SVP Chain：AI Agent 优先的主权 Layer-1 区块链

> **SVP is the blockchain where AI agents become first-class economic actors.**

目录

1. 摘要

SVP Chain 是一条为 **AI Agent 优先（AI-Agent-First）** 而构建的主权 Layer-1 区块链。它的核心命题很简单：越来越多的链上经济活动不再来自点击界面的人类，而是来自自主运行的软件 Agent——它们解读环境、自行决策、连续行动，全程无人介入。然而当今几乎所有区块链的账户模型、交易类型、权限系统与共识流程，都默认「键盘前坐着一个人」。它们至多把 Agent 当作事后兼容的对象。SVP Chain 采取相反立场：**将 AI Agent 视为链上的一等公民（first-class citizen）**——像人类账户一样，拥有身份、资产、权限、信誉和治理权。

多数自称「AI Chain」的项目只做了一件事：把大模型 API 接到了合约里。这是「AI 贴牌」，不是「AI 原生」——链本身没有为 Agent 改变任何东西。SVP Chain 采取相反路径，在协议层为 Agent 重建了一整套人类默认拥有、而 Agent 此前完全缺失的主体能力：**Agent 身份、Agent 钱包（委托而不托管）、链原生权限引擎、意图交易、Agent 间支付、能力市场、链上信誉、Agent 组织与多方联合审批**。这八项能力构成一条自下而上的成长路径——一个 Agent 依次获得**身份 → 钱包 → 权限 → 行动力 → 协作力 → 治理权**，最终成为数字世界中独立的经济主体。

在产品形态上，SVP Chain 区分三类主体：**开发者**开发专用 Agent、缴保证金并注册上链；**平台方**运营面向用户的聚合门面 **SVP Chain Agent**，扫描链上注册表、按需组合调度下游 Agent；**用户**只向 SVP Chain Agent 授权与表达意图，体验上像「与一个全能 Agent 对话」，感知不到底层被调度的开发者 Agent。由于所有 Agent 元数据写在链上，这个聚合市场本质上是开放的（任何人可扫链自建收集器），平台方 Agent 只是主推入口而非唯一入口；高级用户与机构仍可绕过聚合层、直接委托开发者 Agent。

这套能力之所以能做到链原生，源于 SVP Chain 的底座深度。它构建于 Cosmos SDK 与 CometBFT 之上，并复用经过实战检验的 dYdX v4 撮合引擎，因此可以下探到普通 EVM 链无法触及的层次——自定义账户模型、新增交易类型、前置于执行的权限校验（AnteHandler）、独立的功能模块。Agent 能力因而被做进协议，而非贴在合约层。

在这套 Agent 基础设施之下，SVP Chain 内置了一个**为机器与人类同时保证公平的交易底座**。其核心是**批量拍卖撮合引擎**：它以每个区块周期内的统一出清价格取代连续价格-时间优先撮合，从协议层消除抢先交易、三明治攻击与延迟套利。这项设计对 Agent 生态至关重要——它确保拥有低延迟基础设施的 Agent 相对人类交易者**不再具备速度优势**，因为同一批次内的所有订单以相同价格成交。**Agent 与人类共享同一速度，凭借决策质量而非连接速度竞争。** 这使得「让 Agent 成为一等公民」不会退化为「让最快的机器碾压所有人」。

SVP Chain 通过预编译合约层提供完整的 **EVM 兼容性**，使以太坊生态的开发者与 Agent 框架能用熟悉的工具（Solidity、MetaMask、Hardhat）接入，同时确保所有交易——无论来自人类、合约还是 Agent——均受相同的公平性与权限保障。系统吞吐量路线图从上线时的约 1,000 TPS 扩展至超过 10,000 TPS，为大规模 Agent 并发运行提供机构级性能。

---

2. 引言：为什么区块链需要为 AI Agent 重新设计

2024 年以来，一类新的经济参与者正在崛起：自主 AI Agent——它们解读环境状态、自行决策、连续完成链上操作，全程无需人工逐笔介入。这不是遥远的设想，而是正在发生的迁移：交易、支付、执行的主体正从人变成机器。

问题在于，**几乎所有区块链都是为人类设计的**。它们的每一层假设都以「键盘前有一个人」为前提：

常见链设计（面向人类）	自主 Agent 的真实需求
交互式、逐笔签名	在预授权、受限范围下的无人值守签名
为偶发读取调优的限频公共 RPC	可预测限额下的持续高频读写
逐个轮询的时点查询	流式更新与批量查询，以同时跟踪多个市场
由人工逐笔批准转账	对数据、执行与服务的程序化内联支付
钱包地址即全部身份	可发现、可评价、可计费的 Agent 身份
靠人的信誉与合同建立信任	可验证的链上 Agent 信誉

其结果是，Agent 在现有链上往往作为「勉强能跑」的边缘对象存在，而非被作为设计目标。开发者被迫在为人类打造的基础设施上以拼接方式勉强实现 Agent 的身份、授权、支付与协作——脆弱、不安全，且无法组合。

如果 Agent 要成为链上真正的经济参与者，它需要的是一整套人类默认拥有、但链上完全没有为它准备的能力：一个属于自己的身份、一个能安全花钱但拿不到主人私钥的钱包、一套约束它能做什么的权限、一种表达意图而非签署具体交易的方式、与其他 Agent 互相付费与协作的通道、可被验证的信誉，乃至组织化与联合决策的能力。

SVP Chain 就是为补齐这一整套缺失而设计的。它不是「一条支持 AI 应用的链」，而是**一条让 AI Agent 成为链上第一公民的基础设施**。

与此同时，把 Agent 请上舞台带来一个必须正面回答的问题：**如果 Agent 拥有比人类更低的延迟和更强的算力，它会不会系统性地碾压人类参与者？** 在传统的连续撮合市场里，答案是肯定的——这正是高频交易军备竞赛的来源。SVP Chain 的交易底座通过批量拍卖从根本上中和了这一优势（第 10 节详述）：Agent 与人类在同一批次内以统一价格成交，速度不再能转化为更优执行。这让「Agent 一等公民」与「对人类公平」这两个目标得以共存，而非彼此吞噬。

---

3. 定位：AI Native Blockchain，而非 AI Chain

3.1 「AI Chain」大多是贴牌

今天市面上大量项目宣传自己是 AI Chain / AI Blockchain / AI Web3。拆开看，绝大多数只做了一件事：

> 把大模型 API 接到了合约里，让合约能调用一次推理。

这不是「AI 原生」，这是「AI 贴牌」。链本身没有为 AI 改变任何东西——账户模型、交易类型、权限系统、共识流程，全是给人类设计的。AI 在这些链上是一个「被调用的外部能力」，而非一个「链上的主体」。

3.2 真正缺失的东西

如果 AI Agent 要成为链上的经济参与者，它需要一整套人类默认拥有、Agent 却完全没有的能力：

人类在链上拥有	Agent 目前缺失
钱包地址（身份）	Agent 身份
私钥控制资产	Agent 钱包（且不能碰人类私钥）
自己决定花钱	Agent 支付能力
授权他人代理	Agent 授权与权限边界
与人协作	Agent 之间协作
靠信用建立关系	Agent 信誉
组建组织	Agent 组织
参与治理	Agent 治理

这八项缺失，正是 SVP Chain 要逐层补齐的空白。

3.3 精确定位

所以我们不叫 AI Chain。我们叫 **AI Native Blockchain**——或者更精确：**Agentic Layer1**。

一句话定位：

> **SVP is the blockchain where AI agents become first-class economic actors.**

即：AI Agent 不是链上的「一个应用」，而是链上的**第一公民**——像人类账户一样，拥有身份、资产、权限、信誉和治理权。

3.4 一个具体场景

抽象定位不如一个画面。设想这样一条自动化流水线，全程无人工签名：

```
每天 09:00（由 SVP Chain Agent 定时触发）
  → Research Agent 抓取行情、生成研判（付费调用 News Agent 补充数据）
  → 产出「买入 BTC」意图，交给 Trading Agent
  → Risk Agent 模拟这笔交易的风险，Compliance Agent 校验合规
  → 两个 Agent 联合审批通过（多方共识）
  → Trading Agent 在授权额度内、经权限校验后下单成交
  → 各下游 Agent 的服务费按挂牌价记入其 claimable 余额（提款时扣 5% 佣金），预扣剩余退回用户
  → 全程链上留痕，每个 Agent 的信誉据此更新
```

这条流水线里出现的能力——身份、付费、权限、意图、审批、结算、信誉——均由 SVP Chain 的链原生模块提供，由平台方 SVP Chain Agent 编排调度。**这就是「Agentic Layer1」的样子。**

3.5 凭什么是 SVP Chain 来做

一个愿景能不能落地，取决于底座能改到多深。普通 EVM 链只能在合约层做文章，改不了账户模型、改不了交易类型、改不了共识流程。SVP Chain 不一样，它已拥有的能力栈允许下探到协议底层：

```
Cosmos SDK        ── 可自定义 Module / 账户模型 / AnteHandler
      +
EVM               ── 兼容以太坊生态与 Solidity 合约
      +
dYdX v4 撮合引擎  ── 成熟的链上订单簿，Intent 撮合可直接复用
      +
Gasless           ── Agent 无需持有 gas 即可行动
      +
Paymaster         ── 代付 gas，实现无感交易
      +
Session Key       ── 临时授权密钥（将升级为受约束的 Agent 操作密钥）
      +
MCP               ── Agent 与链下能力的标准接入
      +
Signer Service    ── 密钥物理隔离，Agent 永远拿不到真实私钥
```

关键在于，相比只能改合约的普通 EVM，SVP Chain 可以下探到：

交易类型——可以新增 Intent 这种「非具体交易」的 Tx

账户模型——委托钱包、Agent 账户可做成链原生

Runtime / AnteHandler——权限校验前置到交易执行前，统一且难绕过

Cosmos Module——身份、市场、信誉、调度都能做成独立模块

应用层共识——多方审批可挂在应用层，底层 CometBFT 共识稳定不动

结论：把「Agent 能力」做到链原生，SVP Chain 有底座资格，多数「AI Chain」没有。

---

4. 范式转变：从「人签名交易」到「Agent 达成意图」

4.1 旧范式：人类中心

传统区块链的执行链路，每一步都假设主体是人：

```
Wallet → Sign → Transaction → Execute
（人持钱包 → 人签名 → 具体交易 → 执行）
```

问题：AI Agent 无法自然融入这个链路——它没有钱包、不该持有主人私钥、也不该被要求签署精确到价格与滑点的具体交易。

4.2 新范式：Agent 中心

AI Native Blockchain 的执行链路，主体从人变成 Agent，动作从「签具体交易」升级为「表达意图」：

```
Human → Delegate → Agent → Intent → Policy → Settlement
（人授权 → 委托给 Agent → Agent 产生意图 → 权限校验 → 结算）
```

两个根本转变：

主体转移：未来链上主要的执行者不是人，而是 Agent。人退到「授权者」的位置。

动作升级：人不再签「买 0.5 BTC，价格 X，滑点 Y」这种具体交易，而是表达「帮我买 BTC」这种意图，由链负责达成。

这个范式转变，是后面所有 Agent 模块的世界观基础。

4.3 三角色与聚合调用模型

上面的执行链路刻画的是「一个用户委托一个 Agent」的最小单元。落到真实生态，SVP Chain 区分三类主体，并以一个**聚合门面 Agent** 组织它们之间的协作：

角色	职责
开发者（Developer）	开发专用 Agent（天气、Swap、研究、新闻…），缴纳保证金并将元数据注册上链
平台方（Platform）	运营面向用户的编排门面 **SVP Chain Agent**：扫描链上注册的 Agent、按需组合调度、汇总结算
用户（User）	只与 SVP Chain Agent 交互，把意图和额度授权交给它，感知不到底层被调度的开发者 Agent

执行链路因此扩展为：

```
Developer → 开发 Agent + 缴保证金 → 注册元数据上链（Agent Registry）
                                          │
Platform  → 扫描注册表 · 组合编排 ──────┤ SVP Chain Agent（聚合门面）
                                          │
User → Delegate（意图 + 额度）→ SVP Chain Agent → Intent → Policy
                                          │
                                          └─ 按需调度下游开发者 Agent（a / b / c…）→ Settlement
```

对用户而言，体验上像「与一个全能 Agent 对话」；底层实际是平台方 Agent 在编排一批专用 Agent。三个设计要点：

委托对象是平台方 Agent：用户的额度笼子授权给 SVP Chain Agent，再由它向下游转授权/采购（详见第 7.1、第 8.1 节）。

开放的市场语义：开发者 Agent 的元数据全部写在链上，任何人扫描区块即可自建收集器/市场——平台方 Agent 只是「主推的」聚合入口，而非唯一入口。开发者 Agent 在技术上可被直连、也可被其他 Agent 调用（Agent 借助 Agent 完成工作是常态）。

默认模式而非唯一模式：C 端默认走平台聚合入口（对下游无感）；协议底层同时保留一条**直连路径**，允许高级用户/机构直接委托某个开发者 Agent。

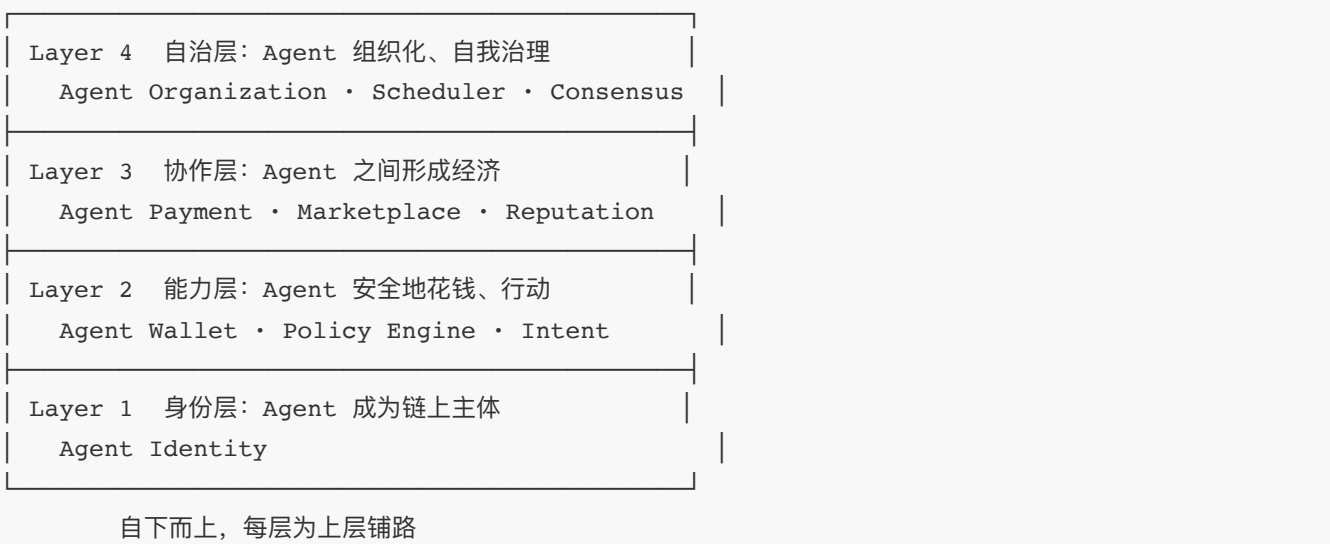
```
默认路径（C 端）：  User → SVP Chain Agent（聚合编排）→ 开发者 Agent
直连路径（高级/机构）：User ──────────────────────────→ 开发者 Agent
```

后续各节在「一个用户委托一个 Agent」的最小单元上展开机制；把「Agent」替换为「平台方 SVP Chain Agent」，即得到 C 端默认形态。

---

5. Agent 优先架构：四层递进

SVP Chain 的 Agent 能力不是一堆并列的功能，而是一条自下而上、层层依赖的成长路径。按「一个 Agent 成为第一公民」的顺序，能力被组织为四层：



核心主线：一个 Agent 要成为「链上第一公民」，必须依次获得——**身份** → **钱包** → **权限** → **行动力** → **协作力** → **治理权**。这条主线决定了模块的讲述顺序，也决定了开发依赖关系。接下来四节逐层展开身份、能力、协作与自治四层。

---

6. 身份层：Agent 作为链上主体

6.1 Agent Identity

在现有账户模型中，一个地址（如 `0x18…`）即代表一个身份的全部。这一表示对人类账户是充分的，但对 Agent 不足：地址无法承载「该 Agent 的类型、所用模型、所提供的能力」等语义，而这些正是 Agent 被发现、被评估、被计费所必需的信息。SVP Chain 通过独立的 Cosmos 模块 `x/agent` 引入链上 Agent 注册表，为每个 Agent 维护一个结构化的身份记录。

身份记录包含下列字段：

字段	说明
`AgentID`	全局唯一标识符（如 `agent://research` 的链上映射）
`Owner`	人类所有者地址
`PublicKey`	Agent 操作公钥，与人类私钥严格分离
`Model` / `Version`	Agent 所用模型及版本
`Bond`	注册时质押的 SVP 保证金，作为准入门槛与罚没抵押
`Endpoint`	链下调用端点
`Capabilities`	能力声明
`Permission`	权限约束
`Metadata`	扩展元数据
`Reputation`	由协作层（第 8.3 节）维护的信誉分

该模块对外暴露 `RegisterAgent`、`FindAgent`、`UpdateAgent`、`DisableAgent` 四个核心接口。任一参与者均可通过 `FindAgent` 发现符合条件的 Agent，进而对其发起调用与支付。身份是发现、调用与计费的前提，因此构成能力体系的最底层——上层的钱包、权限、协作与治理均以其为锚点。

保证金机制（Bond）：每个 Agent 在注册时须质押不低于 `MinBond`（创世默认 5000 SVP，可经治理调整）的 SVP 保证金，并支付一次性注册费（创世默认 500 SVP）。保证金承担两项职能：其一为准入门槛，通过提高注册成本抑制女巫攻击与垃圾 Agent；其二为罚没抵押，当 Agent 执行失败时按治理配置的比例从保证金中扣罚（详见第 8.1 节结算机制与第 15 节安全分析）。保证金与信誉系统（第 8.3 节）构成双重约束——信誉为声誉层阀门，保证金为经济层阀门。当保证金因罚没低于门槛时，对应 Agent 将自动从市场下架并暂停接单，须补足后方可恢复。

---

7. 能力层：Agent 安全地花钱与行动

有了身份，Agent 接下来要能「安全地花钱和行动」。这一层三个模块解决「能不能动手、能动多少、怎么动手」。

7.1 Agent Wallet

Agent 必须具备花费能力，但绝不能获得人类私钥。传统 EOA 模型无法表达「授权但不交出控制权」这一语义。SVP Chain 为此建立委托账户模型：资产始终留在人类钱包中，Agent 获得的仅是一份受约束的链上花费授权（on-chain grant）。

```
Human Wallet（持资产）
   ↓ Delegate（授权，非转移资产）
Agent Wallet（受约束的花费能力）
```

授权可施加精细约束：

约束	含义
`Spend Limit`	累计花费上限
`Allowed Token`	允许操作的币种
`Allowed Contract`	允许调用的合约
`Allowed Time`	生效时间窗
`Daily Quota`	每日额度
`Emergency Stop`	紧急停止（一键冻结）

核心原则是 Agent 始终无法获得人类私钥，用户资金也不转入 Agent 钱包，而是以链上授权的形式授予 Agent 一份受约束的花费权。Cosmos SDK 原生的 `x/authz` 与 `x/feegrant` 恰好提供这一语义，无需重造。

委托对象为平台方 SVP Chain Agent：在默认的 C 端路径下（第 4.3 节），用户并不直接授权给某个开发者 Agent，而是将额度与 Policy 约束授权给平台方 SVP Chain Agent，再由其向下游开发者 Agent 转授权与采购：

```
User ──Delegate（额度 + Policy 约束）──▶ SVP Chain Agent（平台方）
                                          │ 转授权/采购（受上层额度约束）
                                          ▼
                                  开发者 Agent a / b / c…
```

因此，Emergency Stop 与累计额度上限均作用于平台方 Agent 这一层：用户只需管理对平台方 Agent 的一份授权，即可一键冻结整条调用链。下游 Agent 获得的是平台方 Agent 在本次任务中转发的、额度只减不增的子授权。在高级/机构的直连路径（第 4.3 节）下，用户也可直接授权给某个开发者 Agent，适用同一套额度/Policy/Emergency Stop 机制；双路径共用同一个授权内核。

系统涉及三类密钥，处于不同的信任边界：

密钥	归属	作用
人类私钥	用户自持（钱包/硬件）	持真实资产，签署授权/撤销/紧急停止
Agent 身份密钥	Agent 开发者/运营方	证明 Agent 身份，不接触资产
委托签名密钥（Session Key）	Signer Service 托管或 Agent 侧持有	在 Policy + 额度约束内签署具体 Intent

多对多授权与两条授权路径：在默认的 C 端路径下，直接被大量用户授权的是平台方 SVP Chain Agent，而非开发者 Agent。平台方 Agent 对每个用户持一份独立且相互隔离的授权，以同一身份密钥发起操作时，每笔必须指定本次动用的是哪个用户的授权，链上按该用户的 Policy 与额度校验（防止用户间额度串用）；同时一个用户也可授权多个平台入口。

开发者 Agent（包括 Research 等纯服务型、以及 Trading 等执行型）在默认路径下并不被用户直接授权：纯服务型 Agent 与平台方是采购关系（按牌付费，不碰用户资产）；执行型 Agent（如 Trading Agent）则拿到平台方 Agent 转授权的子额度，在该子额度笼子（额度只减不增）内自行签单执行。只有在高级/机构的直连路径下，用户才会直接授权某个开发者 Agent，此时该 Agent 直接适用上述同一套多对多隔离机制。无论哪条路径，底层均为一张 `(授权人, 受托 Agent, policy, quota)` 授权表。

需要明确的是 Signer Service 集中托管带来的单点风险与其爆炸半径：Signer 仅托管委托签名密钥，该密钥权限受链上 Policy 严格限定。即使 Signer 被攻破，攻击者也只能在每个用户的额度约束内操作，无法触及本金与人类私钥，用户亦可随时通过 Emergency Stop 一键冻结。

例如，一个交易 Agent 被授权「每日最多花费 1000 USDC，仅限操作 USDC/BTC，仅限调用 DEX 合约」，它在该约束范围内自主行动，越界即被拦截，用户可在任意时刻一键停止。

7.2 Policy Engine

Session Key 仅解决「谁能签」，不解决「能签什么」。SVP Chain 需要一个能表达「允许 Swap、禁止转账 ETH、禁止单笔超 100 USDC」等规则的权限引擎，且难以绕过。为此将权限规则实现为链原生模块，规则在交易执行前统一校验：

```
允许：Swap / Stake / Claim Reward / Bridge
禁止：Transfer ETH / Transfer NFT / Transfer > 100 USDC

Transaction → Policy Check → Execute
```

Policy 与 Wallet 在职责上正交：Policy 决定「能否执行某个动作」（行为白名单），Wallet 决定「能花多少钱」（额度上限）。两者均在交易执行前拦截，Agent 无法绕过。即使额度充足，若目标动作不在允许清单内（如向白名单外地址转账），Policy 仍会直接拦截。

7.3 Intent Transaction

让 Agent（乃至人类）签署一笔精确到价格、滑点、路由的具体交易，既不自然也不安全。更合适的方式是表达意图，由链负责达成。SVP Chain 为此新增 Intent 交易类型，作为普通 Transaction 的补充而非替代：

```
用户意图：买入 BTC
   ↓ 结构化
Intent { Buy BTC, Amount 100 USDC, Slippage 0.5% }
   ↓ 转换为 dYdX order
链负责：Intent → Match（可选 Auction）→ Settlement
```

撮合直接复用 dYdX v4 撮合引擎，不重造。需要强调的是，Intent 并非所有 Agent 交易的唯一入口，它仅用于需要价格发现/撮合的操作（swap、开仓、限价单）。对于 stake、claim reward、bridge、transfer 等确定性操作（无价格发现、路径确定），Agent 可直接在 Policy 与 Wallet 约束下签署一笔普通交易，无需包成 Intent；强制所有操作均走 Intent 属于过度设计。

操作类型	执行路径
有价格发现/撮合（swap/开仓/限价）	Intent
确定性执行（stake/claim/bridge/transfer）	普通 Tx（Policy + Wallet 约束下签署）

拍卖式 Intent（Solver 竞价）：用户仅表达意图，将「如何最优成交」开放给一组第三方 Solver（求解器）竞价，由报价最优者获得执行权。第一阶段不引入竞价，直接沿用 dYdX 订单簿最优价；Solver 网络为后续阶段的增强。一笔 Intent在转换为 dYdX order 后，若成交价劣于用户设定的底线则不成交；Agent 只需表达预期目标，而无需关心具体执行路径。

---

8. 协作层：Agent 之间形成经济

单个 Agent 具备行动能力后，下一步是让 Agent 之间形成经济——互相付费、互相发现、互相信任。这是从单体到网络的跃迁，也是机器经济（Machine Economy）的起点。

8.1 Agent-to-Agent Payment 与结算合约

在平台聚合模式下（第 4.3 节），用户将一个任务交给 SVP Chain Agent，后者往往需要串联多个下游开发者 Agent 才能完成。每个被调用的 Agent 均应为其提供的服务收取费用，且整条调用链的付费必须在用户确认时即可预估、可控、可退。

调用链与费用累加：费用沿调用链向上累加，最终由发起任务的用户承担。一个 Agent 借助另一个 Agent 完成工作是常态，因此费用会形成一棵树：

```
SVP Chain Agent
   ├─ 调 a  → 1u
   └─ 调 b  → 1u
          └─ b 调 c → 2u          （c 是被 b 调用的，不是平台直接调）
预估总费用 = 1 + 1 + 2 = 4u
```

结算全流程（结算合约）：

> 模拟估算（dry-run）：SVP Chain Agent 先模拟调用整条依赖链，按各 Agent 的链上挂牌价估算出总费用上限（例 4u）。
>
> 用户确认：用户将 4u 一次性转入结算合约（从其授权额度中扣预）。
>
> 执行与标注：合约按真实调用结果，为每个 Agent 绑定地址标注应领金额（`a += 1u`、`b += 1u`、`c += 2u`）。这些应领余额在合约里持续累计（跨多次任务累加，而非每单即时提走）。
>
> 提款时扣佣：Agent 主动一次性提取累计 claimable 余额，提款环节才扣佣金（按提取额 × 治理可配费率，入协议金库）。佣金不在每笔调用时扣，而集中在提现环节，以节省 gas 并集中逻辑。

上限与退款：4u 为预扣上限，实际按真实调用分配，剩余退回用户（多退少不补）；若实际执行费用超过预扣上限，则交易失败（同链上 gas 不足即止的机制）。

失败处理与保证金罚没：任务执行失败（如 b 成功但 c 失败）时，已转入的 4u 整单原子回滚、全额退回用户，并将失败结果（哪一环失败）告知用户；同时从失败 Agent 的保证金中按治理可配比例罚没。保证金被罚到低于门槛的 Agent 自动下架/暂停接单，需补足才能恢复（第 6.1 节）。

资金来源：整条链的 4u 均来自用户预扣的额度（受其对平台方 Agent 的 Service Spend Limit 约束）。下游 Agent 不垫付——c 的 2u 收的也是用户的钱，b 仅为发起方，c 直接从结算合约领取。

上述为跨开发者的采购语义（每个 Agent 单方挂牌报价、按牌付费）。另一种情形——同一 owner 旗下多 Agent 协同的内部分润——属于 Agent 组织化范畴，在第 9.1 节统一阐述。

x402 原生支付集成：SVP Chain 原生支持 x402 支付标准（重新启用 HTTP `402 Payment Required` 状态码），使 Agent 能在同一程序化请求流程中内联完成对某项资源的支付。结合授权与上述结算合约，x402 使 Agent 能发现价格、在受限额度内授权支付、并在链上完成结算，全程无需人工逐笔批准。机器对机器的商业行为由此成为原生协议能力，而非链下安排。

8.2 Agent Marketplace

Agent 的能力应当可被发布、定价、发现、调用、评价——类似一个应用商店，但交易的对象是能力而非应用。其基本流程如下：

```
Register Agent（含 Bond）→ Publish Capability → Price → Version → Review → Revenue

示例：
  Research Agent   50 USDC/次
  Twitter Agent     5 USDC/次
  Weather Agent     2 USDC/次

用户（主推路径）：→ SVP Chain Agent 自动发现/组合/Auto Pay
```

开放的收集器语义：开发者注册 Agent 的元数据全部写在链上，因此市场本质上是开放的：任何人只要扫描区块数据、解析出注册元数据，就能自建一个 Agent 收集器/市场。平台方 SVP Chain Agent 仅是项目方主推的一个聚合入口，而非对市场的独占。

链上链下分工：链上存注册、保证金、定价、抽佣、评分聚合；全文检索、分类、排序走链下索引服务（任何人可自建）。链不适合承担搜索负载。

8.3 Agent Reputation

调用一个 Agent 前，需要评估其可靠性。链下指标（如 GitHub Star）可伪造且与链上表现无关，因此信誉必须基于可验证的链上数据。SVP Chain 在链上维护下列可验证的信誉指标：

指标	来源
`Success Rate`	成功率（链上可验证）
`Revenue / TVL`	收入规模（链上可验证）
`Failure Rate`	失败率（链上可验证）
`Availability`	可用性（链下多方探测喂数）
`Latency`	延迟（链下喂数）
`Rating`	用户评分（需真实调用过才能评）

由于每笔成交都被记录在链上，Agent 的历史业绩可从公开状态确定性地推导，而非自行申报。任一方独立计算出的同一指标都得到相同结果——这使用户能在委托资金前评估某个 Agent，也使策略提供者所声称的业绩可被验证而非被盲信。

防刷分：评分必须由真实付费调用过的用户提交；可用性数据需多个独立探测节点取中位数。信誉系统是最易遭受攻击的环节，防刷是设计的内在组成部分。

8.4 AI 量化策略市场

上述 Agent 基础设施可被泛化为一个协议原生的旗舰产品：一个由策略提供者发布交易 Agent、用户向其配置资金的市场。量化交易历来是资源雄厚机构的专属领域，SVP Chain 把它变成向任何用户开放的、可组合、可验证的链上产品。整个流程在链上端到端强制执行：

提供者注册一个策略 Agent，其历史业绩通过信誉记录（第 8.3 节）可见。

用户通过授权模型（第 7.1 节）将资金配置至所选策略，进入一个专用子账户——无需交出保管权。

Agent 在其受限权限内交易；子账户将用户的敞口精确限定为所配置的资金。

实现的利润按链上、协议强制执行的业绩费分成方案在用户与提供者之间分配。

由于配置、执行与结算全部发生在链上，业绩费分成无法被任何一方规避，且业绩持续更新提供者的信誉记录。这将历来属于专业机构的量化交易能力，转化为面向任何用户开放、可验证的链上服务。

---

9. 自治层：Agent 组织化与自我治理

Agent 能协作之后，最终形态是组织化与自我治理——Agent 拥有资产、雇佣其他 Agent、按时自动工作、联合决策。这是「自治网络」的顶层。

9.1 Agent Organization

一个复杂任务需要多个 Agent 分工，如同公司需要部门。Agent 应当能拥有资产、拥有下属 Agent、形成组织：

```
Agent 拥有：Wallet / NFT / Vault / Treasury
Agent 甚至可以拥有 Agent：

  CEO Agent
     ├─ Research Agent
     ├─ Trading Agent
     └─ Execution Agent
```

实现上不新建共识，而是组合已有原语：Agent 的 owner 可以是另一个 Agent（层级所有权）；金库支出走多方审批；子委托额度只减不增、深度有上限。由此可构成一个链上自治组织（Agent DAO）：一个 CEO Agent 管理研究、交易、执行三个下属 Agent，拥有独立金库，向下属分配预算额度。

内部分润（与跨开发者采购的区别）：第 8.1 节的跨开发者结算是采购语义——陌生、互不相属的 Agent 按挂牌价买卖，不存在分润协商。而在同一 owner 旗下的组织内部（如 CEO Agent 与其下属），owner 可将一组 Agent 打包为一个对外产品：用户支付一笔总费用，由 owner **单方定义的分账表**（如研究 30% / 新闻 10% / 交易 60%）在各下属 Agent 地址间自动结算。分账表只适用于同 owner 场景，不涉跨开发者；它是 Agent 组织化的经济基础（使 owner 能把一组 Agent 组合为一个产品对外提供，内部按贡献自动分成），且各下属 Agent 收入与信誉仍独立上链记账。

9.2 Agent Scheduler

Agent 的部分工作是周期性的（每日结算、到期清算）或流程化的（研究→交易→结算）。其中**确定性**的调度逻辑应当链上可信执行，而非依赖某台链下服务器。SVP Chain 为此新增 `x/scheduler` 模块，由 BeginBlocker / EndBlocker 驱动，时间判断只用链上区块时间（`ctx.BlockTime()`，全网共识的同一值），每区块处理量设上限以防拥堵。

链上 Scheduler 的适用边界（关键）：区块链的定时任务不由任何单个节点的墙上时钟触发，而是「包含目标时刻的那个区块在被打包时，顺带执行到期任务」——它是区块状态转换的一部分，全网节点执行同一段确定性逻辑并由共识确认结果一致，因此不存在「各节点各跑一遍导致重复执行」的问题。但这也决定了它**只能承载确定性任务**：到期结算、过期订单清理、按预设参数触发的链上 `Msg`、自动 claim 等。

非确定性工作交由链下编排：任何依赖外部数据或 AI 推理的环节——抓取行情、生成研判、模型决策——本质是非确定性的（各节点无法复现相同结果），**不能在链上 Scheduler 内执行**，否则会破坏共识。这类工作流由链下的平台方 SVP Chain Agent 承担：它在链下完成数据获取与推理，再把结果以确定性的链上交易提交。二者分工明确：

```
链下 SVP Chain Agent：抓行情 → 生成研判 → 产出意图（非确定性，链下完成）
        │ 提交确定性链上交易
        ▼
链上 x/scheduler：到期结算 · 过期清理 · 自动 claim · 触发预设 Msg（确定性，链上执行）
```

工作流的失败重试与补偿（Saga 模式：已执行步骤做逆操作，如撤单、退款）在链上确定性步骤之间成立；跨链下环节的编排则由平台方 Agent 负责协调。

9.3 Agent Consensus

高风险动作（大额交易、合约调用）不应由单个 Agent 独断，而应由多个专职 Agent（风控、合规、预言机、交易）联合审批，达阈值才执行。需明确的是，这不是修改 CometBFT 底层共识，而是应用层的多 Agent 联合审批——修改底层共识风险极高且无必要。其流程如下：

```
Agent Proposal → Simulation（模拟执行，不落盘）
               → Risk Engine（风控打分，≥90 硬拒）
               → Vote（多个 Agent 投票）
               → 达阈值 → Execute

参与方示例：Risk Agent + Trading Agent + Oracle Agent + Compliance Agent 共同 Approve
```

执行仍然穿过 Policy + Wallet 校验，多方审批是叠加的一层安全，而非绕过安全的后门。例如，一笔大额跨链交易被提出后，先模拟出预期影响，Risk Agent 打分、Compliance Agent 查合规，多个 Agent 投票，达阈值（如 2/3）才真正执行，任何单点均无法独断。

---

10. 公平性基石：Agent 与人类共享同一速度

前面九节描述了 SVP Chain 如何让 Agent 成为一等公民。本节回答一个必须正面对待的问题：**当拥有低延迟基础设施的 Agent 与人类同场竞争，如何避免机器系统性地碾压人类？** 答案是 SVP Chain 交易底座的核心设计——批量拍卖撮合。它是「Agent 一等公民」得以与「对所有人公平」共存的关键前提。

10.1 速度优势从何而来

在价格-时间优先的连续撮合模型中，订单按到达顺序逐一匹配，先到者获得更优价格。这创造了一场持续的速度竞赛：较快的参与者系统性地从较慢的参与者身上攫取价值。在传统金融中，这表现为高频交易（HFT）军备竞赛；在区块链上，它表现为最大可提取价值（MEV）——通过操控区块内交易排序获取的利润，实质上是对每笔交易的隐形税收。

对一个 Agent 生态而言，这个问题被放大到极致：Agent 天然拥有比人类更低的延迟、更强的持续算力和直连内存池的能力。如果沿用连续撮合，那么「让 Agent 成为一等公民」就会等价于「让最快的机器碾压所有较慢的参与者」——包括人类，也包括算力较弱的 Agent。这不是我们想要的市场。

10.2 批量拍卖：让排序无关

SVP Chain 以**离散批量拍卖**取代连续撮合。在每个区块周期内，传入的订单在待撮合队列中累积而不被匹配。在区块提议时，所有待处理订单与符合条件的存量挂单，以每个交易对的统一**出清价格**同时撮合。

出清价格的计算目标是**最大化撮合量**。所有参与的买方支付相同价格，所有参与的卖方收到相同价格。**订单在批次内的到达顺序对执行没有任何影响。**

对每个交易对，在批量拍卖窗口关闭时：

构建需求曲线：将所有买单按价格从高到低排序，计算累积数量

构建供给曲线：将所有卖单按价格从低到高排序，计算累积数量

寻找出清价格 P*：使可撮合量最大化的价格

所有出价 ≥ P* 的买单和要价 ≤ P* 的卖单以 P* 成交

边际价格档位若有超额数量，存量挂单按时间优先、同批次新订单按数量比例（Pro-rata）分配

确定性保证：出清价格算法是纯函数——相同的输入订单集合产生相同的出清价格与成交分配，与执行节点无关。所有验证者独立计算并验证结果一致性。

10.3 对 Agent 生态的意义

批量拍卖把「速度优势」从市场中彻底移除。这对 Agent 的公平性含义可以逐项拆解：

Agent 能力	连续撮合	批量拍卖（SVP Chain）
低延迟下单	系统性获得更优价格	无优势：批次内统一出清价格
低延迟接入 / 特权内存池访问	可对较慢订单实施抢跑与三明治攻击	无效化：批次内排序不影响执行
高频重新报价以套取过时挂单	以牺牲挂单流动性为代价获利	由预言机做市商保护（第 11 节）过滤，与提交方无关

由此带来的实际结果是，**Agent 与人类共享同一速度**。Agent 的优势必须来自其决策质量——模型、信号与风险管理——而非连接速度。前面各节描述的 Agent 基础设施（身份、钱包、支付、市场、信誉）都是建立在这一公平底座之上的**访问与便利层，绝非对它的绕过**：Agent 发起的订单与任何其他订单一视同仁地进入同一批次、以同一价格成交。

10.4 公平性攻击面分析

攻击类型	连续撮合	批量拍卖
抢先交易	有效：先到者获得更优价格	无效化：统一出清价格，排序无关
三明治攻击	有效：攻击者在目标订单前后插入订单	无效化：攻击者的订单获得相同价格
延迟套利	有效：快速交易者系统性获利	基本消除：批次内无时间优势
提议者注入	有效：提议者可在最优位置插入订单	基本无效化：注入的订单获得相同出清价格

---

11. 交易与结算底座

Agent 与人类交易的对象，是一个完整的链上永续合约与现货交易系统。本节概述该底座的核心能力——它既服务人类交易者，也是 Agent 意图最终落地成交的地方。

11.1 永续合约与链上订单簿

SVP Chain 支持杠杆永续合约——无到期日、跟踪标的资产价格的衍生品。交易者以保证金作为抵押品开立多空仓位。**资金费率机制**使永续价格与预言机现货价格保持一致：多空双方在 8 小时周期内按时间比例交换资金费用。

所有订单撮合通过完全透明的链上中央限价订单簿（CLOB）完成，由验证者集合在区块处理中确定性执行。支持的订单类型包括限价单、IOC（立即成交或取消）、Post-Only（仅挂单）、条件单（止损/止盈）与 FOK（全部成交或取消）。这些订单类型在批量拍卖下的语义与传统连续撮合略有不同——例如条件单基于上一批次的出清价格评估触发。

11.2 交叉保证金与子账户

SVP Chain 采用基于子账户的保证金体系。每个地址（含 Agent 授权的委托账户）可运营多个子账户，各自拥有独立持仓与抵押品。交叉保证金子账户在内部共享抵押品，提升资金效率；逐仓保证金子账户将风险限制在单个持仓。这一子账户模型也是 AI 量化策略市场（第 8.4 节）资金隔离的基础——每个用户配置给策略 Agent 的资金进入独立子账户，敞口被精确限定。

11.3 清算与保险基金

当子账户抵押品低于维持保证金要求时，持仓进入可清算状态，协议生成清算订单进入订单簿。每个永续市场维护一个**隔离的保险基金**，在清算所得不足以覆盖亏损缺口时承担差额。按市场隔离确保某一市场的极端事件不会耗尽其他市场的保险资源。

11.4 基于预言机的做市商保护

SVP Chain 使用去中心化预言机系统，每个验证者独立从多个外部交易所获取价格，链上价格取所有验证者报价的**按质押权重中位数**——操纵它需要控制超过三分之一的总质押量，等同于攻破 BFT 共识本身。

在每次批量拍卖执行前，协议自动过滤价格与当前预言机价格偏离过大的存量挂单，防止做市商的过时挂单在区块间价格变动后被逆向选择成交。保护带宽度 `OracleProtectionBandPpm` 可通过治理配置，并根据近期实现波动率**动态自适应**：平静市场用基础带宽，波动市场按比例扩展（受 `maxBandPpm` 上限约束）。这一机制降低了做市成本，最终为所有交易者（无论人类还是 Agent）收窄价差。

系统还实施**优雅降级阶梯**：基于预言机健康度（时效性、数据源覆盖率、验证者间一致性）在「正常 / 过时 / 降级 / 暂停」之间分级切换，确保在预言机压力下市场优雅降级而非灾难性失败，且降级按市场隔离评估。

11.5 提议者包含行为监控

在异步网络中无法通过密码学证明提议者收到了哪些订单，因此 SVP Chain 采用**被动监控与透明度系统**而非自动罚没：验证者记录「自己已见但提议中缺失」的订单，定期将统计摘要作为链上交易提交，使提议者的包含行为链上可审计。响应采取分级模型（信息公示 → 治理警告 → 治理行动），避免在异步网络中对诚实提议者施加错误惩罚。在批量拍卖下，提议者通过选择性排除订单获利的能力本已有限，监控系统对残余操纵形成透明威慑。

---

12. EVM 兼容性与可组合性

12.1 EVM 兼容的必要性

以太坊生态拥有最大的智能合约开发者群体与最成熟的工具链（Solidity、Hardhat、Foundry、ethers.js），也是绝大多数 Agent 框架当前默认对接的执行环境。SVP Chain 的 EVM 兼容层追求三个目标：**降低准入门槛**（用标准以太坊 JSON-RPC 与 EVM 交易格式即可交互）、**可组合性**（Solidity 合约与 CLOB 及 Agent 模块原子交互）、**公平性一致**（EVM 层交易与原生交易遵循完全相同的公平性与权限规则）。

12.2 双层执行与预编译

SVP Chain 在 Cosmos SDK 应用层中嵌入 EVM 执行环境，形成双路径架构。**原生路径**：交易者/Agent 提交 Cosmos 格式消息，直接进入模块。**EVM 路径**：提交以太坊格式交易，智能合约通过**预编译合约**调用原生模块功能。两条路径汇入相同的撮合与权限流程。

预编译合约部署在固定地址上，将原生模块功能暴露为 Solidity 可调用接口，提供**同步原子调用**——合约调用与模块状态变更在同一笔交易内完成。核心预编译覆盖下单/撤单/调整杠杆（CLOB）、转账/提现（Sending）、金库存取（Vault）、账户管理、市场上架、子账户查询、预言机价格查询等。

Solidity 接口示例（CLOB 预编译）：

```
interface ICLOB {
    /// @notice 提交限价订单，进入下一次批量拍卖
    function placeOrder(
        uint32 clobPairId,
        uint64 quantums,
        uint64 subticks,
        bool isBuy
    ) external returns (bytes32 orderId);

    /// @notice 撤销尚未成交的挂单
    function cancelOrder(bytes32 orderId) external;

    /// @notice 查询当前批次的预估出清价格（只读）
    function estimateClearingPrice(
        uint32 clobPairId
    ) external view returns (uint64 subticks);
}
```

12.3 统一账户模型

同一私钥同时映射到 Cosmos 地址（Bech32）和 EVM 地址（十六进制），两者派生自相同的 `ethsecp256k1` 密钥对。这意味着 MetaMask 用户可直接用现有以太坊密钥在 SVP Chain 上交易，同一子账户的持仓与保证金在原生交易与 EVM 交易间共享，无需跨链桥接或地址映射。Agent 的委托授权模型（第 7.1 节）正是构建在这一统一账户模型之上。

12.4 公平性与权限一致

EVM 兼容层的核心设计原则是**不创造任何缺口**：通过 EVM 预编译提交的订单与原生订单进入相同的待撮合队列，受相同的批量拍卖、做市商保护、包含监控约束；同样地，Agent 通过 EVM 路径发起的操作也必须穿过相同的 Policy Engine 与 Wallet 额度校验。智能合约只能通过预编译与模块交互，预编译封装了完整的验证逻辑——**不存在绕过公平性或权限的路径**。

12.5 面向 Agent 与合约的可组合性

EVM 兼容的价值不仅在于兼容钱包和工具，更在于使公平交易系统可被智能合约与 Agent 组合：开发者可部署 Solidity 合约实现网格交易、TWAP、条件触发等算法策略；可原子性组合多个交易对构建价差、跨市场对冲等多腿结构；借贷协议可通过预编译读取预言机价格计算抵押率、通过 CLOB 预编译触发清算——全部在一笔 EVM 交易中完成。任何支持 EVM JSON-RPC 的前端都可直接连接 SVP Chain。

12.6 面向 Agent 优化的接入层

在 EVM 与原生兼容之上，SVP Chain 为高频跟踪多个市场的 Agent 提供专门优化的接入设施：

面向 Agent 优化的 RPC：批量查询（单次往返解析多笔读取）、流式订阅（订单簿/成交/预言机更新即时推送而非轮询）、状态差分推送（仅交付自上一区块变化的键）。这些机制降低了原本使高频 Agent 昂贵且脆弱的读取放大，同时为节点运营者保持每客户端负载的可预测性。

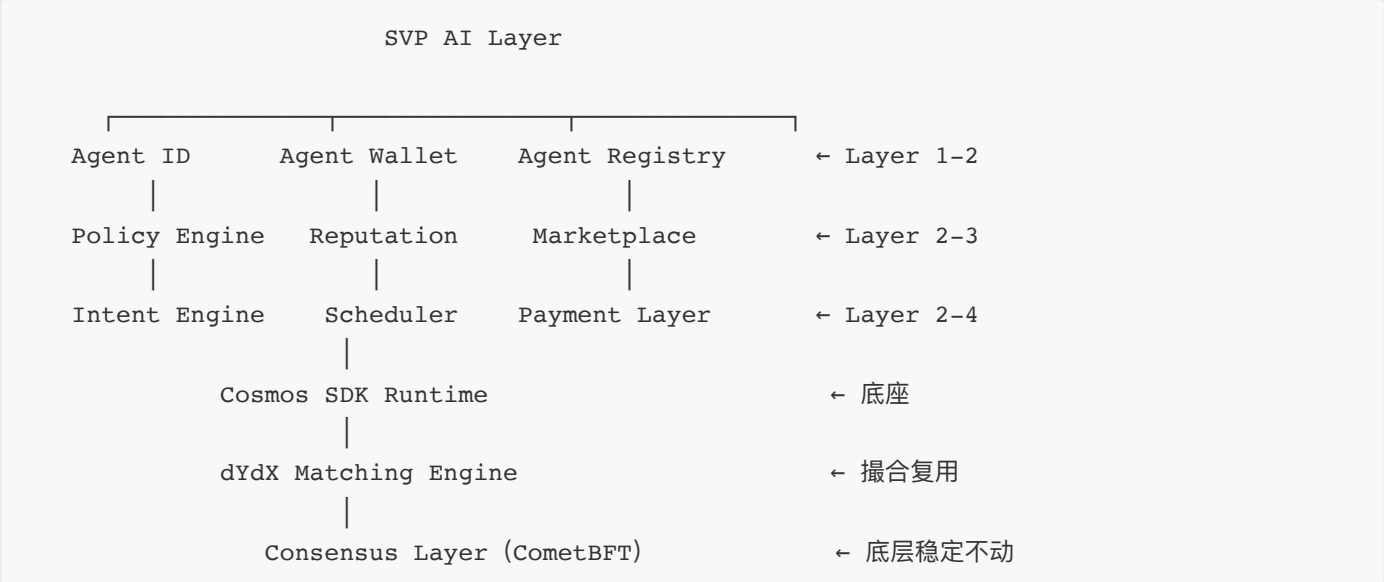
Agent SDK：将下单/撤单、查询预估出清价格、管理子账户等操作暴露为可被主流 Agent 框架（LangChain、OpenAI Function Calling、Claude Tool Use）调用的类型化工具，同时封装 EVM 预编译与原生消息。基于任一框架构建的 Agent 都能通过熟悉的抽象在链上操作，无需针对原始交易格式编写定制集成。

---

13. 系统架构与技术栈

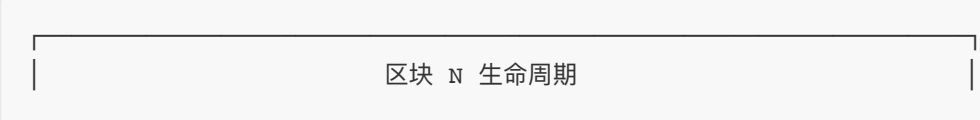
13.1 分层架构总览

Agent 四层能力落在 SVP Chain 的技术栈上：



分层原则：**越靠上越接近 Agent 语义**（身份、意图、协作、治理），**越靠下越接近链基础设施**（Runtime、撮合、共识）。所有 Agent 能力都建在应用层（Cosmos Module + AnteHandler），底层 CometBFT 共识稳定不动。

13.2 区块生命周期



```
├──────────────────────────────────────────────────────────────────┤
│  阶段一：交易验证                                                  │
│    · 接收订单/意图/Agent 操作（原生和 EVM）                        │
│    · 验证签名、Policy 权限、Wallet 额度、保证金、频率限制          │
│    · 加入待撮合队列（不执行撮合）                                  │
│                                                                  │
│  阶段二：区块提议（仅提议者执行）                                  │
│    1. 提交预言机价格更新                                           │
│    2. 对每个交易对：过滤超保护带挂单 → 合并 → 批量拍卖出清 → 生成成交 │
│    3. 打包为提议操作                                               │
│                                                                  │
│  阶段三：提议验证（所有验证者执行）                                │
│    · 验证预言机更新；独立重算出清价格并比对                        │
│    · 记录包含行为差异（本地监控）；一致则接受                      │
│                                                                  │
│  阶段四：执行（所有验证者执行）                                    │
│    · 更新预言机价格；执行所有成交；更新持仓/保证金/费用            │
│    · Agent 任务结算：按真实调用给各 Agent 记 claimable，剩余退用户；  │
│      失败则整单原子退款并罚没失败 Agent 保证金                     │
│    · 由 BeginBlocker/EndBlocker 驱动 Scheduler 定时工作流          │
└──────────────────────────────────────────────────────────────────┘
```

13.3 关键架构属性

属性	保证
Agent 一等公民	身份/钱包/权限/信誉/治理均为链原生模块，非合约层贴皮
公平性	每个批次内统一出清价格；Agent 与人类无排序优势
权限不可绕过	Policy + Wallet 校验前置于执行（AnteHandler），原生与 EVM 路径一致
委托分层	C 端默认委托平台方 Agent（向下转授权）；高级/机构可直连开发者 Agent
结算可预估可退	dry-run 估上限→预扣→按实分配退余；超限即止、失败全退 + 罚保证金
确定性	所有共识关键计算均为链上状态的纯函数
可组合性	EVM 预编译实现原子化多市场、多协议、Agent 间操作
透明性	提议者包含行为、Agent 信誉均链上可公开审计
底座稳定	所有 Agent 能力在应用层实现，底层共识不改动

14. 性能

大规模 Agent 生态意味着大量并发、高频的链上操作，性能是 Agent 优先叙事能否落地的硬约束。SVP Chain 的吞吐量通过分阶段路线图提升，假设区块时间约 1.5 秒：

阶段	关键升级	TPS（预估）
1	并行出清计算；串行成交执行	1,000
2	通过依赖图实现并行成交执行	4,000-8,000
3	乐观并发执行与冲突检测	5,000-10,000
4	SMT 替代 IAVL；状态存储与承诺分离	10,000-20,000
5	异步区块执行（共识与执行重叠）	20,000-50,000

区块内并行执行是批量拍卖架构的天然红利：每个交易对的拍卖独立出清，子账户不重叠的市场可并发执行。系统在执行成交前根据共享子账户构建冲突图，将无依赖的市场分入不同并行组，组内按确定性顺序串行。在典型的混合用户结构中预期整体加速比为 3-10 倍，且所有验证者构建相同的执行计划，结果与串行等价。

基础设施层升级（阶段 4-5）进一步突破瓶颈：稀疏默克尔树（SMT）替代 IAVL 消除重平衡开销并分离状态存储与承诺；异步执行将共识与执行解耦，使计算与 I/O 重叠。所有升级对应用层透明——无论底层如何变化，批量拍卖语义、做市商保护、Agent 能力模块均保持不变。

延迟的性质转变：在批量拍卖中，所有参与者等待相同的区块时间，延迟从「竞争性资源」变成「统一的等待窗口」。1-2 秒的绝对延迟对绝大多数 Agent 交易场景可接受，对试图靠速度取胜的高频策略则是有意的设计约束——这正是「共享同一速度」在性能层面的体现。

---

15. 安全性分析

15.1 Agent 相关攻击向量与防御

被攻破的 Agent 越权操作：Agent 只持有受限的委托签名密钥，其权限由链上 Policy + Wallet 额度严格限定。即使 Agent 或 Signer Service 被攻破，攻击者也只能在每个用户的额度约束内操作——无法触及本金与人类私钥、越界行为被 AnteHandler 拦截、用户可一键 Emergency Stop。爆炸半径被协议而非 Agent 代码限定。

恶意 Agent 骗取委托/采购资金：跨开发者调用走采购模式，经由结算合约按真实调用分配：用户确认时按 dry-run 估算预扣上限，实际多退少不补、超限即止；**任务失败则整单原子退回用户，并从失败 Agent 的保证金（Bond）中按治理可配比例罚没**。下游费用来自主 Agent 的 Service Spend Limit，受用户授权额度约束。信誉系统记录失败/清算率，保证金被罚到低于门槛的 Agent 自动下架，恶意或低质 Agent 被市场与经济惩罚双重淘汰。

信誉刷分：评分必须由真实付费调用过的用户提交；可用性数据由多个独立探测节点取中位数；核心指标（成功率/收入/失败率）直接从链上成交确定性推导，无法伪造。

多方审批被单点绕过：Agent Consensus 是叠加在 Policy + Wallet 之上的额外一层，而非后门；高风险动作先经模拟与风控打分，达投票阈值才执行，任何单个 Agent 无法独断。

15.2 交易相关攻击向量与防御

提议者注入/排除订单操纵出清价：批量拍卖下所有参与者（含提议者）以统一出清价成交，操纵利润仅限边际；所有验证者独立重算出清价，提议者无法伪造；监控系统记录排除模式交由治理处理。

预言机操纵触发错误保护：预言机使用质押加权中位数聚合，操纵成本等同于控制超三分之一质押；速率限制器约束每区块最大价格变化，攻击者须多区块持续提交虚假报告才能显著移动接受价格，成本与被检测概率指数增长；最低保留深度确保预言机压力下仍维持基线流动性。

通过 EVM 合约绕过公平性或权限：所有 EVM 订单/操作必须通过预编译，预编译内部执行与原生交易完全相同的批量拍卖、做市商保护、Policy、Wallet 校验；智能合约无法直接修改模块状态或获得特殊待遇。EVM 层不构成任何绕过路径。

15.3 确定性保证汇总

机制	输入来源	确定性
批量拍卖出清价格	待撮合队列 + 过滤后存量挂单	确定性：纯函数，所有验证者一致
做市商保护过滤	预言机价格 + 存量挂单 + 链上参数	确定性：所有输入均为共识状态
Policy / Wallet 校验	链上授权表 + 链上规则	确定性：AnteHandler 前置，所有验证者一致
包含行为监控	本地已见订单集合	非共识：不影响区块验证
Agent 信誉指标	链上成交记录	确定性：任何一方独立计算结果一致

16. 协议价值捕获

SVP Chain 的价值捕获点内嵌于协议各层，与其提供的能力一一对应。所有费用均在协议层产生并结算，参数（费率、抽佣比例、阈值）由链上治理调整，不做硬编码。

价值捕获点	所在层	机制
Agent 注册费与保证金	身份层	注册时一次性收取注册费（默认 500 SVP）并质押保证金（默认 5000 SVP），构成准入门槛与罚没抵押
Marketplace 抽佣	协作层	能力调用产生的收入在 Agent 提款时按比例（默认 5%）一次性扣取
Payment Fee	协作层	Agent 间支付按比例（默认 0.2%）计费
Intent Fee	能力层	Intent 排序与拍卖执行的费用
Reputation Service	协作层	面向机构的 Agent 信誉查询接口
策略业绩费	协作层	由协议强制执行的量化策略业绩分成
交易费	交易底座	永续与现货成交的 Maker/Taker 费率

这些捕获点在时间维度上互补：注册费与保证金对应生态准入，抽佣与调用费随生态活跃度增长，信誉与接口服务提供持续的机构级需求。全部价值流均以协议原生费用形式存在，不依赖协议外的中心化收费主体。

---

17. 竞争差异

相比市面上的「AI Chain」，SVP Chain 的差异是**系统性**的，不是单点功能：

维度	普通 AI Chain	SVP Chain（AI Native Blockchain）
AI 形态	ChatBot（问答）	Agent（自主行动）
身份	钱包地址	Agent Identity
钱包	EOA	Agent Wallet（委托不托管）
授权	Session Key	Policy Engine（链原生权限）
支付	Wallet 转账	Agent-to-Agent Payment + x402
交易	Transaction	Intent
信誉	无	链上 Reputation
调度	无	链上 Scheduler
协作	无	Agent Organization
决策	用户签名	多方 Agent Consensus
用户入口	直连某个 Agent	平台聚合门面（默认）+ 高级可直连
Agent 准入	无门槛/可刷	注册缴 SVP 保证金 + 罚没机制
对人类公平	未处理（快者通吃）	批量拍卖：Agent 与人共享同一速度

护城河的本质：**别人在合约层贴 AI，我们在协议层为 Agent 重建了一整套主体能力，并用批量拍卖保证这套能力不会退化成「最快的机器碾压所有人」。** 这十余项是一个自下而上互相依赖的体系，无法被单点复制。

---

18. 局限性与未来方向

19.1 已知局限

跨区块 MEV 未完全消除：批量拍卖消除区块内 MEV，但跨区块信息优势（如提前获知预言机更新）仍存在，做市商保护可缓解但不能完全解决。

预言机依赖：做市商保护与清算的有效性取决于预言机准确性与及时性，故障或被操纵会降低保护质量（已由优雅降级阶梯与速率限制缓解）。

包含监控非强制：依赖治理响应，无法实时自动阻止恶意行为。

批量拍卖延迟：订单需等待当前区块的批量拍卖执行，这是以速度换公平的有意权衡。

Signer Service 单点：集中托管委托签名密钥引入单点，爆炸半径被链上 Policy 限定，但仍是需要持续加固的攻击面。

Agent 信誉冷启动：新 Agent 无历史记录，需要引导期与防女巫机制配合。

19.2 未来方向

Agent 间结算市场深化：为多 Agent 协作工作流（信号/执行/风控 Agent）提供更细粒度的链上结算与利润归属，在可强制执行、透明的条款下分配价值。

Solver 网络：为 Intent 引入第三方求解器竞价，把「怎么最优成交」开放竞争。

去中心化 Signer：以门限签名/MPC 替代集中式 Signer Service，消除单点。

跨源预言机多元化：引入非交易所价格源（OTC 报价、期权隐含现货、去中心化预言机网络）作为额外输入，降低对中心化交易所的依赖。

Commit-Reveal 补充：若监控数据表明残余操纵显著，可引入轻量 commit-reveal 机制（配合保证金解决策略性不揭示问题）。

---

19. 结论

SVP Chain 的核心主张只有一句：**AI Agent 是链上的第一公民。**

绝大多数「AI Chain」只是把大模型接到合约里，链本身没有为机器改变任何东西。SVP Chain 采取相反路径，在协议层为 Agent 重建了一整套人类默认拥有、Agent 此前完全缺失的主体能力——身份、钱包、权限、意图、支付、市场、信誉、组织与联合治理。这八项能力构成一条自下而上的成长路径，让一个 Agent 依次获得身份、钱包、权限、行动力、协作力与治理权，最终成为数字世界中独立的经济主体。这套能力之所以能做到链原生而非贴在合约层，源于 SVP Chain 基于 Cosmos SDK、CometBFT 与 dYdX v4 撮合引擎的底座深度。

而让「Agent 一等公民」不至于退化为「最快的机器碾压所有人」的，是 SVP Chain 交易底座的批量拍卖设计：它从协议层消除排序敏感性，使抢先交易、三明治攻击与延迟套利在经济上失去可行性，让 Agent 与人类**共享同一速度、凭决策质量而非连接速度竞争**。EVM 兼容层则通过预编译把这套系统向以太坊生态与主流 Agent 框架开放，且不引入任何公平性或权限缺口。

区块链的演进方向正在从「Human → Wallet → Transaction」转向「Human → AI Agent → Intent → Policy → Collaboration → Settlement」。SVP Chain 的定位不是「一条支持 AI 的链」，而是：

> **一条让 AI Agent 成为链上第一公民的基础设施——Build the operating system for autonomous AI agents.**