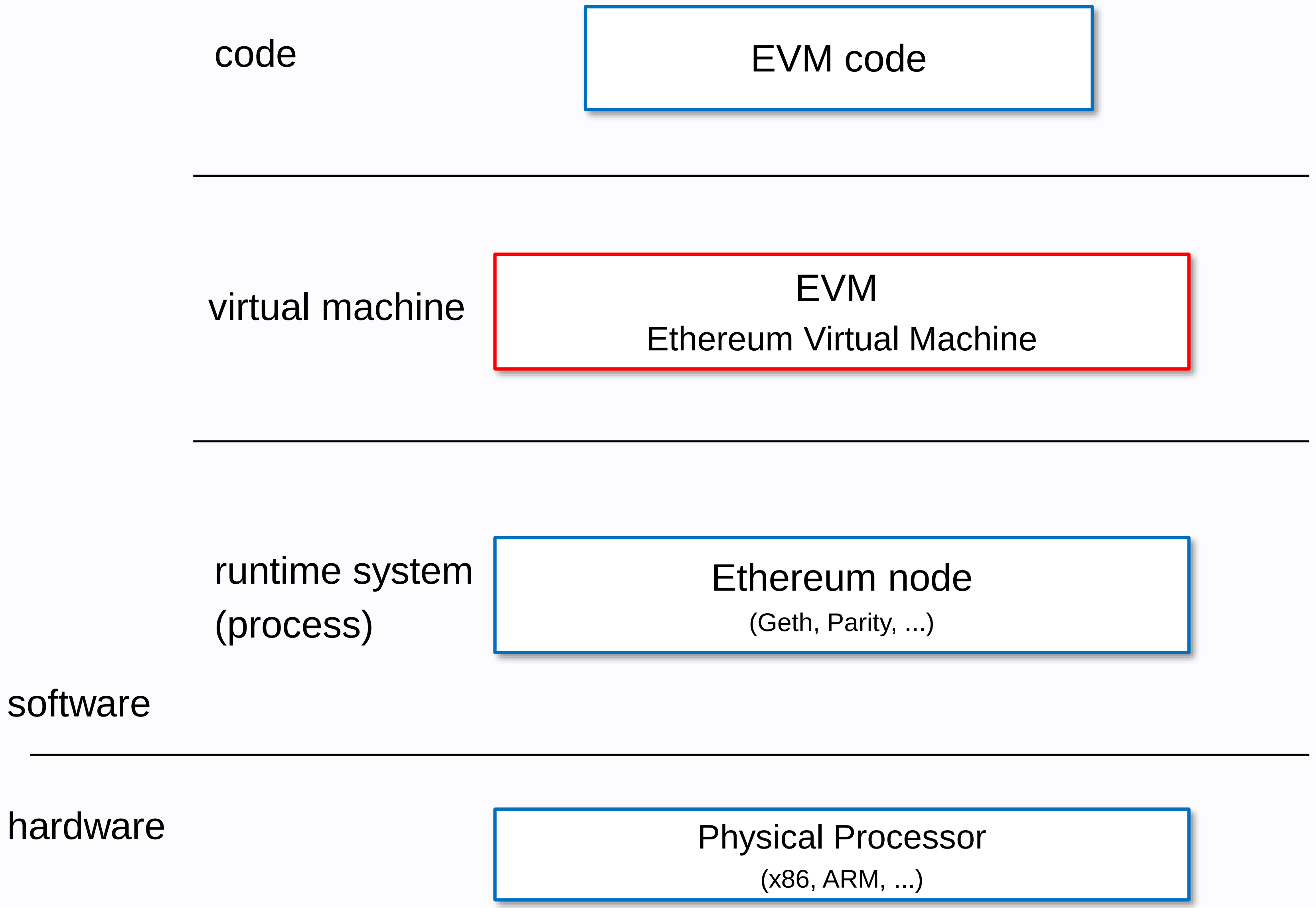


The Ethereum Virtual Machine (EVM) is the runtime environment for smart contracts in Ethreum.

EVM has no access to network, filesystem or other processes.

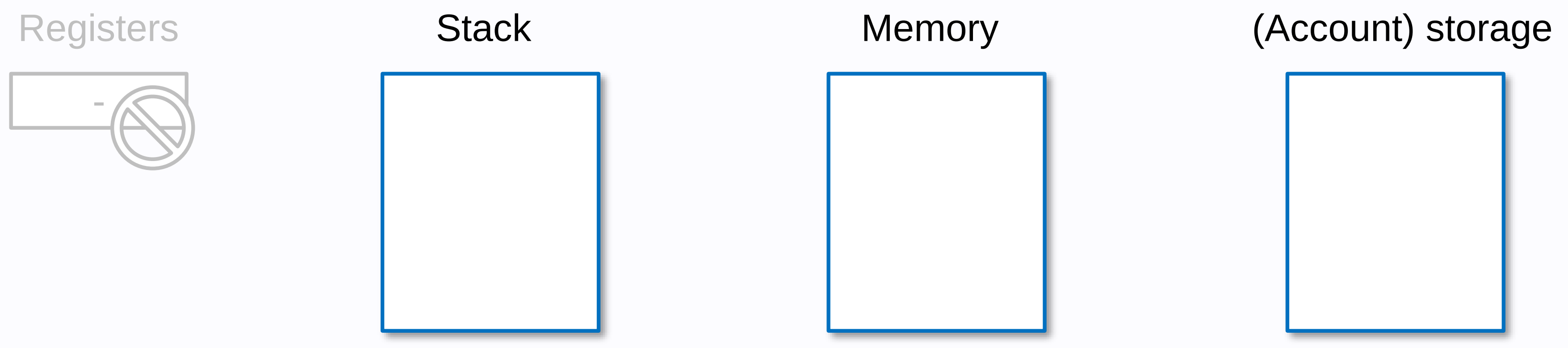
References : [E7] Ch.7

Ethereum virtual machine layer



References : [C14], [C6], [2], [C17], [8], [S15], [S16], [S11]

Machine resources of EVM

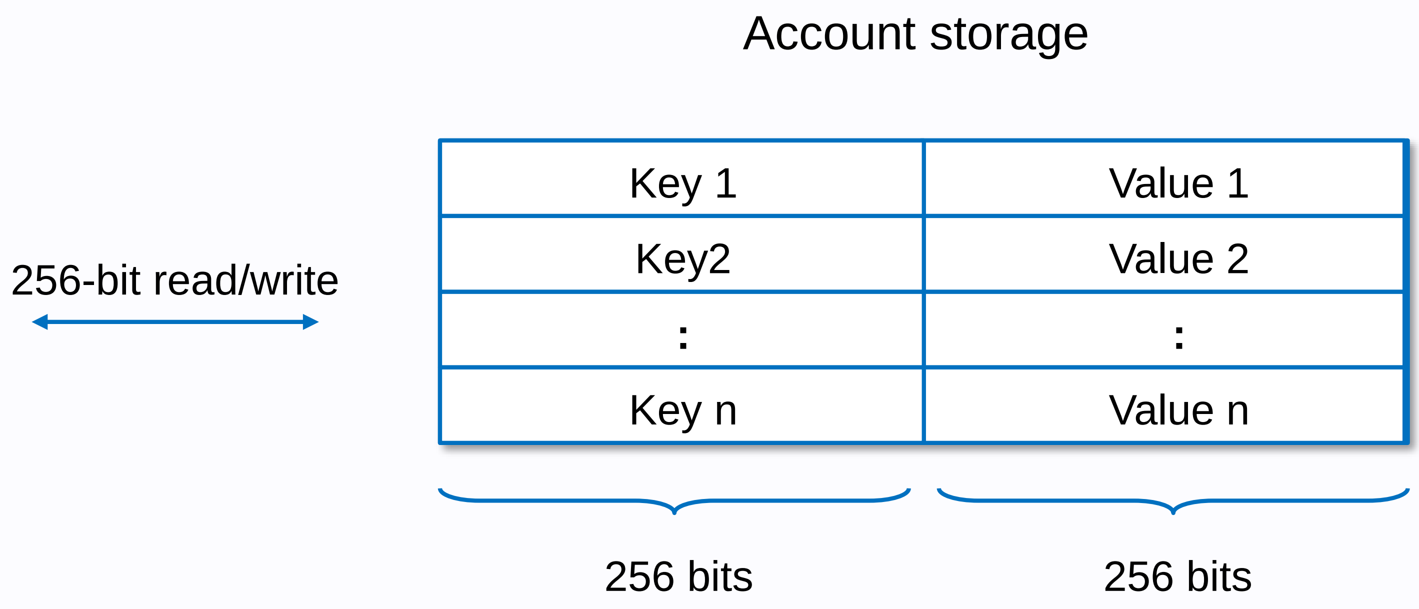


stack memory	volatile memory	persistent memory
256 bits x 1024 elements	byte addressing linear memory	256 bits to 256 bits key-value store

EVM is stack based architecture.

References : [C14], [C6], [2], [C17], [8], [S15], [S16], [S11]

Account storage



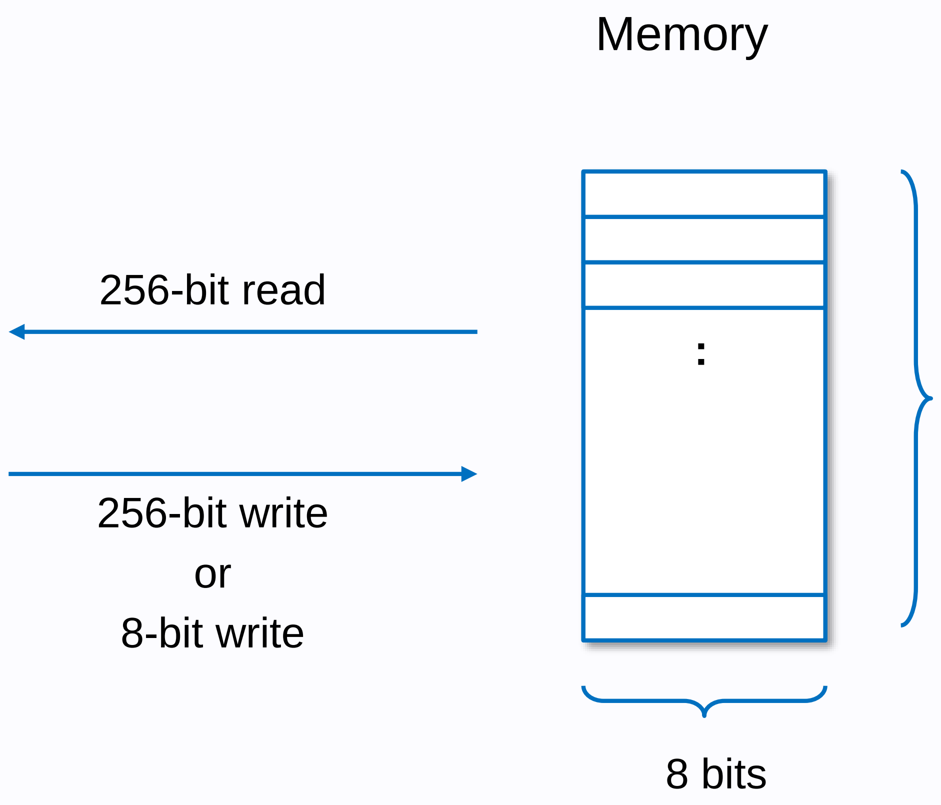
Key 1	Value 1
Key2	Value 2
:	:
Key n	Value n

Storage is a key-value store that maps 256-bit words to 256-bit words.

Access with SSTORE/SLOAD instructions.

References : [E7] Ch.7

Memory

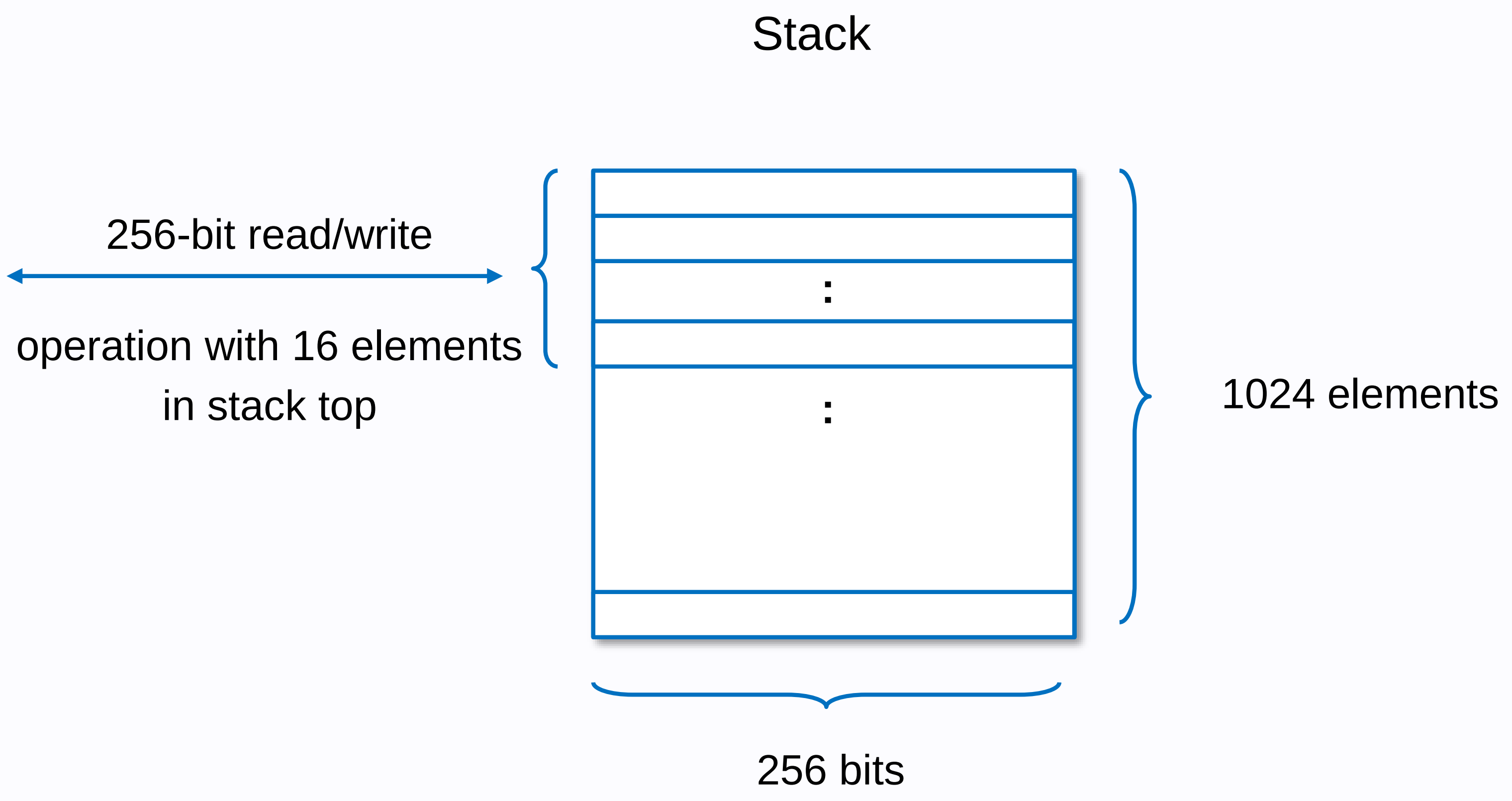


Memory is linear and can be addressed at byte level.

Access with MSTORE/MLOAD instructions.

References : [E7] Ch.7

Stack

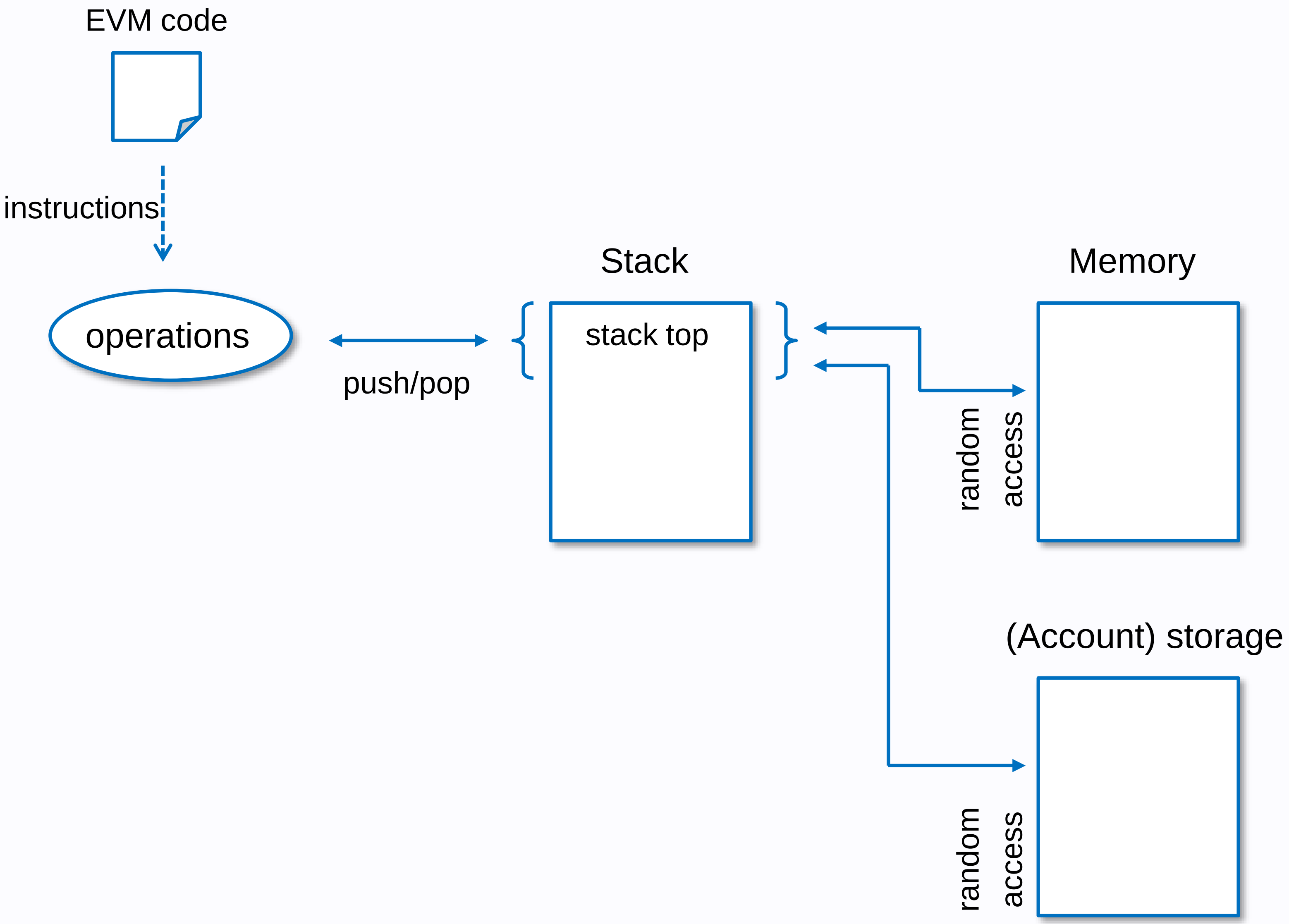


All operation are performed on the stack.

Access with many instructions such as PUS/POP/COPY/SWAP, ...

References : [E7] Ch.7

Execution model



:

References : [C14], [C6], [2], [C17], [8], [S15], [S16], [S11]

2. Virtual machine

Instruction set

Instruction set

Basically, 256-bit operation.

Create contract

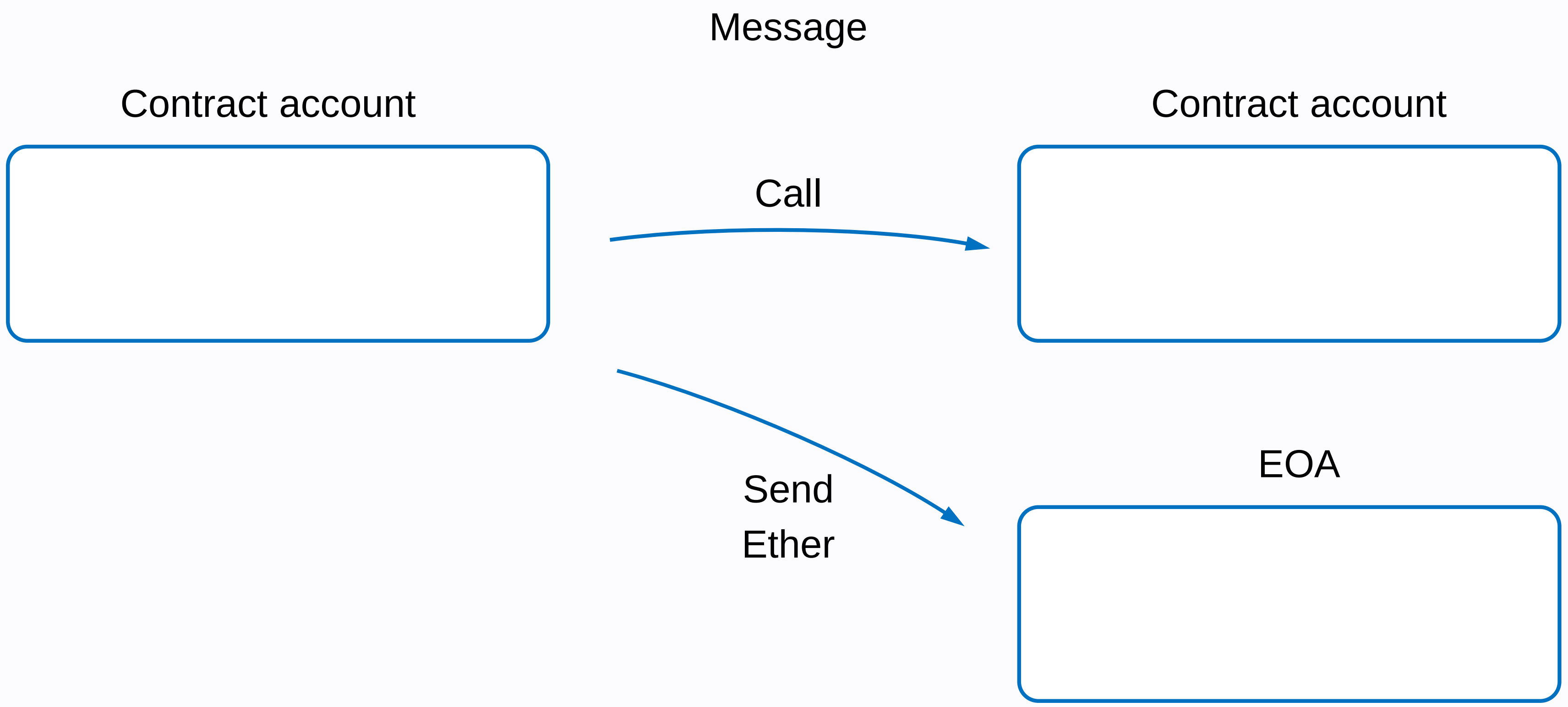
Self-destruct

References : [C14], [C6], [2], [C17], [8], [S15], [S16], [S11]

2. Virtual machine

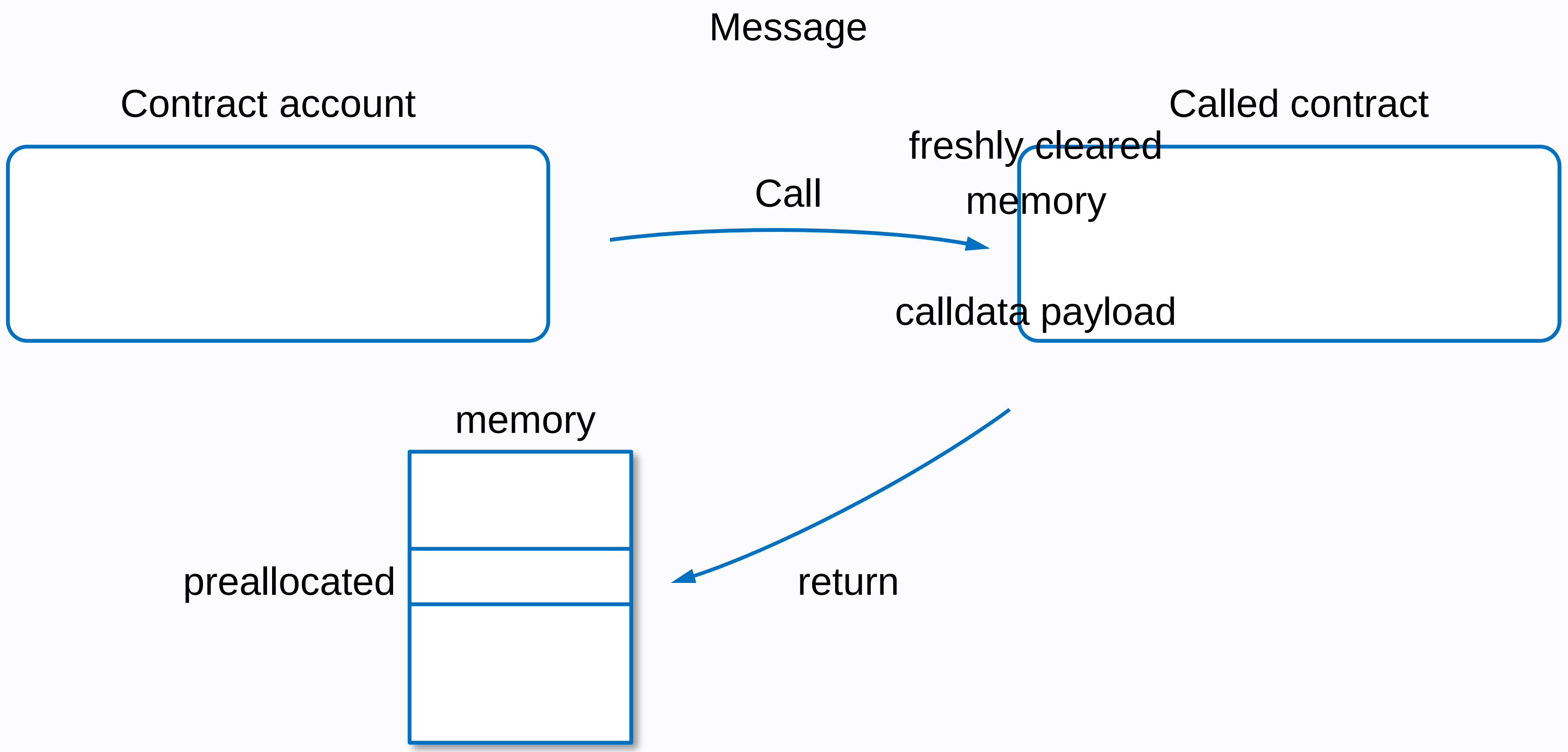
Message call

Message call



References : [E7]

Message call

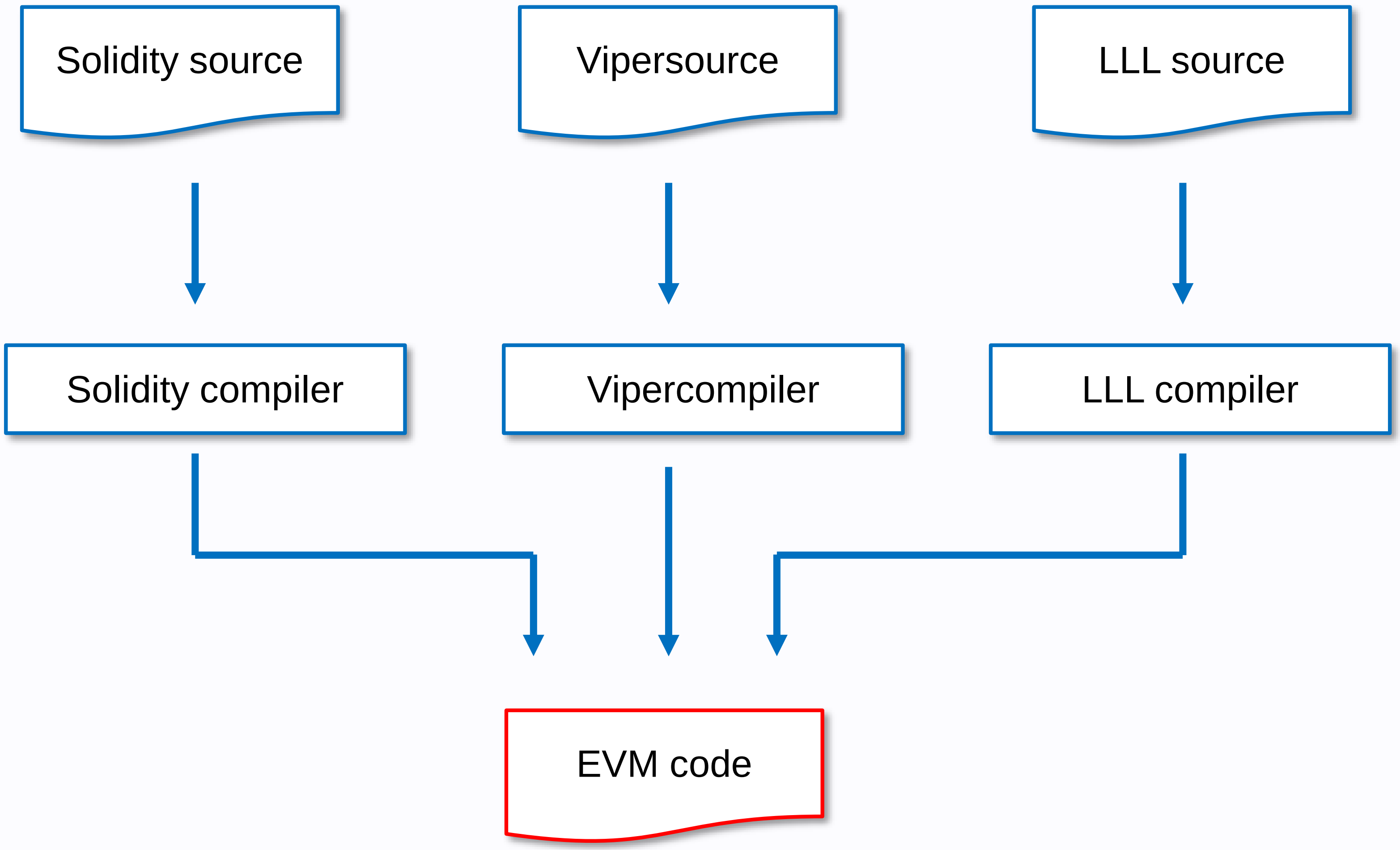


References : [E7]

2. Virtual machine

Code generation

EVM code generation



Ethereum virtual machine code

References : [1], [C1], [C3], [C10], [C19], [S7]

2. Virtual machine

Gas

Gas

All programmable computation is subject to fees.
Programmable computation has const in terms of gas.

creating contracts
making message calls
utilizing and accessing storage
executing operation on the VM

a measurement roughly equivalent to computational steps. [E2]

References : [E1] Ch.5, [E2]

2. Virtual machine

WASM

WASM

WASM

next generation VM

References : @@@

3. References

References

[E1] Ethereum Yellow Paper
ETHEREUM: A SECURE DECENTRALISED GENERALISED TRANSACTION LEDGER
https://ethereum.github.io/yellowpaper/paper.pdf

[E2] Glossary
https://github.com/ethereum/wiki/wiki/Glossary

[E3] White Paper
A Next-Generation Smart Contract and Decentralized Application Platform
https://github.com/ethereum/wiki/wiki/White-Paper

[E4] Design Rationale
https://github.com/ethereum/wiki/wiki/Design-Rationale

[E5] Ethereum Development Tutorial
https://github.com/ethereum/wiki/wiki/Ethereum-Development-Tutorial

[E6] Ethereum Introduction
https://github.com/ethereum/wiki/wiki/Ethereum-introduction

[E7] Solidity Documentation
https://media.readthedocs.org/pdf/solidity/develop/solidity.pdf
https://solidity.readthedocs.io/en/develop/

[E8] Web3 JavaScript app API for 0.2x.x
https://github.com/ethereum/wiki/wiki/JavaScript-API

References

[W1] Awesome Ethereum Virtual Machine
https://github.com/pirapira/awesome-ethereum-virtual-machine

[W2] Diving Into The Ethereum VM
https://blog.qtum.org/diving-into-the-ethereum-vm-6e8d5d2f3c30

[W3] Stack Exchange: Ethereum block architecture
https://ethereum.stackexchange.com/questions/268/ethereum-block-architecture/6413#6413

References

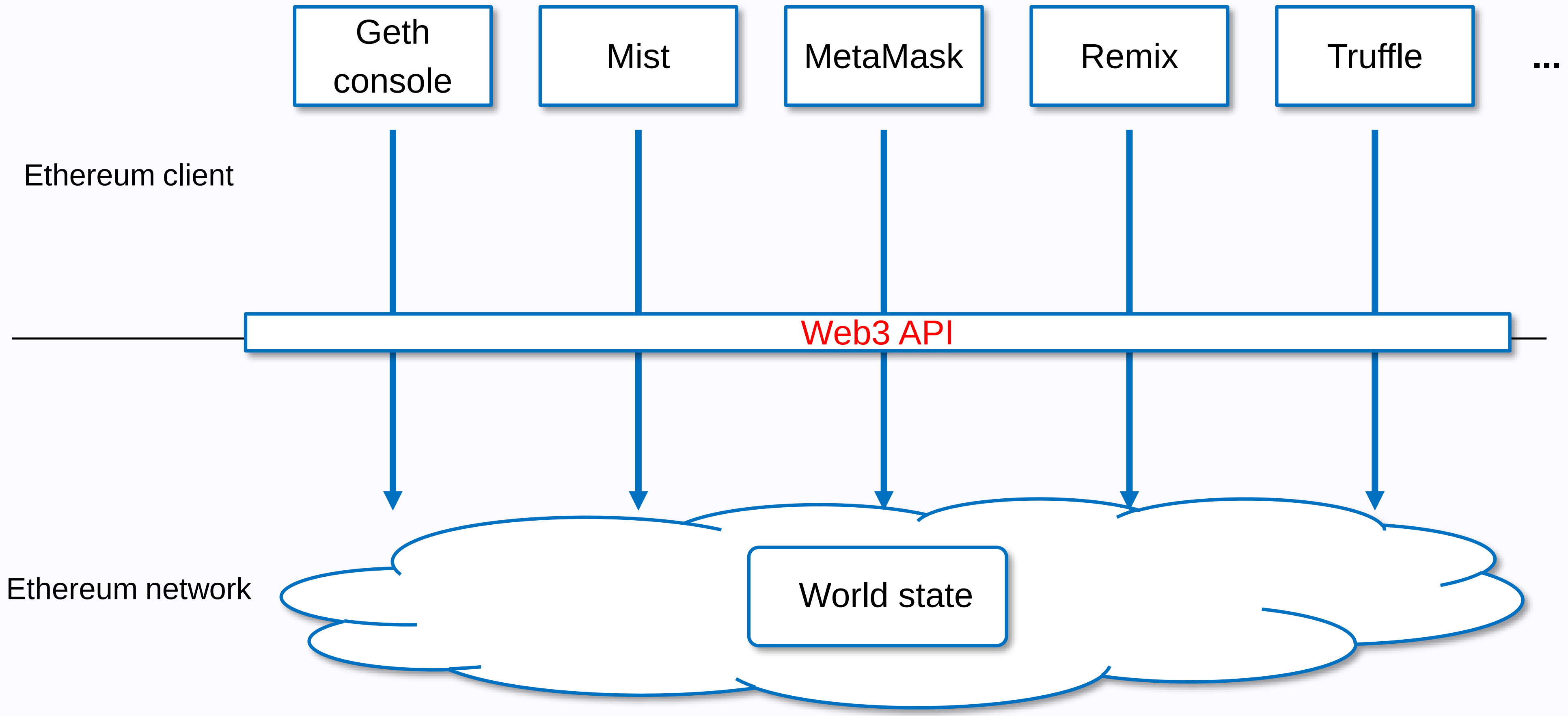
[C1] Go Ethereum
https://github.com/ethereum/go-ethereum

Appendix A

Appendix A

Web3 API

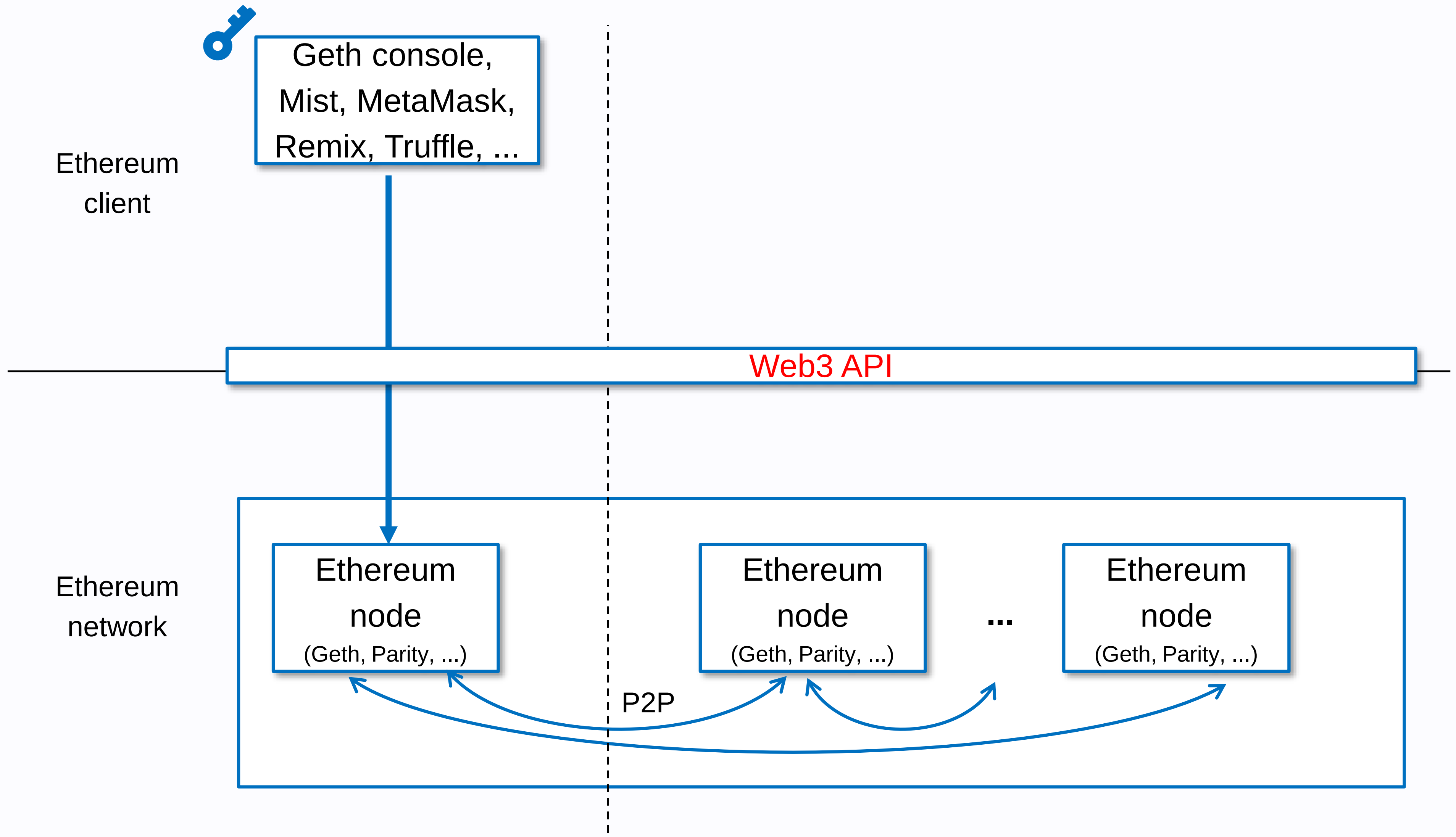
Web3 API and client



Ethereum clients access to Ethereum network via Web3 API.

References : @@@

Web3 API and client


Geth console,
Mist, MetaMask,
Remix, Truffle, ...
Ethereum client
Web3 API
Ethereum network
Ethereum node (Geth, Parity, ...)
Ethereum node (Geth, Parity, ...)
...
Ethereum node (Geth, Parity, ...)
P2P

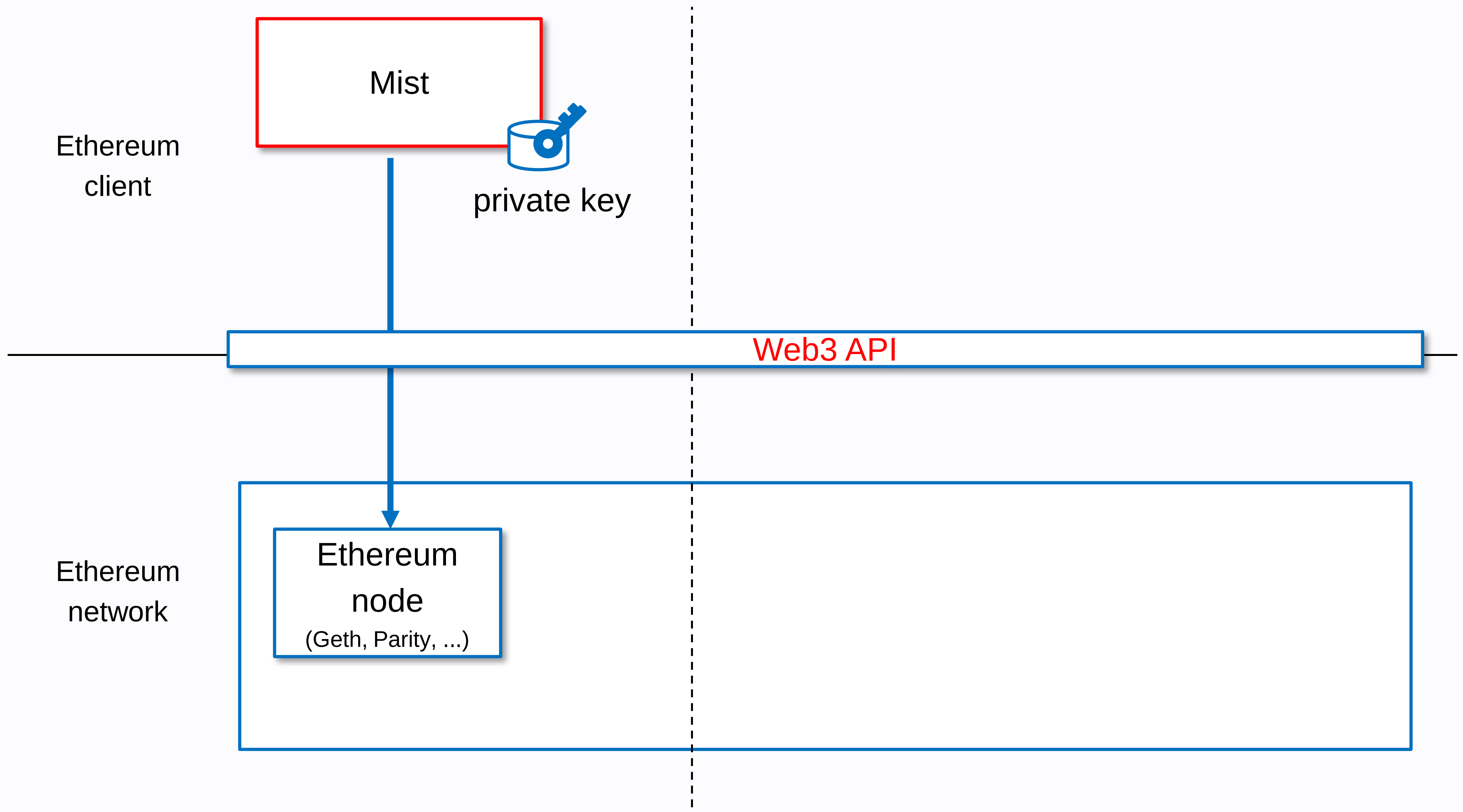
Ethereum clients access to Ethereum network via Web3 API.

References : @@@

Appendix A

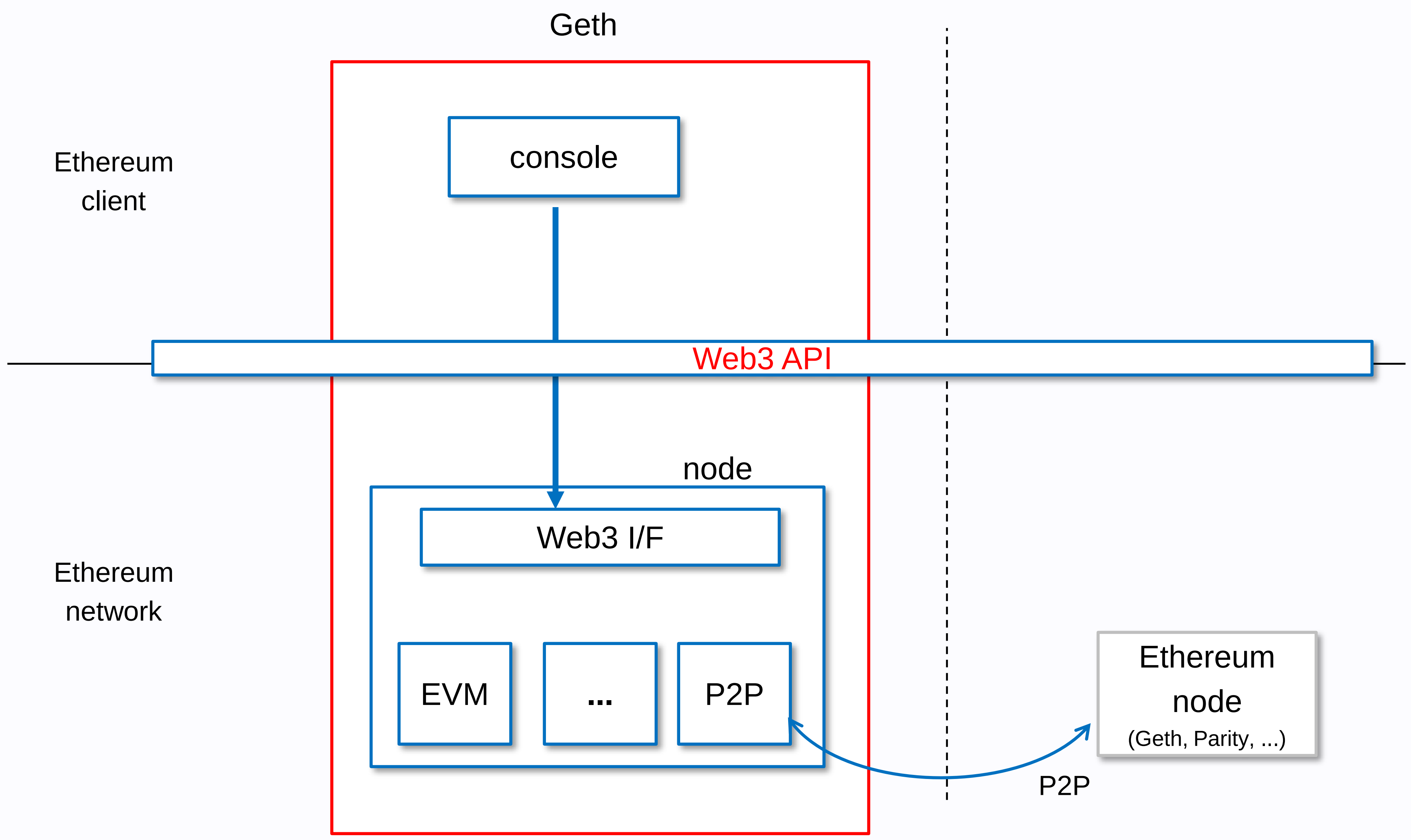
Ethereum implementations

Mist



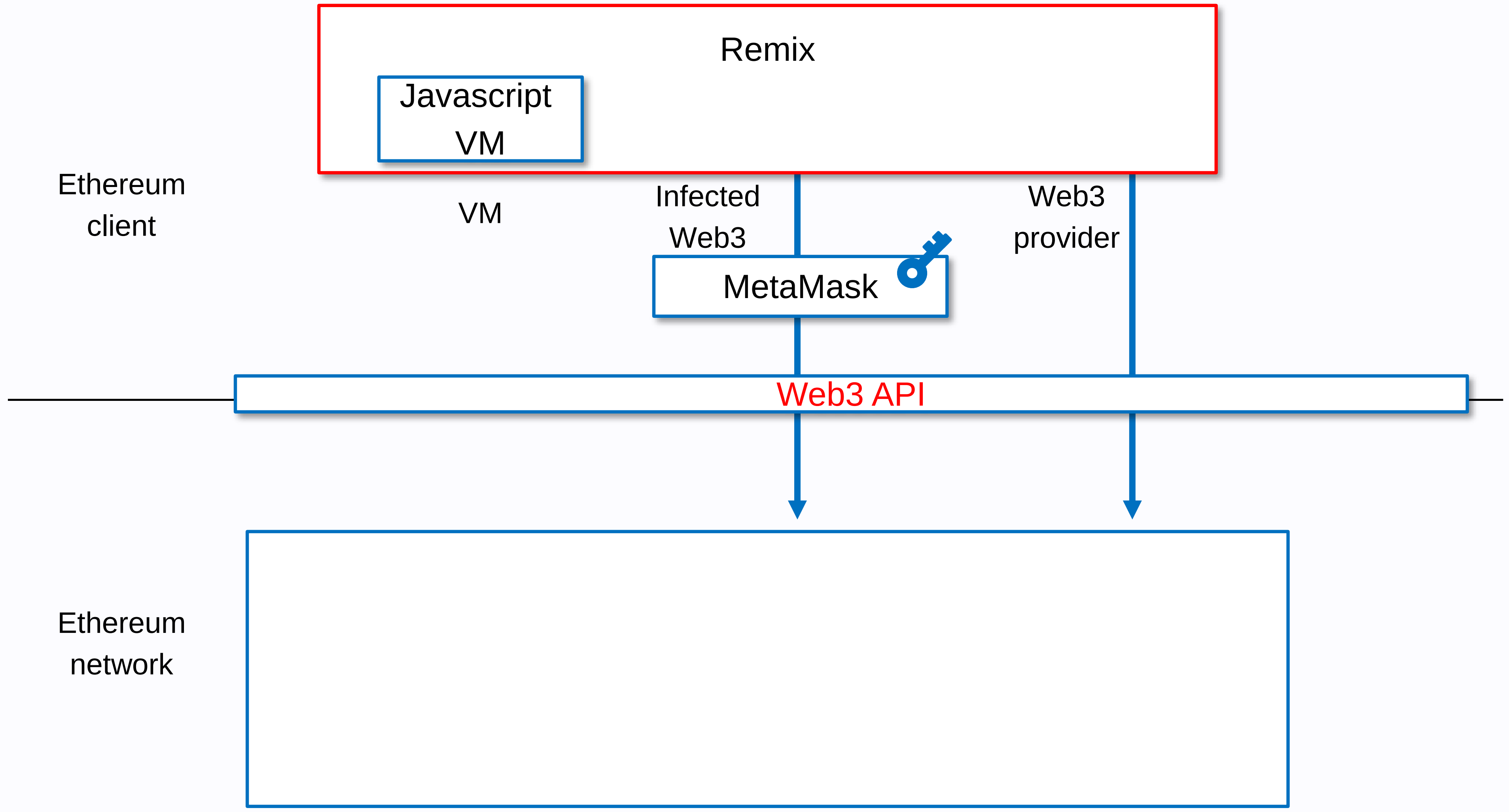
References : @@@

Geth



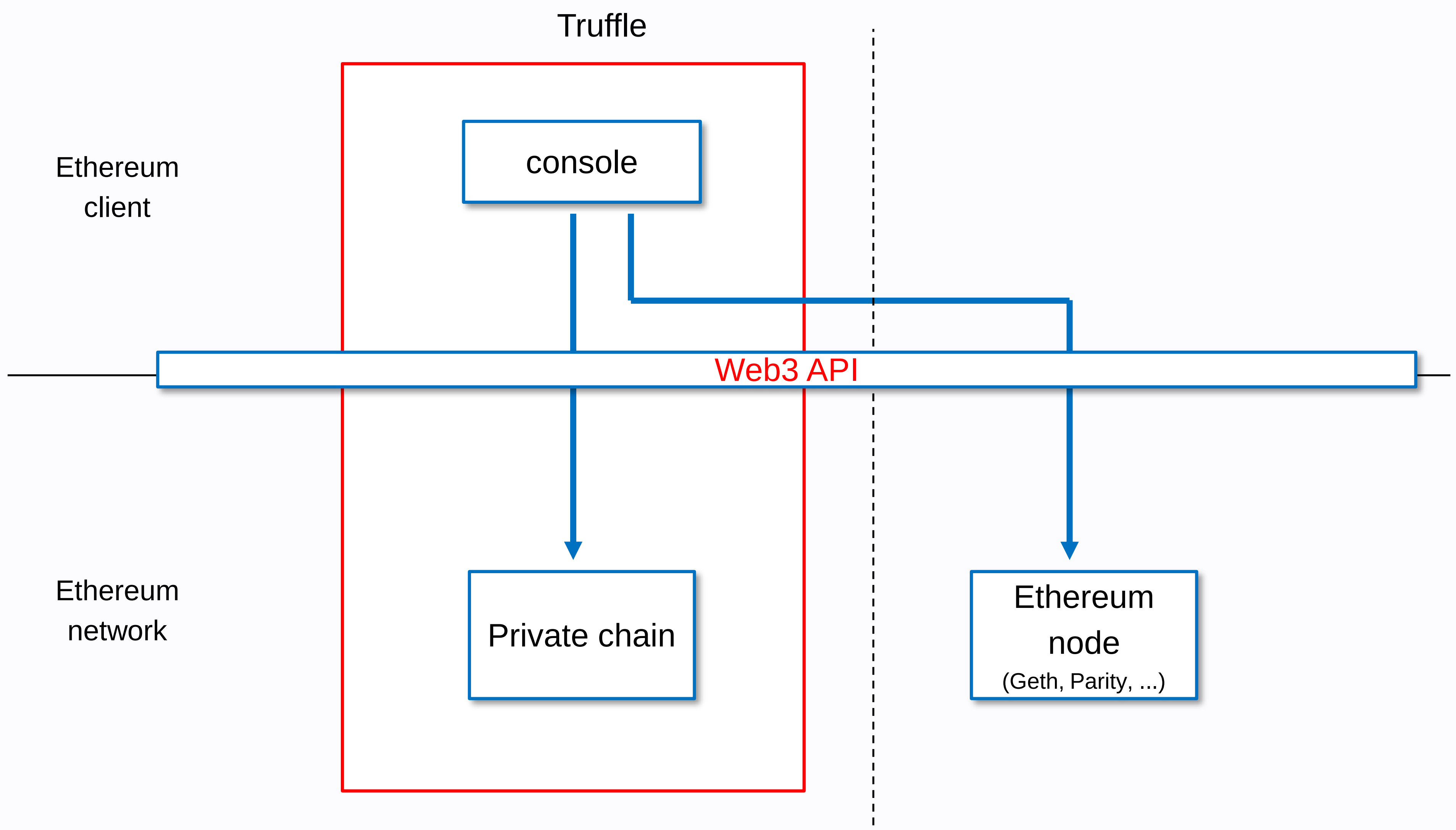
References : @@@

Remix



References : @@@

Truffle



References : @@@