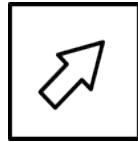


substrate_

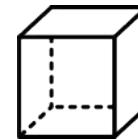


Parity Technologies



Track record

Co-founded by Dr. Gavin Wood (former CTO of Ethereum) in 2015.



Core blockchain infrastructure company

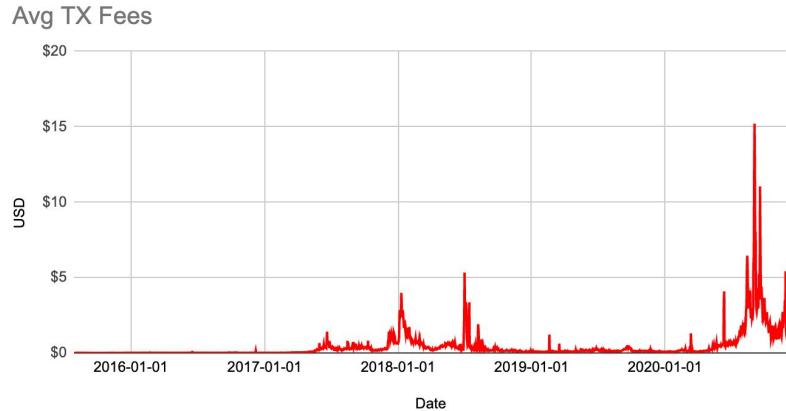
Pioneers of building Ethereum, Bitcoin and Zcash clients in Rust.



Open source

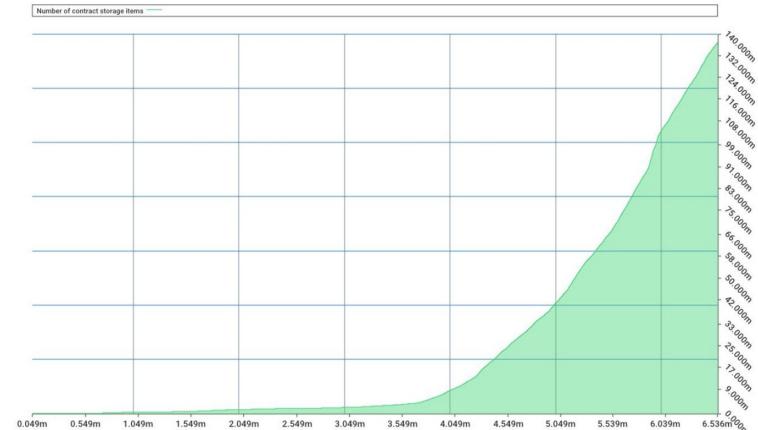
2.1k forks on Github for Substrate, 1.2k forks for Polkadot (Apache 2.0 license).

**Low throughput =
congestion and high
fees**



Average Ethereum transaction fees

**One-time only storage fee =
state bloat**

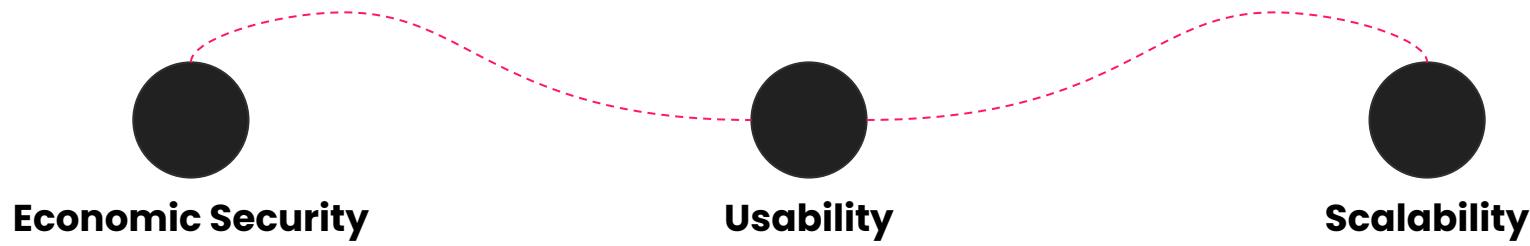


Ethereum State Growth

EVM != complex innovations

Operation	Gas	Description
ADD/SUB	3	Arithmetic operation
MUL/DIV	5	
ADDMOD/MULMOD	8	
AND/OR/XOR	3	Bitwise logic operation
LT/GT/SLT/SGT/EQ	3	Comparison operation
POP	2	Stack operation
PUSH/DUP/SWAP	3	
MLOAD/MSTORE	3	Memory operation
JUMP	8	Unconditional jump
JUMPI	10	Conditional jump
SLOAD	200	Storage operation
SSTORE	5,000/ 20,000	
BALANCE	400	Get balance of an account
CREATE	32,000	Create a new account using CREATE
CALL	25,000	Create a new account using CALL

Limited Ethereum OpCodes



The case for application specific blockchains



Performance

Single app optimised state machine in contrast to bottleneck virtual machine.



Security

Attack-surface of virtual machine is large (mostly due to complexity).



Sovereignty

Not dependent on platform governance.

The challenges of custom blockchains



Overhead

It takes years and \$\$\$ to build a blockchain from scratch.



Isolation

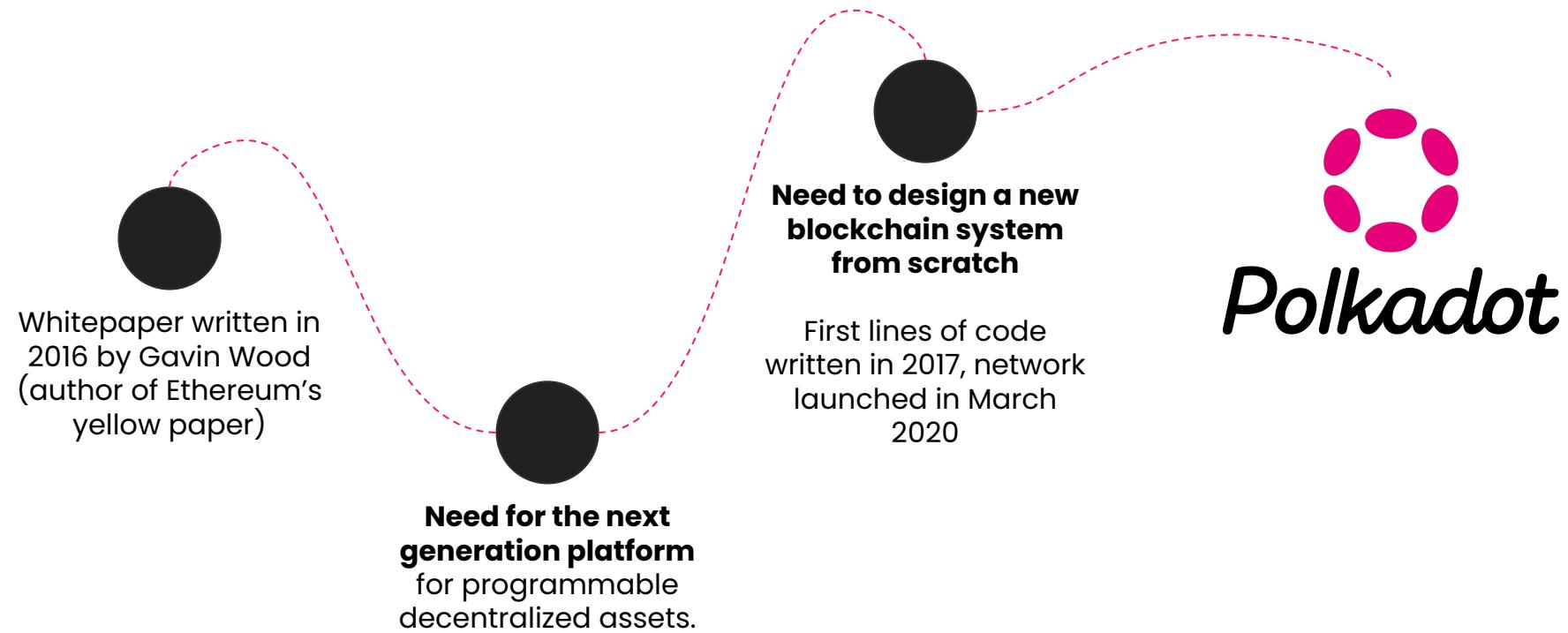
Blockchains couldn't talk to each other without going through centralised services.



Security

Blockchains would naturally compete with each other over security resources; leaving each one of them less secure.

The path to Polkadot



Enter Polkadot: a protocol to pave the way towards Web3 by supercharging blockchain innovation

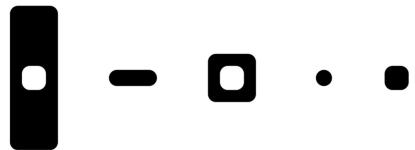
	What was needed	What we built
Benefits of custom blockchains	 Performance High throughput	→ Heterogeneous sharding
Catering to challenges of custom blockchains	 Sovereignty Network with true agency	→ Explicit on-chain governance
	 Security Secure & efficient consensus	→ Nominated Proof-of-Stake
	 Overhead Low cost of building	→ Substrate - modular blockchain SDK
	 Isolation Highly composable	→ Cross-chain messaging
	 Customizable Customizable	→ Runtime development
	 Upgrades Guaranteed network evolution	→ On-chain treasury ensures funding

“Polkadot is **built to connect** private and consortium chains, public and permissionless networks, oracles, and future technologies that are yet to be created.”

Polkadot was launched in May 2020 and had its first parachain connected to it in October 2021.

Kusama (canary network) was launched in August 2019 and its first parachain connected in May 2020.

Polkadot Architecture



Relay
Chain

Validator

Parachain

Collator

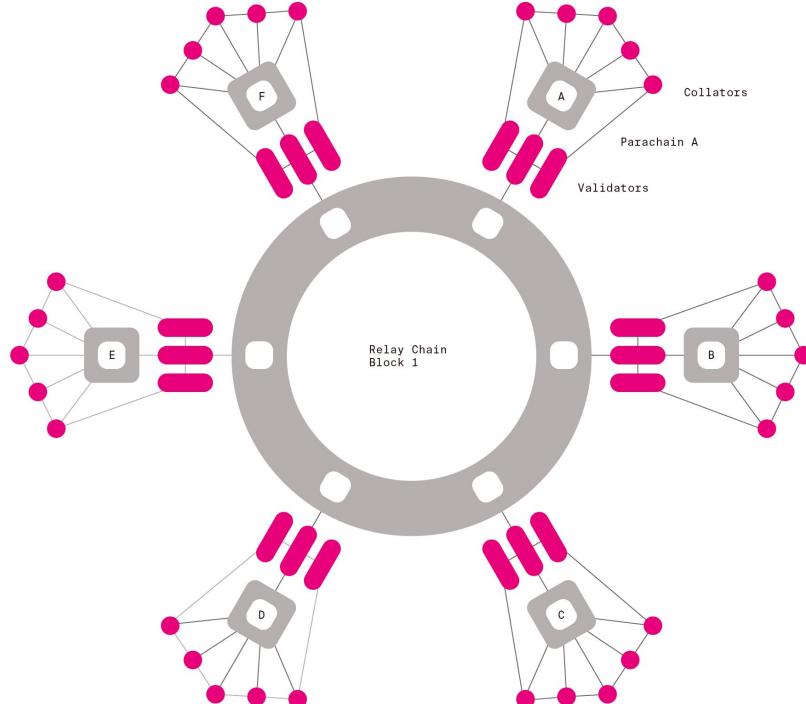
Block

RELAY CHAIN

- The connector chain of Polkadot that provides strong economic security and an interoperability protocol.

PARACHAINS

- App-specific chains that connect to Polkadot for interoperability, scalability, and pooled security.



Interoperability features

Assets teleportation
and remote requests



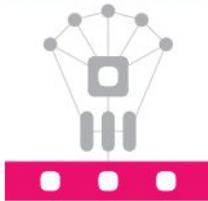
- X-consensus message format for parachains (XCM)
- Extensible
- Future-proof and forwards-compatible

Bridging some chain
with another



- Rarer interactions
- Unknown consensus mechanism
- Expensive to send messages securely

Polkadot network elements



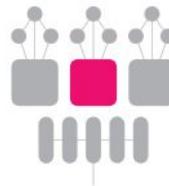
Relay Chain

The heart of Polkadot, responsible for the network's shared security, consensus and cross-chain interoperability.



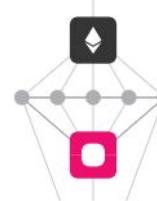
Parachains

Sovereign blockchains that can have their own tokens and optimize their functionality for specific use cases.



Parathreads

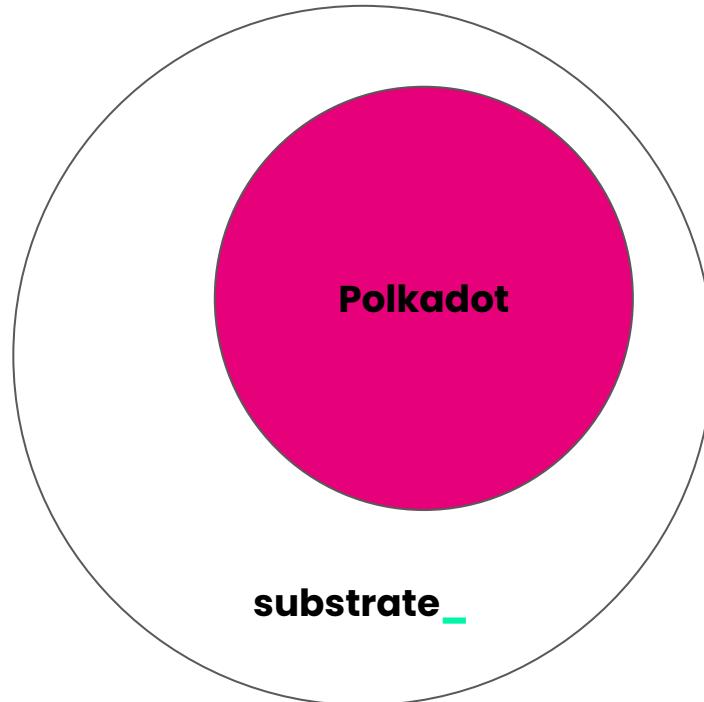
Similar to parachains but with a pay-as-you-go model. More economical for blockchains that don't need continuous connectivity to the network.



Bridges

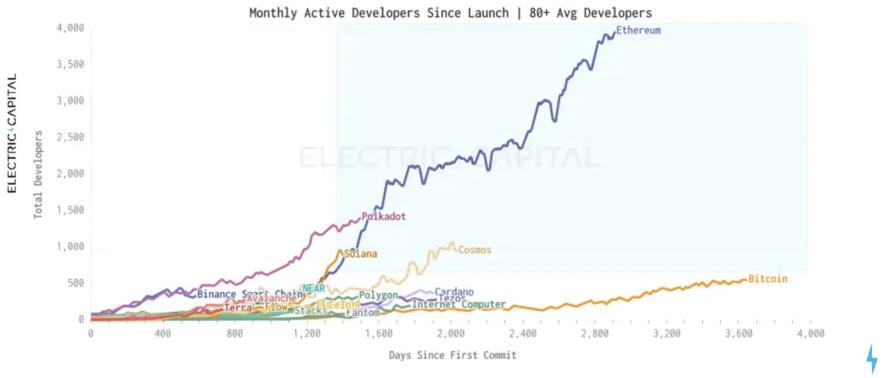
Allow parachains and parathreads to connect and communicate with external networks like Ethereum and Bitcoin.

Polkadot and Substrate

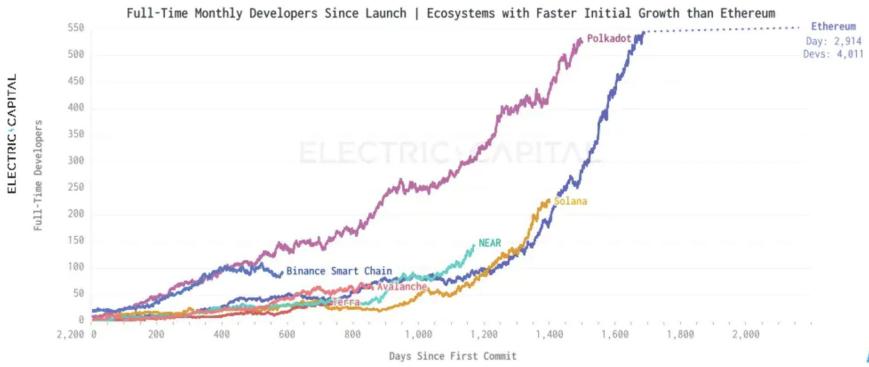


Core developer traction

ETHEREUM, POLKADOT, COSMOS, SOLANA, AND BITCOIN HAVE THE LARGEST ECOSYSTEMS - BUT THEY HAVE TAKEN VERY DIFFERENT PATHS TO GET HERE



FULL-TIME DEVELOPERS: POLKADOT, NEAR, SOLANA, AND BSC HAVE MORE FULL-TIME DEVS THAN ETHEREUM AT THE SAME POINT AFTER FIRST CODE COMMIT



substrate_

The Polkadot Stack for a Multichain Future

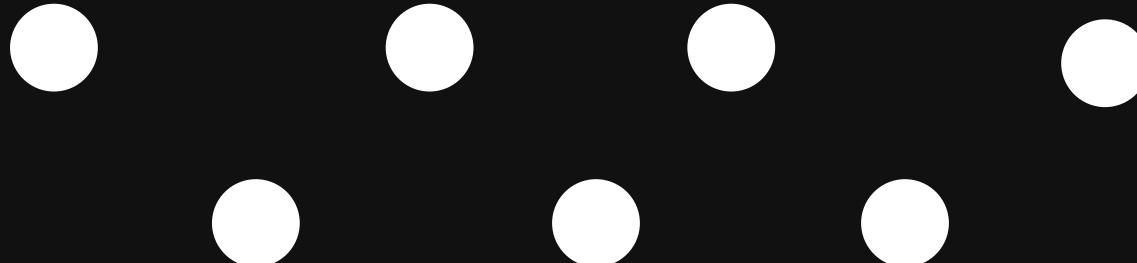
What is Substrate?

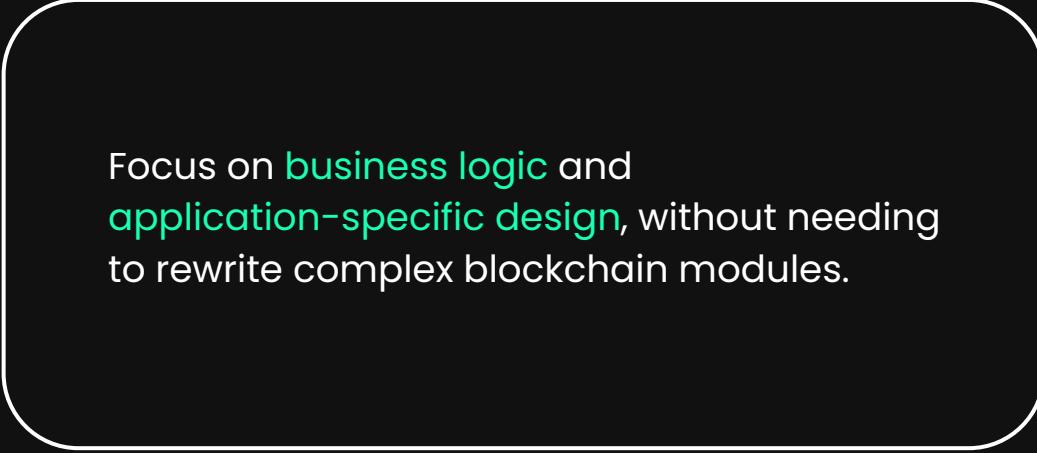
Substrate is a framework **for building blockchains**, initially created for building Polkadot.

Modular

Dynamic

Scalable





Focus on **business logic** and
application-specific design, without needing
to rewrite complex blockchain modules.

Substrate is modular

GOVERNANCE SYSTEMS

Substrate allows you specify rules for on-chain governance, extending existing modules or building new ones.

ASSETS

Substrate unlocks the next generation of programmable assets and payment systems.

CONSENSUS

Consensus mechanisms can be swapped out for different ones or adapted for specific use-cases.

INFRASTRUCTURE

Architectural components can be changed for new ones if needed, from database and networking layers to transaction pool logic.

Substrate is dynamic and adaptable

FORKLESS UPGRADES

Substrate is designed to build blockchains that evolve. If bugs are found in production, Substrate allows enables forkless upgrades.



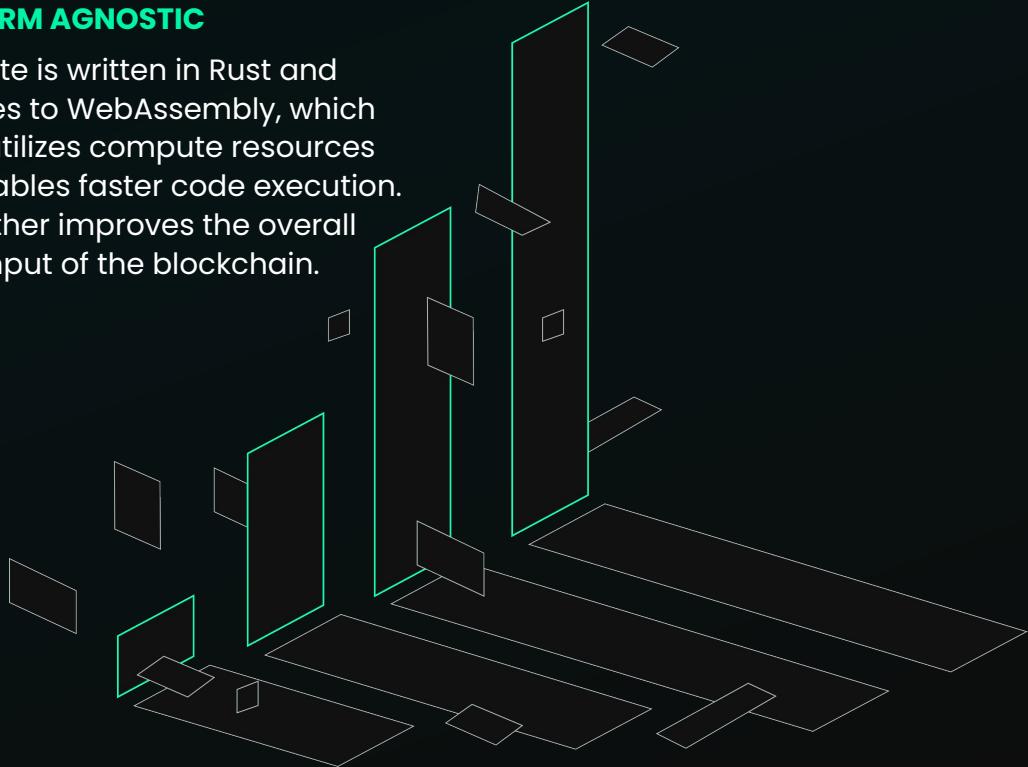
Substrate is built to scale

~1500 TPS

This load test was done on a multi-node, geographically distributed network, with transactions that are non-correlated and independently signed-and-executed.

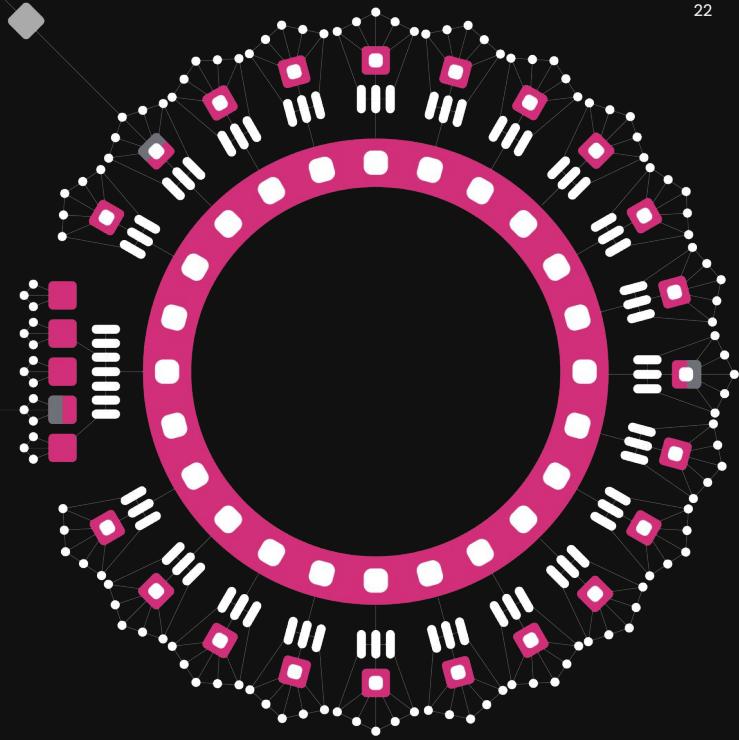
PLATFORM AGNOSTIC

Substrate is written in Rust and compiles to WebAssembly, which better utilizes compute resources and enables faster code execution. This further improves the overall throughput of the blockchain.



What does Substrate provide?

- Database Layer
- Networking Layer
- Consensus Engine
- Transaction Queue
- Library of Runtime Modules



... which are all the core components of a Blockchain.

How it works

Runtime and Storage

State Transition Function

The runtime defines possible state transitions and APIs around them.

Persistence

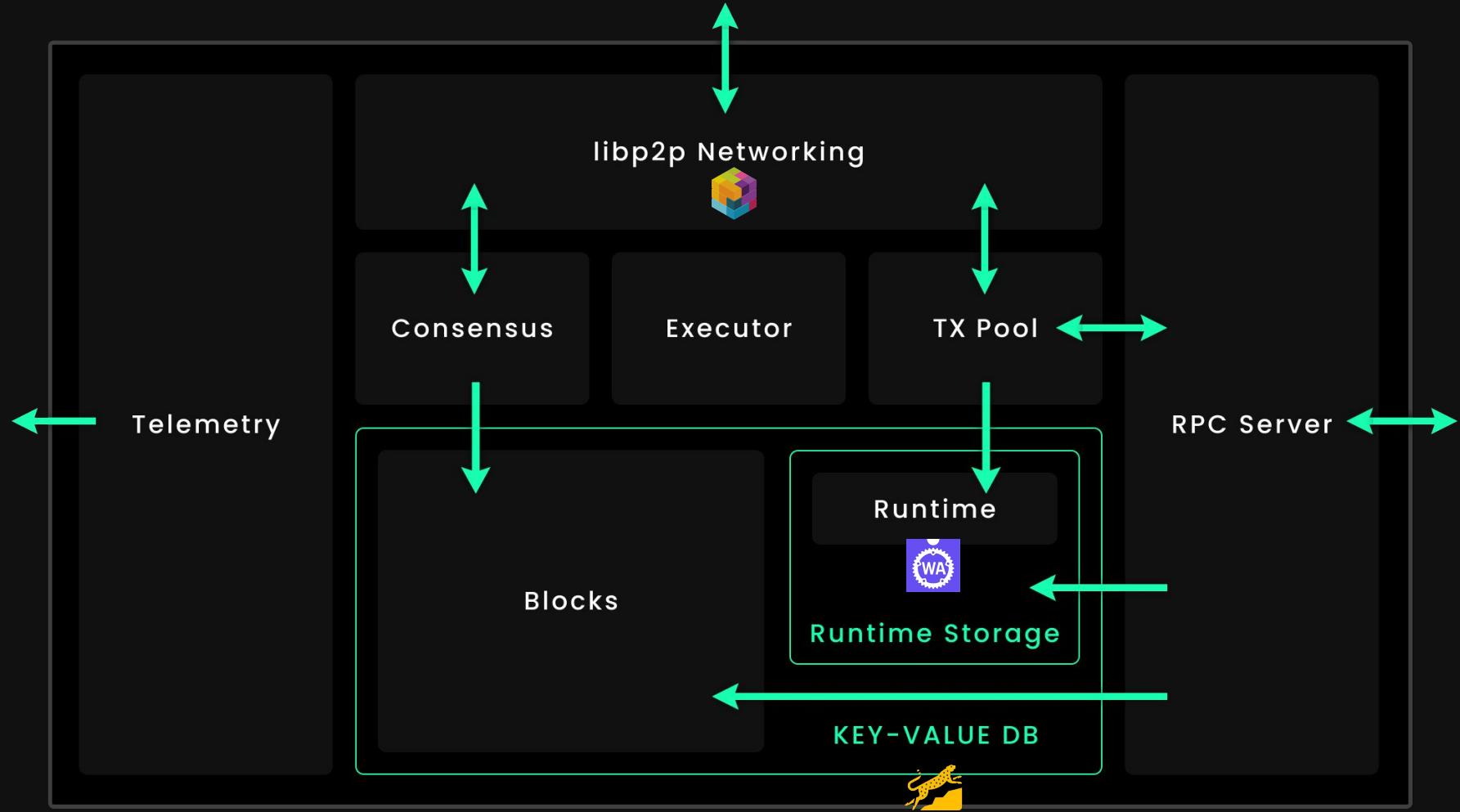
State transitions persist to storage, using Rocks DB's key-value storage layer.

Upgradability

The runtime compiles to Wasm, resulting in a byte-code blob containing all storage and STF logic.

The definition of the runtime is an element in storage.





FRAME Pallets

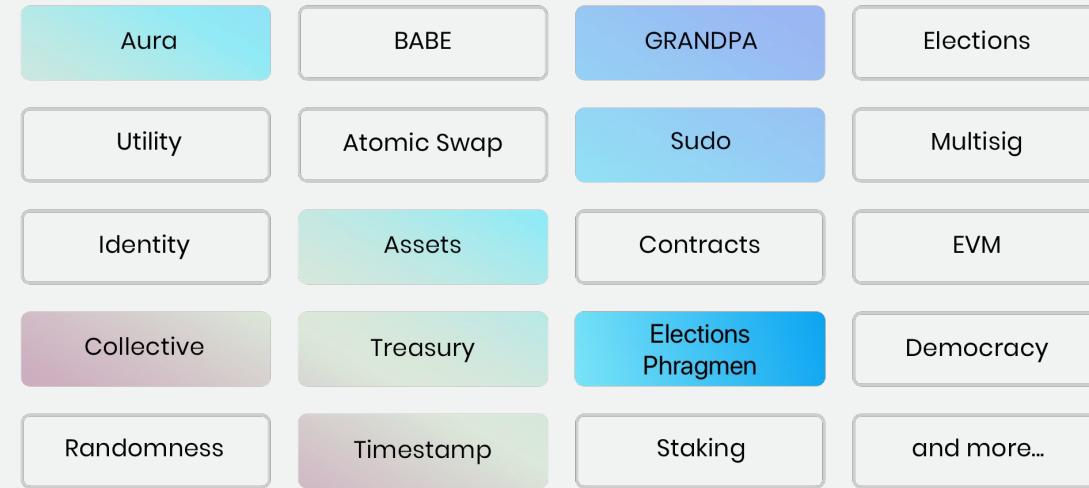
Framework for Runtime Aggregation of Modularized Entities

Substrate runtimes are made of a collection of Rust modules we call **"pallets"**.

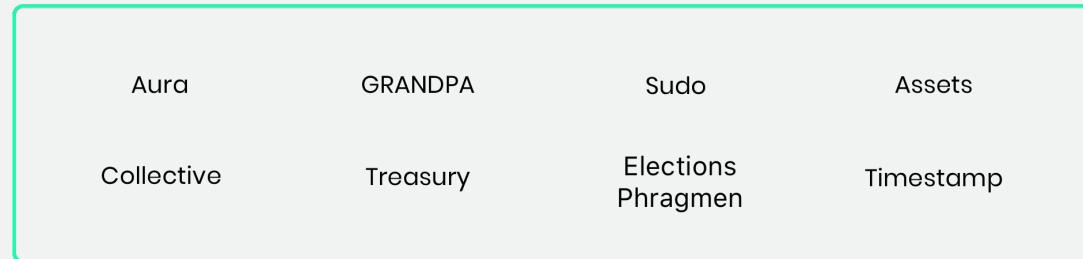
- > Simplifies runtime development.
- > Enables composability.



SUBSTRATE FRAME PALLETS



RUNTIME



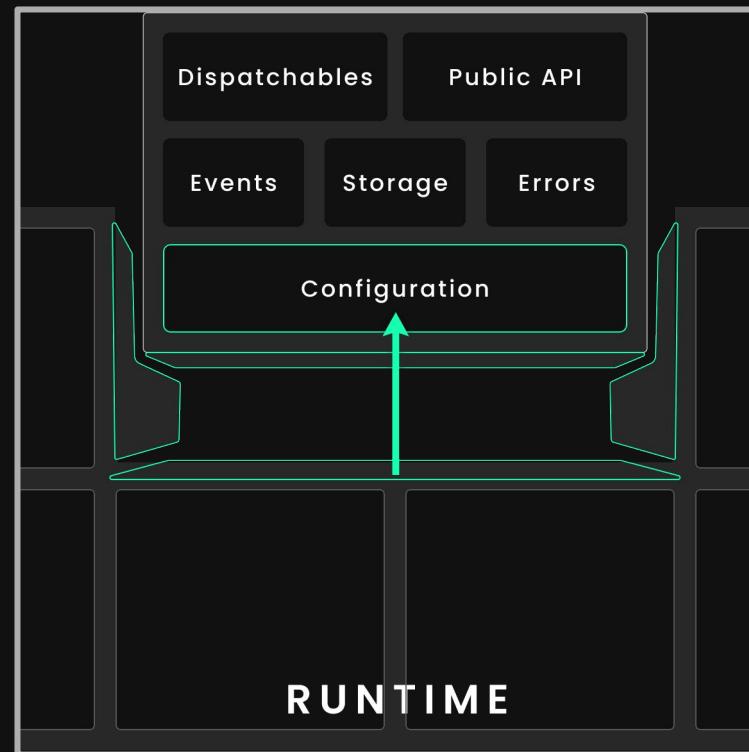
How FRAME works

0—Config

The Configuration trait for all FRAME pallets. Allows developers to configure their pallet by specifying the parameters and types on which it depends.

1—Storage, Events & Errors

Provides an interface to runtime storage, as well as emitting runtime events and errors.



2—Dispatchables

Allows users to make calls into the runtime, such as transactions and [other types of extrinsics](#).

Substrate features a rich set of tools for all stages of your blockchain journey.

With the tools available in the Substrate ecosystem, you can build, deploy and monitor blockchains, smart contracts, and even user interfaces! Here is a quick overview of some of these tools:



BUILD BLOCKCHAINS

- Substrate Framework
- Substrate Node Template
- Benchmarking Framework
- Ink! Smart Contracts
- Frame Pallets
- Zombie Net



BUILD CLIENT APPS

- Polkadot JS API & Apps
- Parity Scale Codec
- Substrate Connect
- Substrate Archive
- Browser Extension
- Signer App
- RPC Libraries



DEPLOY

- End-to-End Deployment Solution
- Docker and Kubernetes Support
- Polkadot Secure Validator (Infra as a code)



MONITOR

- Network Telemetry
- Network Analytics Aggregator
- Built-in Instrumentation
- Prometheus Metrics Support
- Grafana dashboards and visualisation
- Validator Alert System

Developer community



Builder community
& Rust developer
talent pool



Cross-chain
collaboration on
tackling market
verticals

Language Choice

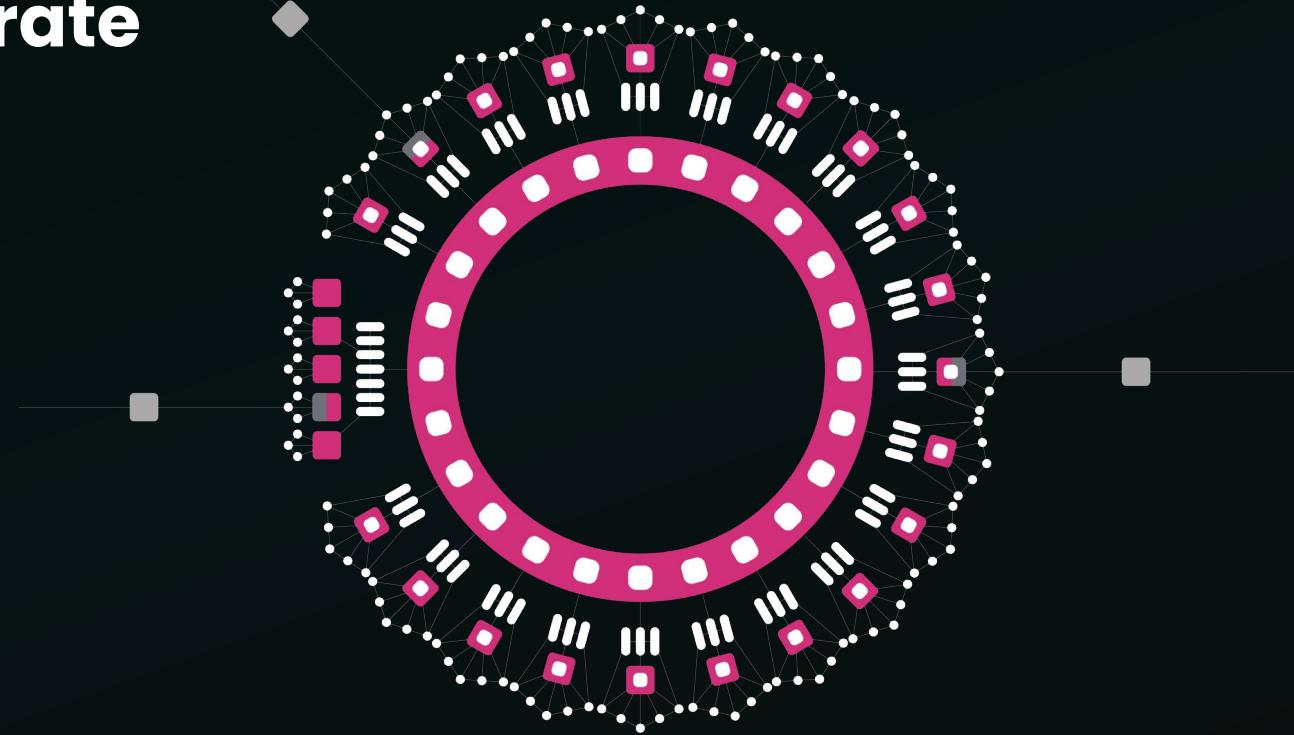
Substrate is built in Rust, but the runtime is in Wasm

Theoretically, any language that compiles to Wasm can be used

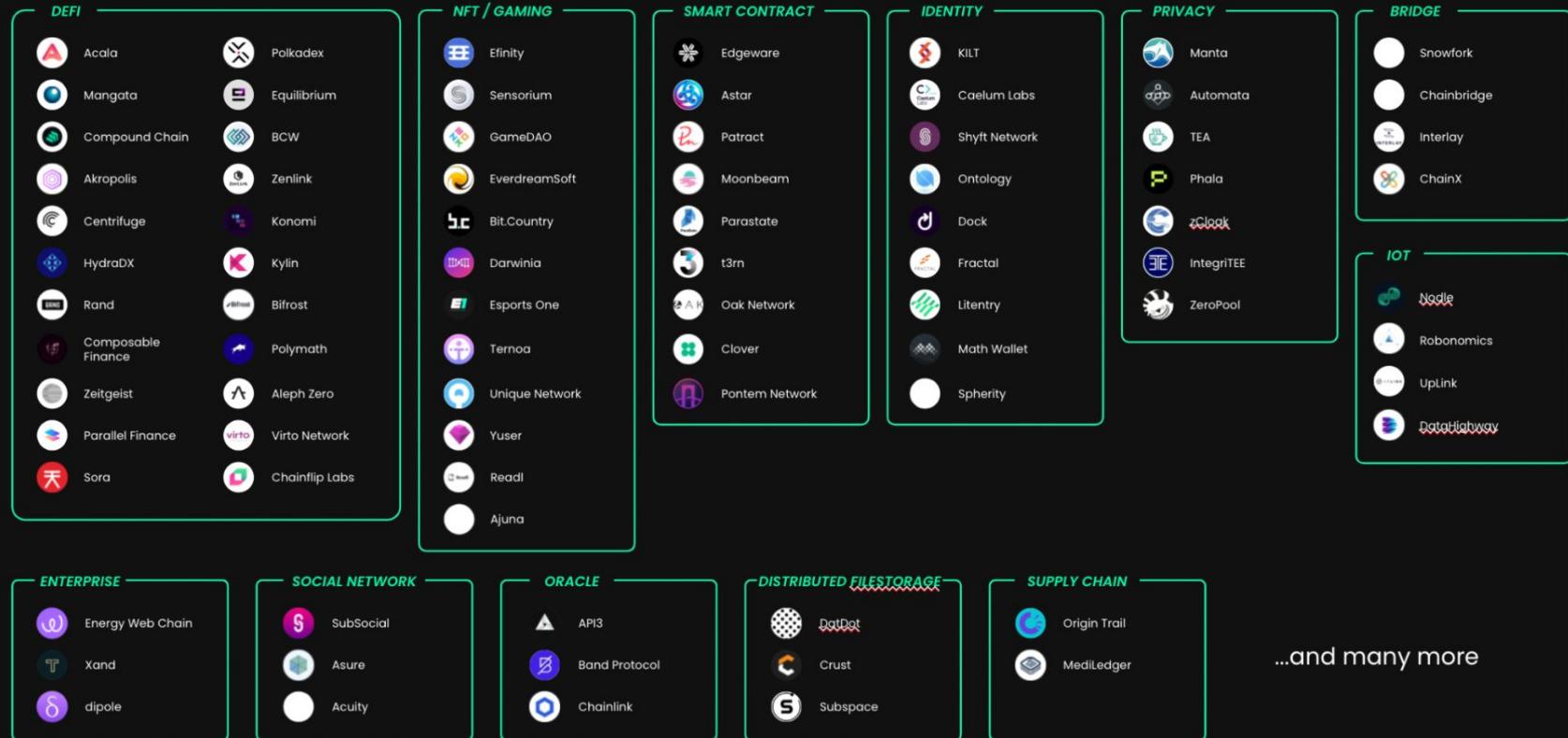
Currently there is a project working to provide support for runtime development in Go

-> Potentially others in the future (AssemblyScript, C++)

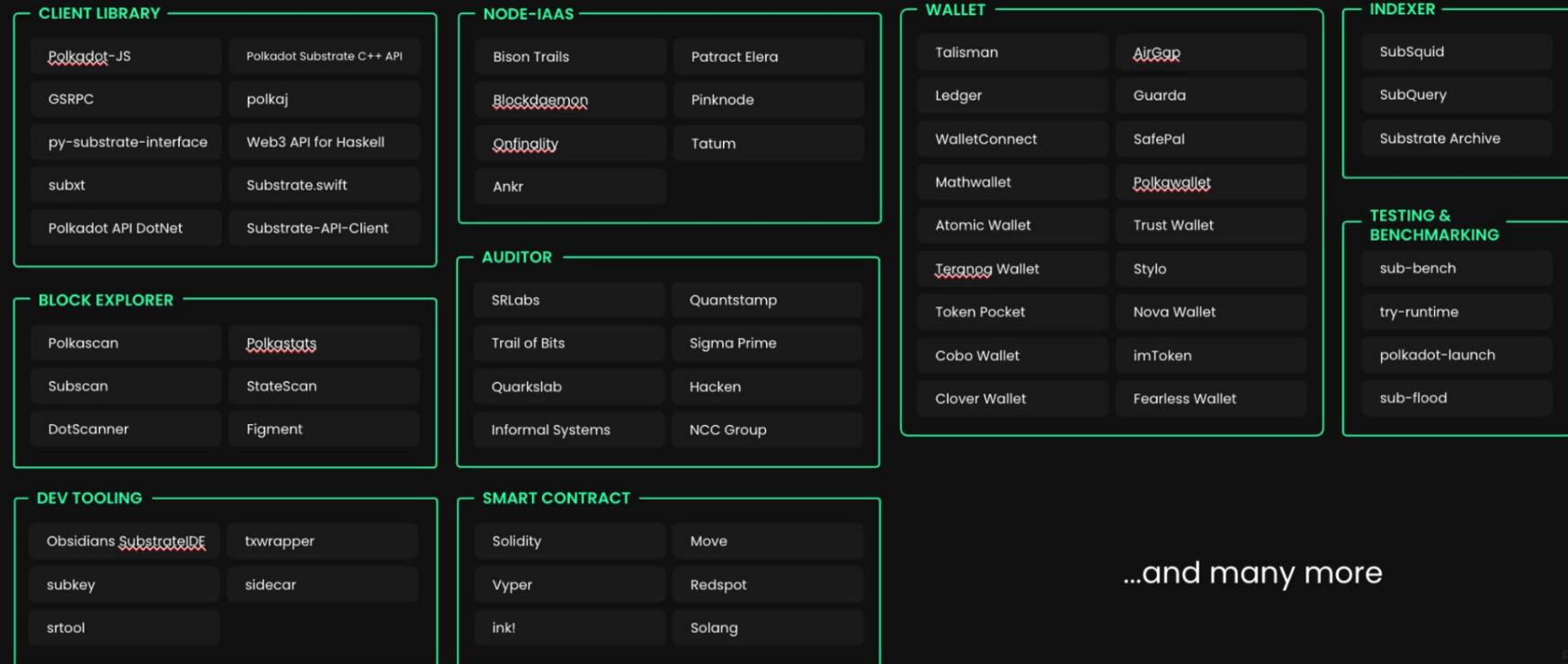
Projects building on Polkadot using Substrate



Substrate-based Chains



APIs, Tools, & Services



Comparative overview



c·rda

cōsmos SDK

 polygon

Comparison

	Ethereum	Hyperledger Besu	Hyperledger Fabric	Corda R3	Cosmos SDK	Polygon	Substrate
Language	EVM bytecode (Solidity, Vyper)	EVM bytecode (compatible w/ Ethereum)	Go, Node (Javascript), Java	JVM (Java, Kotlin, etc.)	Go	EVM bytecode (Solidity, Vyper)	Rust (Go in development)
Granularity	Smart contract	Smart contract	Chain code (similar to smart contracts)	CorDapps (similar to smart contracts)	Runtime (not upgradeable w/o fork)	Smart contracts	Runtime (w/ forkless upgrades)
Sybil resistance / Block production	PoS	PoA	PoA	PoA	PoS	PoA on top of Ethereum PoS	Customizable (out-of-box support for PoA, PoS or PoW)
Permissions	Public	Public, permissioned, or private	Public, permissioned, or private	Private	Public, permissioned, or private	Public or permissioned	Public, permissioned, or private

Comparison

	Ethereum	Hyperledger Besu	Hyperledger Fabric	Corda R3	Cosmos SDK	Polygon	Substrate
Runtime Upgrades	Hard fork	Hard fork	Hard fork	Hard fork	Hard fork	Hard fork	Forkless (runtime code on-chain)
Governance	Off-chain coordination	Off-chain coordination	Off-chain coordination	Chain-customizable, but major changes require off-chain coordination	Chain-customizable, but major changes require off-chain coordination	Off-chain coordination	On-chain (can be configured)
Enactment	Opt-in	Opt-in	Opt-in	Opt-in	Opt-in	Opt-in	Automated
Interoperability	Smart contract-based bridges	Smart contract-based bridges	Smart contract-based	Under development	IBC	Via smart contracts asset movement and L1->L2 asset movement	XCM (arbitrary code execution, fungible and non-fungible assets)

Questions? 