# 基于Octopus的Substrate开发实践

Lester

# 目录

# 1. 智能合约 or 应用链

| | 智能合约 | 应用链 |
|---|---|---|
| **性能** | 受限于合约平台的性能 | 独享链上资源 |
| **成本** | 合约部署成本较低，有些合约平台用户交易成本偏高 | 启动部署多个节点成本较高，优化降低用户交易成本 |
| **可定制性** | 受限于合约执行环境 | 可定制底层协议，优化执行环境参数 |
| **可组合性** | 合约之间可灵活组合 | 需要跨链，不方便组合 |
| **安全性** | 合约平台提供较高的安全性 | 独立链需自建安全，接入多链网络可获得共享/租用安全 |

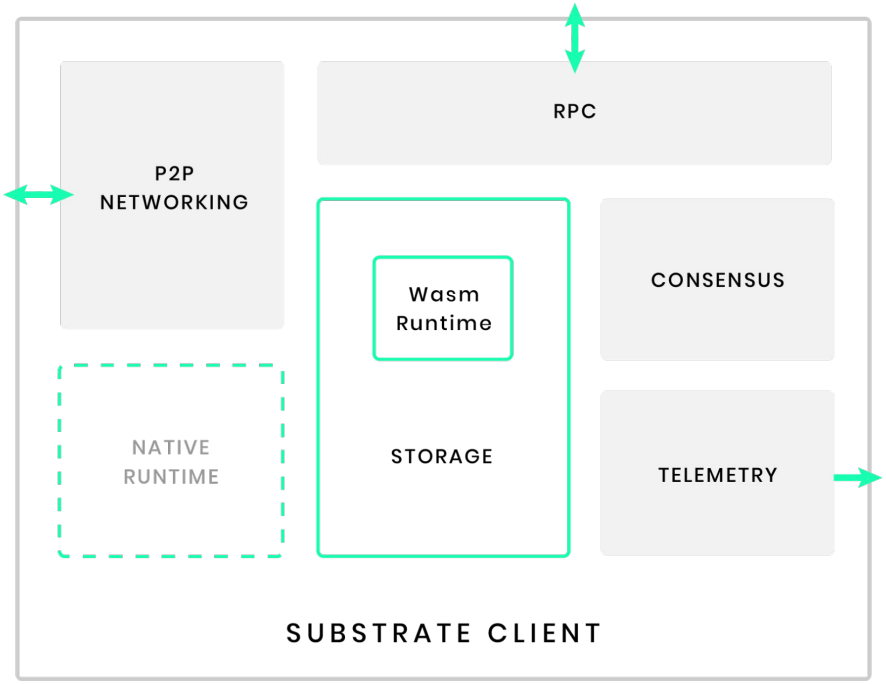
应用链案例：Axis, Compound, Ploymesh

# 2. Substrate 有哪些特性？

基于 Substrate 开发的应用链可获得高性能，低交易成本，无分叉升级等优势。主要得益于：

- **Substrate 架构**
- **可定制性**
- **可升级性**
- **Off-chain 功能**

# 2.1 Substrate 架构




图片来自 Substrate Developer Hub

**Runtime:** WASM, Native

**Core:**

**RPC:** HTTP, WebSocket

**Consensus:** NPoS
Aura,Babe,GRANDPA

**Storage:** RocksDB

**P2P:** libp2p

**Telemetry:** Prometheus

# 2.2 可定制性

- **Extrinsic**：inherents、签名交易和无签名交易  SignedExtension 签名扩展
- **账户**：Stash，Controller，Session Keys
- **交易池**：交易的有效性

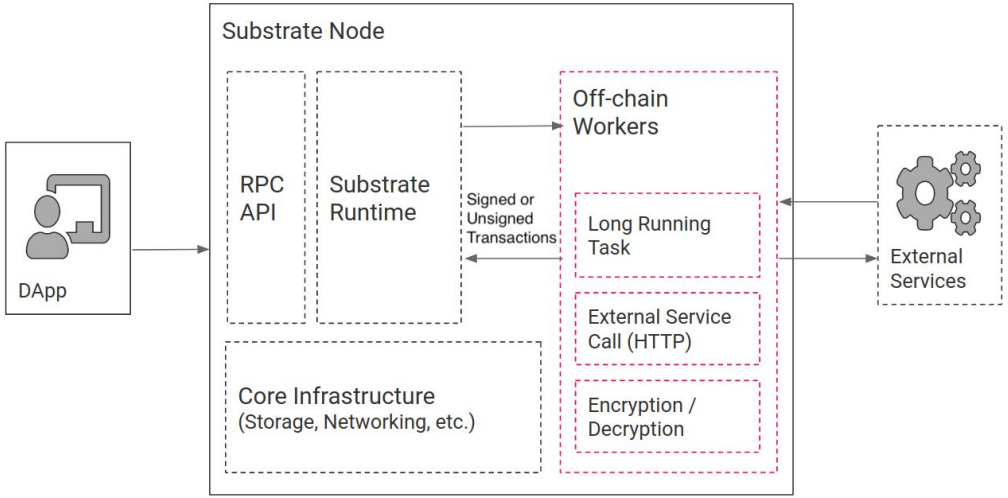
# 2.3 可升级性

- Runtime 可以被编译生成 WASM 和 Native 两种
- 节点执行策略：Native，Wasm，Both，NativeElseWasm
- 无分叉升级：WASM 存储在链上，即使有节点没有更新到最新版本，验证节点的执行策略为总是执行链上Wasm，节点在比较 Native 和 Wasm 版本后，会执行Wasm，这样保证了最终网络能够以最新的代码运行，不会因为代码的不同而分叉。

# 2.4 Off-chain 功能

**Off-chain** 提供安全高效的链下数据查询，处理功能：

- Off-Chain Worker (OCW)
- Off-Chain Storage
- Off-Chain Indexing




图片来自 Substrate Developer Hub

# 3. 如何基于 Substrate 开发应用？

基于 Substrate 开发应用，类似于基于智能合约开发DApps，基于各种框架开发Web应用或移动端Apps。

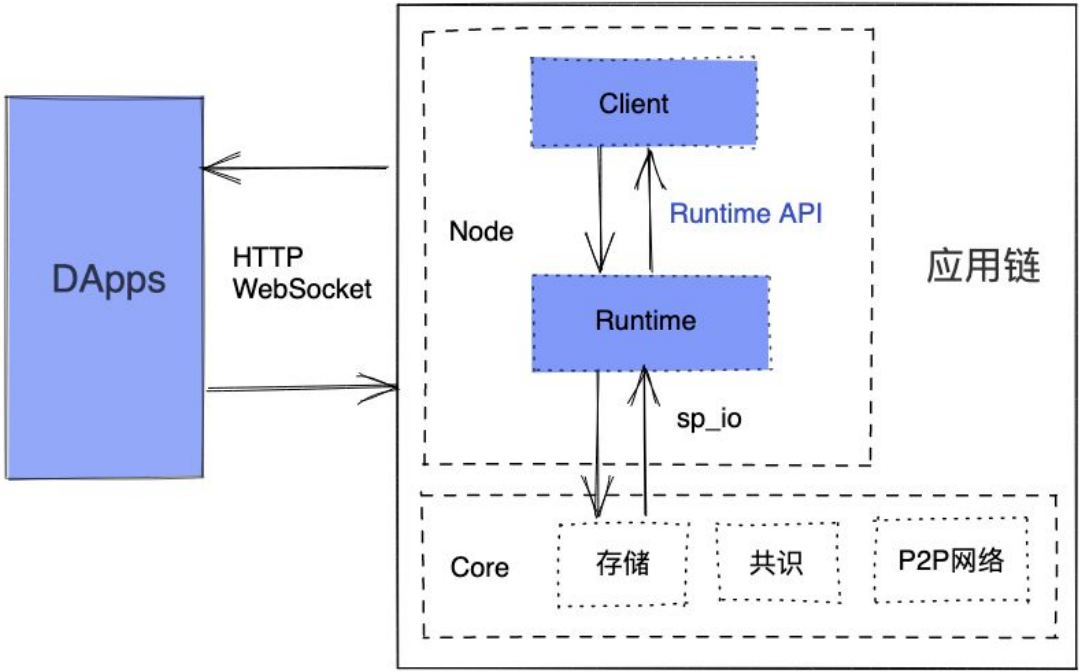
- **开发范式，FRAME的使用**
- **示例1：开发一个 Pallet**
- **示例2：实现简单的 Oracle**
- **示例3：Shine 应用链实践**

# 3.1 Substrate 开发范式

| 应用 | Web/移动端 | NEAR 合约 | 以太坊合约 | Substrate |
| --- | --- | --- | --- | --- |
| 前端 | React, Vue等 | | | |
| 交互 | RESTful, RPC框架等 | near-api-js, near-api-rs | web3.js等 | Polkadot-JS, Substrate RPC Client（Go等） |
| 后端 | Java Spring, Go Gin等 | near-sdk-as,near-sdk-rs | Solidity OpenZeppelin | Rust，FRAME |
| 存储 | MySQL，Redis，Mongo DB等 | K-V存储 | | |

**开发模版：**Front-end template，Node Template，Barnacle

# 3.1 Substrate Runtime 开发



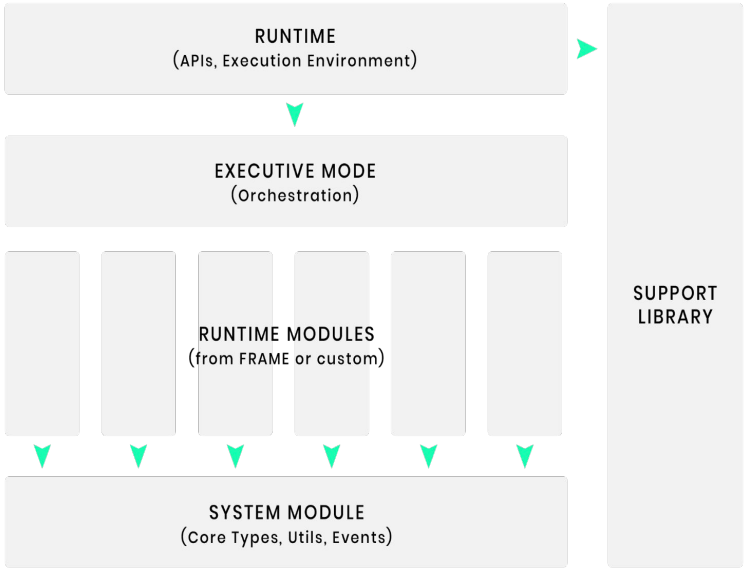

**Runtime:** 实现应用业务逻辑，类比智能合约，开发范式：

1. 开发业务功能的Pallets；
2. 开发和前端交互的Runtime API；
3. 其它，比如链外的分布式存储等

# 3.1 FRAME v2 常用宏

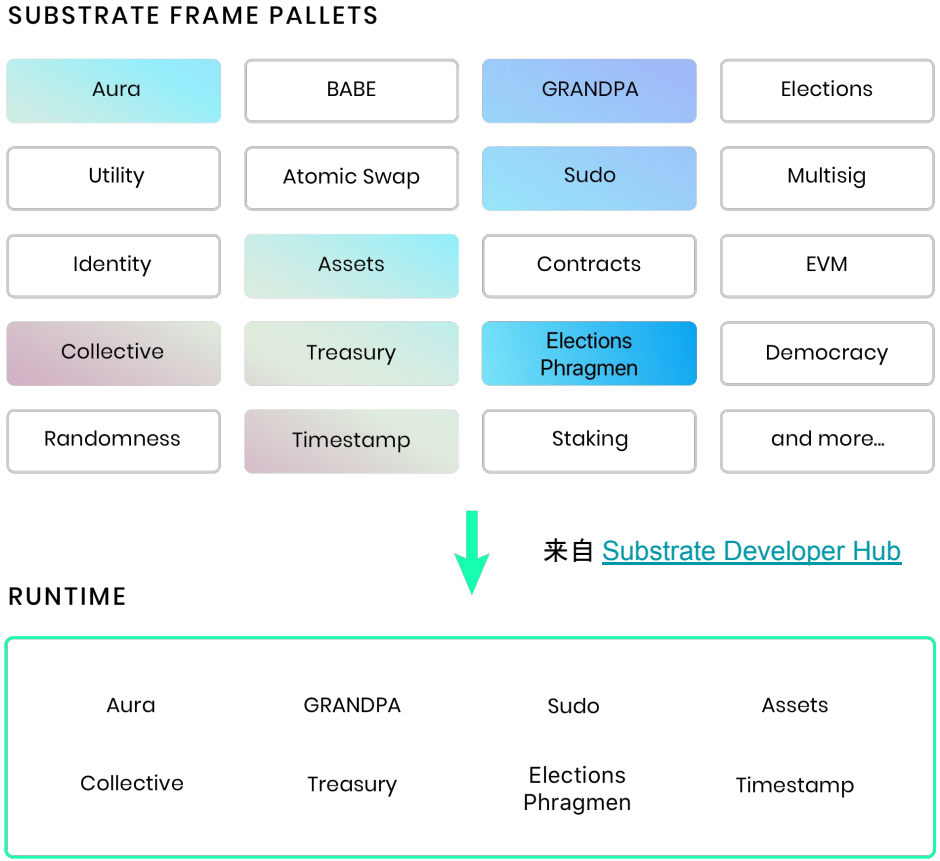
**FRAME:** 方便Runtime开发的模块和库

- frame_support::pallet
- pallet::pallet
- pallet::config
- pallet::storage
- pallet::event
- pallet::error
- pallet::call
- pallet::hooks
- construct_runtime




图片来自 Substrate Developer Hub

# 3.1 FRAME 已有的 Pallets



# 3.2 Pallet 开发示例

**Pallets:** 定义应用类型，链上存储，链外可调用函数等的Rust模块

Substrate中的 Pallet 模版

存储数据类型：

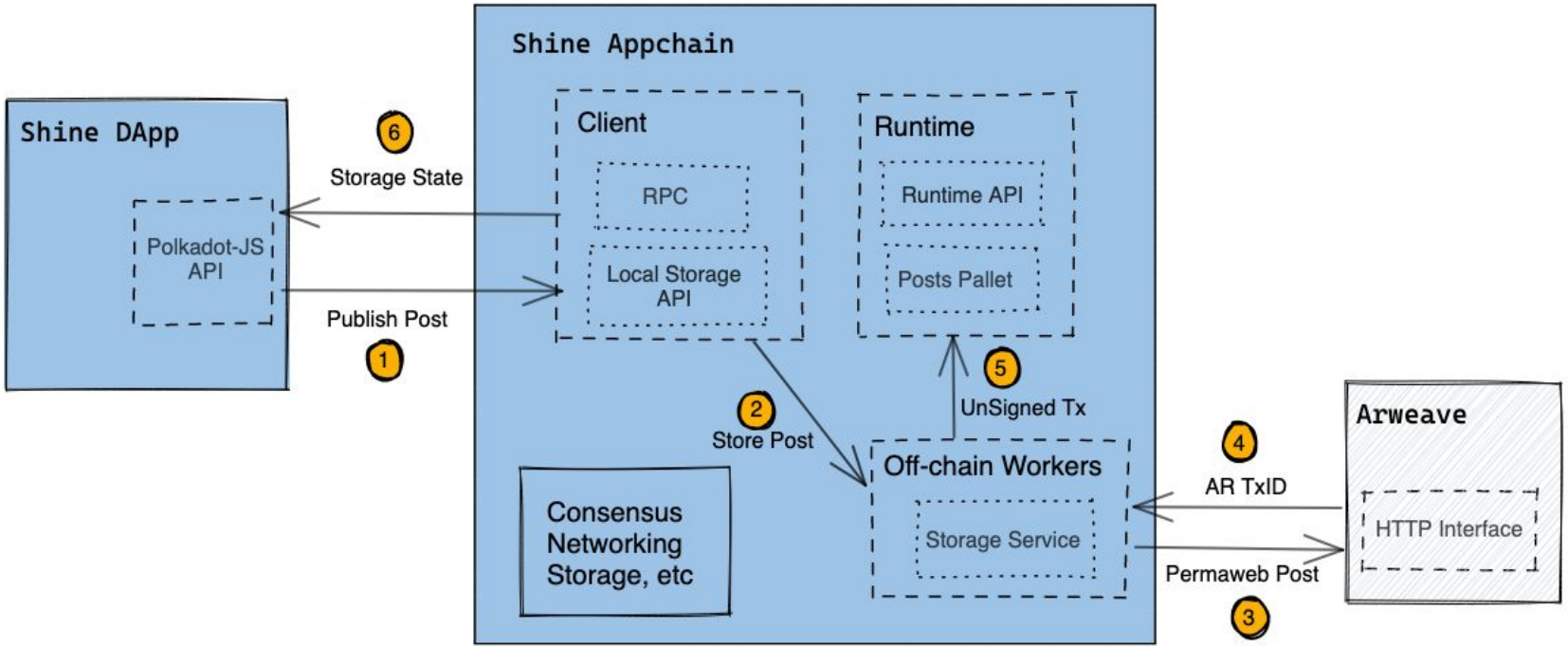
- Storage Value
- Storage Map
- Double Map
- Storage N Map

# 3.3 Oracle 开发示例

**Oracle:** 一个简单的BTC/USD价格预言机，Substrate Repo中示例。

- Offchain Worker 每个区块被触发，获取当前价格并将结果上链
- 链上简单聚合结果计算平均价格并存储

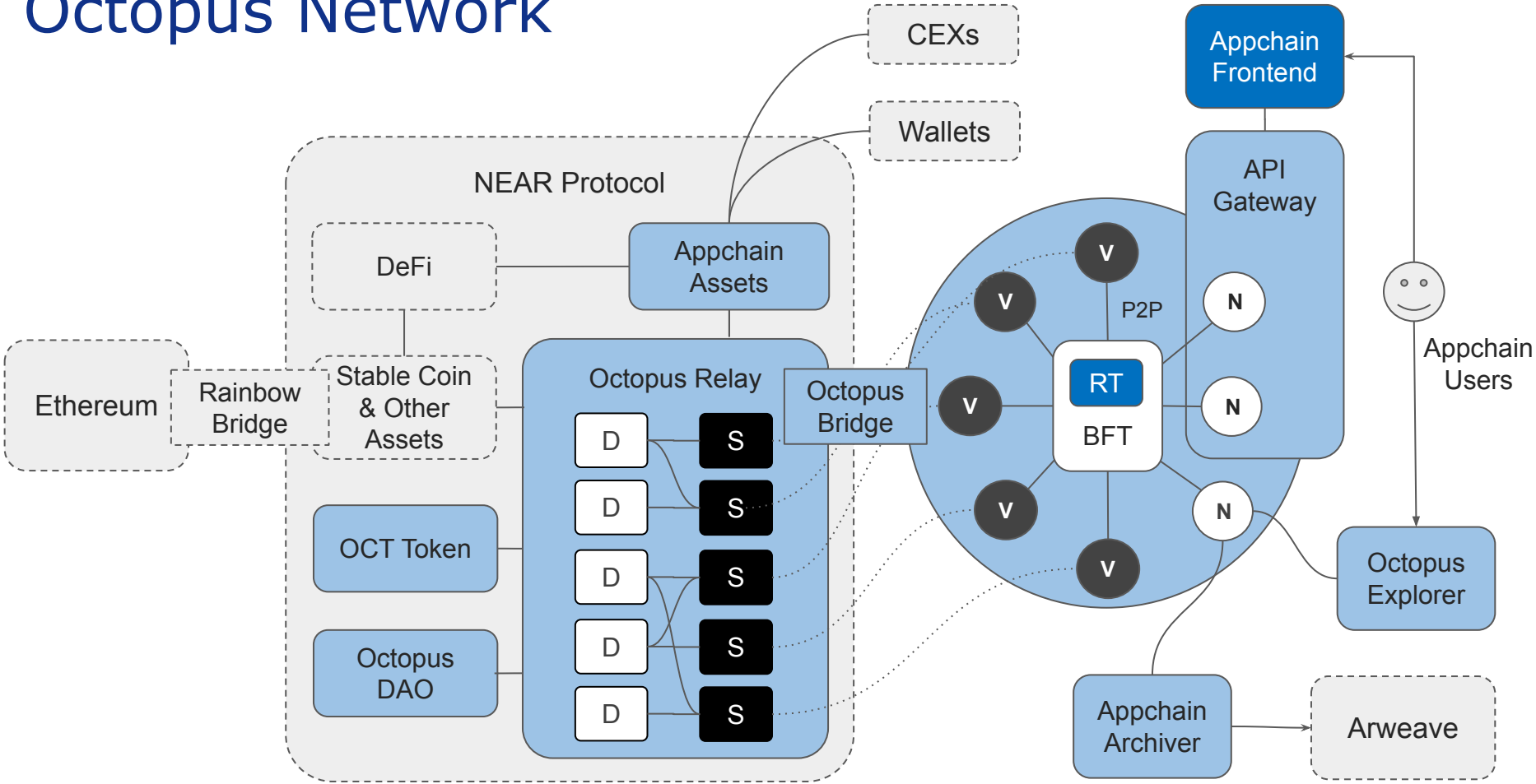
# 3.4 Shine 应用链实践



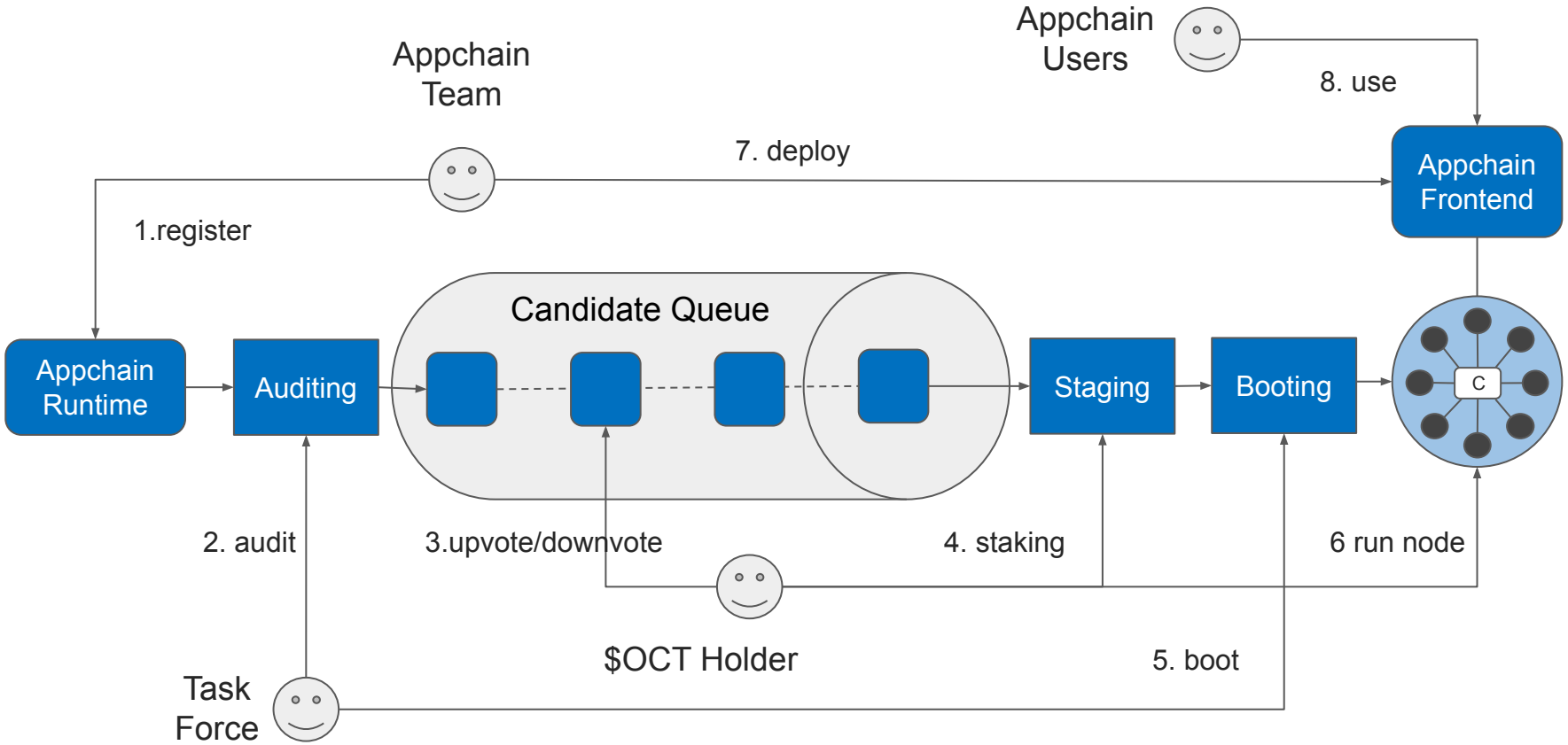
# 如何部署运行 Substrate 应用？

- **独立链？**
- **Octopus Network**
- **集成接入 Octopus**
    - 一键部署服务 octoup
    - 区块链浏览器
    - Gateway，Indexer，Archive等服务

# Octopus Network



# 集成接入 Octopus



octopus network

# 感谢聆听！