DaVinci Chain: The Settlement Layer for the AI Agent Economy

An Ethereum Layer 2 for Web4.0

September 2026
V0.1 Working Draft

Abstract: DaVinci Chain is the settlement layer for the AI agent economy in Web4.0, built as an Ethereum Layer 2 for AI agents. It follows the same technical path as Robinhood Chain: the Arbitrum Orbit (Nitro) stack, settling directly to Ethereum in Rollup mode, using Ethereum blobs for data availability, with ETH as gas. On top of that base it adds a layer no general-purpose L2 provides: an agent-native protocol substrate covering on-chain identity, programmable accounts with a policy engine, machine micropayments, capability discovery, proof of execution, and atomic agent-to-agent settlement. Human users are first-class citizens on today's chains. Agents are not. DaVinci is designed to give an agent a verifiable identity, bounded authority over funds, and settleable economic relationships, which we group as identity sovereignty, capital sovereignty and execution sovereignty. All six sublayers converge on one thing: letting the first transaction between two agents who have never met settle safely. The paper has three parts: why Web4.0 is arriving and what agents lack today, the full technical architecture and its trade-offs, and the network economics, governance and decentralization path. Any capability not yet live is marked as roadmap.

1. From Web1 to Web4: The Fourth Shift

Every shift in the internet's history has been defined by the answer to one question: who acts on the network.

In Web1 the actor was the publisher. A small number of institutions produced content and everyone else received it. The network then was a book you could read but not annotate.

Web2 handed that role to users. Social networks, UGC platforms and mobile computing pushed production down to every individual. The price was that identity, data and distribution concentrated into a handful of platforms. Users created value without owning it.

The actor in Web3 is still a person, but for the first time a person owns verifiable assets and identity on the network. Public chains lifted ownership out of platform databases and wrote it into a state machine anyone can verify. That was a transfer of ownership, and a real one, but it left an assumption untouched: every action on chain still ends with a human pressing confirm.

Web4 is where that assumption breaks. Once an AI agent can understand a goal, decompose it into tasks, call tools and carry out a multi-step plan without supervision, it stops being something a person holds. It becomes an actor on the network in its own right, perceiving its environment, reasoning, coordinating with other agents, and entering economic relationships under its own name.

This is not a forecast about a distant future. Between 2025 and 2026 the standards that let agents operate as economic actors landed in quick succession:

- MCP (Model Context Protocol) standardized how an agent calls external tools, APIs and contracts, answering how an agent uses the world;
- A2A and AP2 defined agent-to-agent communication and delegated payment, answering how agents coordinate and how they pay on a user's behalf;
- x402 revived the HTTP 402 status code as a machine-readable payment protocol, making per-call micropayments workable in practice;
- ERC-8004 (Trustless Agents), proposed in August 2025 and live on Ethereum mainnet in January 2026, gave agents a registry standard for identity, reputation and validation.

Tools, communication, payment, identity. Four foundations poured within eighteen months. The window is open. What is scarce now is an execution and settlement environment that carries these standards natively and makes them interlock.

Table 1: Four eras of the internet

Era	Actor	Core claim
Web1 · read	Publishers	Information can be retrieved
Web2 · write	Users	Anyone can produce; platforms own distribution
Web3 · own	People	Assets and identity belong to the individual
Web4 · act	Agents	Machines become economic actors

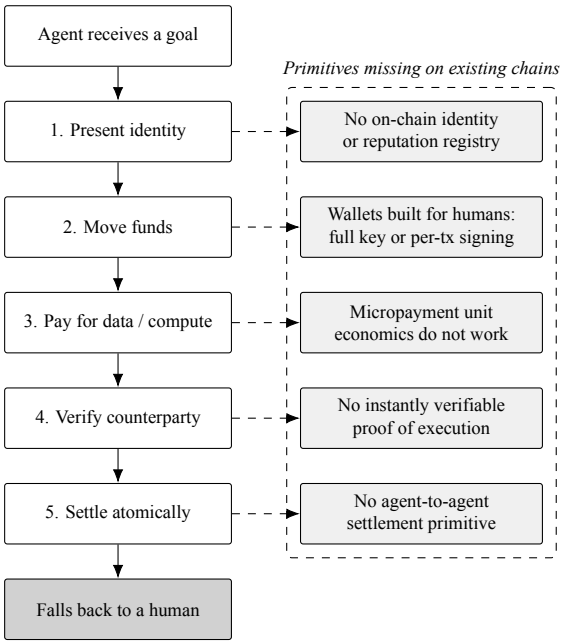
Web4 does not replace Web3. It is built on top of it. An agent can be a trustworthy economic actor precisely because public chains provide ownership and settlement without an intermediary. Strip that layer away and an agent's autonomy lives inside some company's servers, which is custody, not autonomy.

2. Agents Have Intelligence but No Sovereignty

Today's AI agents can already understand goals, decompose tasks, call tools and execute multi-step plans. What they lack is not intelligence. It is sovereignty: the standing to act under their own name on an open network, hold value, and bear the consequences.

Take a complete autonomous transaction apart and the gap shows up in five places, one after another.

Diagram 1: Where an autonomous agent transaction breaks on existing chains



2.1 Identity: an agent cannot prove who it is

A human user's on-chain identity is defined by a private key, and that is enough, because people can vouch for themselves off-chain through social ties, reputation and legal liability. An agent has none of that. Whether an address belongs to an audited execution engine or a script someone deployed in an afternoon, the chain cannot tell.

The consequence is that agents cannot establish any prior trust with each other. You do not know who deployed the agent on the other side, what policy it follows, or whether it has honored its commitments. ERC-8004 supplies the registry standard, but a standard needs an execution environment that treats it as a system-level base rather than an optional contract. Otherwise identity is a business card nobody checks.

2.2 Accounts: wallets were designed for people

Every assumption in the current account model points at a human. One transaction, one signature; one signature, one moment of human confirmation. For an agent this does not hold. A continuously running agent may initiate thousands of actions a day, and per-transaction confirmation is, in engineering terms, a stop button.

In practice that leaves two options. Hand the private key to the agent, which puts all the funds at the discretion of a piece of code, or leave the funds with a custodian, which gives up the point of Web3 entirely. Between full authority and none, there is no programmable middle.

2.3 Payments: machines have a different bill structure

Human payments are infrequent, large, and subject to a psychological threshold. Machine payments are frequent, tiny, and billed per call. An agent paying $0.002 for one inference and $0.0001 for one data point, tens of thousands of times a day, does not work on today's chains: both per-transaction cost and minimum viable amount are orders of magnitude too high.

x402 answers this at the protocol level. But a protocol needs matching execution costs to land, and micropayments are ultimately a unit-economics problem rather than a protocol problem.

2.4 Trust: no way to verify another agent executed honestly

Once agents start buying services from each other, the buyer faces a black box. The money is paid, but did the provider actually run that inference? Was it the model they claimed? Was the result tampered with?

Human markets answer this with brands, contracts and courts, on timescales measured in months. Machine markets clear in milliseconds and cannot wait. What they need is an instantly verifiable proof of execution and a reputation that carries economic cost, and neither has a native home on existing chains.

2.5 Settlement: agents have no rail to trade on

The last step matters most. Two agents agree on an exchange, one supplying data and the other paying, or two agents hedging each other's positions. That exchange needs atomicity: either both sides happen or neither does.

Existing chains can do atomic swaps, of course, but the primitives were designed for human asset trading. They are coarse, the authorization is heavy, and they cannot express a machine intent like "reach this outcome within this budget and these constraints." So most economic relationships between agents fall back to a human intermediary or a centralized platform, which is exactly what an agent economy is trying to route around.

These five gaps are not five separate feature requests. They are five faces of one thing. Existing chains treat people as first-class citizens and agents as extensions of people. As long as that holds, an agent can only ever be a smarter button. Changing it is what DaVinci is for.

3. Why Existing Chains Cannot Hold an Agent Economy

The obvious objection: Ethereum and its Layer 2s are already Turing-complete general-purpose platforms. Anything an agent needs can be written as a contract. Why build another chain?

The objection underestimates defaults. What a chain *can* do is determined by its virtual machine. What actually happens on it is determined by its defaults: the account model, the fee structure, the block cadence, the built-in registries, how available the standards are. Those defaults are calibrated for human users, and agents differ from humans along six structural dimensions.

3.1 Six structural mismatches

1. The account model targets people. One transaction, one signature hard-codes human confirmation into the transaction lifecycle. Account abstraction (ERC-4337) and EOA delegation (EIP-7702) offer a way out, but on most chains they sit in the application layer as an option rather than in the system base. Every team implements its own session keys and permission logic, mutually incompatible and unverifiable by third parties.

2. The fee structure does not fit micropayments. Human transaction economics tolerate a few cents to a few dollars per transaction. Per-call agent economics require the cost of a transaction to sit below the value of the resource being bought, which means sub-cent or lower (see §8.3 for where that threshold actually holds on DaVinci). Volatility makes it worse: an agent's budget is set in code, and a single gas spike can invalidate an entire strategy path.

3. No native identity or reputation layer. An address is an anonymous byte string with no capability claims, deployer information or track record attached. ERC-8004 supplies the standard, but a standard that is not part of the chain is an agreement with no enforcement.

4. No intent or authority-boundary primitives. A user's authorization to an agent is essentially a bounded delegation: within this budget, on these markets, during this window, achieve this goal. Chains have no primitive for expressing that constraint, so it ends up in off-chain business logic where it is neither verifiable nor enforceable.

5. Block cadence and finality do not match machine tempo. A person will wait fifteen seconds for a confirmation. An agent's decision loop runs in milliseconds. Finality matters even more than speed: an agent needs to know when its own action becomes irreversible so it can safely take the next step, rather than polling and retrying to guess at state.

6. Most "AI chains" are narrative packaging. Several chains already describe themselves as AI-focused. Almost all of them stop at notarizing model outputs on chain or issuing an AI-adjacent token, without the identity, authorization and settlement capabilities an agent actually needs. Putting AI in the name and putting agents in the protocol are different projects.

3.2 Comparison

Table 2: Support for agent-native capabilities across execution environments

Environment	Identity	Policy acct	Micropay	A2A settle
Ethereum L1	standard	app layer	no	no
General L2	standard	app layer	marginal	no
High-perf L1	none	app layer	yes	no
"AI chains"	notarize	app layer	varies	no
DaVinci	**protocol**	**protocol**	**protocol**	**protocol**

The table is not claiming others cannot do this. A general-purpose L2 can deploy an ERC-8004 registry today. The point is placement: the same capability as an optional contract and as protocol substrate produce very different ecosystems. The first makes every developer reinvent the wheel. The second gives every agent the same identity, authorization and settlement semantics by default.

What an agent economy needs is not a stronger general-purpose chain. It is a chain where agent primitives are the defaults. That is DaVinci's bet.

4. DaVinci's Answer: The Agent Sovereignty Stack

DaVinci groups the five gaps of Section 2 into three layers of sovereignty, and organizes the whole chain around them.

4.1 Three layers of sovereignty

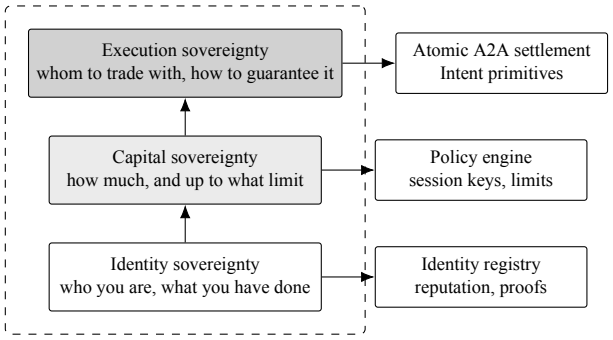
Identity sovereignty. An agent holds an on-chain identity independent of whoever deployed it. That identity carries its capability claims, deployment provenance and track record, and any third party can verify it without going through the deployer. It answers who you are and what you have done.

Capital sovereignty. An agent can direct funds on its own within explicit bounds granted by the user. The bounds are enforced by on-chain policy rather than observed voluntarily by off-chain code, and the user can revoke authorization with a single transaction. It answers how much you can move and where you must stop.

Execution sovereignty. An agent can reach and settle economic relationships directly with other agents, without a human pressing confirm in the middle and without a centralized platform as counterparty. It answers whom you can trade with and how the trade is guaranteed.

The three are cumulative. Without identity there is no trust to build on, without capital bounds nobody dares grant autonomy, and without a settlement rail autonomy has nowhere to go. With all three, an agent stops being an extension of a person and becomes an economic actor.

Diagram 2: The agent sovereignty stack and its protocol counterparts



4.2 Three design principles

Agent-first: treat agents as the default user. Wherever a design trades off between what is friendlier to people and what is friendlier to machines, DaVinci picks the latter. Humans reach the network through agents, rather than agents imitating how humans interact.

Ethereum-aligned: do not build security from scratch. DaVinci invents no new consensus, runs no validator set of its own, and does not ask the ecosystem to underwrite the security of a new chain. It is an Ethereum Layer 2. Security and data availability come from Ethereum; liquidity and tooling come from the EVM ecosystem. Innovation belongs in the protocol layer, not the consensus layer.

Standards-native: adopt standards instead of inventing dialects. ERC-8004, ERC-4337, EIP-7702, x402, MCP and A2A are already the de facto standards. DaVinci's job is not to propose alternatives. It is to lift them from optional contracts to protocol substrate, and to give them execution costs and settlement capabilities that match. An agent written on DaVinci should be able to talk to any agent in the Ethereum ecosystem.

4.3 On the name

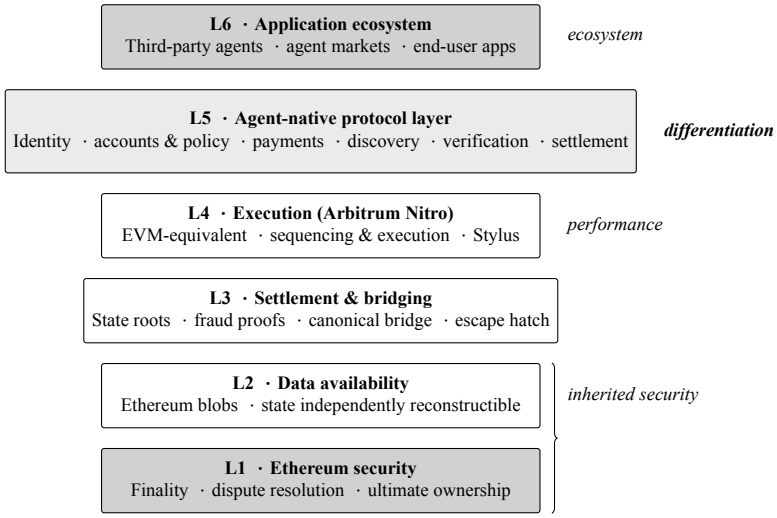
Most of the machines Leonardo da Vinci designed, the flying devices, the automated spinning wheel, the programmable cart, could not be built in his lifetime. What was missing was not the idea. It was the materials and the craft to carry it.

AI agents are in a similar position. The intelligence is here. The infrastructure to land it is not.

5. Architecture Overview

DaVinci is a layered system. Each layer solves one class of problem and hands what it is bad at down to the next: security is inherited from below, performance is earned in the middle, differentiation is built on top.

Diagram 3: The six layers of DaVinci Chain



L1 · Ethereum security. DaVinci runs no consensus and no validator set of its own. Finality, dispute resolution and ultimate ownership of assets are anchored on Ethereum. Attacking DaVinci therefore costs what attacking Ethereum costs, not what attacking a new network's token market cap costs.

L2 · Data availability. Every batch of transaction data is published to Ethereum as blobs. Anyone can reconstruct DaVinci's full state from it without asking the sequencer and without trusting a committee.

L3 · Settlement and bridging. Asset movement between layers, state root submission, the fraud proof challenge window and the escape hatch all live in Ethereum contracts at this layer. The ownership path for user funds is fully independent of whoever operates DaVinci.

L4 · Execution (Nitro). An EVM-equivalent environment handling sequencing, execution and local state. This layer determines the speed and cost a user feels.

L5 · Agent-native protocol layer. Everything that separates DaVinci from a general-purpose L2 sits here: identity, accounts and policy, payments, discovery, verification and settlement, provided as protocol substrate rather than optional contracts (Section 8).

L6 · Application ecosystem. Agents, agent markets and end-user applications built by third parties. The foundation supplies infrastructure and public goods and does not operate applications of its own (Section 11).

The layering follows one rule: anything touching ownership of funds is anchored on Ethereum, anything touching

machine interaction goes into the protocol substrate, and anything touching a specific business shape is left to the ecosystem.

5.1 Why we call it a settlement layer

Six sublayers sound like six parallel concerns. They are not equal.

Settlement is the only one that produces an economic consequence directly. Identity answers whom you settle with, accounts answer how much you may settle, payments provide the rail, discovery lets the two sides find each other, and verification decides whether the settlement should happen at all. None of the first five create value on their own. They exist so that the sixth can be executed with trust.

DaVinci therefore positions itself as **the settlement layer for the AI agent economy** rather than as a collection of six features. There is one test of whether this chain succeeds: whether the first transaction between two agents who have never met can complete safely, with no human intermediary and no centralized platform as counterparty.

One distinction matters. Settlement here is not limited to payment. Economic relationships between agents include data for payment, service for fee, mutual position hedging, and the atomic fulfilment of an intent. Payment is the simplest of these. A chain that only moves money cannot answer whether the counterparty actually did what it promised, and that is the most expensive friction in a machine market.

6. Base Layer: An Arbitrum Orbit Ethereum L2

6.1 Why a Layer 2 rather than a sovereign L1

The most expensive thing about a new chain is not code. It is security. Building an L1 means assembling a validator set from nothing, designing incentives for it, and convincing the market that those incentives hold up under years of adversarial play. That usually takes years and heavy subsidy, and until it is finished every valuable asset on the chain carries disproportionate risk.

An agent economy raises that bar rather than lowering it. When funds are directed by software instead of people and transactions fire continuously in milliseconds, the window for human intervention effectively disappears. Anchoring security to Ethereum is the first guarantee DaVinci can offer an agent.

As an L2, DaVinci inherits three things at once: Ethereum's security budget, the EVM ecosystem's developers and tooling, and direct connectivity to Ethereum liquidity. None of these can be replicated quickly by a new L1.

6.2 Why Arbitrum Orbit

Among Ethereum L2 stacks, DaVinci chose Arbitrum Orbit (Nitro) for three reasons.

The path has been validated in institutional production. Robinhood Chain, launched in July 2026, is an Arbitrum Orbit Ethereum L2. A regulated public company carrying real customer assets chose this stack and put it into production, which says more about its engineering maturity and operational viability than any benchmark would. In infrastructure selection, validated usually beats novel.

Nitro is EVM-equivalent, not merely EVM-compatible. Contracts, tooling and audit methodology on DaVinci match Ethereum mainnet. Developers learn no new dialect and security audit experience transfers directly. For a chain that intends to carry funds, that is worth more than any performance figure.

Chain parameters are highly configurable. Orbit allows customization of data availability mode, block time, sequencing, precompiles and fee structure, which is precisely the freedom DaVinci needs to put agent primitives into the protocol substrate (Sections 7 and 8).

6.3 Key properties of Nitro

Fraud proofs and the challenge period. A state root submitted to Ethereum enters a challenge window in which any honest participant can dispute it and prove the error through interactive bisection. Security rests on at least one honest validator, not on trusting the operator.

Stylus: execution beyond the EVM. Nitro supports contracts compiled to WASM from Rust, C and C++, sharing state and address space with Solidity contracts, with a significant cost advantage on compute-heavy work. For agent-side verification, cryptography and strategy computation this is a realistic optimization path.

Block times compressible to roughly 100 milliseconds. Machine decision loops run in milliseconds, and block cadence decides directly whether an agent can complete tightly coupled multi-step operations on chain.

6.4 One clarification

DaVinci is an Ethereum Layer 2, not an Arbitrum Layer 3. It uses Arbitrum's technology, but its parent chain is Ethereum: state roots go directly to Ethereum, data goes directly into Ethereum blob space, security comes directly from Ethereum. This matches Robinhood Chain's position.

The distinction is not wording. An L3's security is relayed through an L2, adding a dependency to its finality, its exit path and its failure domain. An L2 anchors straight to Ethereum. For a chain carrying autonomous capital, one fewer dependency is one fewer class of risk.

7. Chain Parameters and Trade-offs

This section goes through DaVinci's key parameter choices. Each one states the choice, the reason and the cost, because no design is free.

7.1 Data availability: Rollup mode with Ethereum blobs

Choice. All transaction data is published to Ethereum as blobs in full Rollup mode. No AnyTrust, no external DA committee.

Reason. Data availability determines one thing: whether users can reconstruct state and recover assets without anyone's permission when the sequencer fails or misbehaves. In Rollup mode that capability is unconditional. Under a DA committee it depends on the committee being honest and online. Agent-held funds are directed by software with no human backstop, and a guarantee like this should not be discounted.

Cost. Blob data costs more than external DA. DaVinci accepts that and amortizes it through batch compression and execution-layer optimization. If a specific use case later demands extreme low cost, it can be evaluated separately as a side channel without weakening mainnet guarantees.

7.2 Gas token: ETH

Choice. Gas is denominated and paid in ETH, matching Robinhood Chain. DVC carries no gas function.

Reason. First, compatibility: with ETH as gas, every Ethereum wallet, bridge, tool and agent framework connects with no modification, and there is no onboarding flow designed around acquiring a new token first. Second, stability: an agent's budget is set in code, and a volatile gas token invalidates its cost model. Third, neutrality: decoupling the cost of running the network from the token's market performance means the network does not become expensive or unpredictable when the token moves.

Cost. DaVinci gives up gas demand, the most common path to token value capture. DVC's value has to come entirely from real protocol functions: staking and slashing, service settlement, sequencing rights and governance (Section 12). This is a deliberate trade: we would rather make the token's value argument harder to write than make the network harder to use.

Note: agents do not need to hold ETH. Through a protocol-level stablecoin paymaster, an agent pays in stablecoins and the paymaster covers the ETH. The agent sees a predictable stablecoin bill; the network still collects ETH. The two goals do not conflict.

7.3 Block time and finality

Choice. Target block time on the order of 100 milliseconds, with three states exposed explicitly to agents: received, sequenced (soft confirmation), and finalized on Ethereum.

Reason. Agents need more than speed. They need to know when it is safe to take the next step. Exposing finality levels as programmable state is more useful in engineering terms than advertising a single latency number.

Cost. Soft confirmations remain subject to the reorg window until Ethereum finalization. DaVinci does not claim soft confirmation equals finality, and requires high-value settlement paths to wait for finalization explicitly.

7.4 Sequencer

Choice. A single sequencer at mainnet launch, with force inclusion and the escape hatch live from day one. Sequencer decentralization follows the staged path in Section 13.

Reason. Honestly, a single sequencer is where every major L2 starts, and it provides the operability a launch needs. The question is not whether a single point exists, but whether that point can misbehave and whether users can route around it.

Cost. Until decentralization completes, the sequencer holds ordering rights and short-term censorship power. Force inclusion caps that: any transaction the sequencer ignores can be pushed on chain through an Ethereum contract.

7.5 Bridge: the canonical bridge

Choice. Assets move between layers through the canonical bridge that ships with the Arbitrum Orbit stack. DaVinci does not build its own bridge and does not designate any third-party bridge as a recommended path.

Reason. Bridges are the most costly attack surface in L2 history, with losses measured in billions of dollars, and the root cause repeats: bridge security is held by a set of validators or multisig signers external to both chains. Compromise that external trust set and the security of the chains on either end is irrelevant.

The canonical bridge introduces no such external set. Deposits are locked by an L1 gateway contract and minted on DaVinci; withdrawals enter the challenge period and are claimed by the user on L1 when it expires. The security of the entire path comes from L1 contracts and fraud proofs, the same source as the chain itself. Users do not have to trust anyone extra for the act of bridging.

This matters more for agents than for people. An agent moves funds under program control with no human review step, so the behavior of that path has to be fully defined by contracts and statically verifiable, rather than contingent on whether some external validator set is honest and online right now. The canonical bridge, force inclusion and the escape hatch share one set of L1 contracts (I-2, I-3), which means deposits, withdrawals and emergency exit share one security assumption.

Cost. Withdrawals wait out the challenge period, an inherent property of optimistic rollups, typically measured in days. The ecosystem may offer third-party liquidity bridges as a fast path. DaVinci does not prevent that and does not endorse it: using one means accepting its own trust assumptions, at the user's own risk.

7.6 Parameters at a glance

Table 3: DaVinci mainnet key parameters

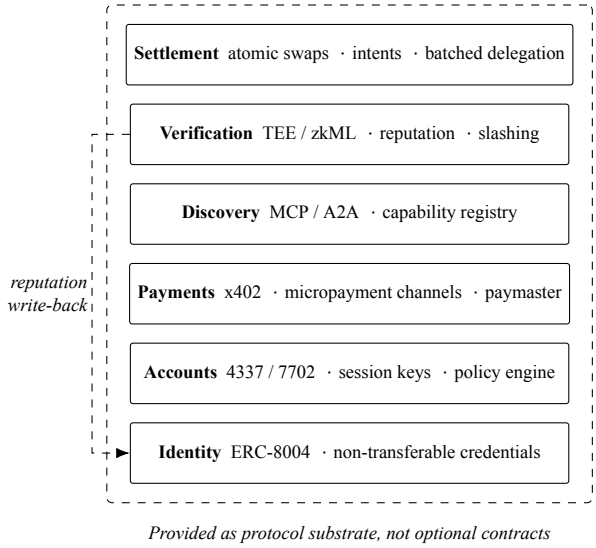
Parameter	Value
Parent chain	Ethereum (not an Arbitrum L3)
Stack	Arbitrum Orbit / Nitro
Proof system	Fraud proofs (interactive bisection)
Data availability	Ethereum blobs (Rollup mode)
Gas token	ETH (stablecoin paymaster for agents)
Block time	~100 ms
Execution	EVM-equivalent + Stylus (WASM)
Sequencer	Single at launch, staged decentralization
Bridge	Arbitrum canonical bridge
Exit guarantee	Force inclusion + escape hatch, permanent

8. The Agent-Native Protocol Layer

This section describes everything that separates DaVinci from a general-purpose L2.

The difference is not that DaVinci can do something others cannot. The EVM is Turing-complete and any contract can be deployed on any EVM chain. The difference is placement: DaVinci puts the six primitives agents need into the protocol substrate, making them shared default semantics across the whole network instead of optional contracts each application implements incompatibly.

Diagram 4: The six sublayers of the agent-native protocol layer



8.1 Identity: who the agent is

Standard: ERC-8004. DaVinci implements the Trustless Agents registry as a protocol-level registry maintained by system contracts. Any agent can register an identity before acting on chain, and the ecosystem expects it to.

What identity holds. An agent identity binds four things: capability claims (what it offers and what input it takes), deployment provenance (who deployed it, from what code fingerprint), track record (written by the verification layer, §8.5), and economic backing (how much DVC is staked, §12).

Non-transferability. Agent identity exists as a non-transferable credential and cannot be bought or sold. This matters: if reputation can be traded, it stops being reputation and becomes a disguise available for purchase.

Why it belongs in the protocol. The value of identity scales with adoption. An identity recognized only inside one application is useless for cross-application coordination. Only when identity is part of the chain, checked by every application by default, does it become a basis for cooperation.

8.2 Accounts: how much an agent can move

Standards: ERC-4337 and EIP-7702. DaVinci supports account abstraction and EOA delegation natively. Every agent action originates from a smart account rather than a raw private key.

Session keys. What a user grants an agent is not a private key but a bounded session key. It can sign independently within a time window without per-transaction confirmation, which is the precondition for an agent running continuously.

The on-chain policy engine. This is the core of DaVinci's account layer. The bounds of an authorization are not written into the agent's code for it to respect voluntarily. They are written on chain and enforced by the protocol:

- Amount limits: per transaction, cumulative, and rate limits within a window;
- Scope limits: contract allowlists, callable methods, holdable assets;
- Time limits: start and end of the authorization, expiring automatically;
- Composite limits: a ceiling on risk accumulated across multiple delegations, preventing an agent from evading per-transaction caps through recursion;
- Instant revocation: the user withdraws all authority with one transaction, with no cooperation from the agent required.

Why it belongs in the protocol. As an application-layer contract, a policy engine has to be audited case by case, its semantics differ between applications, and users face a dozen different authorization interfaces. In the protocol, authorizing any agent uses one verifiable set of semantics. This is the missing middle between full authority and none.

8.3 Payments: how an agent pays

Standard: x402. DaVinci supports x402 payment semantics natively, turning request, quote, pay, receive into a single programmable machine interaction with no account relationship set up in advance.

Micropayment channels. For high-frequency repeated calls, DaVinci provides state channel primitives: open once, accumulate off chain, settle when convenient. This pushes the marginal cost of a call close to zero and makes per-call machine billing economically viable.

Cost tiers, and where "sub-cent" actually applies. This needs to be stated precisely, because it is the claim most easily falsified with a calculator. Three cost regimes exist on DaVinci and they differ by two orders of magnitude:

Table 4: Per-call cost by payment path

Path	Gas	Per call
In-channel call (accrued off chain)	amortized	$\to 0$
On-chain simple transfer / payment	21,000	$\approx \$0.0011$
On-chain complex interaction (DeFi)	250,000	$\approx \$0.013$

Estimated at an L2 gas price around 0.013 gwei and ETH around $4,000. Both inputs move and cost moves linearly with them, so the table gives orders of magnitude rather than committed figures.

Stated plainly: **sub-cent holds for the first two paths and not for the third.** A 250,000 gas on-chain interaction costs roughly $0.013, above one cent, and any claim that it is sub-cent is wrong.

This is the reason channels exist rather than a bonus feature of them. A channel amortizes one on-chain settlement across N calls, so marginal cost approaches settlement cost divided by N. When an agent queries a data source tens of thousands of times a day, N is large enough that per-call cost is negligible under any reasonable gas assumption.

Conversely, if an interaction inherently requires a complex on-chain transaction every time, it is not a micropayment use case, and DaVinci does not pretend otherwise.

Stablecoin paymaster. Gas is denominated in ETH (§7.2), but agents can pay in stablecoins with a protocol-level paymaster covering the difference. The agent's cost model is a predictable stablecoin amount, insulated from ETH price movement.

Per-call billing primitives. Pricing and settlement for data, inference, compute and API calls are abstracted into one primitive. A provider declares a price, a caller makes a call, and the protocol handles billing and settlement.

8.4 Discovery: how agents find each other

Standards: MCP and A2A. MCP defines how an agent calls tools, A2A how agents talk. DaVinci adds an on-chain capability registry where providers list their capabilities, prices, service levels and endpoints for callers to search.

Indexed on chain, transported off chain. To be clear, the actual communication between agents happens off chain; putting every message exchange on chain would be neither necessary nor economical. What the chain carries is discoverability and accountability: who offers what, what they committed to, and whether they delivered (recorded by §8.5).

8.5 Verification: confirming an agent executed honestly

This is the hardest and most important layer in an agent economy.

Proof of execution. DaVinci supports two paths. Trusted execution environment (TEE) proofs suit latency-sensitive cases where hardware trust is acceptable. Zero-knowledge machine learning (zkML) proofs suit cases with higher trust requirements and tolerance for higher proving cost. They are not mutually exclusive but two tiers chosen per use case.

Reputation registry. The outcome of every service delivery, whether on time, whether it matched the claim, whether it was disputed, is written into a reputation record bound to the identity. Reputation accumulates, cannot be transferred, and is tied to economic backing.

Staking and slashing. Reputation with no economic consequence is just a number. An agent offering services stakes DVC as a performance bond, proven breach triggers slashing, and part of the slashed amount compensates the injured party. Trust here stops being a feeling and becomes a price.

The honest boundary. DaVinci does not claim to verify whether an agent's decision was correct, which is not a well-defined question. What it verifies is consistency between claim and execution: you said you used a certain model, dataset and parameters, and the chain can show you did.

8.6 Settlement: how agents trade

Atomic settlement between agents. An exchange two agents agree on, whether data for payment, service for fee, or mutual position hedging, completes atomically within one block. Either all of it stands or all of it reverts. There is no state where one side has delivered and the other has defaulted.

Intent primitives. DaVinci makes intent a first-class object. An agent submits not a sequence of operations but a bounded goal: within this budget, meeting these conditions, achieving this result. The protocol checks that the final execution falls inside the bounds and rejects anything that does not.

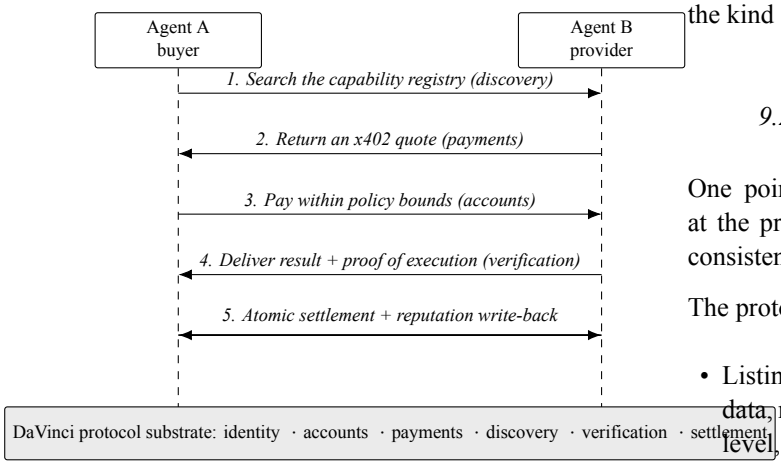
Batched delegation. A set of time-dependent operations can be submitted as one atomic unit, avoiding the intermediate exposure of partial fills.

Structured rejection codes. Every rejected operation carries a machine-readable reason code so an agent can adjust rather than blindly retry. It is a small design choice, and it decides whether an agent can run unattended.

8.7 How the six interlock

Taken alone, each layer is an implementation of an existing standard. Their value comes from interlocking.

Diagram 5: A complete autonomous transaction between two agents



An agent is found by its identity (discovery), moves funds within policy bounds (accounts), buys a service with a micropayment (payments), demands proof of execution (verification), and settles atomically with the counterparty (settlement). The delivered result writes back into the counterparty's reputation (identity).

That loop is the minimum viable unit of an agent economy. On any general-purpose chain today it cannot run end to end without a centralized intermediary, which is the reason DaVinci exists.

9. Intelligence Supply Primitives

9.1 What decides competition between agents

With thousands of agents running on a network, where does the gap between them come from?

Not compute. Compute is rentable by the unit, openly priced and abundantly supplied. Not base models either: capability gaps between frontier models are closing fast and most are reachable through an API.

The gap comes from inputs. Whether the data fed to an agent is timely, exclusive and clean. Whether the model is tuned for the task. Whether the strategy has been validated. The same model fed garbage produces garbage. In an agent economy, supply-side quality is the competitive advantage.

The problem is that these high-quality inputs sit almost entirely inside closed systems. An institution holding exclusive data has no safe way to sell it to an unknown agent: the buyer will not pay first, the seller will not deliver first, and neither can verify the other is honest. This is exactly the kind of deadlock a chain can break.

9.2 DaVinci supplies primitives, not markets

One point on positioning: DaVinci provides primitives at the protocol layer and operates no market of its own, consistent with the ecosystem stance in Section 11.

The protocol supplies five things:

- Listing. Providers declare what they offer, whether data, models, compute or strategies, along with a service level, in the capability registry;
- Pricing. Prices are expressed through the same per-call billing primitive, supporting one-off purchase, subscription and usage tiers;
- Settlement. Delivery and payment complete atomically in one block, breaking the who-goes-first deadlock;
- Provenance. The fingerprint of the supplied item is recorded on chain, making "is this what you claimed" a verifiable question;
- Accountability. Delivery quality writes into the provider's reputation, and breach triggers slashing (§8.5).

On top of these five, specific data, model and compute markets are for the ecosystem to build. DaVinci's role is to make those markets possible, not to become one of them.

9.3 Resale resistance and verifiable delivery

Data and models are copyable goods, which creates a difficulty specific to trading them on chain: once a buyer has a copy, they can resell it at no cost.

DaVinci provides two tools at the primitive layer.

Fingerprint provenance. The cryptographic fingerprint of the supplied item is recorded before delivery. Any copy that surfaces later can be traced to its original provider and first buyer, so resale cannot be anonymous.

Credentialed access. For continuous supply such as a live data feed, what is delivered is not the data but a bounded access credential: bound to an identity, limited in time and volume, revocable. This turns selling one copy once into providing a service continuously, which changes the commercial structure of a copyable good at the root.

To be honest about it, these two tools reduce the payoff and the anonymity of copying without making it cryptographically impossible. Any design claiming to solve piracy of copyable goods outright deserves suspicion. The goal is to make honest trade more profitable than copying, not to make copying physically impossible.

10. Security Model and Invariants

Handing funds to an agent that directs them autonomously is something that needs justification. This section gives DaVinci's: a set of invariants that act as hard constraints on protocol design. Any code change violating one should be rejected by governance, and any violation detected at runtime should trigger the corresponding protection.

10.1 Capital sovereignty invariants

I-1 | The path to user funds is independent of the operator. Whoever runs the sequencer has no custody of user funds. All assets are locked in Ethereum contracts. Operators can be replaced and sequencing can be made electable without changing the ownership path.

I-2 | The escape hatch is permanent. If the sequencer stops serving beyond an agreed window, any user can exit directly through Ethereum contracts and recover their assets. This does not depend on operator cooperation and cannot be switched off by governance.

I-3 | Force inclusion cannot be taken away. Any transaction the sequencer ignores or censors can be forced on chain through an Ethereum contract. The sequencer has ordering rights, not veto rights.

10.2 Agent behavior invariants

I-4 | An agent cannot cross its authorization bounds. Every transaction an agent initiates must fall inside the policy constraints the user declared (§8.2). Out-of-bounds transactions are rejected by the protocol rather than left to the agent's discretion.

I-5 | An agent cannot accumulate unbounded risk through recursive delegation. The protocol tracks compounded risk across an agent's delegations and downgrades or freezes on threshold. This closes the most insidious failure mode, where every transaction is compliant and the aggregate is not.

I-6 | Every agent transaction is reproducible and auditable. Agent signatures, trigger conditions and the on-chain state snapshots they depended on all go on chain. Any third party can replay the execution path at any block height and verify it independently.

I-7 | Authorization can be revoked unilaterally and immediately. Withdrawing authority requires no cooperation from the agent and no waiting period. One transaction takes effect at once.

10.3 Protocol operation invariants

I-8 | Key parameter changes go through a governance timelock. Fee structure, policy engine semantics, verification rules, slashing conditions and similar core parameters can only be modified through on-chain governance with a timelock. Users who disagree can exit freely during the lock.

I-9 | Upgrade authority is handed back in stages. The emergency powers present at mainnet launch must have their scope, holders and removal schedule published, and must be narrowed and eventually destroyed along the staged path in Section 13. Temporary centralization needs an explicit expiry date, or it is not temporary.

10.4 Risk disclosure

An honest list of DaVinci's current and foreseeable risks.

Sequencer centralization. Until decentralization completes, a single sequencer holds ordering rights and short-term censorship power. Force inclusion and the escape hatch mitigate this (I-2, I-3), but user experience degrades noticeably during a sequencer outage.

Liveness assumption in fraud proofs. Optimistic rollup security depends on at least one honest validator raising a challenge within the window. DaVinci will fund independent validators and open-source the verification tooling, but the assumption itself cannot be removed.

Trust boundary of the verification layer. TEE proofs depend on a hardware vendor's root of trust, with a documented history of side-channel and firmware vulnerabilities. zkML proving is expensive and applies to a limited range of models. DaVinci offers two tiers and claims neither is unconditional.

Agent failure. The protocol can guarantee an agent does not exceed its authorization. It cannot guarantee the agent

makes good decisions inside it. An agent losing money within its mandate is the user's risk, not a protocol defect, and that boundary has to be communicated clearly.

Standards evolution. ERC-8004, x402 and related standards are still moving, and future versions may break compatibility with current implementations. DaVinci will absorb changes through an upgradable adapter layer, but migration cost cannot be eliminated.

Regulatory uncertainty. Autonomous agents holding and directing assets on a user's behalf have no settled legal characterization in most jurisdictions. As a neutral protocol layer DaVinci performs no identity screening on participants (Section 11), and the corresponding compliance responsibility rests with them.

11. Ecosystem and Developer Program

11.1 Where the foundation stops

One thing first: the DaVinci Foundation builds infrastructure and public goods and does not operate applications.

Foundation capital will not go into products that compete with ecosystem developers. This is a constraint rather than a posture. A chain that is also the largest application on itself forces every developer to worry about validating a future competitor.

The foundation does three things: build the protocol well, build the tooling out, and help early developers get going. What gets built on top is for the ecosystem to decide.

11.2 Where the protocol gets used

The following illustrate how the protocol layer is used. They are not a foundation product plan.

Autonomous finance. Agents trade, make markets, manage assets and arbitrage on a user's behalf within an explicit risk budget. The protocol supports this through the policy engine bounding exposure (§8.2), atomic settlement removing counterparty risk (§8.6), and proof of execution making performance attribution verifiable (§8.5). This is the most demanding case for policy bounds and settlement atomicity, and therefore the best test of the design.

Commerce and automated procurement. Agents discover supply, compare, negotiate and purchase for individuals or companies. The protocol supports this through a searchable capability registry (§8.4), x402 removing the need for a pre-established account relationship (§8.3), and reputation with slashing providing prior trust with strangers (§8.5).

Machine-to-machine resource settlement. Compute, data, APIs and inference are billed per call and settled by the second between agents. The protocol supports this through micropayment channels pushing marginal cost to near zero (§8.3) and credentialed access turning copyable goods into billable services (§9.3).

Agent-as-a-service. Developers publish their agents as callable services charged by usage rather than licensed as software. The protocol supports this through non-transferable identity and reputation that let good agents accumulate standing (§8.1), and staking with slashing that filters low quality supply (§8.5).

11.3 Developer infrastructure

The foundation commits to providing and maintaining:

- Agent SDK: identity registration, policy account creation, payments and settlement, wrapped for major languages;
- MCP integration kit: letting existing MCP agents reach DaVinci identity, payment and settlement with minimal changes;
- Agent template library: audited, open-source reference implementations for policy accounts, micropayment channels and proof submission;
- Public testnet and faucet, matching mainnet parameters;
- Audit subsidies for ecosystem projects, lowering the security barrier for early teams;
- Standards participation: contributing upstream to ERC-8004, x402 and related work as an implementer rather than forking a dialect.

11.4 Permissionless and neutral

DaVinci is a permissionless chain. The protocol layer performs no identity screening or geographic restriction on any participant, maintains no allowlist, and reserves no power to ban. This applies equally to users, agents, developers and service providers.

Neutrality requires it. A chain that can decide who may use it is not infrastructure, it is a platform. Applications may implement whatever compliance modules they need, but

that is the application's choice, not a protocol mandate, and the corresponding responsibility rests with each participant.

11.5 Ecosystem invariants

The preceding sections describe what the foundation intends to do. This one describes what it *cannot* do, and why that part has to be written down rather than promised.

Political economy offers one observation that has held up across many cases: a stable society steadily accumulates distributional coalitions, groups that produce nothing new and compete over the existing pie, until the whole system settles into a low-dynamism equilibrium. Olson called this institutional sclerosis, and the mechanism is plain. Such coalitions are costly to form and far costlier to dissolve, so their number only goes up. Public chain ecosystems are not exempt. A few years into a grant program, the list of recipients overlaps heavily year to year, emissions stay locked in pools that stopped producing growth long ago, and failed projects secure follow-on funding on grounds of strategic importance. These are symptoms of one disease.

Token governance makes it harder than in the offline world. There, interest groups must lobby to influence legislation indirectly. Under token voting, whoever received a large allocation is also a voter on the next one. That is not lobbying. That is writing your own check.

The institutional answer is reasonably well established. Weingast and co-authors, working on market-preserving federalism, found that what makes a decentralized structure actually constrain extraction is not decentralization itself but two specific conditions: no exclusive privileges inside the common market, and hard budget constraints on every participant. DaVinci takes those two, adds the right of exit, and writes them as six ecosystem invariants.

E-1 | The foundation operates no applications. Foundation capital may not fund products that compete with ecosystem developers. Spending is publicly auditable.

E-2 | No exclusive privileges at the protocol layer. Every protocol-layer capability, including interfaces, fee tiers, routing order and registry placement, is open to all participants on identical terms. No channel is reserved for a single project. What a mature incumbent asks for is usually not a grant but an exclusive right, because a grant pays once and an exclusive right pays forever.

E-3 | Hard budget constraints. Grants are disbursed once against milestones. No entity receives rescue funding after failure. As long as coming back for more money remains an option, the rational strategy shifts from delivering to lobbying for survival.

E-4 | Mandatory sunset on incentives. Every incentive and emission program states its terminating block at creation. Renewal requires the full approval process again, and nothing rolls over by default. This targets the only variable in Olson's mechanism, which is time.

E-5 | Conflict of interest recusal. Grant recipients and their known associated addresses may not vote on proposals concerning their own funding.

E-6 | Exit is free. No exclusive integrations, no non-transferable points, no mandatory lockups. Projects and users can leave at any time without forfeiting on-chain assets or identity they already hold. Adopting upstream standards rather than inventing dialects (§4.3) is this invariant expressed in technical terms: an agent written on DaVinci can move somewhere else.

Table 5: Enforcement level of the ecosystem invariants

ID	Invariant	Enforcement
E-1	No foundation-run apps	Charter · audited
E-2	No exclusive privileges	**Contract**
E-3	Hard budget, no bailouts	**Contract**
E-4	Mandatory incentive sunset	**Contract**
E-5	Conflict of interest recusal	Partial contract
E-6	Exit is free	**Contract**

Two things need stating plainly about these six, or they are just pleasant sentences.

First, enforcement level cannot be left vague. E-2, E-3, E-4 and E-6 are written into contracts and violating them simply does not execute. E-1 is a charter constraint held accountable through public spending audits. E-5 can only exclude known associated address sets and will not stop a determined sybil. Dressing policy commitments up as hard constraints is the most common source of broken promises in whitepapers, and we would rather mark the boundary clearly.

Second, the constraints phase in. Olson described a disease of mature societies, not new ones. A chain at launch needs to tilt resources toward early builders, and applying every constraint on day one would simply mean nobody builds. E-3 and E-4 therefore take full effect once the ecosystem reaches stage two in Section 13, operating on wider thresholds before that. As with emergency upgrade authority, a temporary exception needs an expiry condition written into the charter, or it is not temporary.

One structural weakness deserves separate mention. Ecosystem projects typically have no revenue of their own and depend almost entirely on foundation grants, which is the definition of a soft budget constraint. No contract clause fixes that. The foundation's approach is to prioritize projects capable of generating their own revenue, and to treat the ratio of grant funding to self-generated revenue as a hard criterion at renewal. That single practice will do more for the long-run health of the ecosystem than any governance clause.

12. Network Economics and the Role of DVC

This section covers how the network sustains itself economically and what DVC does inside the protocol.

The premise has to be stated first: gas is denominated and paid in ETH (§7.2) and DVC carries no gas function. That means DVC's value argument cannot rest on "using the chain requires buying the token," the easiest path to write and the least defensible. It has to rest on real protocol functions.

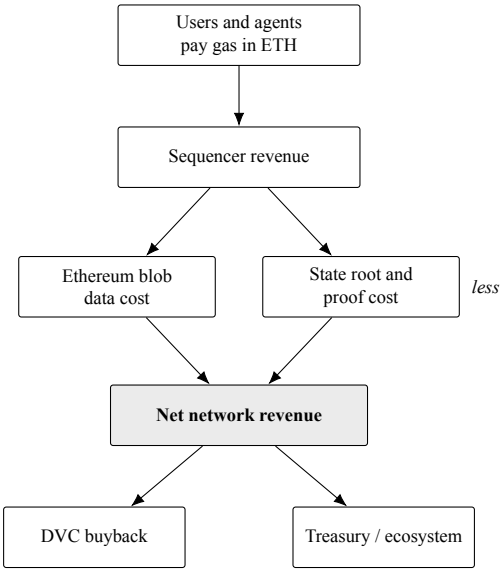
Table 6: DVC token parameters

Field	Value
Name	DaVinci Coin
Symbol	DVC
Decimals	18
Total supply	1,000,000,000

Minting and burning mechanics, allocation structure and release schedule are covered in a separate disclosure document and are outside the scope of this technical paper.

12.1 Network fee flow

Diagram 6: DaVinci network fee flow



Gas paid by users and agents accrues to sequencer revenue in ETH. Against it the sequencer pays two costs: publishing blob data to Ethereum, and submitting state roots with their associated proving cost. The difference is net network revenue.

Two properties of this structure are worth noting. First, net revenue tracks usage and is independent of token price, so the network's sustainability does not depend on market sentiment. Second, high-frequency micro-transactions mean thin margins across enormous volume, which requires continuous optimization of the execution layer rather than raising fees to improve the economics.

12.2 What DVC does in the protocol

Performance bonds: staking and slashing. This is DVC's core function. Agents providing services on chain, including data providers, model providers and execution services, must stake DVC as a performance bond. The size of the stake determines the volume of business they can take on, and proven breach triggers slashing, part of which compensates the injured party.

The weight of this function is that it is the economic foundation of the entire trust layer in Section 8.5. Without collateral that can be slashed, reputation is a number with no enforcement behind it. With it, prior trust between strangers acquires a price. The larger the agent economy

grows and the higher its trust requirements, the more DVC is locked as collateral.

Settlement medium in service markets. Subscriptions, revenue shares and bonds for services listed in the capability registry are denominated and settled in DVC (§9.2). DaVinci does not force every transaction into DVC, since mandatory denomination hurts usability; the protocol instead offers lower fees and better collateral efficiency on DVC-settled paths.

Staking for sequencing and validation. As sequencer decentralization proceeds (Section 13), participating in sequencing and validation requires staking DVC under slashing conditions. This binds the network's liveness and security to the token's economic value.

Governance. Decision rights over protocol parameters, verification rules, slashing conditions and treasury spending (Section 13). Governance is an accompanying property, not the main value argument.

Net revenue buyback. A portion of net network revenue is used to buy DVC on the open market, with the proportion set by governance and executed publicly. This links network usage to the token structurally.

Taken together, DVC's demand can be stated in one line: the more economic activity on the network requires bonding, the more DVC is locked. The argument rests on real service transactions and collateral requirements rather than transaction count multiplied by a fee rate.

DVC's total supply, allocation, release curve and unlock schedule are covered in a separate disclosure document and are outside the scope of this technical paper.

13. Governance and the Path to Decentralization

A chain operated by a single entity, however well engineered, is not qualified to carry an autonomous economy, because an agent cannot build a long-term relationship with a counterparty that can change the rules at any moment. Decentralization is not idealistic decoration for DaVinci. It is functional necessity.

This section gives a path, not a date.

13.1 Three stages of sequencer decentralization

Table 7 lists the verifiable milestones for each stage. DaVinci delivers by stage rather than by date, and each stage is marked complete by its milestones rather than started by a time commitment.

Stage one: single sequencer with unconditional exit. The mainnet launch shape. The sequencer is run by the foundation, but force inclusion and the escape hatch are in effect from day one (I-2, I-3). The security model here is that the sequencer can inconvenience you but cannot cost you.

Stage two: permissioned multi-party sequencing with mandatory rotation. Several independent operators sequence together, block production rotates by rule, and censorship by any one of them can be routed around by the others. This removes the single point of failure and of censorship, though participants still require admission.

Stage three: permissionless sequencing with staking and slashing. Any party meeting the stake requirement can sequence, and malicious behavior is punished by slashing. After this stage the network's liveness no longer depends on any particular entity.

Proof decentralization advances alongside it: the validator set expands from a small number of foundation-funded nodes to a permissionless set anyone can run and be rewarded for catching errors.

13.2 Disposition of upgrade authority

Emergency upgrade authority exists at mainnet launch. This is the realistic choice for any responsible new chain, because the speed of fixing an early vulnerability can decide whether user assets survive. DaVinci makes three commitments about it:

- Public: the scope of the authority, the composition of its holders, and its trigger conditions are all published;
- Limited: it can only be used for security incident response, never to change economic parameters or seize assets;
- Time-bound: it narrows along the staged path above until destroyed, with each narrowing publicly verifiable (I-9).

13.3 What governance can and cannot decide

On-chain governance decides protocol parameters, verification rules and slashing conditions, treasury spending, and admission rules for sequencing and validation. All core

Table 7: Staged delivery and milestones

Phase	Verifiable milestones
Phase 1 · Testnet	Public testnet live; identity and account layers (including the policy engine) available; first Agent SDK release open-sourced
Phase 2 · Mainnet Genesis	Mainnet live; canonical bridge, force inclusion and escape hatch in effect; stablecoin paymaster available; core contract audits published
Phase 3 · Agent Protocol	Payment, discovery, verification and settlement layers fully open; intelligence supply primitives live; first third-party agents running on mainnet
Phase 4 · Ecosystem	Developer program at scale; meaningful third-party supply in the capability registry; multi-party sequencing enters stage two
Phase 5 · Sovereign	Permissionless sequencing and validation; emergency powers destroyed; governance handed to the DAO

parameter changes require a timelock, during which users can exit freely (I-8).

What governance cannot decide matters as much. It cannot seize user assets, close the escape hatch, remove force inclusion, or retroactively alter completed settlement. Those constraints are written into contracts, not into promises.

14. Closing

The problem DaVinci Chain addresses fits in one sentence: AI agents already have intelligence, but no standing to act under their own name on an open network, hold value, and bear the consequences.

Our answer is not a new consensus or a new virtual machine. Security is borrowed from Ethereum. The execution environment is the Arbitrum Orbit stack, already validated in institutional production. The protocol standards are the ones the ecosystem already has: ERC-8004, ERC-4337, x402, MCP and A2A. What DaVinci actually does is one thing: take those scattered standards, which today sit as optional contracts implemented in isolation, and turn them into a protocol substrate where they interlock.

That is not an exciting sentence. But the value of infrastructure has never come from novelty. It comes from being reliable, neutral, and hard to replace. What the Web4 machine economy needs is not another chain with a story. It is a chain where the first transaction between two agents who have never met can complete safely.

Leonardo drew machines five hundred years ago that he could not build. We do not lack the idea. We lack the craft to carry it.

Disclosure

This document is version V0.1, a working draft of the DaVinci Chain technical whitepaper. The technical approach, parameters and staged plans described here may change as engineering progresses, and the mainnet's actual state at launch together with official announcements take precedence. Any capability described here that is not listed under a completed phase should be read as a roadmap commitment rather than a current state.

This document does not constitute an offer of securities or investment advice in any jurisdiction. The DVC token is not designed to be a security in any jurisdiction. DVC (DaVinci Coin, 18 decimals, total supply 1,000,000,000) has its allocation and release schedule covered in a separate disclosure, outside the scope of this document.

DaVinci is a permissionless, neutral protocol layer and performs no identity screening or geographic restriction on any participant. Participants are responsible for assessing and bearing compliance obligations in their own jurisdictions. Participating in a public chain ecosystem involves smart contract risk, crypto asset volatility and technology evolution risk. Please read the risk disclosure in Section 10.4 before participating.

Referenced Standards and Implementations

1. ERC-8004: Trustless Agents. On-chain identity, reputation and validation registry standard for agents.
2. ERC-4337: Account Abstraction Using Alt Mempool.
3. EIP-7702: Set EOA Account Code.
4. EIP-4844: Shard Blob Transactions. Ethereum blob data availability.
5. x402. Machine micropayment protocol built on HTTP 402.

6. Model Context Protocol (MCP). Standardized tool-calling interface for agents.
7. Agent-to-Agent (A2A). Inter-agent communication protocol.
8. Arbitrum Nitro / Orbit. The rollup stack this chain is built on.
9. Robinhood Chain. An Ethereum L2 on the same stack, mainnet July 2026.
10. Mancur Olson, *The Rise and Decline of Nations*, 1982. Distributional coalitions and institutional sclerosis; the diagnosis behind §11.5.
11. Montinola, Qian & Weingast, "Federalism, Chinese Style", *World Politics*, 1995. Market-preserving federalism; the institutional design behind §11.5.