ethereum

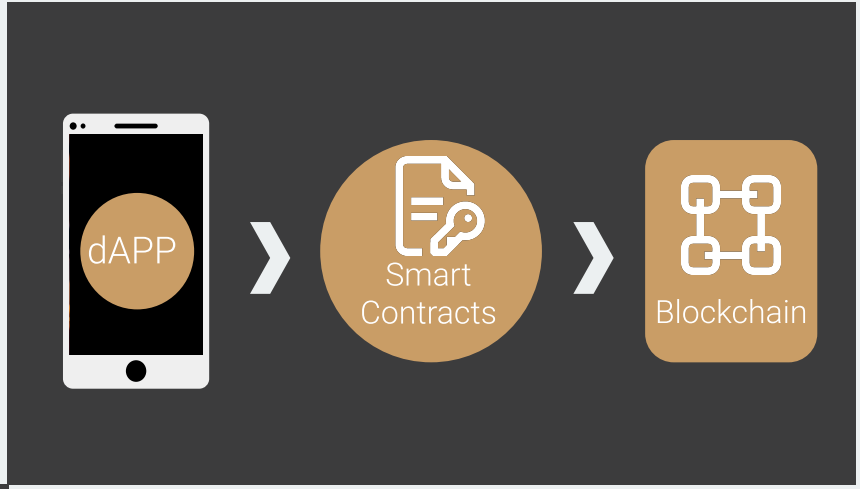
dApp Development

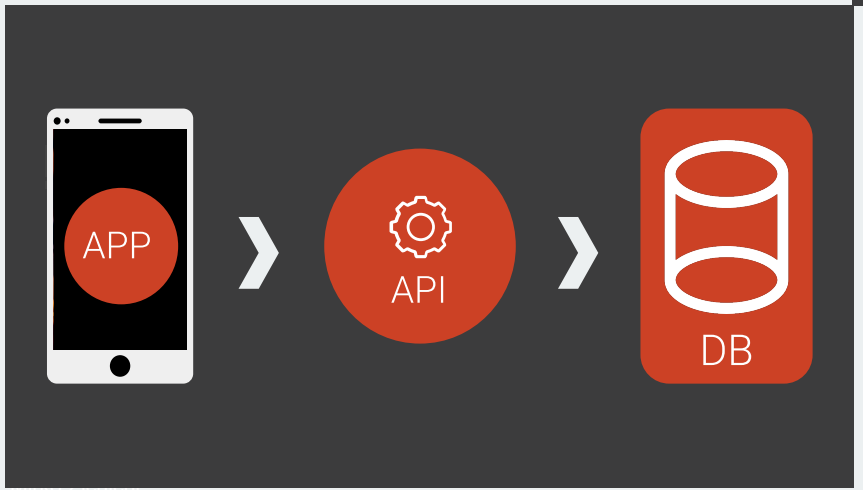
Mirko Zichichi - mirko.zichichi2@unibo.it

Slides by Mirko Zichichi and Michele Contu

DECENTRALIZED APPLICATIONS

Blockchain-based user-facing interfaces which connect the end user to the technology through a combination of underlying Smart Contracts.





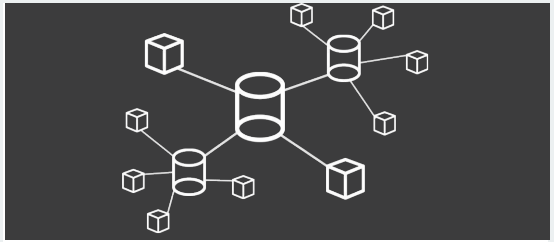
The relationship between dApps, Smart Contracts and the Blockchain is similar to traditional web applications. They render a particular page by using particular platform API to access its database.

Similarly, dApps use Smart Contracts in order to connect to the particular Blockchain upon which they are based.

DEFINITION OF DECENTRALIZED

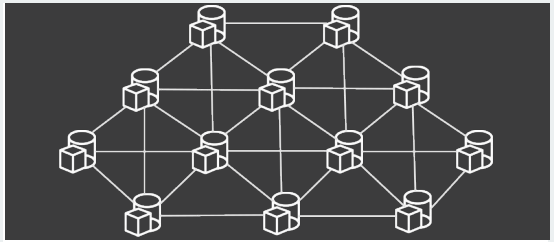
Centralized

One node does everything



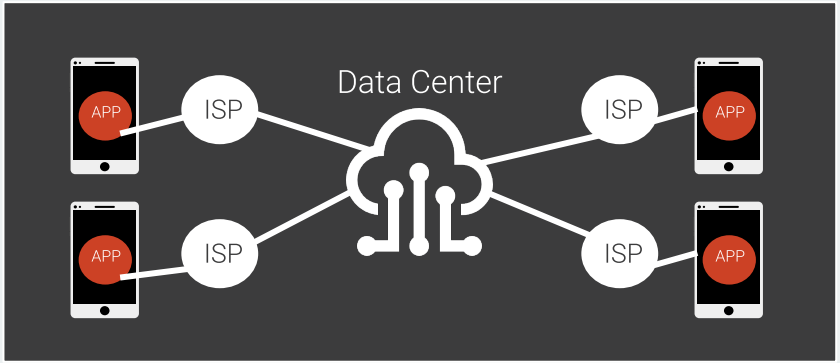
Distributed

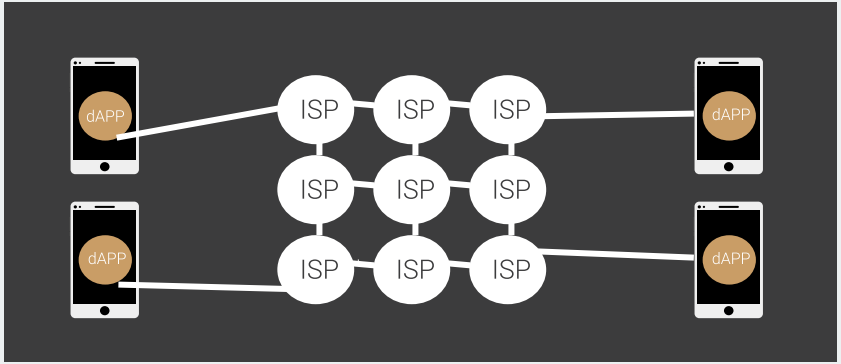
Nodes distribute work to sub-nodes



Decentralized

Nodes are only connected to peers





WEB 3.0 Stack

L4

(9) Protocol-extensible user-interface cradle ("browser")

L3

(8) Protocol-extensible developer APIs & languages

L2

(7) Second layer protocols

(7.1)	(7.2)	(7.3)	(7.4)	(7.5)	(7.6)	(7.7)	(7.x)
State channels	Plasma protocols	Encrypted storage	Storage incentivisation	Heavy computation	Distributed secret management	Oracles	...

L1

(5) Zero/low-trust interaction protocols

(3) Data distribution protocols

(4) Zero/low-trust interaction platforms (shared security)

(6) Transient data pub/sub messaging

L0

(1) Peer-to-peer (p2p) internet overlay protocols

(2) Platform-neutral computation description language

Brave

MetaMask

Web3.js

Solidity

Rust

Different optional protocols

(5) Bitcoin

(5) Ethereum

(3) IPFS

(1) libp2p

(2) Ethereum Virtual Machine

USE CASES FOR DAPPS AND PROJECTS

Dapps and Smart Contracts can **build on each other**. The ecosystem can grow exponentially instead of just linearly.

EtherDelta

Decentralized Exchange

Augur

Prediction Market

Golem

Distributed Computing

uPort

Identity Management

Digix

Digital Asset Management

Slock.it

Internet of Things

Technologies/Frameworks

- **Truffle** and **Open Zeppelin** for contracts development
- **Ganache** as Blockchain Emulator
- **Metamask** as hot-wallet
- **Django** for the implementation of the platform
- **Web3.js** and **Web3.py** for frontend calls to deployed contracts

Truffle Framework

Truffle is a world class development environment, testing framework and asset pipeline for blockchains using the Ethereum Virtual Machine.

- Built-in smart contract compilation, linking, deployment and binary management.
- Automated contract testing for rapid development.
- Scriptable, extensible deployment & migrations framework.
- Network management for deploying to any number of public & private networks.
- Interactive console for direct contract communication.
- Configurable build pipeline with support for tight integration.
- External script runner that executes scripts within a Truffle environment.

Open Zeppelin

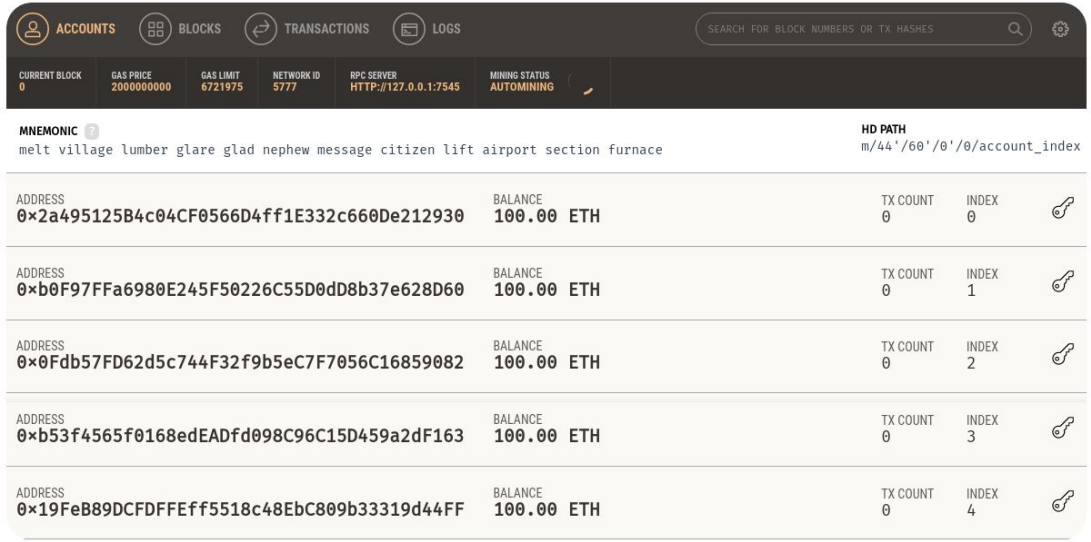
OpenZeppelin

OpenZeppelin is a battle-tested framework of reusable smart contracts for Ethereum and other EVM and eWASM blockchains.

- Focused on security: using industry standard contract security patterns and best practices.
- Modular approach: simple code, only basics. Easy collaboration and auditing.
- Open source: community driven. Used by multiple organizations and individuals.

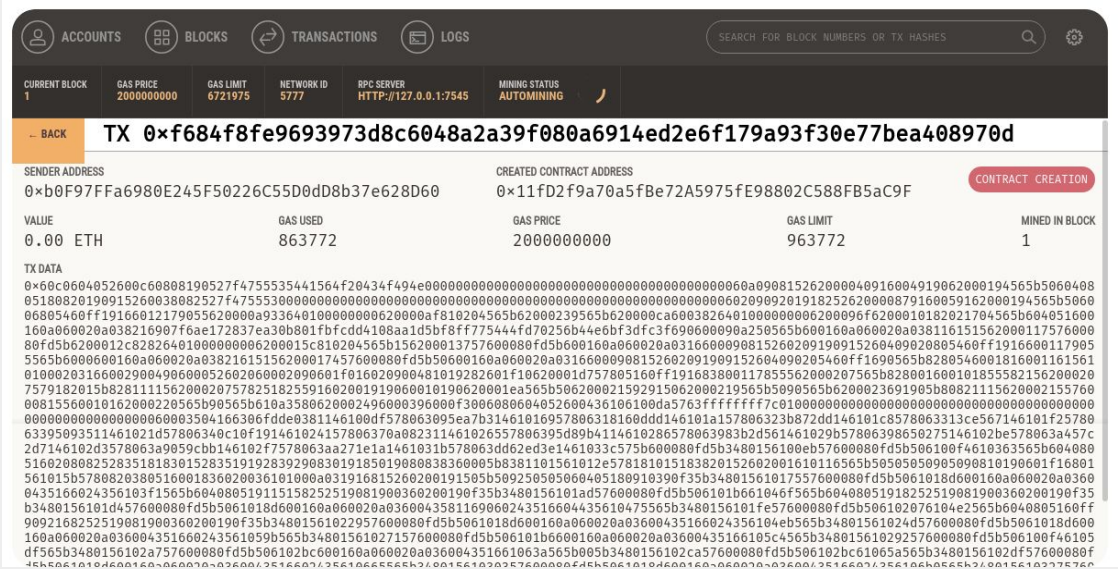
Blockchain emulator: Ganache

Ganache is a standalone full Blockchain emulator that runs locally. When is runned, it's possible to connect to the network through the socket: http://127.0.0.1:7545, with the help of Web3.



Blockchain emulator: Ganache

Ganache is very useful for developing and testing, it's possible to examine all blocks and transactions present in the blockchain. Also offers the possibility to see the log output of the entire operations that occurs into it.



An example of a transaction deployment into Ganache

The Metamask hot wallet

MetaMask is an extension for accessing Ethereum enabled distributed applications by the normal browser.

The extension injects the Ethereum web3 API into every website's javascript context, so that dapps can read from the blockchain.

MetaMask also lets the user create and manage their own identities, so when a Dapp wants to perform a transaction and write to the blockchain, the user gets a secure interface to review the transaction, before approving or rejecting it.

Web3.js & Web3.py

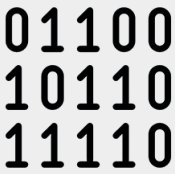
The web3.js library is a collection of modules which contain specific functionality for the ethereum ecosystem. In particular:

- The web3-eth is for the ethereum blockchain and smart contracts
- The web3-utils contains useful helper functions for Dapp developers.

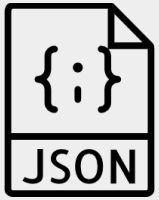
Web3.py is instead a Python library for interacting with Ethereum. Its API is derived directly from the Web3.js Javascript API.

Contract's files

Contract Definition: Formal definition in high-level code (e.g. Solidity).



Compiled Contract: the contract converted to byte-code to run on the Ethereum Virtual Machine (EVM). Note the function names and input parameters are hashed during compilation. Therefore, for another account to call a function, it must first be given the function name and arguments.



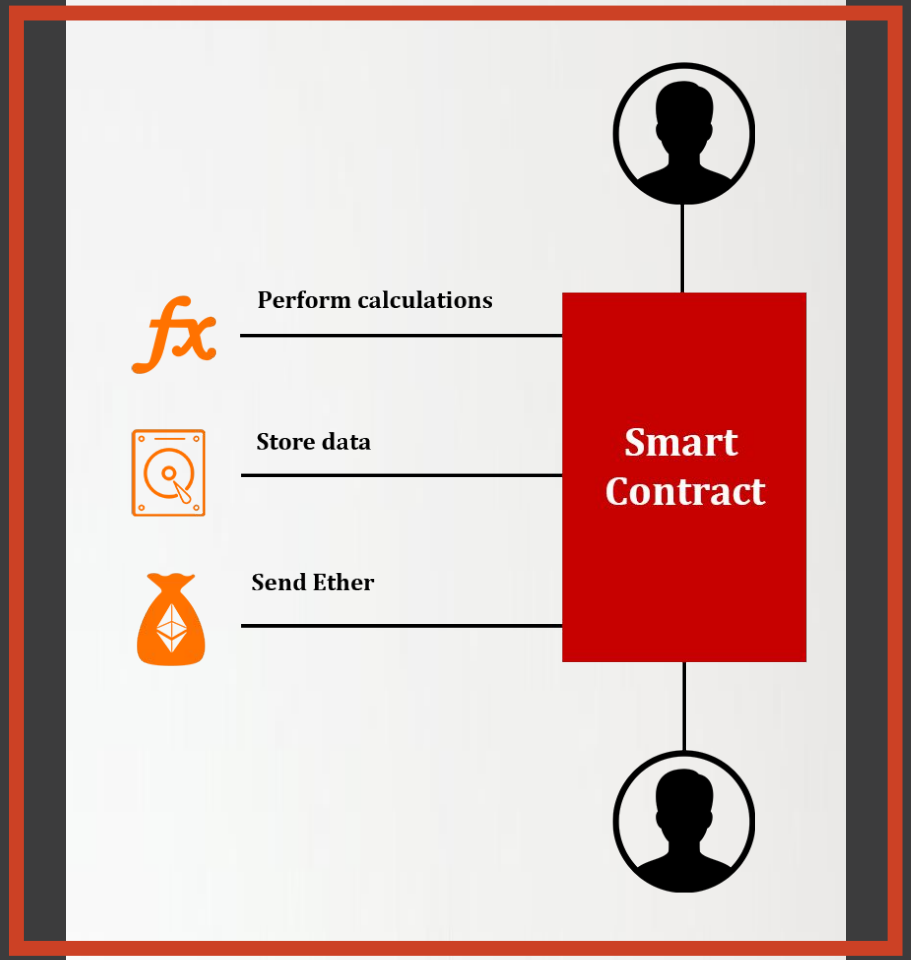
Application Binary Interface - ABI: A list of the contract's functions and arguments (in JSON format). An account wishing to use a smart contract's function uses the ABI to hash the function definition so it can create the EVM bytecode required to call the function. This is interpreted by the EVM with the code at the target account (the address of the contract).

Smart Contracts Peculiarities

Smart contracts are autonomous.

There are not controlled by anyone.

They self-executed based on a set of instructions that two parties have agreed to.



IDEA:
We are used to think that a Like cannot help an African child.

IDEA:
We are used to think that a Like cannot help an African child.

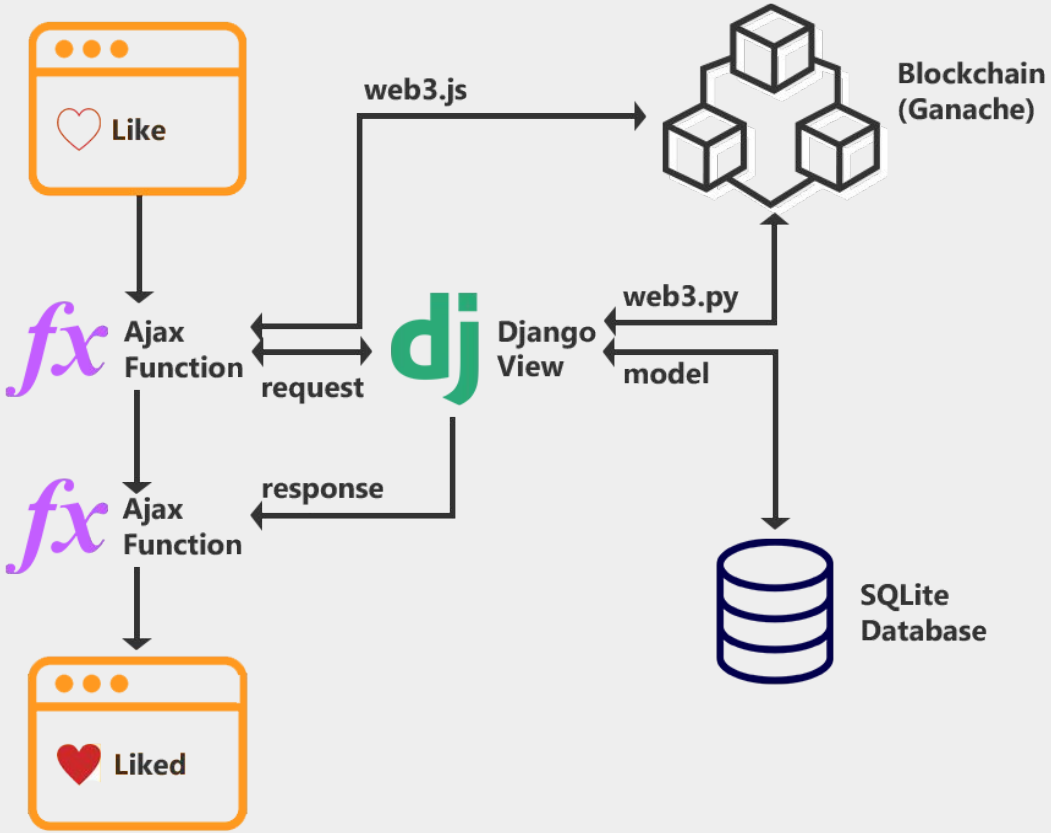
What if it can?

LikeStarter

Social Network that allows users to raise funds for other users through a simple Like.

https://arxiv.org/abs/1905.05560

Anatomy of communication



LikeStarter Contracts

Likoin

Standard ERC20 Token used for Crowdfunding.
It acts as a Token but also as a Share, expressing the ownership relationship between the crowdfunding beneficiary and the token holder.

Crowdsale

Simple contract for Crowdfunding that allows to buy Likoin token transfering ETH to the beneficiary.

ArtifactsManager

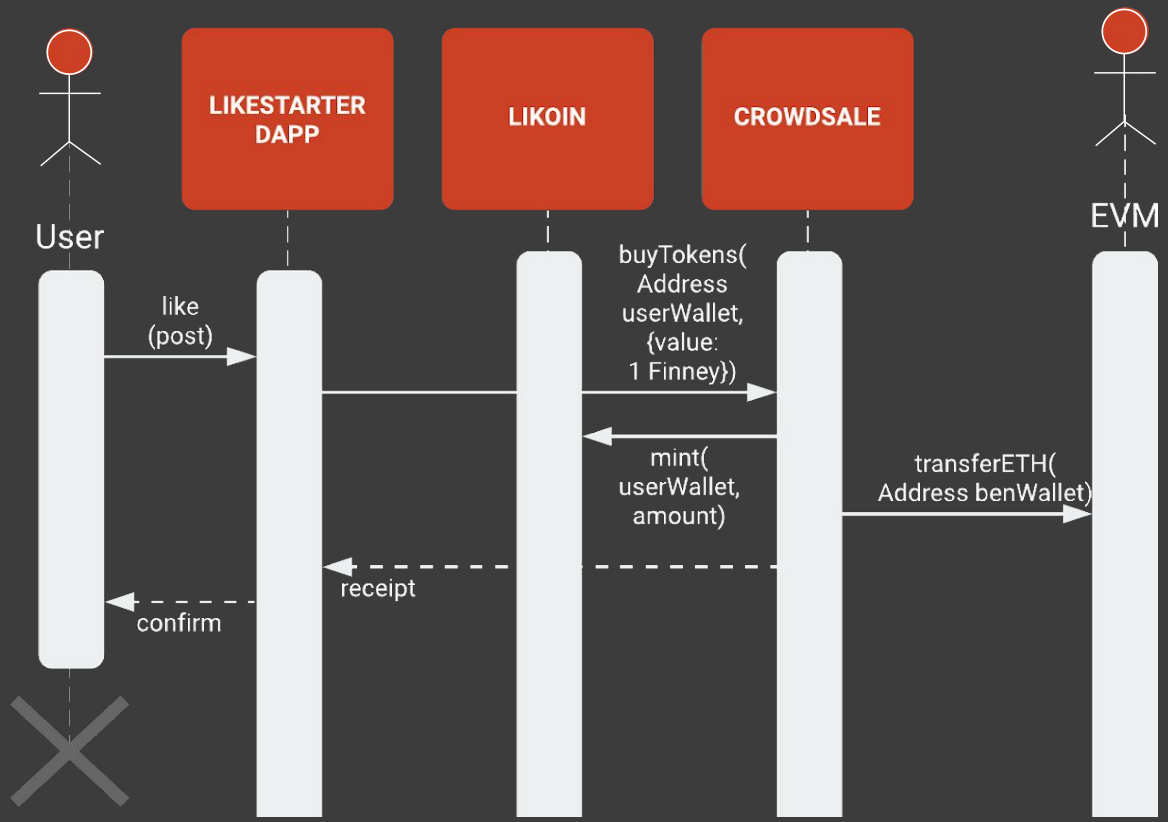
A contract that allows the crowdfunding beneficiary to offer artifacts that can be traded for Bucks.
Buck is another type of token that can be only traded with an artifact and has no monetary value.

Voting

A Decentralized Autonomous Organization implementation, to allow Likoin holders to vote for a price to give to an artifact.

https://github.com/flamel13/eth-crowdsale/tree/master/LikeStarterTruffle/contracts

Like


User
LIKESTARTER DAPP
LIKOIN
CROWDSALE
EVM
like (post)
buyTokens(Address userWallet, {value: 1 Finney})
mint(userWallet, amount)
transferETH(Address benWallet)
receipt
confirm