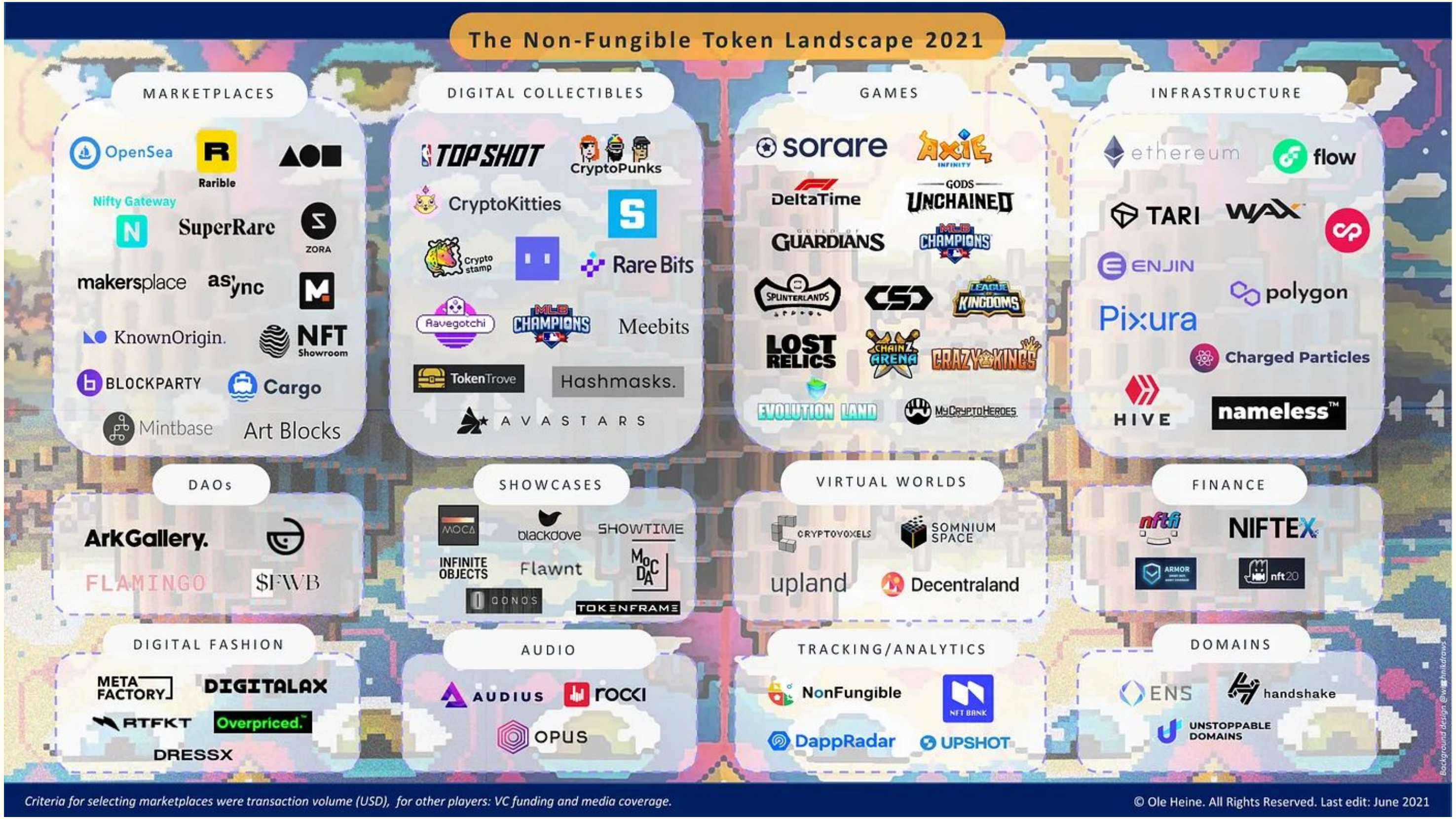
Lecture 7

NFTs & Account Abstraction

NFT Landscape



Criteria for selecting marketplaces were transaction volume (USD), for other players: VC funding and media coverage.

© Ole Heine. All Rights Reserved. Last edit: June 2021

NFT Standards explosion

Use Case

ERC-6066: Signature Validation Method for NFTs (Access Verification)

ERC-4955: Vendor Metadata Extension for NFTs (3D models)

ERC-4519: Non-Fungible Tokens Tied to Physical Assets (IOT)

Manipulating Fungibility

ERC-1155: Multi Token Standard

ERC-3525: Semi-Fungible Token

ERC-721: Non-Fungible Token Standard

Marketplace

ERC-2309: ERC-721 Consecutive Transfer Extension

ERC-2981: NFT Royalty Standard

ERC-4910: Royalty Bearing NFTs

Rental and Services

ERC-4907: Rental NFT, an Extension of EIP-721

ERC-5007: Time NFT, ERC-721 Time Extension

NFT Speedrun

The ERC721 Standard

https://eips.ethereum.org/EIPS/eip-721

```
function name() public view returns (string)

function symbol() public view returns (string)
```

```
function balanceOf(address _owner) external view returns (uint256);
function ownerOf(uint256 _tokenId) external view returns (address);
function safeTransferFrom(address _from, address _to, uint256 _tokenId, bytes data) external payable;
function safeTransferFrom(address _from, address _to, uint256 _tokenId) external payable;
function approve(address _approved, uint256 _tokenId) external payable;
function setApprovalForAll(address _operator, bool _approved) external;
function getApproved(uint256 _tokenId) external view returns (address);
function isApprovedForAll(address _owner, address _operator) external view returns (bool);
```

```
event Transfer(address indexed _from, address indexed _to, uint256 indexed _tokenId);
event Approval(address indexed _owner, address indexed _approved, uint256 indexed _tokenId);
event ApprovalForAll(address indexed _owner, address indexed _operator, bool _approved);
```

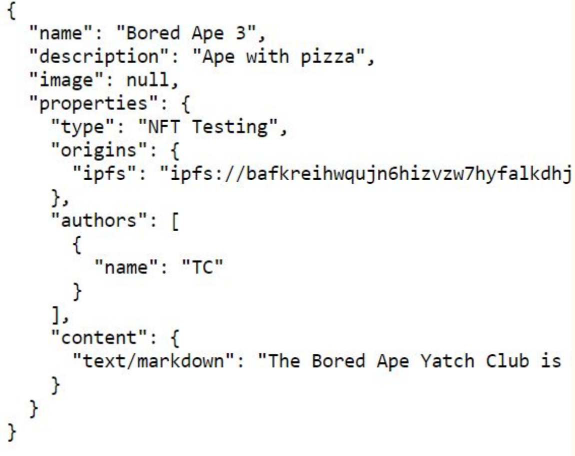
NFT Architecture - A visual

_setTokenURI()

Onchain?
IPFS?
HTTPS?

Metadata

Must be raw!



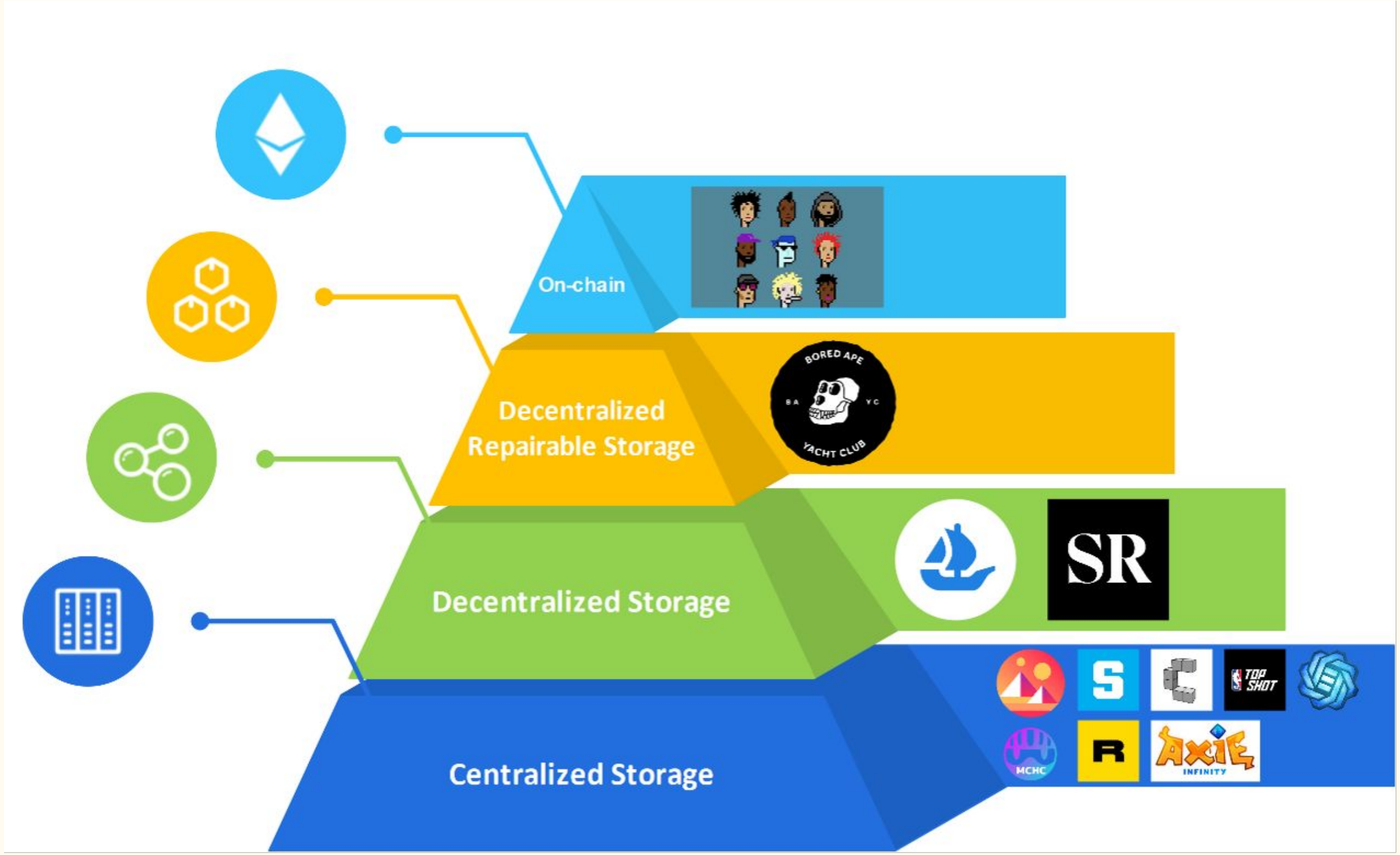
```
{
  "name": "Bored Ape 3",
  "description": "Ape with pizza",
  "image": null,
  "properties": {
    "type": "NFT Testing",
    "origins": {
      "ipfs": "ipfs://bafkreihwqujn6hizvzw7hyfalkdhj
    },
    "authors": [
      {
        "name": "TC"
      }
    ],
    "content": {
      "text/markdown": "The Bored Ape Yatch Club is
    }
  }
}
```

Frontend

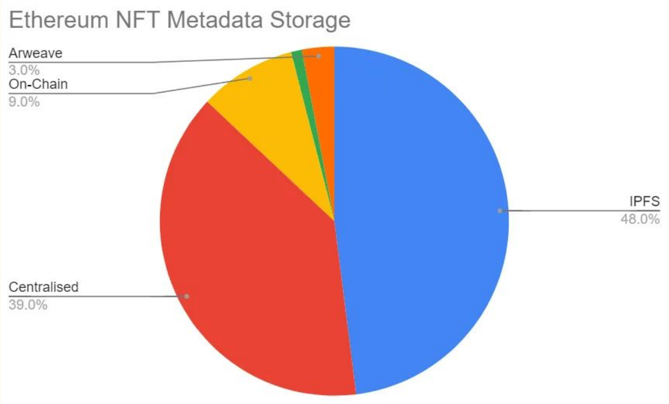
OpenSea

NFT Storage

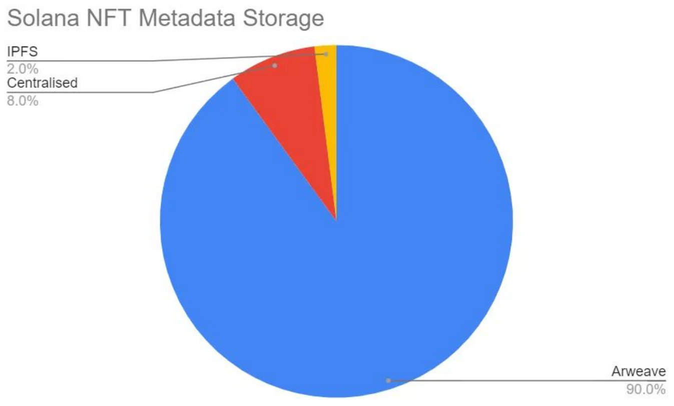
Storage Types



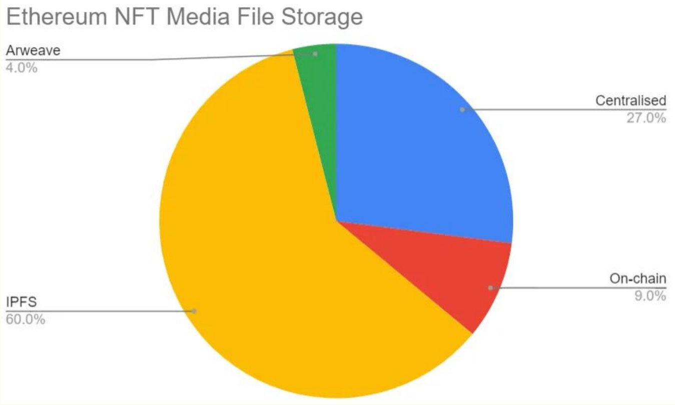
Storage Comparison



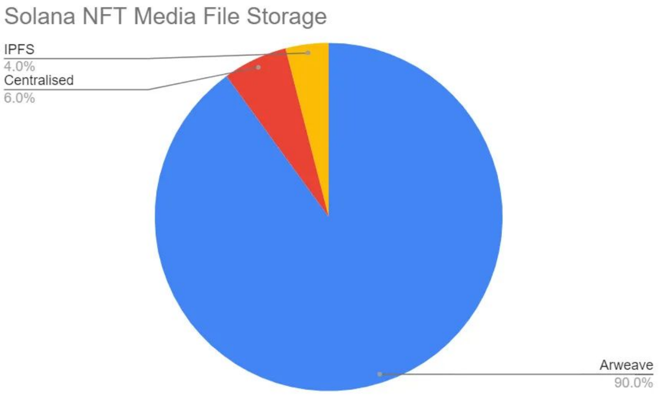
Ethereum NFT Metadata Storage



Solana NFT Metadata Storage



Ethereum NFT Media File Storage



Solana NFT Media File Storage

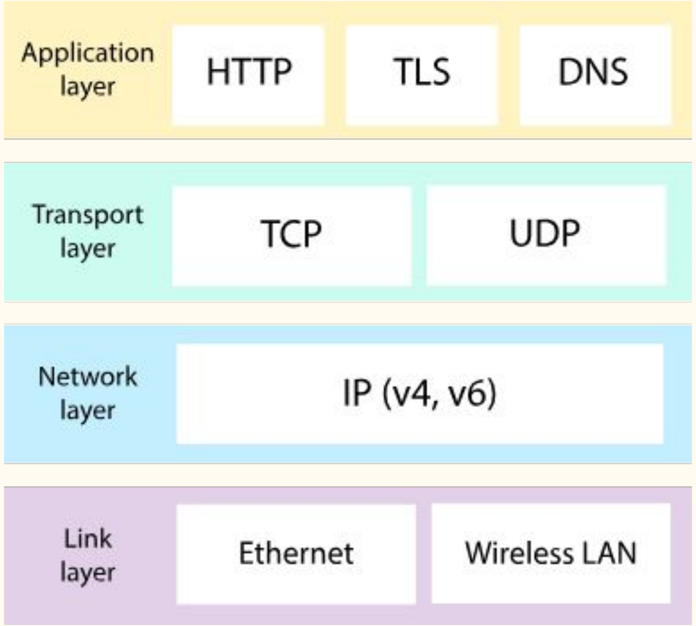
IPFS - Interplanetary Filesystem

Content Addressing + Network Agnostic

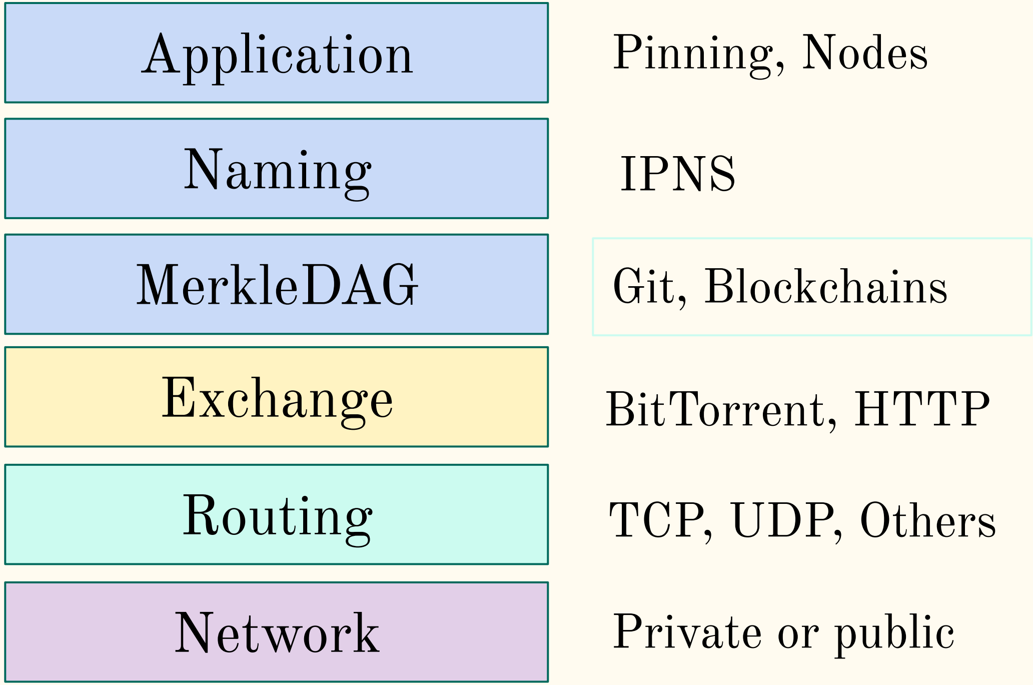
An IPFS Implementation:

- *MUST* support addressability using CIDs.
- *MUST* expose operations (eg. retrieval, provision, indexing) on resources using CIDs.
- *MUST* verify that the CIDs it resolves match the resources they address, at least when it has access to the resources' bytes.
- *SHOULD* name all the important resources it exposes using CIDs.
- *SHOULD* expose the logical units of data that structure a resource (eg. a CBOR document, a file or directory, a branch of a B-tree search index) using CIDs.
- *SHOULD* support incremental verifiability, notably so that it may process content of arbitrary sizes.
- *MAY* rely on any transport layer. The transport layer cannot dictate or constrain the way in which CIDs map to content.

Internet vs IPFS - Centralized vs Decentralized Web

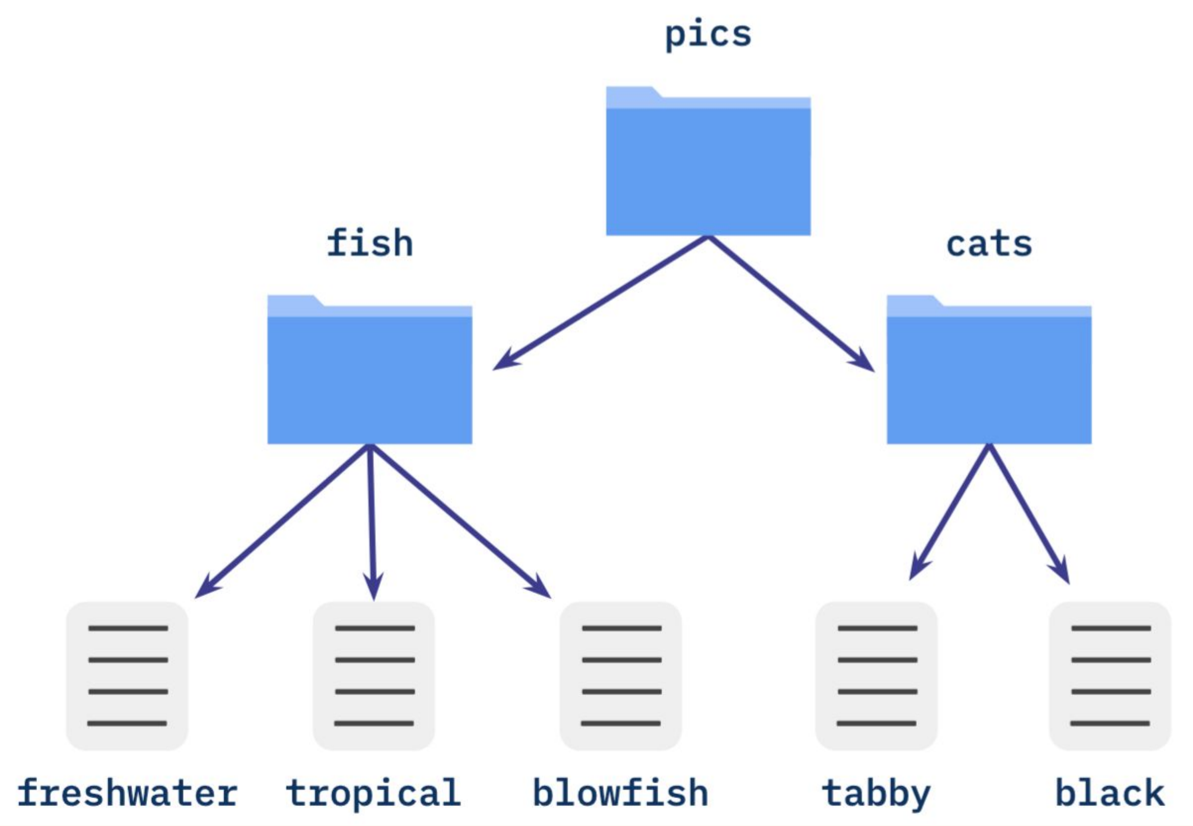


Application layer	HTTP	TLS	DNS
Transport layer	TCP	UDP	
Network layer	IP (v4, v6)		
Link layer	Ethernet	Wireless LAN	



Application	Pinning, Nodes
Naming	IPNS
MerkleDAG	Git, Blockchains
Exchange	BitTorrent, HTTP
Routing	TCP, UDP, Others
Network	Private or public

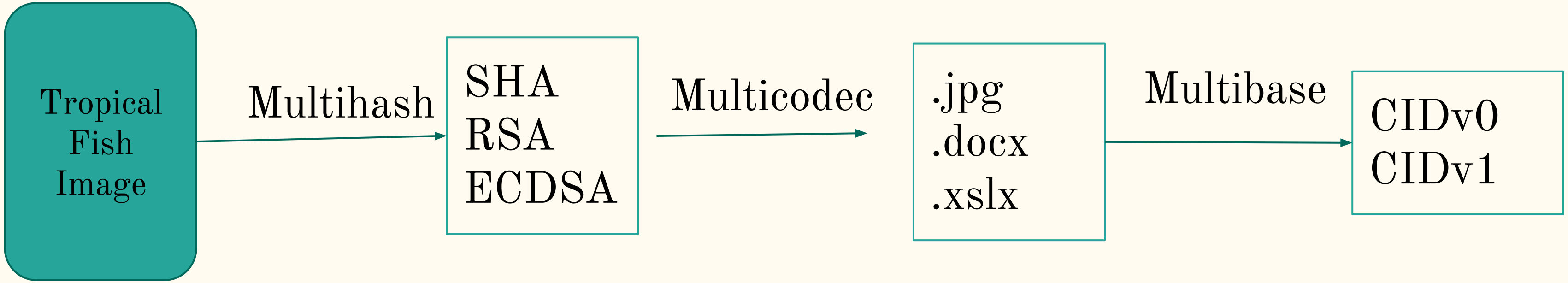
IPFS - Directed Acyclic Graphs (DAG)



Directed - Has definite direction / path
Acyclic - No circles, all paths are unique
Graph - Tree like data structure

Merkle Trees are hashed trees: CIDs!!

IPFS - CID



Tropical Fish Image

Multihash

SHA
RSA
ECDSA

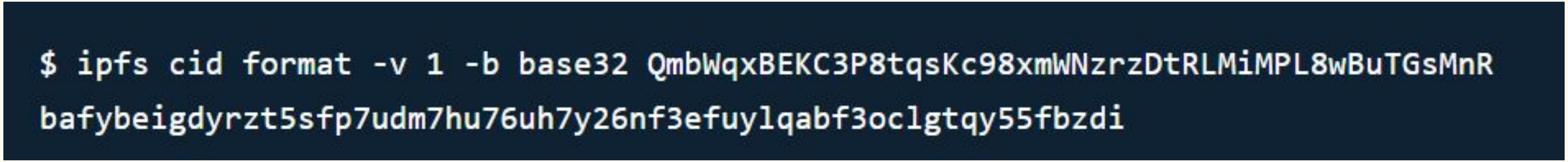
Multicodec

.jpg
.docx
.xslx

Multibase

CIDv0
CIDv1

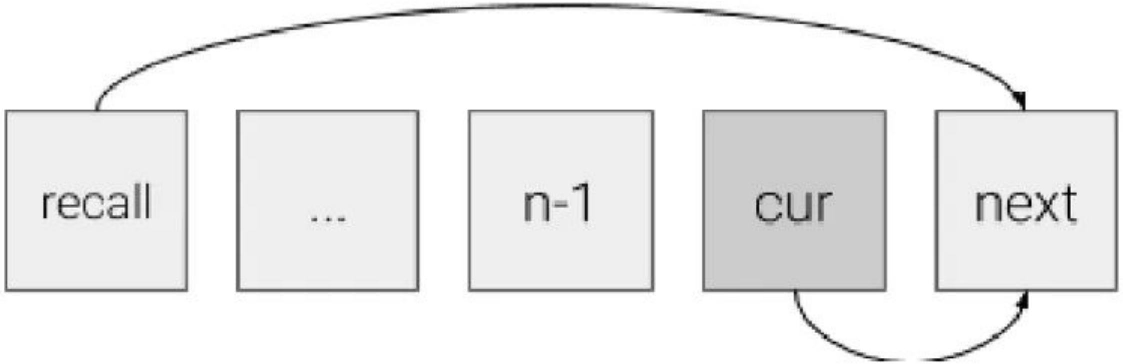
From V0 to V1:



```
$ ipfs cid format -v 1 -b base32 QmbWqxBEKC3P8tqsKc98xmWNzrzDtRLMiMPL8wBuTGsMnR
bafybeigdyrzt5sfp7udm7hu76uh7y26nf3efuylqabf3oclgtqy55fbzdi
```

Arweave - Permanent Data Storage

Figure 3: Blockweave structure and recall blocks



Recall block in blockweave structure

Source: Aweave whitepaper

- Users pay miners a once off fee
- Miners choose what information to incorporate.
- Proof of Access + blockhash list to prevent downloading entire ledger

NFT Extensions

Fractional, Soulbound, Composable

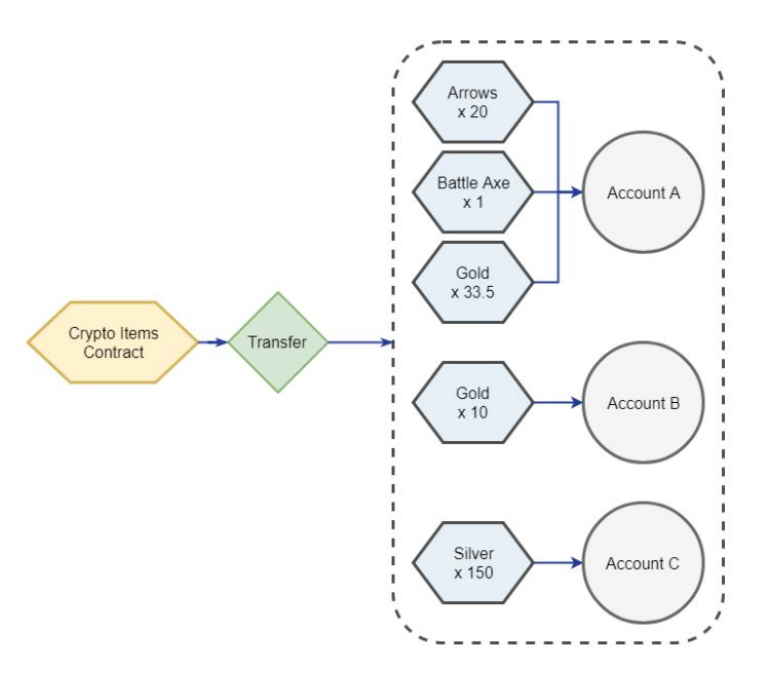
ERC1155 - Fractional NFTs

```
interface ERC1155 /* is ERC165 */ {

●    safeTransferFrom()
●    safeBatchTransferFrom();
●    balanceOf(address _owner, uint256 _id)
●    balanceOfBatch(address[] calldata _owners,
     uint256[] calldata _ids)
●    setApprovalForAll()
●    isApprovedForAll()
}
```

Limit contract sprawl - 450k contracts deployed

One contract for FT, NFT and SFT/fractional



```
interface ERC1155Mintable {

●    create()
     ○    Sets supply
●    mint()
     ○    Track supply
     ○    singleTransfer
●    setURI()
}
```

Soul Bound NFTs

- ERC721 - Just delete the transfer function
- ERC1155 - Similarly disable transfer
- ERC5484 - Consensual Soulbound Tokens
  - Agree to burn. May be issuer or owner triggered
- ERC5114 - Soulbound Badge
  - In review
  - Can be attached to existing NFTs
  - Makes normal NFTs soulbound.

- A non transferable and immutable record
  - Medical Records
  - Governmental Records
  - Certifications
- Ownership and verification
  - Loss?
  - Authenticity?
  - Value

cNFT / dNFT - Mutable NFTs

Composable - A base NFT which may have items added onto it.

Eg. A game character which gets weapons and armour

Dynamic - A general change which occurs to a base NFT

Automatic - Example, AI aging or growing an NFT animal

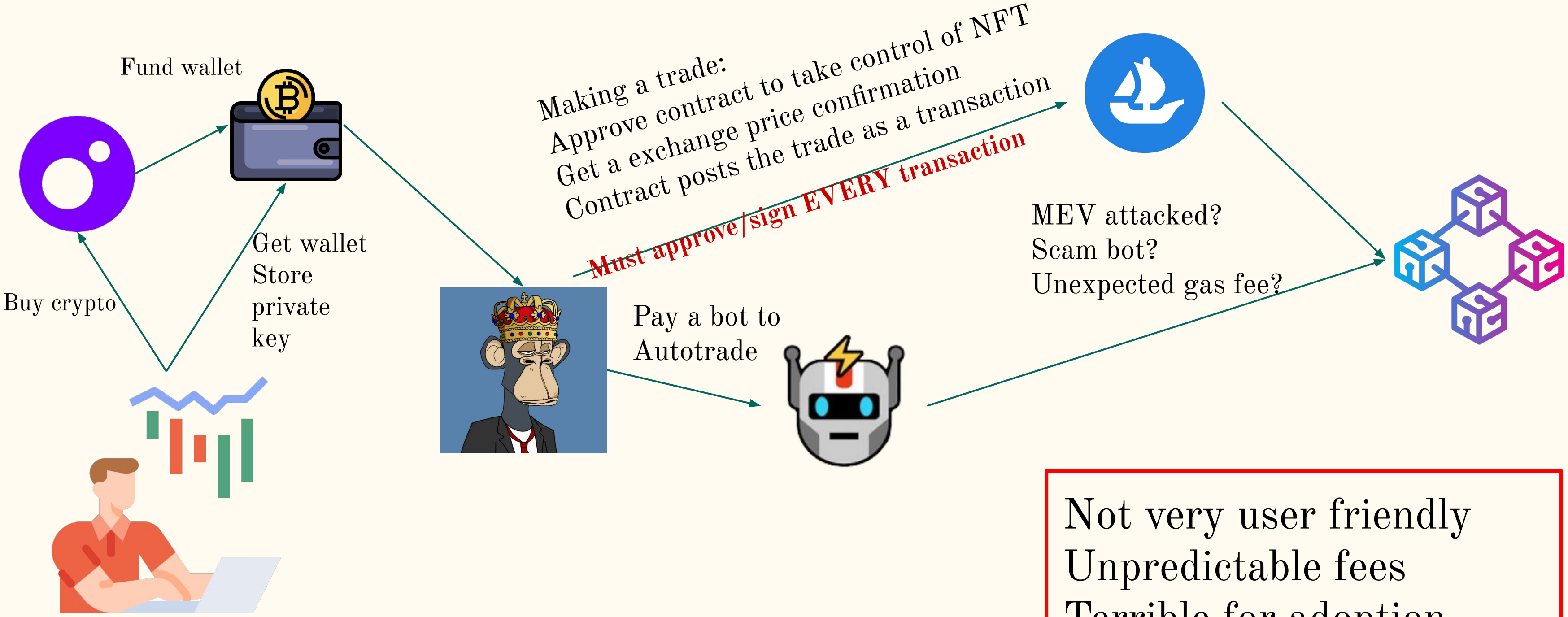
Interactive - User performs an action, NFT upgrades

Onwards to Eth Hackathons!

Towards User Onboarding

Account Abstraction & Gasless Transactions

From the eyes of a NFT trader



Fund wallet

Buy crypto

Get wallet
Store
private
key

Making a trade:
Approve contract to take control of NFT
Get a exchange price confirmation
Contract posts the trade as a transaction

Must approve/sign EVERY transaction

Pay a bot to
Autotrade

MEV attacked?
Scam bot?
Unexpected gas fee?

Not very user friendly
Unpredictable fees
Terrible for adoption

Driving towards mass adoption - EIP 4337

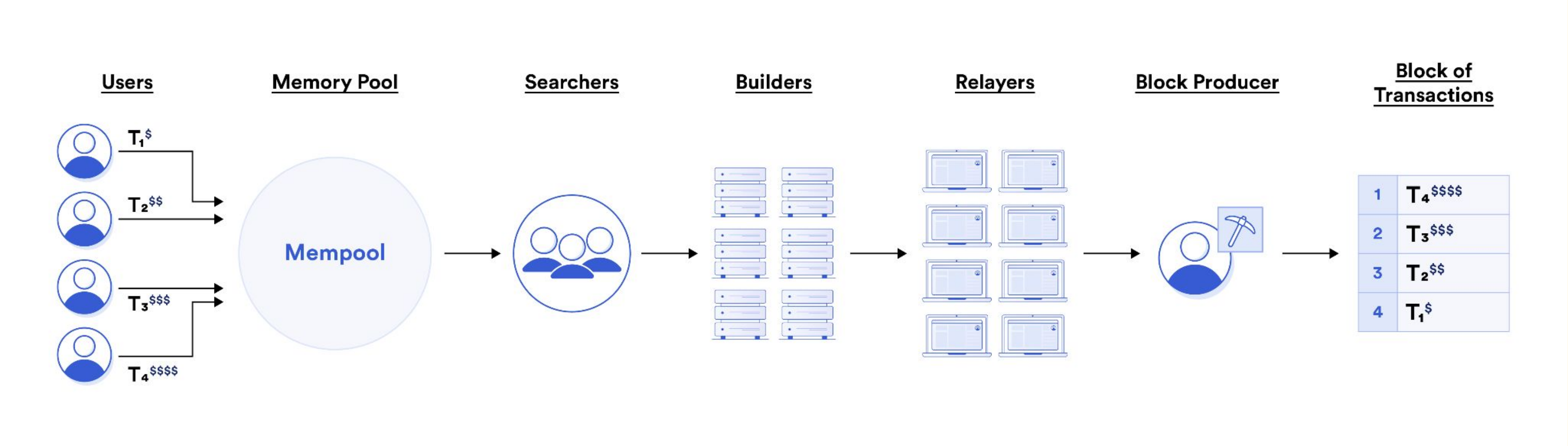
1. Massively improve ease of use for users
   1. Abstract away the need for wallets
   2. Abstract away the need to store private keys securely
   3. Allow more traditional financial user operations - continuous payments (prelevement à la source)
2. Easier user onboarding experience for crypto projects - Web2.5
   1. Sponsored gas payments by projects instead of user payments - paymaster
   2. Wallet projects offers familiar Auth mechanisms - MFA, session cookies, password, social media logins...
   3. Wallet recovery!!!
3. Avoid a hard fork!!!
   1. Hard forks are extremely intensive - remember the adoption of the Euro?
   2. General interface for contracts to make payments - no more difference between wallet and contract
   3. Introduce new players into the transaction value chain

Recap - EOAs vs Smart Contract

- Has private and public key
- Randomly generate seed phrase → generate private key → Public key on elliptic curve → Address
- Able to sign transactions with private key
- No code execution

- Contains executable code
- No keys - cannot initiate its own transactions!
- Execution passed around as the bytes in the data field of a transaction

Recap - Transaction lifecycle

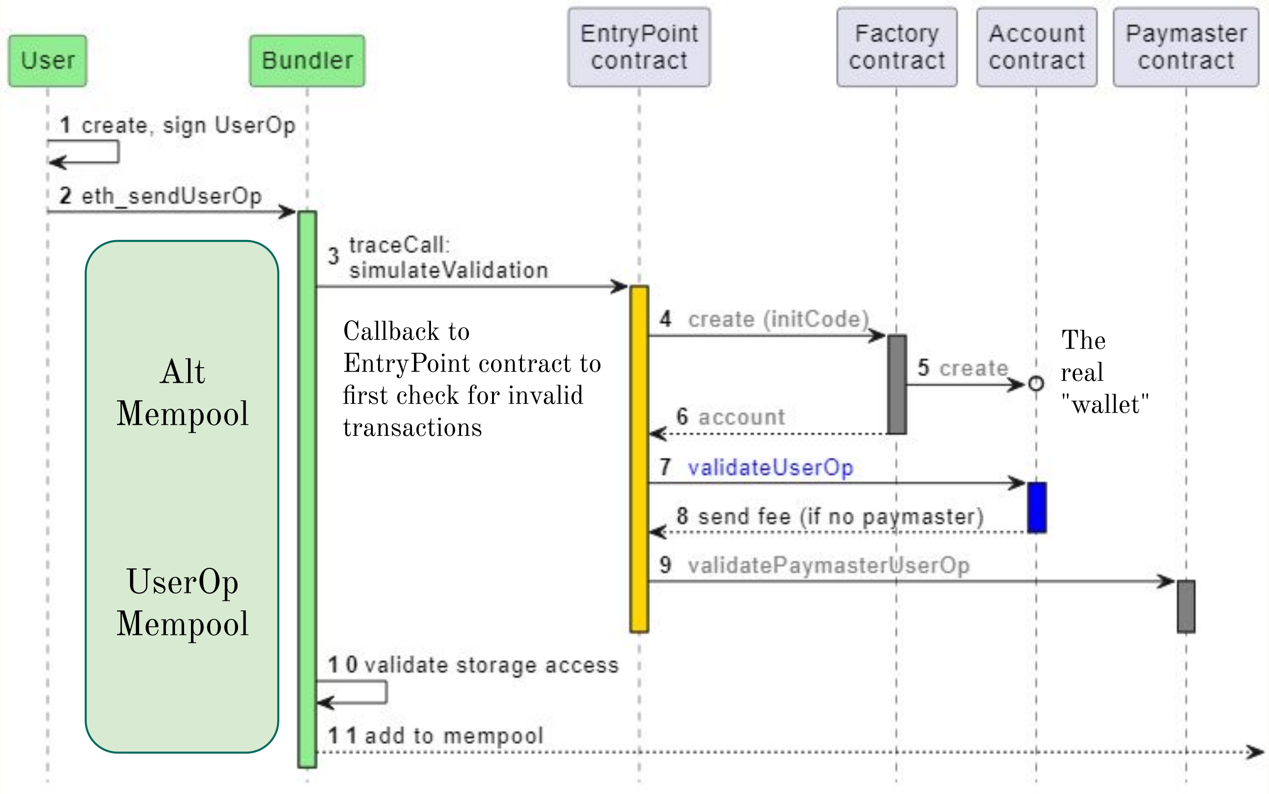


1	$T_4^{\$\$\$\$}$
2	$T_3^{\$\$\$}$
3	$T_2^{\$\$}$
4	$T_1^{\$}$

EIP 4337 - Architecture (Approximate)

A proxy collecting
many transactions.
Batching!

RPC call to bundler



Alt
Mempool

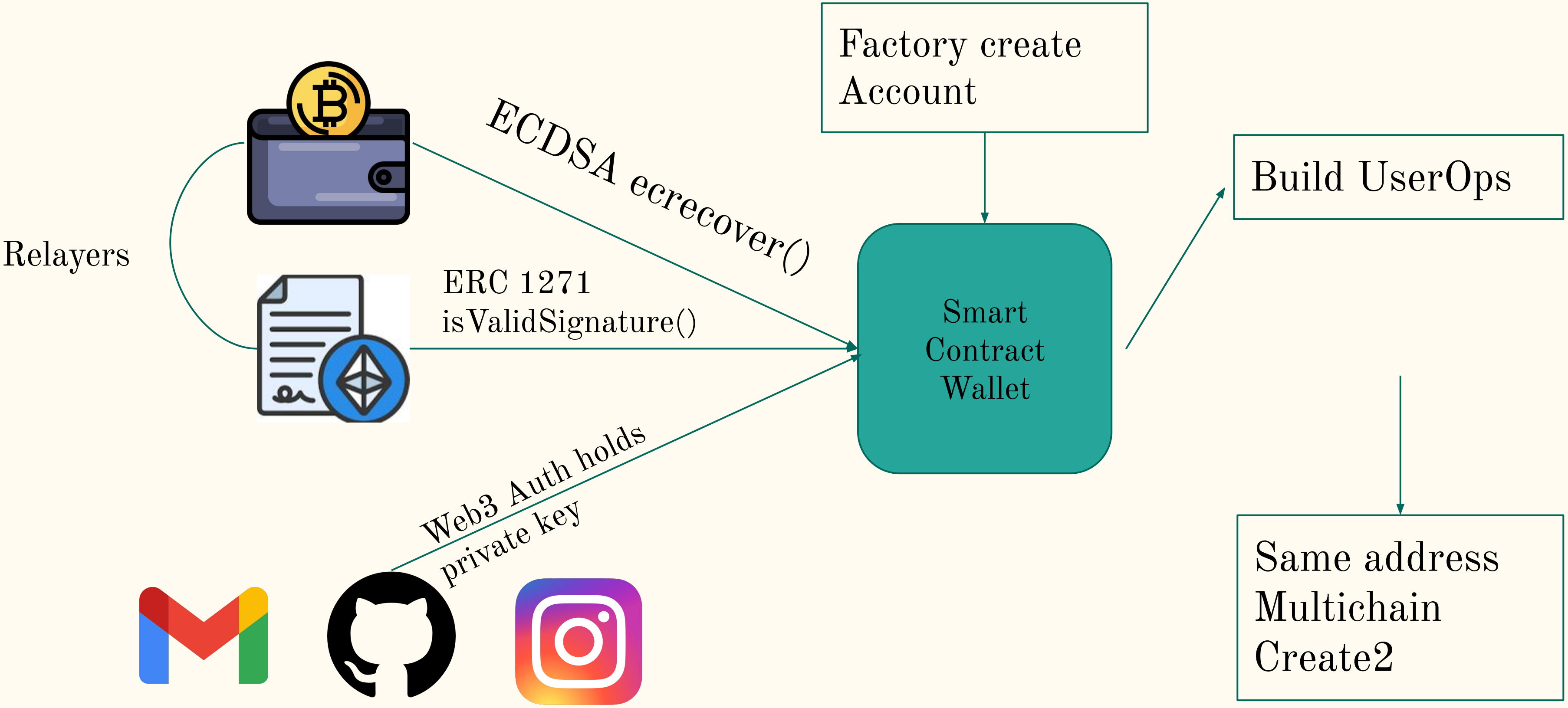
UserOp
Mempool

Callback to
EntryPoint contract to
first check for invalid
transactions

The
real
"wallet"

Back to normal
transaction flow.
No protocol
changes!

User - Bringing Abstraction



Relayers

ECDSA ecrecover()

ERC 1271
isValidSignature()

Web3 Auth holds
private key

Factory create
Account

Smart
Contract
Wallet

Build UserOps

Same address
Multichain
Create2

EIP 4337 - UserOperation / Pseudo-transaction

Signed Transaction

```
{
  nonce: "0x0",
  maxFeePerGas: "0x1234",
  maxPriorityFeePerGas: "0x1234",
  gas: "0x55555",
  to:"0x07a565b7ed7d7a695fe0",
  value: "0x1234",
  data: "0xabcd",
   v: "0x26",
   r: "0x223a7c9bcf20e",
   s: "0x28cc7704971491663",
   hash: "0xeb0d46df3870f8a30e"
}
```

sender	The address of the smart contract account
nonce	Anti-replay protection; also used as the salt for first-time account creation
initCode	Code used to deploy the account if not yet on-chain
callData	Data that's passed to the sender for execution
callGasLimit	Gas limit for execution phase
verificationGasLimit	Gas limit for verification phase
preVerificationGas	Gas to compensate the bundler
maxFeePerGas	Maximum fee per gas
maxPriorityFeePerGas	Maximum priority fee per gas
paymasterAndData	Paymaster Contract address and any extra data required for verification and execution (empty if self-sponsored)
signature	Used to validate a UserOperation along with the nonce during verification

EIP 4337 - Contract Interactions

Initiators call *target* contract to perform the execution and return the result

<u>EntryPoint.sol</u>

- Singleton contract!
- validateUserOp();
  - Bundler
- handleOps(){

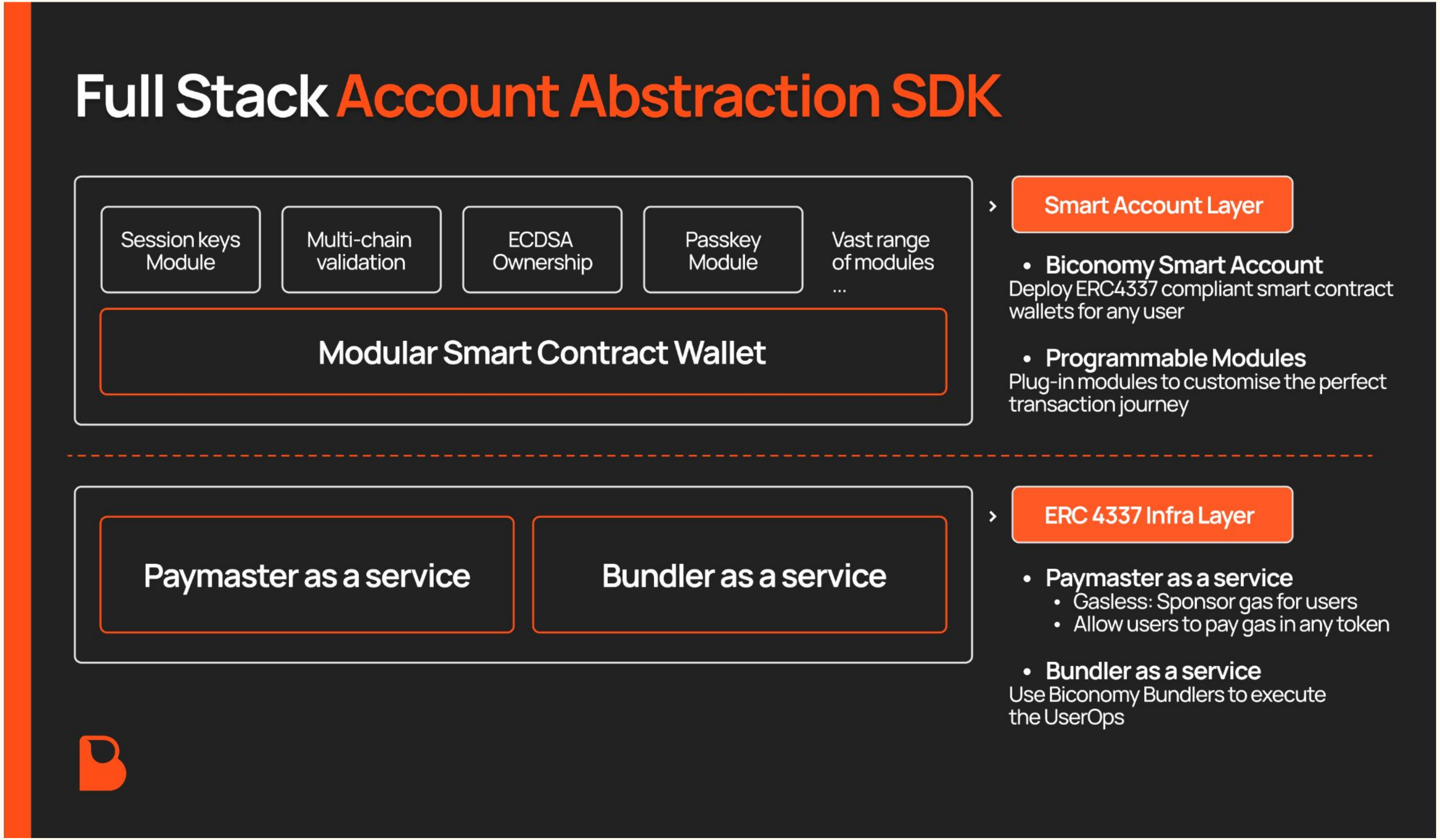
```
validateUserOp();
validatePaymasterOp();
}
```

- handleAggregatedOps();

<u>Paymaster.sol</u>

- validatePaymasterOp();
  - EntryPoint.sol
- deposit();
- withdrawTo();
- stake();
  - Proof of Stake
- postOp();
  - EntryPoint.sol

A new payment system



Full Stack Account Abstraction SDK

Session keys Module

Multi-chain validation

ECDSA Ownership

Passkey Module

Vast range of modules …

Modular Smart Contract Wallet

Smart Account Layer

- **Biconomy Smart Account**
  Deploy ERC4337 compliant smart contract wallets for any user
- **Programmable Modules**
  Plug-in modules to customise the perfect transaction journey

Paymaster as a service

Bundler as a service

ERC 4337 Infra Layer

- **Paymaster as a service**
  - Gasless: Sponsor gas for users
  - Allow users to pay gas in any token
- **Bundler as a service**
  Use Biconomy Bundlers to execute the UserOps