Reinventing the ISP

Request for Comments

Revision 0.1

March 21, 2026

Contents

1 Introduction

The rise of hyperscale platforms like Google or Facebook exemplifies the power of Metcalfe's law: their value explodes as user networks grow, amplified by the borderless reach of online services that can address the entire globe from a single point. This dynamic has created immense economic concentration, where platforms capture value through direct user relationships and proprietary user data silos which together power advertisement revenues and subscriptions.

This proposal introduces a setup that harnesses an even stronger set of dynamics and network effects than that of the hyperscalers: by reinventing Internet Service Providers (ISPs) as federated Payment Clearing House of the Internet, Data Commons Aggregator and Internet Service Distributors, we position them as the central economic and distribution hub that spans geography and application verticals. The resulting economical liquidity and data assets dwarf any single hyperscaler platform. All this with the aim to establish metered, pay-as-you-go monetization model as the superior model of the new Internet.

Individually, hyperscalers appear unbeatable, fortified by their massive scale and entrenched network effects. Although end users constitute the largest mass amongst all Internet actors, they are structurally weak, fragmented, and essentially do not battle efficiently for their own good. If we however place economically motivated institutions (not individuals) that are fundamentally more aligned with what is good for the users and we sufficiently coordinate them, we can create a force that is competitive with the hyperscalers.

Internet Service Providers are the only universal counterparties that every online interaction already touches. Yet they are treated as commodity bandwidth vendors, structurally excluded from the economics of services that traverse their networks. This is historically understandable — ISPs lacked a neutral way to participate without becoming yet another platform. This is despite the fact that very physics is on ISPs side because it funnels all traffic through them; now the advancements in cryptography and decentralized protocols finally let them cooperate. This together opens up an opportunity to create a new business model for next genaration of Internet applications which will be able to tap into larger money pool and build applications will less friction and superior UX, all that while still allowing app builders to actually make money.

In the proposed model the user's sole financial counter-party is the Internet Service Provider; applications never invoice, token-gate, or collect cards from end-users. Users pay their ISP, creating an aggregated fund for all digital services; apps earn revenue based on cryptographically proven metered usage, bypassing direct user billing. This financial decoupling fosters seamless pay-as-you-go access, broadening markets to casual users and niche creators.

ISP collected and protocol-federated data aggregagation as well as jobs executed on aggregated edge computing capacity in turn unlocks new revenue streams which serves as a counterweight for advertiser money which in the old Internet funds free services.

When payment and telemetry migrate to the last mile, the strategic footing of over-the-top platforms erodes and the economic gravity of Internet services flips: rather than users paying dozens of apps, a single recurring payment to the ISP funds everything they touch making the new Internet user experience completely ubiquitous. Rather than a few platforms owning the world's telemetry, thousands of federated providers aggregate a dataset no company could

assemble alone while simultaneously making it available to everyone.

This setup does not expect users to behave like idealised cypherpunks, champion digital self-sovereignty or be moral agents - users remain delightfully ordinary. They still pay their connectivity bill and click on whatever entertains them. The heavy lifting — metering, settlement, revenue sharing, privacy enforcement — occurs behind the scenes, faciliated by the protocol and executed at the edge. ISPs, incentivized by new revenue streams and strong position in the new economy propel adoption — users gain from reduced costs, wider and seamless app access as the system expands.

At the heart of this transformation is a neutral, decentralized protocol - the Edge Settlement Protocol (ESP). Just as TCP/IP revolutionized computing by federating disparate networks and converting a patch-work of incompatible networks into one routable fabric - into the Internet, ESP federates ISPs into a planetary-scale economic force. Where TCP/IP's federation created universal data exchange, ESP's federation creates universal value exchange for Internet services. Just as TCP/IP elevated ISPs from isolated network operators to interconnected Internet gateways, ESP builds on this foundation by further elevating them from mere commodity pipes to essential distributors for Internet services.

The protocol enables emergence of next generation Internet services designed to capitalize on the new monetization paradigm. By leveraging new economic model, these applications can generate revenue directly proportional to actual usage—whether measured in API calls, data transfers, or computational cycles—without the need for user registrations, bundled offerings, or compromises that dilute the core user experience (displaying ads). This approach enables developers to focus on core functionality, usability, and efficiency, leading to services that effectively transform casual user engagement into tangible value. Consequently, the system favors applications optimized for user experience and broadens market reach to encompass casual and global audiences while ensuring equitable compensation for creators based on demonstrated utility, thereby cultivating a more vibrant and user-centric digital landscape.

Another important premise for why a universal pay-as-you-go model can become the new economic foundation of the Internet as well as many design choices made in the proposed setup stems from how the AI revolution is fundamentally reshaping application dynamics and shifting the balance of power. In the post-AI landscape, the nature of digital services is fundamentally changing - AI is transitioning from being a distinct, standalone product into a ubiquitous, foundational utility (eg. electricity). Because of this transition, we will likely see more and more initiatives to nationalize infrastructure that provides this fundamental utility and metering based distribution network capitalizes on this shift.

Further, the AI models and generative services are rapidly commoditizing software creation. The cost of generating code, text, and business logic is trending toward zero. In this future, the traditional software moats—unique features, slick user interfaces, and proprietary application logic—will evaporate.

The envisioned ecosystem is heavily rooted in the moast that are believed to be the only remaining moats in a post-AI world:

1. **Physics:** The actual physical infrastructure, and facts like that compute must happen on

actual computer placed in actual building, actual communication cables must be wired to actual households etc.

2. **Ownership:** Ownership of digital or physical assets, contracts etc.
3. **Policy:** The rules of engagement, neutrality, and the governance frameworks that dictate how services are distributed and monetized. Social fabric.

1.1 Core Objectives and Ecosystem Goals

The overall goal is to achieve universal payg model for the Internet and enable building new generation of Internet applications which are better for end user from every possible angle including user agency and as a result let new kind of physical infrastructures emerge naturally as a result of structurally different business and delivery models leveraged by the new applications - a kind of physical infrastructures which DePINs aspired but failed to deliver. For this to happen a number of core objectives is identified:

- **Significantly strengthen ISPs position in the Internet economy:** Make ISPs the gravitational centers of the Internet, not the hyperscalers. Evolve them to more comprehensive role with larger responsibilities and bigger capture.
- **Remove distribution from hyperscalers:** The protocol must actively dismantle walled-garden effects. By decoupling the payment and distribution layer from the application layer, and by dividing power and aligning incentives among disparate actors, it ensures open access and prevents any single entity from monopolizing the user relationship.
- **Remove the data moat from hyperscalers:** Data is users and users should benefit if data is being used, provide means of socializing the value of data commons
- **Significantly advance the state of the art of Internet applications:** Applications where user agency, UX and monetization are aligned and superior to the current state of the art.
- **Be low footprint, minimal and neutral:** The protocol itself is not designed to make money, nor should it bear an inherent tax or take a cut of the economy. Wealth should be generated the hard way by "hard business"—ISPs building physical connectivity, data centers provisioning compute, and developers writing valuable software. The protocol acts simply as a neutral public utility that unlocks and coordinates these opportunities.
- **Do not hinder real tech innovation:** Disaggregating distribution must have no negative impact on true technological advancement. It must not block innovative teams to build innovative products and compete with them on the market.
- **Evolve cypherpunk thinking for the real world:** The protocol must always frame key actors (like ISPs and Application Site Operators) as profit-driven, legally recognized entities, not overly rely on individuals and tech fans. The structural pillars of the network always operate as accountable, legal entities and should be businesses. No legal vacuums.
- **End user privacy must be paramount:** The participation and interactions of end-users must remain private

- **Lessen capital investor influence and empower builders:** the system lessens the influence of pure capital and empowers the actual builders
- **Ensure small innovative teams thrive:** The ecosystem must guarantee that small, agile teams can succeed. By providing a neutral, global distribution and monetization layer, a small team can instantly reach millions of users and get paid for their utility without needing to build massive billing, compliance, and distribution networks from scratch.
- **Solidify physics, ownership, and policy as the only moats:** In a post-AI world where software logic is infinitely reproducible, the protocol aims to make the physical reality of the network—the trifecta of physics (cables and compute), ownership (the customer relationship), and policy (the social contract and protocol rules)—the only defensible moats.
- **Ensure a permissionless entry:** The protocol must remain fundamentally permissionless, allowing any developer, service provider, or infrastructure operator to join, contribute, and monetize without seeking approval from a central authority or gatekeeper.
- **Prioritize metering over full-blown transactions:** For real-time pay-as-you-go to be viable at Internet scale, the architecture relies on lightweight, high-throughput usage metering rather than heavy, consensus-driven transactions, avoiding unnecessary overhead for continuous micro-interactions.
- **Capitalize on always-on, on-premises compute:** As more powerful compute hardware is deployed on-premises and remains powered 24/7, the protocol must to capitalize and support this trend

1.2 Note on DePIN

In recent years, Decentralized Physical Infrastructure Networks (DePIN) attempted to disrupt this entrenched model by crowdsourcing infrastructure deployment. The promise was alluring: use blockchain tokens to incentivize individuals to deploy antennas, compute nodes etc., creating a decentralized competitor to traditional ISPs and cloud providers.

However, DePIN largely failed to achieve mainstream, sustainable adoption because its economic model relied heavily on speculative tokens rather than real-world value generation. The tokenomics created misaligned incentives. The supply side (node operators) was motivated by artificial yields and the hope of token price appreciation, rather than actual demand for network usage. When the speculative fervor cooled and token prices inevitably crashed, the economic viability of operating the infrastructure collapsed with it.

Furthermore, DePIN networks were often too retail-focused. They relied on hobbyists and individuals, failing to provide the enterprise-grade reliability, Service Level Agreements (SLAs), and robust business-to-business frameworks required to support mission-critical applications. It became clear that infrastructure cannot be sustained purely by retail speculation; it requires a foundation of actual utility and stable economic exchange.

Interestingly, the Internet itself functions as the largest and most successful Decentralized Physical Infrastructure Network (DePIN) in existence, with the Internet Service Provider (ISP)

as its primary building block.

The historical centralization of the Internet, characterized by the emergence of closed ecosystems or "walled gardens," was driven primarily by the transition toward data-centric applications. Because data necessitates physical storage in data centers, it inherently gravitates toward centralization. This architectural convergence was not born of monopolistic intent but was a pragmatic response to technical limitations. While early protocols such as SMTP and HTTP were inherently stateless and aligned with a peer-to-peer topology, the evolving demands of user experience required persistent state management, real-time search capabilities, and sophisticated algorithmic curation. During the formative years of the web, coordinating state across a highly distributed, heterogeneous network presented an intractable computer science challenge. Consequently, the most efficient solution was to route traffic through centralized computational hubs—the precursors to modern hyperscale data centers—where data aggregation, indexing, and monetization could be executed efficiently.

Under this paradigm, the underlying network was relegated to a commoditized transit layer. ISPs committed massive capital expenditures to deploy physical infrastructure—including subsea cables, terrestrial fiber networks, and cellular towers—yet the overwhelming majority of economic value was captured at the application layer. The infrastructure providers absorbed the capital risk, while the entities controlling the centralized data silos captured the outsized returns.

Crucially, the technological constraints that originally mandated this centralized architecture are being challenged. Recent advancements in applied cryptography, distributed consensus mechanisms, data replication, zero-knowledge proofs and edge computing are making it possible to build competitive, global applications which are more distributed and decentralized in nature.

The proposed protocol does not merely pursue decentralization as an ideological goal; rather, it corrects a multi-decade architectural compromise. ISPs, functioning as federated nodes within a unified Data Commons and Payment Clearing House, embody the ultimate realization of the DePIN concept. They already operate deployed, robust networks; they command recurring revenue streams from established consumer bases; and they maintain the critical physical connection to end-users globally. They do not require the issuance of speculative tokens to bootstrap a network. Instead, they require a standardized, neutral protocol designed to unlock the economic potential of their existing physical assets.

2 ESP protocol: the Internet's clearing house

Edge Settlement Protocol establishes ISPs as the sole financial intermediary between users and applications. The protocol revolves around three economical relationships that define the flow of value: Users to ISPs for the actual usage, ESP to applications for service revenue, and ISPs to ESP for aggregated usage clearing.

The foundational relationship is between users and ISPs, where users remit a single, recurring connectivity bill that covers both network access and all downstream application usage. This dynamic benefits users by eliminating the need for multiple subscriptions or payment methods, creating a seamless "pay once, use everything" experience that reduces administrative burden and financial fragmentation. For ISPs, it strengthens customer relationships by positioning them as the trusted gateway to the entire Internet.

Note that user-to-ISP relationship is not formalalized in the protocol itself, the user-ISP relationship is fully in hands of ISPs in traditional way. It has to be mentioned here however to get a complete picture of the environment for which the ESP is designed.

The ESP-to-applications relationship handles service revenue disbursement, where applications interact solely with the protocol as their global counter-party, insulating them from the variability or unreliability of individual ISPs. When users initiate service consumption, these events are recorded through the user's parent ISP into a tamper-proof ledger of user consent and usage. Applications submit periodic disbursement requests—batched claims listing the service events they fulfilled, including correlation keys, timestamps, and cryptographic signatures proving they served the requested units.

The ISPs-to-ESP relationship enables aggregated usage clearing, where ISPs maintain pre-funded balances and periodically top up escrow proportional to projected traffic. ESP debits ISP accounts based on matched service-to-disbursement pairs. ISPs can verify ESP demands against their own logs and the ledger records posted through their network. If balances fall below thresholds or mismatch rates exceed policy limits, ESP enforces compliance through collateral slashing, temporary exclusion from revenue sharing, or other financial penalties. This setup provides predictable cash-flow tools for ISPs and integrates clearing revenue into core operations.

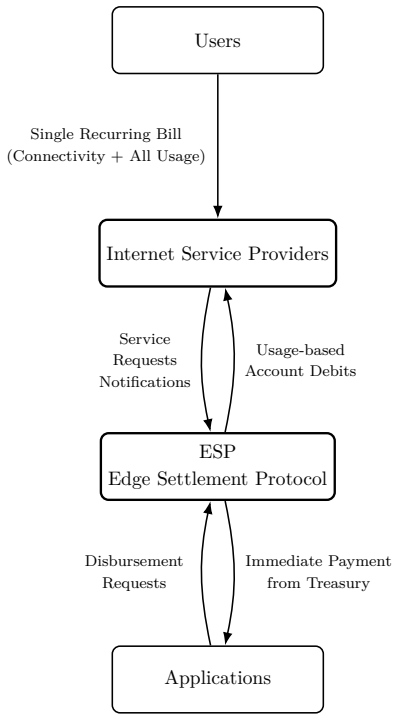

Figure 1: ESP relationships among Users, ISPs, ESP, and Applications

2.1 ESP Economy

The ESP architecture introduces a fundamental shift in Internet economics: the transition from fragmented, bilateral transactions to a unified, aggregated Liquidity Pool. This is the economic engine that makes the entire ecosystem viable, mimicking the proven financial strategies of nation-states, insurance conglomerates, and corporate federations.

2.1.1 Dynamics of Aggregate Pooling

In the current digital economy, every interaction is a discrete financial event. A user pays $10 for Netflix, $0 for Google (paying with data), and $5 for a Substack. This fragmentation creates immense friction: mental transaction costs ("is this article worth $1?"), subscription fatigue, and inefficient capital allocation.

The ESP model replaces this with a single, aggregated, global "Liquidity Pool" managed by the ISP collective. The user pays a single bill—their connectivity subscription—which ISPs use to feed this pool. Simultaneously, the pool is fed by non-user revenue streams: the commercialization of shared data assets and the leasing of distributed edge computing resources.

This aggregation creates a "Sovereign Wealth Fund" for the digital citizen. The money pool blends these distinct revenue streams into a single reservoir of value, decoupling the *cost of service* from the *price of access*.

The strength of having such a pooled economy (Wealth Fund) lies in diversified inputs and blending them together. In the ESP model, revenue is a blend:

- **Connectivity Cost:** The monthly connectivity fee.

- **Paid internet services:** Paid internet services (equivalents to Netfix, Youtube, AI tokens etc. - the heavy duty services)
- **Monetization of data and compute:** Revenue from Data Commons (web indexing, mapping) and Edge Compute jobs.

On the expenditure side, the pooling yields effects known from insurance funds, corporate bundles and public budget allocations. Microsoft bundles Word, Excel, and Teams not because every user uses every tool equally, but because the *aggregate value* prevents churn and captures the total surplus. Governments use tax pools to fund public utilities (roads etc.). A citizen who walks still pays taxes that pave the highway, because the highway drives the economy that employs the walker.Similarly, the ISP bundle covers "Basic Internet" (Search, Email, News, Video). Heavy users of Search subsidize heavy users of Video, and vice versa. The friction of "paying per search" is eliminated, unlocking usage that would otherwise be suppressed by transaction costs. The pool ensures these foundational public goods are funded by the aggregate economy, not by charitable donations or ad-tech surveillance.

This pooling mechanism stabilizes the economy for builders. Instead of chasing viral hits to sell ads, developers compete for a share of the stable, recurring Liquidity Pool based on verifiable utility.

This allows for the "Long Tail" of software to survive. A niche application used by only 10,000 people cannot survive on ads (CPM is too low) or subscriptions (conversion friction is too high). But in a pooled economy, if those 10,000 users find it useful, the app claims its fair share of the pool.

2.1.2 Monetization of Data and Compute

It is vital for the ESP-shared Wealth Fund to not rely solely on user connectivity fees for its income, the network must generate systemic revenue from its own operations. This is achieved by transforming latent network capabilities, specifically, anonymized data aggregation and distributed compute, into marketable assets.

In the legacy model, user data is extracted by centralized platforms to fuel targeted advertising. The new paradigm restructures data monetization into a privacy-preserving ecosystem. By securely aggregating macroscopic trends, network traffic patterns, generalized web indices, etc. at the ISP level, the network produces valuable, anonymized datasets.

Similarly, the network contains vast amounts of underutilized computational power at the edge, spanning from ISP-managed facilities to consumer-grade hardware. By pooling these resources, the network can lease distributed compute capacity for external workloads—such as content delivery, machine learning inference, or decentralized processing. This effectively turns the network's decentralized physical footprint into an active, revenue-generating resource.

Together, the commercialization of aggregated data and the leasing of edge compute create a robust, non-extractive revenue stream. This systemic income helps to subsidize the overall cost of generalized internet access.

2.1.3 Reframing Value: Savings, not Profit

The Web3 and crypto space has seen countless attempts to build decentralized user networks by promising participants that they can "earn" native tokens in exchange for their data, bandwidth, or compute. This approach has consistently failed to achieve mainstream adoption. Earning a few volatile, often useless tokens feels like a speculative chore rather than a reliable benefit, and the friction of managing wallets and liquidating these assets far outweighs the negligible financial reward.

Instead, the focus must shift from earning to saving. As "profit" (a few dollars of side income) it feels trivial and administratively costly; as "savings" (a visible discount on an existing, recurring connectivity bill) it becomes salient and attractive. Anchoring micro-earnings against a fixed monthly spend reframes them as a cheaper Internet bill, not as pennies earned, improving perceived value while reducing cognitive load. ISPs can pass a share through as automatic bill credits while retaining the remainder to fund commons services. This framing turns a psychologically insignificant profit into a meaningful saving.

However, individual users are neither altruistic volunteers nor motivated data entrepreneurs. Even when monetized, the marginal value of one person's telemetry is tiny; payouts feel like "coupon math" that sits below attention thresholds. Participation therefore has to be orchestrated and "pushed" by ISPs as managed, default infrastructure. ISPs can operate verifiable edge software, and can aggregate across their entire subscriber base—where the signal becomes statistically powerful and the revenue line item becomes significant.

2.1.4 The "Free" Services

The key economical thesis is that universal, commitment-free access results in household's willigness to spend more on Internet services. Today, millions avoid per-app subscriptions because the commitment risk outweighs occasional value; the long tail of curiosity and casual usage remains unmonetized. When access becomes seamless across an immense catalog, users willingly accept a slightly higher blended monthly spend (compared to their current total they pay for Internet services now) in exchange for optionality, lower cognitive load, and fairness—they pay for what they actually touch, not for every service they might someday use.

The effect is familiar from city-wide sport facility "multipass" programs. Few will maintain two full memberships they seldom use; many will pay a modest premium for a pass that covers all gyms and pools. Aggregate revenue rises: light users begin contributing, regular users consolidate spend, and income becomes more stable and predictable.

This economical thesis is followed by the critical budget thesis which underpins economy of the ESP environment for services which users are accustomed to get for free:

The combined extra from the universal-access surplus (the increase in blended spend when users pay once for access to everything) combined with extra income which ESP protocol faciliates via its ability to monetize data collection and edge compute workloads exceeds the cost of maintaining the core "free" Internet services—email, search, video, and news—at high quality and without ads.

When this economic surplus is blended with the connectivity bill, it enables ISPs to provide

a perception of services being free. At the same time users gain universal access to "heavy duty" online services and have their overall Internet spend consolidated, better controlled, and ultimately lower. While builders remain free to leverage advertising, doing so opens up a direct UX battle where competitors can offer the same product without ads, instantly gaining a superior user experience.

While the underlying mechanics of the ESP protocol are strictly pay-as-you-go (every API call, every query, every byte has a price), the user experience is designed to feel indistinguishable from the "free" internet they are accustomed to. This is a deliberate psychological and economic strategy orchestrated by the ISPs.

It is up to the ISP to construct the "illusion of free" by leveraging effects of the pooled economy. Because the pool aggregates revenue from connectivity fees, data commons sales, and edge compute jobs, the ISP can offer generous "allowances" or "unlimited tiers" or monthly credits for core services. To the end user, Google-grade search, 4K video streaming, and ad-free news appear as zero-marginal-cost utilities included in their monthly internet bill. They do not see the ticker of micro-payments occurring in the background; they simply see a service that works without a paywall.

This abstraction is crucial for adoption. Users will not accept a regression in user experience where they must approve every micro-transaction. By absorbing the variance of usage into the aggregate pool, the ISP effectively subsidizes the "basic living standards" and maintains the frictionless experience of the free web while replacing its toxic ad-based funding model with a sustainable, transparent economy.

2.2 Software as franchise

Although ESP protocol provides universal means for clearance, it contains subset of primitives tailored for one specific application deployment model and that is the model envisioned as the end game of post-AI future. The architecture introduces a fundamental split between product building and the operations, drawing some similarities to a business franchise model rather than the monolithic SaaS model of Web 2.0. In this "Software as a Franchise" paradigm, there are two roles defined and economically aligned by the protocol:

The **Founder Team** functions as the "Headquarters". Application Builders focus on building: ie. product vision, innovation, brand building, and software development. They produce the "product"—the application code, the user interface, and the core logic—but they do not host it.

The **Site Operator** (hosting entity often affiliated with some ISP) functions as the "Franchisee." They focus on execution - they provide the physical infrastructure (datacenter or edge capacity), handle local regulatory compliance, manage the "last mile" of customer support, and ensure service availability. Possibly also, they act as a marketing/distribution, leveraging their direct access to subscribers to drive local adoption and marketing.

The ISP protocol offers specific primitives that enable this structural split in a tangible and enforceable way.

3 ESP protocol architecture

ESP protocol provides a shared ledger, settlement rules, and a common base primitives that allow many competing ISPs to behave like a single global clearing and distribution layer. The protocol does not replace an ISP's traditional relationship with its subscribers; instead it standardizes how economic activity observed at the last mile is reported, priced, and settled so that applications can be paid reliably without integrating with thousands of billing systems.

At a high level ESP implements four core capabilities:

- **A central, shared, clearing fund at the center of all economic activity** for stablecoin-denominated settlement between ISPs and applications.
- **General economic activity metering that is verifiable on blockchain** from user/household devices on behalf of the ISP that serves the endpoint.
- **Permissionless marketplace for advertising edge compute and data collection work** where ISPs advertise their capabilities and external entities can request work, enabling decentralized data commons and distributed edge computation at planetary scale.
- **Auxiliary primitives needed for economic activity and capital formation** ensuring robust liquidity and financial operations within the ecosystem.

ESP is envisioned to run as either a dedicated L1 chain purpose-built for settlement and metering, or a sovereign rollup with its own execution and governance. In both cases, the key property is that ISPs collectively run the validating nodes and enforce rules by executing code. Consequently, protocol rules are strictly enforced on-chain, ensuring that network policies and subsequent upgrades are collectively ratified by the participating ISP operators.

All value in the ESP ecosystem flows through the same Master Clearing fund but expresses itself in three distinct activity types:

- **Service activity:** Service activities are the primary consumer-facing unit of value. The billable unit is deliberately small: a service is decomposed into chunks such that missing a few payments is operationally tolerable, and disputes can be scoped narrowly. For example, a search engine might price a single query; a video application might price a 30-second segment; a generative model might price a short completion.
- **Data activity:** Data activity is expressed as paid requests for aggregates, not as raw user data extraction. ISPs can advertise which aggregates they can produce (under strict privacy constraints) and at what price. Data is collected server-side from application sites running in Site operator datacenters, observing traffic for users the ISP serves. Buyers submit requests, ISPs produce and post proofs that the aggregate was computed from eligible user sessions and time windows, and payouts clear from the master clearing treasury.
- **Job activity:** Job activities represent edge workloads executed on user-premises infrastructure (including home edge devices when applicable) if users enable it (for ISP to route jobs there). ISPs advertise capabilities, buyers submit job requests, and completion is proven via attested execution outputs. This creates a market for ISP-operated compute and measurement.

All economic activity within ESP — services, data, jobs — is structured around a universal pattern of **Offers** and **Requests** - producers (applications, ISPs) publish offers declaring what they provide and at what price; consumers (users, apps, data buyers, job requesters) emit requests committing to purchase at those terms. The protocol's role is to match offers and requests, meter, and settle payments.

- **Offers:** Providers of services, data, or compute publish standardized Offers to the network. An Offer defines the capability being sold, the billing unit, the price, and the cryptographic or operational proofs required to verify delivery. We distinguish three Offer types - *Offer for Service (OfS)*, *Offer for Data (OfD)*, and *Offer for Job (OfJ)*.
- **Requests:** Consumers of these resources (such as users accessing a service, or buyers commissioning data jobs) initiate activity by issuing Requests against specific Offers. A Request signals a commitment to pay the stipulated price upon successful execution. We distinguish three Request types - *Request for Service (RfS)*, *Request for Data (RfD)*, and *Request for Job (RfJ)*.

Economic activity in the ESP-based Internet is propelled by the continuous emission of Rf* events—they set things in motion, trigger work, and unlock payments. If Rf* events cease, the system idles; if they surge, the system starts spinning. This event-driven architecture aligns with pay-as-you-go principles: payment is tightly coupled to actual demand, not pre-committed subscriptions or speculative allocations.

Together with verifiable metering and automated settlement, this Offer/Request lifecycle eliminates the need for bespoke bilateral agreements, allowing any participant to seamlessly discover, consume, and pay for digital resources across the global network.

3.1 Account model and financial flow

ESP maintains a small set of well-defined on-chain account types:

- **Master Clearing fund account:** the heart of the protocol - holds the stablecoin float that backs all economic activity.
- **ISP Accounts:** each ISP maintains a funded balance (or escrow) that the protocol can debit based on cleared usage.
- **Application Site Operator Accounts:** each application site receives payments from the master account as it serves user requests. Application site is identified by IP address (prefix).
- **Stablecoin Issuer Account:** the on/off-ramp counterparty that mints and redeems the settlement asset.
- **Job/data buyer accounts:** one account per buyer of data or jobs, used to fund job and data activities by paying stablecoins into the clearing fund.
- **Founder Accounts:** represents the legal owner and creator of an application, tied to its DNS domain. This account configures royalty parameters within the protocol and receives automatic revenue shares from each service interaction disbursement

- **Fund raise Accounts:** a temporary, capped account created during an application's capital formation phase. It receives a defined portion of service revenue to pay back initial backers, dissolving once the maximum cap is reached.

The financial cycle begins when ISPs purchase stablecoins from the stablecoin issuer account—a regulated or protocol-managed entity that mints and redeems stablecoins pegged to fiat currency. ISPs deposit these stablecoins into their protocol-controlled ISP accounts, effectively pre-funding their obligations for downstream usage. As metered activity occurs—users consume services, data is collected, jobs are executed—RfS, RfD, and RfJ events are posted on-chain, creating an immutable record of economic activity.

Applications interact exclusively with the master clearing fund: when they submit disbursement requests that match RfS events recorded on-chain, the protocol transfers stablecoins from the clearing fund to the application's account. This insulates apps from ISP-level credit risk or variability; from the application's perspective, ESP is a single, reliable counterparty with instant settlement.

This creates a closed-loop settlement circuit: ISPs front liquidity into the treasury, usage clears against that liquidity, and apps exit via stablecoin redemption. In practice ISPs can manage balances with policy (minimums, auto-topups, or collateral requirements) enforced by the chain.

3.2 Service activity: pay-as-you-go application usage

Service activity represents the most user-facing economic flow: consumption of Internet applications measured in discrete, billable units. The protocol's design goal is to enable seamless, metered usage where users never manually pay apps, yet apps are compensated fairly for every request served.

The billable unit is intentionally granular—a single API call, page render, media stream chunk, or computational query. These units must be small enough that occasional non-payment or fraud is tolerable noise rather than existential risk, yet large enough to amortize proof overhead.

The client-side initiates service activity by emitting a **Request for Service (RfS)**—a cryptographic acknowledgment that the user consents to consume a specific billable unit. When a user loads an application, the delivered HTML and JavaScript contain an embedded **Offer for Service (OfS)**: a structured declaration of what is being offered (e.g., "search query execution"), the unit price, the app identifier, and a validity window. The browser's JavaScript, upon user interaction (e.g., clicking a search button), generates an RfS referencing the OfS, signs it with an ephemeral session key, and transmits it both to the application server and to the local Customer Premises Equipment (CPE) device.

The CPE device listens for RfS events and logs them locally in a secure, append-only ledger with a timestamp and correlation key (e.g., request hash or nonce). These locally logged RfS events are then pulled by protocol scrapers operated by other ISPs (as described further in the pull-based scraping model), which submit them to the blockchain, creating an immutable, on-chain record of user consent. Critically, the CPE does not independently generate RfS events—it merely witnesses and logs user-initiated requests that the browser JavaScript (loaded application's code) explicitly emits them, preventing ISPs from fabricating phantom usage while

ensuring all legitimate activity is recorded.

The application server receives the same RfS that has also been collected by the CPE device, validates it, fulfills the service, and stores the RfS as part of its own accounting log. Periodically, the app bundles fulfilled RfS records into a **disbursement request** and submits it to ESP—a batch of signed, structured records asserting "I served these specific RfS events, please pay me." Each entry references the on-chain RfS by its correlation key, timestamp, ISP identifier, and app identifier.

ESP's settlement engine queries the metering data for RfS events matching the application's disbursement request. For each RfS that exists on-chain (posted by a CPE device) and appears in the app's disbursement list, ESP verifies the match on correlation keys, timestamps, and identifiers. Only matched RfS-to-disbursement pairs trigger payment: ESP immediately transfers stablecoins from the master clearing fund to the application's account. The corresponding ISP's obligation to replenish their account for the usage attributed to its users is handled separately through periodic settlement and top-up mechanisms. Unmatched requests enter a reconciliation queue; persistent mismatches degrade reputation and may trigger collateral slashing or temporary exclusion.

3.2.1 Protocol schematics: Offer for Service and Request for Service

Offer for Service (OfS)

- **ofs_id:** deterministic content-hash of the canonical OfS payload (referenced by RfS events).
- **protocol_version:** data structute version.
- **provider_id:** the Site Operator (or legally responsible app operator) protocol identity.
- **provider_jurisdiction:** country code (or jurisdiction identifier) for jurisdictional unambiguity checks.
- **provider_sig:** signature by a protocol-registered key for provider_id, covering the canonical OfS payload.
- **app_id:** global app identifier.
- **user_ip:** IP address of the user for which this OfS was generated.
- **service_id / sku:** identifies the billable unit inside the app.
- **unit_definition:** precise definition of one billable unit (e.g., "one API call to endpoint X", "one page render", "one streamed minute", "one inference", etc.). This is what prevents "weird metering disputes."
- **unit_price:** price per unit.
- **settlement_asset:** which stablecoin / unit of account.
- **payee_account:** protocol account that receives disbursements (usually the provider_id, but it can be explicit).

- **valid_from / expires_at:** offer validity window (or a TTL). Without this, old offers can be replayed indefinitely.
- **terms_hash or terms_uri:** human-readable terms pointer + hash (so terms can be referenced consistently in disputes without being huge).

Request for Service (RfS)

- **rfs_id / correlation_key:** globally unique per billable unit (e.g., 128-bit random).
- **ofs_id:** links the RfS to an immutable offer (price + terms).
- **app_id:** prevents cross-app replay of the same RfS blob.
- **timestamp:** when the request was made (used with validity window + dispute windows).
- **units:** quantity of units being requested (often 1, but make it explicit).
- **client_sig:** signature created on the user device

CPE witness + metering envelope

- **cpe_id:** hardware-rooted device identity (public key / identifier).
- **parent_isp_id:** the ISP identity in provisioning metadata
- **cpe_seq:** monotonic sequence number per CPE log (anti-replay + append-only checks).
- **cpe_timestamp:** witness time (helps detect weird client clocks; also used for liveness windows).
- **cpe_sig:** CPE signature
- **prev_log_hash or merkle_root:** lets scrapers/validators prove the log is append-only and detect deletions/rewrites across pulls.

3.3 Job activity: distributed edge computation

Job activity represents the execution of computational workloads on ISP-managed CPE devices—tasks like web crawling, map imagery processing, ML inference, or CDN-style content caching. These jobs are requested and orchestrated by external entities (data commons protocols, application builders, CDN operators) but executed on the ISP's distributed edge infrastructure.

ISPs advertise their compute capacity using **Offers for Job (OfJ)**, published to a protocol-managed job marketplace. An OfJ specifies the job types supported ("web crawler job,", distributed inference, car dashcam processing etc.), available capacity (aggregate CPU/RAM/bandwidth across the CPE fleet), geographic distribution, uptime SLOs, and unit pricing (e.g., per crawl request, per gigabyte processed, per hour of runtime).

Job requesters browse the marketplace (mostly in trully automated way) and emit **Requests for Job (RfJ)** when they need capacity. An RfJ is a work order: "I authorize to pay to run workload W on N nodes in region R for duration D at price P - based on the specific OfJ published by the ISP."

The ISP receives the RfJ and deploys the corresponding containerized workload to its CPE fleet using the secure orchestration mechanisms (image verification, resource limits, attestation). The CPE devices execute the job, generate outputs (crawl data, processed imagery, computed results), and submit them in ways, specific to particular job type (particular edge-computing application which gets deployed).

While the protocol does not govern the intent of these tasks, the ISP collective does exercise consensus-driven oversight over the execution environment. Specifically, the collective approves a registry of permissible binaries and application containers, establishing which software is technically "possible" and authorized to be executed as a Job activity across the network. This collective approval functions as a security and operational baseline. However, this system of authorization does not equate to a mandate for execution; individual ISPs retain complete operational sovereignty and are fully in charge of deciding whether they want to host and execute any particular approved Job within their own environments.

ESP does not dictate what jobs produce or where outputs go. External protocols are responsible for implementing CPE-compatible job containers, defining their own data schemas and collection logic, and managing their own dataset repositories. ESP simply provides the economic substrate—matching ISP capacity with requester demand, settling payments based on verified completion, and enforcing execution attestations. A web indexing protocol implements its own crawler container, and by submitting RfJ puts the entire workflow in motion: ISPs deploy the job across their CPE fleets, execute it, generate proofs; the protocol pays for execution and aggregates results into its own publicly accessible web index. ESP enables this at scale but does not own or govern the resulting dataset.

Job activity powered datasets are envisioned datasets for shared, non-rival information assets—web indices (deduplicated crawl data, link graphs, content metadata), geographic mapping data (street imagery, road geometry, place metadata), network quality telemetry, and other infrastructure-layer observability enabled by ESP's infrastructure but managed by external protocols and initiatives. The protocol's role is strictly job distribution - External entities are responsible for implementing CPE-compatible job workloads, submitting RfJ to activate ISP execution, collecting and processing outputs, maintaining the resulting datasets, and ensuring public access. ESP provides the economic coordination layer (ISPs advertise compute capacity, requesters commission jobs, settlement flows automatically), but dataset governance, schemas, quality control, and distribution remain under the jurisdiction of the external protocols that create and maintain them. This separation allows specialized protocols to optimize for their domain (web crawling, mapping, network measurement) while leveraging ESP's federated ISP infrastructure for execution at scale.

Note that strong focus on legal accountability within ESP protocol extends equally to the Job Activities. When an ISP accepts a Request for Job and deploys a workload to a user's CPE device—whether it be a web crawler or a distributed inference task—they assume legal responsibility for that code's execution - not the user in premises where it got executed. If an ISP were to recklessly deploy a malicious workload (e.g., a DDoS script or illegal surveillance tool), they cannot claim ignorance; their digital signature is on the deployment manifest, and they are the legal entity liable for the damages. This ensures that the distributed computing power of the network is not an anonymous botnet, but a regulated fleet of resources where every

cycle of execution is attributable to a licensed operator.

3.4 Data activity: aggregating the commons

Data activity encompasses the collection, anonymization, and sale of user telemetry (user data). User data has special support in the form of **Offers for Data (OfD)** because, unlike other tasks, it is not executed on the user's device (the edge), but is instead collected server-side.

ISPs announce their data collection capabilities through OfD: structured declarations posted to a protocol-managed marketplace, specifying what data types they can collect, at what granularity, with what privacy guarantees, and at what price per unit (e.g., per thousand anonymized records, per gigabyte of aggregated telemetry).

Buyers emit **Requests for Data (RfD)** referencing specific OfD. An RfD is a cryptographic purchase order: "I commit to fund the collection of X units of data type Y from ISP Z at the offered price."

The ISP, upon receiving an RfD, activates the corresponding data collection workflow. Data collection happens server-side at the Site operator operating the given application (eg. Youtube). These sites observe application-layer telemetry—server-side logs, API call metadata, anonymized user interactions, request patterns, and session aggregates—for the traffic they serve. The ISP has verifiable rights to collect and aggregate this data from sites for the users the ISP "owns" (this is determined from the metering dhata). The ISP aggregates this server-side observability across all sites serving its users, applies privacy-preserving transformations (anonymization, aggregation, differential privacy), and packages it into protocol-compliant datasets.

The collected data is not delivered to the buyer. Instead, it is fed into a public user dataset. The buyer is merely someone motivated to have a fresh portion of specific data present in this shared, public user dataset.

Payment flows through the ESP (Master Clearing Fund Account): the buyer's RfD is matched against the ISP's delivery attestation—a cryptographic proof confirming dataset delivery to the public repository with payload hashes, timestamps, and quality metrics. The ESP verifies the match and transfers stablecoins from the buyer's account to the master clearing fund account. Fulfilling these requests makes the ISP eligible to receive a regular "dividend" from the master clearing fund which is paid from RfD service revenues.

3.5 Metering and scraping architecture for Service Activities

The protocol employs a **pull-based scraping model** to collect metered activity from CPE devices.

When a CPE device witnesses economic activity — an RfS emitted by the browser - it logs the event locally in a secure, append-only ledger backed by the device's hardware enclave. These logs include timestamps, correlation keys, ISP identifiers, app/job identifiers, and cryptographic signatures.

Rather than the CPE pushing meters directly to its parent ISP, the protocol mandates that meters are **pulled by isp scrapers**. Specifically, CPE devices accept meter requests only from

scraper nodes whose identity differs from the device's parent ISP. If the CPE device were to push via the parent ISP, the ISP could selectively suppress events that trigger payment obligations or usage accounting, and neither the user nor downstream applications would have independent evidence of the suppression unless they could inspect the device's local logs.

A global registry, maintained on-chain or as a protocol-provided service, announces the set of active ISP scraper identities. Each CPE device subscribes to this registry and enforces IP whitelisting: it will respond to meter requests only from scrapers listed in the registry and explicitly excluding its own parent ISP's scraper.

The ESP metering infrastructure can be understood as an architectural inversion of the RPC server pattern common in permissionless blockchain systems such as Ethereum and Solana. In traditional blockchain architectures, users submit transactions to the network via RPC (Remote Procedure Call) servers. The data flow is push-based: the user constructs and signs a transaction locally, then actively transmits it to an RPC endpoint, which relays the transaction to validator nodes for inclusion in the canonical chain. The user relies on the RPC server to faithfully relay the transaction without censorship, modification, or undue delay. While users can mitigate RPC-layer censorship by operating their own full node or by switching to alternative RPC providers, the default interaction pattern assumes trust (or at least hope) that the chosen RPC server will act honestly. RPC servers lower the barrier to blockchain participation by abstracting away peer-to-peer networking, mempool management, and state synchronization, making transaction submission accessible to lightweight clients.

ESP's scraper nodes serve an analogous intermediary role—they bridge the gap between edge devices (CPE) that generate metered economic events and the blockchain validator nodes that record those events in the canonical ledger. However, the data flow is inverted: rather than CPE devices actively pushing events to a relay, CPE devices passively log events locally in a secure, append-only ledger, and scraper nodes actively pull those logs from the devices and submit them to the blockchain on the devices' behalf.

While the pull-based model securely decouples event submission from the parent ISP, the actual operational flow of scraping is propelled by those with the most direct economic incentive: the Site Operators.

Because Site Operators interact with the user's browser, they are the first to receive the application-side Request for Service (RfS). However, they know they cannot claim disbursement until the corresponding CPE-signed meter log appears on-chain. Therefore, it is the Site Operators who actively drive the scraping process. Upon serving an RfS, the Site Operator identifies the user's parent ISP and consults the global protocol registry to select a scraper node operated by a different, competing ISP. The Site Operator then dispatches a targeted "wake-up" signal to this selected scraper, notifying it that uncollected economic activity is waiting at a specific CPE endpoint. This event-driven approach relieves scrapers from having to continuously and blindly poll millions of edge devices, optimizing overall network efficiency.

To prevent signaling floods and redundant scraping operations, the protocol relies on a cooperative delay mechanism among applications. Because a CPE device maintains a single, unified append-only log of all user activity, a single scrape operation pulls the pending RfS records for all applications the user has recently consumed. Consequently, applications are designed to employ

a strategic waiting period before actively signaling a scraper. This creates a functional "game of chicken" among recently used services: an application will hold off on requesting a scrape in the highly probable event that another service—perhaps one the user interacted with moments earlier, whose patience has expired—will trigger the scrape first. When that earlier application inevitably triggers the pull, the entire batch of recent user activity is submitted to the blockchain, securely placing everyone's RfS events on-chain with minimal redundant overhead.

3.6 Consensus, voting power and proof of IP endpoint ownership

A critical primitive underpinning this system is the provable notion that an ISP "owns" a given IP endpoint. Metered volume of economic activity is sampled from blockchain data based on RfS events and determines voting power for individual ISPs in regular intervals.

Concretely, every RfS posted to the protocol includes an ISP identifier (the parent ISP of the originating CPE). The protocol maintains a rolling window (e.g., the past 30 or 90 days) of activity volume per ISP, denominated in a normalized metric such as total stablecoins cleared or total RfS events successfully matched and settled. Each ISP's voting weight in governance proposals is proportional to its share of this activity.

This mechanism ensures that ISPs with large, active user bases—those actually moving value through the network—have proportional consensus weight. It prevents dormant or Sybil ISPs from accumulating power, and it naturally adjusts over time: if an ISP grows its subscriber base or increases usage density, its governance weight rises; if it shrinks or becomes inactive, its weight decays.

Governance decisions enacted through this voting system include protocol upgrades (e.g., deploying new smart contract logic, adjusting settlement parameters), fee structure changes (e.g., the percentage retained by the clearing fund vs. passed to apps), dispute resolution policies, and the onboarding or offboarding of stablecoin issuers. Because ESP operates as either an L1 blockchain or sovereign rollup, these decisions are executed as on-chain code changes voted on by ISP-operated validator nodes. There is no external foundation or committee with veto power; the protocol is governed by those who sustain it through actual usage.

This "proof of ownership" notion extends beyond ESP and has wider ecosystem reach and implications: it grants ISPs permission to monetize user-associated data in external protocols, provided they can demonstrate ownership of the originating endpoint. Consider the case of a Request for Data (RfD) where a third-party data consumer wishes to purchase anonymized user behavior metrics or engage in bandwidth sharing (e.g., decentralized web crawling networks like Grass). The ESP protocol enforces that only the ISP cryptographically linked to the specific CPE generating that traffic can successfully authorize and settle the RfD. If ISP A attempts to sell data generated by an IP endpoint belonging to ISP B, the transaction will fail to match on-chain because ISP A lacks the required cryptographic proof—specifically, the device-level signatures—from that endpoint. Consequently, an ISP is strictly constrained to monetizing only the data and bandwidth explicitly generated by its own subscriber base.

As the network grows and the governance process becomes more complex, maintaining active participation from all ISPs could become challenging, particularly for smaller operators with limited resources. To address this, the protocol anticipates the eventual need for a delegation

mechanism. This would allow smaller ISPs to delegate their accumulated voting power to larger, more active ISPs or specialized governance entities whose interests align with theirs. Liquid delegation ensures that the collective voting weight of smaller participants is not lost due to voter apathy or operational constraints, while still maintaining the fundamental principle that governance power is strictly derived from actual network usage and verified IP endpoint ownership.

3.6.1 Power of Consensus for off-chain governance

One of the most profound innovations of the ESP architecture is the elevation of *revenue allocation* to a consensus-level primitive. In traditional commerce, "who gets paid" is a matter of legal contracts, enforced by courts and subject to friction, breach, and opacity. In the ESP model, "who gets paid" is a matter of protocol consensus, enforced by cryptography and subject to immediate, immutable execution.

The shift from "Contract Law" to "Protocol Law" fundamentally alters the power dynamics of the digital economy. It allows us to decouple the *operation* of software from the *monetization* of software, while maintaining a rigid, unbreakable link between the two. This decoupling is the mechanism that allows the "Software as a Franchise" model to function without a central authority.

This deterministic revenue control provides a powerful mechanism for exerting *off-chain pressure*. Because the flow of capital is inextricably bound to protocol consensus, the protocol itself functions as an enforcement layer for off-chain agreements, regulatory compliance, and governance frameworks. In the legacy web, mitigating non-compliant behavior—whether involving the distribution of malware, persistent intellectual property infringement, or violations of local data sovereignty laws—often requires fragmented enforcement actions across various hosting providers, DNS registrars, disjointed payment gateways and various legal authorities. Entities can frequently circumvent these measures by migrating to more permissive infrastructure or alternative payment processors, allowing their operations to continue largely uninterrupted.

Under the ESP architecture, this form of jurisdictional and infrastructural arbitrage is structurally neutralized. The revenue stream is not an external, easily swappable service decoupled from the application; it is an intrinsic operational requirement for the application's distribution over the network. If an entity violates the legal requirements of their matched jurisdiction, breaches a franchise agreement, or engages in established non-compliant behavior, the network of infrastructure providers can, via formalized governance mechanisms, sever the routing of funds to that entity's protocol identity. This creates a robust economic compliance mechanism. It empowers local jurisdictions, regulatory bodies, and decentralized governance structures to enforce off-chain rules with on-chain finality.

The threat of immediate, mathematically guaranteed cessation of revenue acts as a primary deterrent, often enforcing compliance more efficiently than traditional, protracted legal injunctions. By elevating revenue allocation to the consensus layer, the protocol ensures that software operates not in isolation, but in strict, enforceable adherence to the legal and regulatory frameworks of the physical jurisdictions it serves.

For example consider a case where an application is definitively identified as distributing malware

or facilitating coordinated malicious activities, network governance can execute a protocol-level revenue cessation. Rather than attempting the technically complex task of blocking dynamic IP addresses or revoking domain names, the infrastructure providers sever the application's underlying financial utility. The operation is thereby economically starved, rendering its continued deployment structurally unsustainable.

3.7 Jurisdictions

The introduction of jurisdiction as a core concept in the ESP protocol addresses a fundamental challenge in global digital systems: the tension between borderless technology and bordered laws. Internet services operate across countries, but legal responsibilities, regulations, and enforcement mechanisms are tied to specific jurisdictions. Without clear rules on where an activity "happens," disputes can lead to complex cross-border conflicts, high compliance costs, and uncertainty for all parties. By embedding jurisdiction matching into the protocol, the aim is to create predictability and reduce these risks, making the system more robust for real-world adoption.

To resolve this, the ESP protocol introduces the concept of **Jurisdictional Unambiguity**. This is a protocol-level constraint that explicitly aligns the flow of digital value with the boundaries of physical law. The core axiom is: **Economic settlement on the protocol is only possible within a matched jurisdiction.**

In practice, this means the protocol's settlement engine will only disburse funds for a Request (RfS, RfD, or RfJ) if the corresponding Offer (OfS, OfD, or OfJ) originates from a provider legally domiciled in the same jurisdiction (Country) as the consumer. A German user's request for service must be matched by an offer from a German-compliant site operator; a US-based data buyer can only purchase data via offers compliant with US law.

By enforcing jurisdiction matching at the atomic level of the transaction, we eliminate the complexity of cross-border legal conflicts. An ISP operating in France does not need to worry if they are inadvertently violating the laws of Brazil or Singapore. They serve French users under French law. The protocol turns compliance from a complex, fuzzy risk management problem into a binary, deterministic state. If the jurisdictions don't match, the transaction simply does not clear.

This mechanism gives nation-states enforceable oversight over their digital economies. If a country passes a law regarding data privacy or digital taxation, that law is naturally enforced because the economic activity of its citizens is processed by entities within its legal reach. This restores the social contract between the state and the market, ensuring that the digital economy contributes to the local society rather than extracting value to offshore havens.

For the ISPs and Site Operators—who are physical entities with assets and employees—this provides a liability shield. They are not acting as global intermediaries for unknown traffic; they are strictly local businesses serving a local market. This clarity reduces legal exposure and insurance costs, making it safer for mainstream businesses to participate as infrastructure providers.

3.8 Roaming

ESP does not model end users as first-class protocol participants. All protocol interactions occur between ISPs, other ISPs, and applications. In particular, the protocol does not require a user account, a user wallet, or any user-signed on-chain message. A user's relationship with an ISP (authentication, billing, KYC, customer support) remains off-protocol and continues to use the ISP's existing systems.

Nevertheless, users are physically mobile. A user may leave their home network and connect from a third-party access network such as a hotel or coffee shop. In the default case, nothing special is required: the *parent ISP of the originating CPE* (the access network) is the consumer-of-record on ESP, and the jurisdictional matching rules ensure that the local Site Operator is the legal and operational counterparty for the traffic. The access network can then recover costs from the user off-protocol (e.g., captive-portal payment, bundled Wi-Fi pricing, or venue-sponsored connectivity).

This section describes an *optional* extension for cases where the user is presented an option (eg. captive-portal option) to bill user's native ISP. The serving access network and the local Site Operator remain the entities responsible for service execution and local compliance; a foreign ISP merely agrees to fund the clearing obligation for a bounded session.

3.8.1 Roamed RfSs

Roaming is implemented entirely through ISP-to-protocol mechanics. The user only participates at the captive portal by presenting a credential issued by their home ISP. The credential can be a simple ISP-signed voucher, or it can embed a zero-knowledge proof of an active subscriber relationship (to avoid revealing a stable subscriber identifier to the access network). In either case, the output of the captive-portal flow is a **sponsorship authorization** that the access network can attach to metered activity.

Operationally:

1. The user joins the access network and is redirected to a captive portal.
2. The portal offers "Pay locally" or "Use home ISP contract." If the user chooses the latter, the portal verifies a **sponsorship authorization** issued by the user's home ISP (the *payer ISP*).
3. Once verified, the access network binds the authorization to the session (e.g., MAC address, DHCP lease, or tunnel endpoint) and enables Internet access.
4. As the user consumes applications, the browser continues to emit standard RfS events to the local CPE. The CPE logs these RfS events as usual, but additionally attaches the sponsorship fields.
5. Scrapers pull the meter log from the CPE and submit the extended RfS records on-chain. Settlement debits the payer ISP's protocol balance for those sponsored RfS records, while jurisdictional matching and operational liability remain determined by the local access network and local Site Operators.

Roaming does *not* change where an activity "happens." Jurisdictional matching remains anchored to the local access network and the local Site Operator serving the request. The payer ISP is merely funding a local transaction; it does not become the serving provider and does not inherit operational or compliance responsibility for the traffic.

Similarly, governance sampling and proof-of-endpoint-ownership remain anchored to the originating CPE's parent ISP, not the payer override. Sponsored RfS records still constitute metered activity observed at the last mile, and therefore still count toward the access ISP's endpoint ownership proof and activity volume metrics. The sponsor mechanism only affects *who replenishes the clearing obligation*, not *who owns the endpoint* or *who served the jurisdiction.*

3.9 User data dataset

User data dataset is a dataset managed by ISP collective. These consist of privacy-preserving, anonymized aggregates derived from server-side telemetry. User data datasets are populated through **Data activity (RfD)**. When external entities submit RfDs, ISPs activate server-side collection workflows (as described in the Data activity section), aggregate telemetry from application sites serving their users, apply anonymization and differential privacy guarantees, and contribute the results to ESP-managed public dataset repositories. The protocol enforces data provenance, privacy guarantees, quality standards, and open access. User data products are freely accessible to anyone: AI training pipelines, market research firms, product analytics platforms, or advertising optimization engines can all access the same aggregates, eliminating the proprietary data moat that currently advantages hyperscale platforms.

The ISP protocol maintains a dedicated account type called the **User dataset maintainer account**. This account is bound to a legal entity responsible for operating the infrastructure that stores, serves, and refreshes the user data dataset, and it exists to cover the real operational costs of maintaining that commons (storage, delivery, compliance audits, and reliability).

This introduces a practical amount of centralization, and that is acceptable for two reasons. First, decentralized collective of ISPs (under consensus) still control the signal: they facilitate the data collection itself, so the maintainer cannot fabricate or withhold data without the ISPs noticing because the input stream originates from ISP-operated sites and devices. Second, the maintainer account is governed by ISP consensus: ISPs collectively decide which legal entity holds the User dataset maintainer account. If the maintainer misbehaves, overcharges, or fails to serve the dataset, ISPs can vote to appoint a different entity and redirect both payments and data flow accordingly. In effect, ISPs can stop paying and stop sending data to the old maintainer, preserving operational continuity while preventing lock-in.

Dataset is intentionally designed as raw, minimally processed assets. ESP does not prescribe specific use cases or derivative products; instead, it provides the economic and technical substrate for third-party companies to build businesses around these datasets—indexing services, recommendation engines, predictive models, or vertical-specific analytics platforms. The protocol's role is to ensure data provenance, enforce privacy guarantees, coordinate population activities, and maintain open access; the value-add layer is left to market participants.

The user data dataset follows a distinct economic loop (due to its sensitive nature it is a "snowflake" in the whole concept). The dataset is free to access for everyone, but the maintainer

incurs real operational costs. Those costs are covered from the master clearing fund via the User dataset maintainer account, so the expense is socialized across the network. ISPs set OfD prices for RfD activity such that the clearing fund remains net positive when RfDs settle; the protocol therefore converts demand for user data into surplus that sustains the commons and earns net income to master account. External consumers who build businesses on top of the dataset have a direct incentive to keep RfDs flowing: they top up accounts and emit RfDs to trigger data collection and keep dataset fresh.

The open nature of the dataset engenders a cooperative funding paradigm akin to the financing of public infrastructure. Because the utility of the dataset is non-rivalrous and non-excludable, the operational costs of its maintenance—incurred via the emission of RfDs—can be distributed across a heterogeneous coalition of beneficiaries. This structure incentivizes the formation of multi-stakeholder consortia, where governments, non-profits, and commercial entities effectively pool capital to subsidize the requisite data collection. By coordinating to underwrite these costs, participants socialize the preservation of the digital commons, decoupling the sustainability of the dataset from the balance sheet of any single centralized entity.

As reminder note, other commons datasets (web index, mapping, network telemetry) are primarily maintained through RfJ-funded jobs. Applications that depend on these datasets are incentivized to commission continuous crawling, mapping, or measurement work via RfJ so the datasets stay fresh and comprehensive. ISPs execute the jobs, get paid via ESP settlement, and contribute outputs to the repositories owned by a company who requested the job execution.

3.10 ESP marketplaces

The protocol operationalizes economic coordination through two distinct marketplaces, each structured around the Offer/Request pattern but serving different roles in the ecosystem:

The edge Jobs Marketplace serves as ESP's primary interface to the broader decentralized compute ecosystem. ISPs advertise their distributed compute capabilities via Offers for Job (OfJ), specifying job types supported, available capacity, geographic distribution, pricing, and SLOs. External protocols and entities— application builders, research institutions, or any party requiring edge compute—browse available OfJ and submit Requests for Job (RfJ) to commission work. The marketplace is permissionless and competitive: any ISP can post OfJ entries, any entity can submit RfJ, and settlement flows automatically through ESP as jobs complete and proofs are verified.

The User Data Marketplace is an ESP-native marketplace where the protocol takes full responsibility for the data lifecycle. Crucially, while the raw telemetry and usage data originates from the applications running on infrastructure managed by Site Operators, the protocol grants the exclusive right to collect, aggregate, and monetize this data to the ISPs. ISPs perform this function on behalf of the Site Operators, but strictly and exclusively for the subset of users who are their direct subscribers. ISPs publish these user data aggregation capabilities via Offers for Data (OfD), declaring data types available (e.g., anonymized search queries, video consumption aggregates, session quality telemetry), granularity, privacy guarantees (differential privacy parameters, aggregation thresholds), pricing per unit, and delivery terms.

Both marketplaces operate as on-chain or protocol-adjacent registries (discoverable via blockchain

queries or marketplace front-ends), ensuring transparency, auditability, and permissionless participation. ISPs post capabilities; external entities commission work by submitting requests; ESP handles matching, settlement, and enforcement. The fundamental difference is governance and integration: the Jobs Marketplace is a neutral coordination layer enabling external protocols to build their own data products, while the User Data Marketplace is a fully integrated ESP service producing protocol-managed public dataset. This layered approach allows ESP to own the user data product (where protocol-level privacy and governance are critical) while remaining agnostic and enabling for the diverse ecosystem of commons initiatives that can leverage ISP infrastructure via RfJ.

3.11 Fund raise accounts

The protocol introduces a native primitive for capital formation designed to align the interests of builders and early backers without creating permanent rent-seeking equity structures. This mechanism allows founders to raise upfront capital from retail participants to fund development, securitized against the future revenue of the application itself.

Emission and Sale: Founders create Fund raise account and define a **Maximum Cap** (e.g., $5M) of this account and emit a fixed supply of **Tokens** that represent a claim on future revenue in form of proportional claim on balance of this account. These tokens are sold to the public to raise initial funds.

Revenue Routing: When the application goes live and begins processing Requests for Service (RfS), the protocol's settlement engine enforces a three-way split of each reibrusment:

- **Operational Revenue** to the ISP/Site Operator.
- **Royalty Revenue** to the Founder Account.
- **Fund raise Revenue** to the Fund raise Account (e.g., 20% of the transaction value).

Yield Claiming: Holders of Fund Raise Tokens can claim their accrued share of the revenues from the Fund Raise Account at any time. The token acts as an amortizing cashflow claim rather than a burnable asset. The claimed value is a proportional share of the accumulated funds in the Fund Raise Account, bounded by a strict per-token lifetime cap.

$$\text{Payout} = \text{Accured Balance} \times \frac{\text{Tokens Held}}{\text{Total Supply}}$$

Upon claiming, the token is *not* burned. The holder simply withdraws the yield generated so far, and the token remains in their wallet, continuing to accrue future revenue until it reaches its individual maximum payout.

This creates a self-regulating, fundamentals-driven market. Investors do not have to completely exit the ecosystem to realize returns; they receive continuous cashflow as the application scales. These tokens can be traded on the open market at any time, allowing secondary markets to price them rationally based on the discounted value of their remaining unpaid cap.

These Fund raise Tokens are not meant to represent equity nor governance. They convey no voting rights and no perpetual claim on profits. They represent a "temporary burden"—a finite

debt obligation incurred to bootstrap the application. Once the total inflows to the Fund raise Account reach the maximum account cap, the routing logic permanently terminates. The application's debt obligation is fulfilled, the Fund raise Account is dissolved, and the revenue cut is removed from its cost structure. The tokens themselves, having paid out their maximum allotted cashflow, become economically exhausted and hold no further value. This ensures that the cost of capital is paid exactly once and does not become a permanent tax on the ecosystem.

4 Blueprint for Next Gen Web3 Applications

The prevailing narrative of Web3 has often forced a painful trade-off between decentralization and user experience. "On-chain" applications, constrained by the latency of global consensus and the friction of wallet interactions, struggle to compete with the seamless, instant performance of centralized hyperscale alternatives and solutions offered so far have failed to match the operational velocity of the incumbents. A decentralized Twitter that takes ten seconds to post is not a competitor; it is a downgrade.

This blueprint assumes, that "Web3" is invisible—a settlement layer that hums in the background while the foreground experience rivals vertically integrated Web2 applications. This requires abandoning the dogma that "everything must be on-chain" and instead embracing a hybrid architectures: trustless settlement combined with trusted, high-performance local execution.

Focus is on decoupling *data agency* from *application execution*. It envisions a network where applications run with the full speed and power of modern datacenters and vertical integration on datacenter level — while remaining structurally decentralized in their operation and economically aligned with their creators through the protocol. It is a vision where the datacenter is not dismantled, but federated.

At its core, this architecture embraces a following principle:

Applications must be global in perception but radically local in execution.

To the user, the application presents as a cohesive monolith—a seamless global brand with a singular identity. Beneath this surface, its physical reality is deliberately fractured. Operations are anchored to diverse, sovereign jurisdictions, processed by local entities, and bound by local laws. By grounding the digital application in the tangible boundaries of human geography, decentralization ceases to be an artificial protocol constraint. It becomes an organic, natural outcome as inescapable property of the network.

Ultimately, the aim is to enable emergence of new generation of builders who want to challenge incumbents. The protocol gives them the instruments to do so—providing the economic primitives, distribution leverage, and monetization models needed to compete with hyperscale platforms.

4.1 Anycast: The foundation of Distribution Leverage

Central to the architecture is **IP Anycast**, an addressing scheme that allows an ISP to translate its inherent routing authority directly into economic leverage.

Applications utilize global Anycast VIPs, enabling simultaneous prefix advertisement from diverse network edges delegating the selection of the serving instance to the routing layer.

Anycast grants the ISP "distribution leverage." Because the ISP controls the routing tables for its own subscribers, it dictates where packets destined for that Anycast IP actually go and thus who (which site operator) will earn money by processing user's request. This creates a key economic choice for the ISP:

- **Delegated Routing:** If an ISP chooses not to host a specific application (not be a site operator for given application), they route traffic to an external operator. This decision

still preserves economic agency nevertheless: the ISP can steer this volume to specific partners—upstream providers or peer networks—who are willing to pay for the traffic feed or share a portion of the resulting operational revenue, effectively monetizing the routing decision itself.

- **Local Execution:** The ISP instantiates the application locally and advertises the Anycast prefix within its own autonomous system. By keeping the traffic on-net, it directly captures the full operational revenue, maximizing the value extracted from their subscriber base.

4.2 Software as franchise

The architecture introduces a fundamental split between product building and the operations, drawing some similarities to a business franchise model rather than the monolithic SaaS model of Web 2.0. In this "Software as a Franchise" paradigm, the roles of the Application Builder (Founder) and the Site Operator (ISP) are distinct, specialized, and economically aligned by the protocol. The ISP protocol offers specific primitives that enable this structural split in a tangible and enforceable way. Central to this is the **Founder Account** and the configurable *royalty* share applied when disbursing RfS payouts.

4.2.1 Founders

The Founder Account is the account that:

- Is tied to application domain name (DNS) and thus to the legal entity owning that domain.
- Is the only actor that can configure the application's royalty policy in the ISP Protocol.
- Receives royalty payouts when economic activity clears (economic activity is always unambiguously attributable to the application via its DNS domain name).

While the application code itself can be (and should be) open source, the *monetization* of that code is enforced by the protocol's settlement layer. When an ISP operates a site and serves a user, the economic activity is recorded on the blockchain via Request for Service (RfS) events. The settlement engine then automatically splits the payment:

1. **Operational Revenue** goes to the Site Operator for the physical work of running the infrastructure and serving the traffic.
2. **Royalty Revenue** is automatically routed to the Founder Account as defined by the protocol configuration.

Incentives in such setup are fully aligned. Founders are incentivized to build and maintain high-quality open-source software because the protocol guarantees they capture a share of the value it creates, regardless of which ISP operates the infrastructure. The application infrastructure becomes a federated resource, built and maintained by the very entities that connect users to it, while the product vision remains coherent and driven by the founders.

This alignment extends beyond mere operations into distribution and growth. Because ISPs capture operational revenue from every unit of service consumed by their subscribers, they acquire a direct financial stake in the application's success. This incentivizes ISPs to leverage their existing

customer channels to market and bootstrap adoption for high-quality applications—effectively acting as a distributed sales force. Just as a franchise owner promotes the brand locally, ISPs promote the application to drive network traffic and utilization.

This structural separation of concerns significantly alters the economics of application development, particularly for smaller, builder-centric teams. By delegating the operationally intensive components—infrastructure scaling, regulatory compliance, and end-user support—to a distributed network of Site Operators, Founder teams can direct their resources entirely toward product engineering and innovation. This model reduces the organizational overhead typically required to service a global user base, allowing compact engineering teams to deploy planetary-scale applications. The result is a competitive landscape where efficiency and product quality, rather than operational scale or capital depth, become the primary determinants of market success.

4.2.2 Site Operators

Site Operators represent a fundamental departure from the anonymous infrastructure providers typical of Web3 (validators). In traditional decentralized networks, validators are often faceless entities running code in obscure data centers, shielded by the pseudonymity of a cryptographic address. In the ESP model, Site Operators are registered legal entities—typically ISPs or their affiliates—operating within specific physical jurisdictions with full accountability.

This distinction transforms the Validator from a mere instruction processor into a full-fledged legal shield for the entire ecosystem. By leveraging IP Anycast addressing, traffic is deterministic and routed to infrastructure located within the user's specific legal jurisdiction. This creates an unambiguous link between the user and the physical infrastructure, making legal responsibility local, enforceable, and tangible. A user in the Czech Republic is served by a Czech Site Operator under Czech law, operating on Czech soil. This structural localization avoids the complex cross-border jurisdiction issues that plague globalized platforms, where enforcing local laws against a foreign entity is often impossible.

The Site Operator acts as the primary regulatory interface. They handle KYC/AML compliance, tax reporting, and dispute resolution for their local market, and they are compensated for this heavy regulatory burden via the protocol's operational revenue share. Identity verification is strictly bound to this local provider: since the ISP already maintains a verified billing relationship with the subscriber, they can perform checks once and issue Zero-Knowledge (ZK) proofs to applications. This allows global applications to remain compliant with local regulations without ever holding sensitive personal data themselves. The Site Operator absorbs the "messiness" of the real world, allowing the application layer to remain pure and globally consistent.

This transition to legally accountable operators resolves the fundamental "accountability gap" that has hindered institutional adoption of decentralized infrastructure. In a permissionless system run by anonymous nodes, there is often no recourse for malicious behavior beyond protocol-level slashing. By anchoring the network in licensed Site Operators, the protocol introduces a layer of real-world recourse.

Consider a Decentralized Exchange (DEX): In the ESP model, the "Validator" is not just a server confirming transactions, but a regulated local entity serving users in a specific jurisdiction.

This operator is responsible for handling the entire legal interface—KYC/AML compliance, tax reporting, and dispute resolution—and is compensated for this regulatory burden. If a user has a complaint, they do not shout into the void of a DAO governance forum; they contact the specific Service Provider (ISP/Validator) who served them. Identity verification (KYC) is similarly bound to this provider: since the ISP already maintains a billing relationship with the user, they can perform checks once and issue Zero-Knowledge (ZK) proofs to the application layer, allowing the DEX to remain compliant without ever holding the user's personal data. The "Validator" thus becomes the legal shield for the protocol, handling the messy reality of local law while the protocol handles the clean logic of global settlement.

4.3 Data Architecture: Replication, not Consensus

Consensus is and will remain expensive, and the blueprint argues that the bulk of user's agency and sovereignty should be built on replication principles, not consensus principles. Early decentralized architectures often encountered scalability bottlenecks by attempting to persist high-velocity application data directly on consensus layers. To resolve this, the proposed framework establishes a structural distinction between *Digital assets* and *User data*.

Consensus is for Digital assets: Applications rely on the blockchain for data that requires absolute global consensus: primarily money, digital property and sovereign identities. These are high-value, low-velocity state transitions where consensus based verification is paramount.

Replication is for User data: The vast majority of internet traffic—videos, social media posts, comments, photos, and game state—does not require global consensus. It requires high availability, speed and user agency. This data is stored in user-owned fashion using non-blockchain, replication-based technologies.

The architecture assumes emergence of decentralized storage layer where data is addressed by some form of public handle (Content ID). In this blueprint, the user retains the authoritative handle to their data. The application does not "ingest" the data into a proprietary silo. Instead, the ISP operates a datacenter-local infra of the underlying decentralized storage network (present and future technologies not dissimilar to Filecoin/IPFS or others) co-located within the same datacenter as the application instance. Data replication is handled natively by the storage protocol, syncing content to the ISP's local node. The application then accesses this data directly via the high-speed local interface of the co-located storage node. This ensures that while execution benefits from datacenter-grade locality, the underlying data remains on the open, user-controlled storage network.

This model provides specific degrees of freedom for legal compliance without granting absolute platform power over user state. A site operator can:

1. Decide not to replicate certain data into its site (e.g., categories of content illegal in its jurisdiction).

2. Decide not to propagate or display certain content (e.g., local moderation rules, court orders, or safety policy).

However, the operator cannot delete user data from existence. Because the data layer is replication-based and user-owned, deletion is not a unilateral power of any site. At most, a site

can refuse to host, replicate, or serve particular content within its jurisdiction, while users retain the ability to carry their data elsewhere. This is the key balance: local compliance is achievable, but global lock-in via data captivity is structurally resisted.

4.4 Vertical Integration: The Datacenter as a First-Class Citizen

A fundamental limitation of current Web3 architectures is the absence of the "datacenter" as a coherent unit of execution. In the peer-to-peer paradigm, nodes are treated as isolated islands, often connected over high-latency public networks. This fragmentation makes it impossible to build complex, data-intensive applications that require tight coupling between compute, storage, and caching layers.

To match the performance of hyperscalers, the notion of the datacenter—analogous to an AWS Virtual Private Cloud (VPC)—must be retained and elevated. Full-blown internet applications require the colocation of multiple moving parts within the same high-bandwidth local network. An AI-powered search engine, for instance, needs its vector database to sit millimeters away from its inference engine, not on a different continent.

In this blueprint, the "Site" is the atomic unit of deployment. The ultimate goal for this architecture is that there is **zero limitation** on vertical integration within the datacenter. The Site Operator can run the exact same high-performance stacks as AWS or Google—Redis caches, Postgres clusters, vector databases for LLMs—without the overhead of blockchain transactions for every read/write. This ensures the User Experience (UX) is indistinguishable from, or superior to, centralized alternatives.

4.5 Summary of key design principles

The next generation of Internet applications must break the false dichotomy between the seamless user experience of centralized platforms and the sovereignty of decentralized protocols. Following core principles try to capture the key design principles for new applications:

- **Pragmatic Decentralization:** Advocating for a structural separation of concerns where "Founders" build the software product and "Site Operators" (ISPs) run the infrastructure. This operational layer is further dissected by jurisdiction, ensuring that physical infrastructure is always legally accountable to the local laws where it resides.
- **User-Controlled State:** All stateful data must remain in the hands of users. High-value assets (money, identity) live on consensus-based storage (blockchain), while high-velocity content (media, social graph) lives on replication-based storage. The application never "ingests" data; it merely views it.
- **Stateless Applications:** Applications must be architected as stateless logic layers that operate over user-controlled data. Because the application does not own the data, there is no lock-in; users can bring their data state to any compatible interface or operator.
- **Vertical Integration at Datacenter Level:** To match hyperscale performance, embrace the datacenter as a coherent unit of execution. Compute, caching, and database layers should be vertically integrated and co-located for speed, provided they remain operationally federated and run open-source stacqs.

- **Jurisdiction as a First-Class Constraint:** Treat jurisdictional variability not as an edge case, but as a core design constraint. Site Operators must have the flexibility to toggle features or filter content to comply with local regulations (e.g., GDPR, local moderation laws).
- **No Data Moats:** hoarding of data should not yield a competitive advantage. Foundational datasets (web indices, maps, user data aggregates) should be treated as public infrastructure—Data Commons—accessible to all builders, ensuring competition happens on service quality, not data monopoly.
- **Pay-As-You-Go Economics:** Replace the opacity of ad-funded models and the friction of subscriptions with granular, pay-as-you-go metering. Users pay only for the resources they consume (compute, bandwidth, data), creating a direct, transparent economic link between cost and value.

5 Blueprint for ISP-centric infrastructure

The evolution of the physical and logical architecture of the Internet, primarily the AI-led disruptive forces is what drives the economic and operational shifts proposed by the ESP protocol. The current Internet treats the ISP strictly as a dumb pipe—moving packets between a passive household router and centralized hyperscale clouds. Moving toward an ISP-centric infrastructure results in structural changes in roles and responsibilities of actors in the Internet.

5.1 Reimagining user premises - Active Edge Infrastructure

The technical foundation enabling ESP's metering, and distributed execution results in a profound reimagining of the role of edge and user premises infrastructure (households). It starts with the need for reliable, device-level metering of Request for Service (RfS) events at the user premises which is essential for tracking and settling pay-as-you-go application usage. An upgrade of a CPE device is required here which must act as an impartial witness to the user's consumption. Building on this core capability, the infrastructure naturally evolves to support Request for Job (RfJ) execution as well, enabling distributed edge workloads like web crawling or compute tasks to run on household hardware.

Together, these drive a reimagining of edge and household infrastructure into what we call Active Edge Infrastructure. While traditionally the user premises domain has been passive, just serving as entry points for data to reach the wider Internet, now it becomes an active layer that takes part in creating value, and running computation. By adding cryptographic tools, verifiable execution setups, and economic incentives, household devices change from simple endpoints into a connected network of useful nodes. This change taps into unused resources at the network's edge, supporting decentralized tasks, privacy-focused applications, and smooth service delivery, all while keeping user control and security intact.

5.1.1 Incentive inversion: why distributed apps win

ESP changes incentives for the actors who already sit at the last mile—primarily ISPs. In the current Internet, centralized SaaS and hyperscale cloud services concentrate execution and value capture in remote infrastructures. ISPs forward packets and collect a fixed connectivity fee; they have no structural path to participate in application economics beyond selling bandwidth.

In an ESP environment, distributed architectures as contrast with cloud centric architecures, expand the surface area where ISPs and other distributed actors can have a **stake** in the flow. The ISP motivation for pushing such architectures is twofold. First, certain applications are directly monetizable via Request for Job (RfJ) mechanisms or by earning native protocol tokens. Rewards are collected by the ISP and distributed to users via the connectivity bill (with the ISP retaining a percentage for management). Second, these applications drive households to acquire more powerful hardware, thereby increasing the pool of latent compute resources which the ISP can monetize by running more demanding jobs.

Once distributed applications are the established path for creating value, a flywheel emerges: better premises capability enables more valuable local workloads; more valuable workloads justify subsidies for better hardware; subsidies accelerate adoption; adoption expands the edge fabric and attracts more applications.

The accumulation of capable hardware at user premises is not solely the result of ISPs pushing to monetize the edge. It is equally driven by the users' own growing desire for resilience and independence. This trajectory finds a strong parallel in the energy sector, where a growing segment of users pursues self-sufficiency—solar panels, batteries, smart inverters—as a value in itself, not merely for cost optimization. A similar motivation drives the demand for "intelligence self-sufficiency": households capable of providing local inference, coordination, and privacy-preserving services even when the cloud is unavailable or untrusted. Crucially, once this powerful edge capability is deployed for personal autonomy, it creates a surplus resource that is often idle. Just as prosumers in the energy market are accustomed to selling excess power back to the grid, the owners of these new edge nodes will naturally seek to monetize their spare capacity, turning a sunk cost into a revenue stream.

Because ESP provides a policy and accounting substrate, this spare capacity can be pooled under explicit user constraints (schedules, resource caps, data scopes) and commissioned for external jobs (RfJ) without turning the household into an unmanaged botnet. This converts the aforementioned "intelligence self-sufficiency" into a dual-purpose asset: prioritizing the user's local needs first, while safely sharing any unused capability with the broader network.

Why this aligns with users: users also want "as local as possible" for privacy, control, and reliability reasons; they just cannot practically run everything locally today because deployment friction, heterogeneous hardware, and lack of standard runtimes push most services into centralized clouds. ESP makes the locality path commercially viable and operationally standardized. The ISP's incentive to activate locality (to have a stake) pulls in the same direction as the user's preference for local-by-default.

Contrast to SaaS/cloud: SaaS and hyperscale cloud incentives point the other way. Centralizing execution and data makes it easier for a platform to capture value, enforce lock-in, and compound proprietary datasets. That centralization structurally reduces local participation opportunities: the user becomes a thin client, the household becomes a dumb endpoint, and the last mile remains economically inert. ESP inverts this: value capture increases when more of the system can be decomposed into accountable local roles.

5.1.2 Open hardware ecosystem and the ESP software stack

ESP does not mandate a single device design or restrict participation to approved manufacturers. Instead, it defines functional and security requirements—secure boot, isolated execution, hardware-rooted keys where available, and sufficient compute/storage for edge workloads—that many OEMs can implement. Devices based on open platforms such as OpenWrt, or OEM firmware stacks conforming to ESP specifications, can participate. This openness prevents vendor lock-in and allows the edge fabric to evolve as hardware improves.

The capabilities that transform a standard router into Active Edge Infrastructure are delivered as an installable, open-source **ESP software package** maintained by a community-governed foundation or technical working group. Releases are versioned, signed, and distributed via standard update channels; devices verify provenance before installation and reject packages lacking valid cryptographic signatures.

The initial ESP package would include three modules:

RfS witnessing and scraping interface: implements the client-side metering logic for service activity settlement. The device logs RfS events in a secure, append-only ledger and exposes them to authorized scraper nodes operated by *other* ISPs under the pull-based scraping model (explicitly excluding the parent ISP to prevent suppression). The module enforces anti-replay constraints, validates signatures, and provides an auditable trail that users can inspect for billing transparency.

Edge runtime for RfJ workloads: enables execution of containerized jobs (web crawling, measurement, caching, inference) inside a locked-down runtime with signed images, SBOM verification, and strict resource policies. Outputs are attested with device identity so requesters can attribute completion and ESP can settle value share.

Local management dashboard: a reserved local UI (e.g., `192.168.1.1/esp-dashboard`) that exposes participation controls and transparency. It shows metering logs, running jobs, resource usage, scraper accesses, earnings attribution, and privacy parameters for any data aggregation workflows. Advanced users can inspect immutable image digests, SBOMs, and network egress policies; casual users get simple toggles and high-level summaries (bill offsets, uptime, and job schedules).

ISPs serve as distributors and onboarding facilitators (shipping preconfigured devices or assisting installation), but do not require privileged administrative access after setup. “Managed access” is deliberately narrow: ISPs can see opt-in status and aggregate capacity needed for marketplace SLOs, but cannot execute arbitrary commands, inspect user traffic, modify meters, or override user participation settings without invalidating attestation and losing settlement eligibility.

5.2 Distribution for national AI Infrastructure

To the extent that Artificial Intelligence matures from a novelty into a central engine of economic productivity, the perception of AI infrastructure is likely to shift from "commercial software" to "strategic national asset." While not every model will require such classification, governments and regulators may increasingly view the compute clusters, training datasets, and inference engines not merely as private products, but as critical infrastructure analogous to power plants, telecommunications grids, and highway systems. In this potential geopolitical reality, the concept of "Sovereign AI"—the ability of a nation to control its own artificial intelligence capabilities without total dependence on foreign tech giants—becomes a significant policy consideration.

The current hyperscale model is structurally ill-suited for this potential paradigm. A centralized AI model hosted in a massive data center in Oregon or Dublin, serving the entire globe via opaque APIs, presents a sovereignty risk in a world where AI is critical infrastructure. It creates a dependency where a nation’s critical information processing capacity is subject to the whims, laws, and outages of a foreign entity. Furthermore, if AI regulation (such as the EU AI Act) tightens, the "black box" nature of hyperscale AI services becomes a liability. Nations may need assurance that their citizens’ data is processed locally, that models comply with local cultural and legal standards, and that the underlying compute resources cannot be unilaterally severed.

This is where the ISP-centric model offers a decisive strategic advantage should this trend solidify. ISPs are, by definition, the most "jurisdiction-aware" entities on the Internet. They are licensed, regulated, and physically grounded within specific national borders and quite literaly in the soil.

They operate the fiber, the exchanges, and the local datacenters that constitute the physical internet of a country. By repositioning ISPs as the distributors and sometimes even operators of AI infrastructure, we align the technological topology of AI with the political topology of the world.

In the ESP framework instead of routing user prompts to a global API endpoint, the ISP can route, meter and settle usage of localized instances of foundation models within a domestic datacenters (or on the active edge). These models can be:

- **Localized Mirrors:** Licensed copies of global frontier models (e.g., Llama, Mistral) running on local hardware, ensuring that inference data never crosses national borders.
- **Nationally Aligned Models:** Fine-tuned versions of models that adhere to specific local laws, languages, and cultural norms, filtered and moderated according to local jurisdiction.
- **Strategic Compute Reserves:** A distributed fleet of GPUs managed by the ISP collective, available for national priorities (research, emergency response) during crises, much like a strategic petroleum reserve.

In the traditional software paradigm, the 'engine' was bundled inside the application code. In the AI era, applications are increasingly like electric appliances—a vacuum cleaner or a toaster—that are inert until plugged into a grid. The application defines the logic and the interface (the appliance), but the 'power' that drives it—the tokens generated by a foundation model—is drawn from the infrastructure.

The ESP is well suited for such separation - It provides the necessary distribution mechanism and primitives for consuming tokens from the "intelligence grid," where tokens are metered and the user plugs in whatever application they choose. This decoupling allows applications to be lighter, cheaper, and more diverse, while the heavy lifting of intelligence generation is handled—and metered—by the AI infrastructure. Moreover, this model solves the licensing deadlock: global model creators (Founders) are incentivized to allow their models to run in these restricted, sovereign environments because the protocol guarantees they receive a royalty for every single token generated. Sovereignty no longer means isolation; it becomes a compliant, monetized franchise of the global AI economy.

This would transform the ISP's role in AI economy from a passive pipe into a strategic partner for the state. Just as the state relies on utility distributor companies (electricity, water, gas) to keep the households on, it may come to rely on ISPs to keep the national intelligence infrastructure running. This alignment opens up vast new avenues for public-private partnership, where ISPs can access government subsidies, tax incentives, and guaranteed contracts to build out the high-performance compute infrastructure required for the AI era, all while providing a service that is faster, more private, and more accountable to the end-user.

5.3 Inside the Datacenter

A crucial questions is: can decentralized ISPs really match the operational efficiency of AWS? To answer this, we must contrast the prevailing cloud model with a new architectural approach which will be reffered to as **Portable Cloud Paradigm**.

In the traditional Cloud Computing paradigm, exemplified by hyperscalers like AWS, the cloud provider offers a proprietary catalogue of distinct infrastructure services (EC2, RDS, EKS, SQS). Developers build applications by deploying isolated chunks of logic into these managed silos. While this offers convenience and integration, it creates deep vendor lock-in; the application is not a self-contained unit but a dispersed set of configurations tied to a specific provider's API surface.

The **Portable Cloud Paradigm** inverts this model. Here, a neutral portable substrate - very likely a derivation of Kubernetes - acts as the true Data Center Operating System (DCOS). Middleware and infrastructure services—databases, message queues, caches—are not external proprietary services but are deployed directly alongside the application logic as Kubernetes Operators. In this architectural view, cloud capabilities are encapsulated within the application deployment itself, transforming infrastructure from a static destination into a portable, application-defined dependency bundle.

Attempts to do this are known from history but today, thanks to rapid reshaping of the power game thanks to the AI revolution, this approach deserves a second chance. Historically, this approach faced a fragmentation challenge. Third-party middleware authors (e.g., the creators of Kafka/Strimzi or Yugabyte) produced excellent individual operators, but they operated in isolation. There was no single entity ensuring that the Postgres operator played nicely with the Redis operator or the monitoring stack. Hyperscalers had the advantage of the "single umbrella"—they guaranteed that all their proprietary services integrated seamlessly, absorbing the complexity of interoperability.

With rapid advancements in AI based software engineering the "AWS DC automation" moat can be challenged. We can now leverage AI to replicate the hyperscaler's integration advantage with a meaninful amount of resources despite large horizontal size of service catalogue. AI enables the rapid generation, testing, and maintenance of a coherent "catalogue of operators" under a single umbrella. Instead of manually stitching together disparate open-source tools, the **Portable Cloud Paradigm** uses AI to maintain a unified suite of operators that are continuously tested to work well with each other.

By treating the entire infrastructure stack—from the database to the load balancer—as a unified software artifact managed by operators, key properties can be achieved: the portability of open source with the "it just works" automation of a managed cloud. This allows an ISP to run a complex, multi-service application with the same operational ease as a hyperscaler, but without building a proprietary service empire. The ISP simply runs the "Site Template," and the bulk of operational excellence is distilled into Kubernetes operators.

An application in this model is conceived as a portable "datacenter site" in the form of a deployable template that can be instantiated by many operator-grade datacenters and routed to by many ISPs, while remaining a single product defined by its builder. This paradigm effectively reimagines the application as a "datacenter-scale software package." Just as a package manager on a single machine resolves and installs all necessary libraries and dependencies, the Portable Cloud treats the entire infrastructure stack as a single, deployable unit. Kubernetes Operators and Custom Resource Definitions (CRDs) encode live operational knowledge and make operations reproducible and automated across different datacenters operated by different

teams. The database, the caching layer, the message queues, and the application logic are bundled together, pre-wired and pre-tested.

As the site templates become truly plug-and-play, the operational experience becomes frictionless: a builder publishes; the operator deploys and brings the site to life; probes verify Service Level Objectives (SLOs); and the site begins emitting proofs and earning. For the ISP, "rolling out" a new search engine or video platform becomes as operationally straightforward as installing a package: the complexity is internal to the bundle, while the interface to the datacenter is simple, standardized, and reproducible.

Hyperscalers also incur massive complexity costs because they solve for generic multi-tenancy—running millions of unrelated workloads side-by-side. In the ISP-Franchise model, this is simplified further: deployment is inherently single-tenant, an ISP runs a *specific* instance of an application for its *own* subscribers. This eliminates the hardest class of multi-tenancy security and isolation problems.

5.4 Site operator dynamics

Operating many sites creates compounding know-how. Each incident, rollout, and moderation workflow improves tooling and playbooks, lowering the marginal cost of the next deployment. Over time, operators specialize: some become excellent “finance operators” (strong KYC/AML, reporting), others “media operators” (moderation, copyright), others “AI operators” (GPU scheduling, model serving). This specialization is a healthy market outcome: applications can be present everywhere, but they are served by the operators best suited to the local execution and legal constraints. Because operators earn operational revenue per unit, they are directly incentivized to improve uptime, latency, and cost efficiency. At the same time, they retain hard power: they can refuse to deploy unsafe templates, delay risky upgrades, or route traffic away from misbehaving sites. This creates a credible feedback loop for founders: publish high-quality, operable, legally adaptable templates or lose on-net placement and thus performance and growth. The result is a market that rewards not only good products, but also good operational citizenship.

The operational role of the Site Operator (typically the ISP, or an ISP-affiliated entity) fundamentally transforms in the ESP ecosystem. Instead of merely transporting packets, the operator becomes:

- a multi-application hosting business,
- a fluid hardware orchestrator inside a jurisdiction-bound datacenter footprint, and
- the local legal and support interface for end-users.

5.4.1 Managing multiple applications as a portfolio

A capable operator does not run “one flagship app”; they run a *catalogue*. This diversification is rational: demand is volatile, and unit-based settlement means revenue is a stream of small events with real variance. By hosting many sites (search, video, social, AI inference, games, fintech, etc.), the operator reduces revenue concentration risk and improves utilization of fixed costs (racks, peering, on-call staff, compliance). Portfolio management becomes a first-class discipline:

- **Onboarding as due diligence:** before deploying a template the operator evaluates expected local demand, unit economics (payout per unit vs. cost per unit), and operational risk (attack surface, incident history, data sensitivity).
- **Fleet operations:** every site is integrated into the same observability and incident-response surface (metrics/logs/traces, standardized SLOs, canary rollouts, rollbacks), so the marginal operational burden of the next application is low.
- **Capacity-aware routing:** because Anycast makes serving location a routing choice, operators continuously decide which services to keep on-net and which to delegate upstream, based on latency, cost, and legal risk.

5.4.2 Hardware fluidity and datacenter repurposing

The operator's datacenter is a shared pool, not a collection of silos. Under ESP, repurposing is economically rewarded because improving cost-per-unit directly increases margin:

- **Burst absorption:** when one site experiences a spike (e.g., a local sports event driving video demand, or a new model launch driving inference), the operator can reallocate CPU/GPU capacity across sites instead of over-provisioning each site individually.
- **Off-peak monetization:** during troughs, the operator can spin down excess capacity from latency-sensitive workloads and redirect it to profitable background activities (RfJ jobs, batch indexing, training runs, data aggregation), turning idle capex into continuous yield.
- **Specialization and accelerators:** operators with GPUs, high-speed NVMe tiers, or specialized networking can preferentially host the sites that best exploit them (LLMs, vector search, transcoding), creating a market for "operator capability" rather than a race to the cheapest generic VM.

5.4.3 Scaling and managing the legal site

A "site" is not only compute; it is a *jurisdictional service perimeter*. As traffic scales, the operator scales the legal and compliance layer alongside the technical layer. Because the serving operator is a registered entity in the user's jurisdiction, obligations are local and unambiguous (lawful process, consumer protection, taxation, local record keeping). Crucially, compliance becomes reusable across the entire portfolio:

- **Shared compliance utilities:** KYC/AML checks, age-gating, sanctions screening, VAT/tax handling, and audit trails can be implemented once by the operator and exposed to many applications via standardized interfaces and zero-knowledge proofs.
- **Risk-tiering:** operators can classify applications into risk bands (payments-heavy, content-heavy, regulated verticals) and apply differentiated retention, monitoring, and escalation policies without requiring founders to rebuild compliance for every country.

5.4.4 Efficient local moderation and customization

Site templates are deployed with a strict separation between the founder-provided core logic and operator-owned local policy:

- **Local moderation:** operators build or contract moderation capacity tailored to their language and legal standards, combining automated classifiers with human escalation for appeals and court orders.
- **Jurisdictional feature toggles:** sensitive features (content categories, recommendation policies, certain financial instruments) can be enabled/disabled per country without forking the global codebase.
- **Localization:** language packs, local pricing presets, tax/VAT rules, and culturally appropriate defaults are layered as configuration, keeping the product globally consistent while locally compliant.

Operating many sites creates compounding know-how. Each incident, rollout, and moderation workflow improves tooling and playbooks, lowering the marginal cost of the next deployment. Over time, operators specialize: some become excellent "finance operators" (strong KYC/AML, reporting), others "media operators" (moderation, copyright), others "AI operators" (GPU scheduling, model serving). This specialization is a healthy market outcome: applications can be present everywhere, but they are served by the operators best suited to the local execution and legal constraints. Because operators earn operational revenue per unit, they are directly incentivized to improve uptime, latency, and cost efficiency. At the same time, they retain hard power: they can refuse to deploy unsafe templates, delay risky upgrades, or route traffic away from misbehaving sites. This creates a credible feedback loop for founders: publish high-quality, operable, legally adaptable templates or lose on-net placement and thus performance and growth. The result is a market that rewards not only good products, but also good operational citizenship.

5.4.5 Inter-operator interfaces: cooperation and data replication

A local site does not exist in isolation. For most successful applications, multiple Site Operators will host the same template across more sites and/or jurisdictions. As a result, operators need a way to *cooperate* in the moments where user behavior or application semantics naturally span more than one operator. The most important cooperation surface is data replication: deciding what state can move, how it moves, who remains authoritative, and how replication is paid for.

Cross-jurisdiction replication is permitted, but the local serving operator still owns compliance and still claims (or doesn't claim) settlement. Replication is an operator-to-operator cooperation channel. It can cross borders if it doesn't become a backdoor for serving users "from abroad" while still claiming local payouts, or shifting legal responsibility away from the local operator-of-record. Compare this to settlement - "Jurisdictional unambiguity" constrains who can claim protocol disbursements (processed RfS/RfD/RfJ) and, by extension, who is the operator-of-record for a user session in that jurisdiction.

For routine replication and state synchronization, operators generally engage in settlement-free peering, recognizing that mutual synchronization benefits all participants by keeping the global application functional. However, if a severe imbalance occurs—for instance, a small operator

hosts viral content that is heavily pulled by a massive Tier-1 operator—the ESP protocol's metering infrastructure can track these cross-operator data retrievals. Operators can establish predefined peering thresholds; if cross-operator bandwidth exceeds the threshold, the protocol can automatically route a micropayment from the consuming operator's treasury to the hosting operator, compensating them for the outbound transit. This ensures that smaller operators are not financially penalized for hosting popular content.

6 Blueprint for Data Commons

The term "Data Commons" is defined as actively collected, non-rivalrous information about the virtual and physical world: open web indexing (fresh, deduplicated content and link graphs), road and place mapping (street-level changes, POIs, opening hours), and network quality telemetry. Equally important are privacy-preserving aggregates of user-side signals: anonymized intent distributions, content-popularity curves, session-level QoS, anonymized Internet search queries, and general user behavior on the public Internet. This private user data is considered fundamentally non-rivalrous because it is not an exclusive secret held solely by a central server. Every interaction is observed by both the client and the server—there are always two sides that see the exact same data. It is just that the client end is unaggregated and thus hard to have any value as a dataset.

Today, these assets are enclosed within the walled gardens of hyperscale platforms. This enclosure stifles innovation, forcing majority of value creation to occur strictly within the boundaries set by incumbent platforms. The current state of the Internet sees this data largely privatized, giving hyperscalers an immense competitive advantage for their products, which they then lock behind walled gardens.

The fundamental illness of the present arrangement is the privatization of what is effectively public infrastructure. A web index or a street map is not a creative invention of the platform that hosts it; it is a derivative record of the collective output of humanity and the physical world. By allowing hyperscalers to treat these observational datasets as proprietary capital, the enclosure of a shared digital and physical reality has been permitted. This creates a dangerous conflict of interest: when the entity that organizes the world's information also relies on selling ads to survive, the incentive is not to show the most accurate map, but the most profitable one. The "truth" of the internet becomes distorted by commercial extraction. This data belongs in the commons not just for economic efficiency, but to restore the integrity of the shared information environment.

The "Data Commons" blueprint proposes a radical inversion: unbundling these foundational datasets from the applications that use them. Instead of a single company owning the map to sell ads on top of it, the network itself—coordinated by ISPs and edge devices—collaboratively builds and maintains the map, making it available to any application developer for a metered fee. This transforms data from a competitive moat into a public utility. By federating the collection of this data—using the vast, distributed infrastructure of ISPs and residential gateways—datasets can be built that are fresher, more comprehensive, and more neutral than any single corporation could achieve.

Crucially, this is not a charitable endeavor but a structural economic necessity. The revenue generated from running the Data Commons serves as the "subsidy replacement" for the advertising model. In the current web, services like search and email are "free" because they harvest user data to sell attention.

This structural shift unlocks a new wave of innovation by lowering the barrier to entry for complex, data-intensive applications. Currently, if a team wants to build a new search engine, they must first replicate Google's multi-billion dollar crawling infrastructure before they can

even begin to innovate on the ranking algorithm or user interface. In the Data Commons model, the "heavy lifting" of ingesting the original signal is done once by the network and licensed as a commodity. This means a small team of engineers can access the global web index dataset and focus entirely on building a superior vertical search product.

Furthermore, this distributed approach to data collection offers inherent advantages in quality and resilience that centralized incumbents cannot match. A centralized crawler is easy to block, easy to spoof, and prone to blind spots. In contrast, a Data Commons built on millions of residential ISP endpoints sees the web exactly as users do. It bypasses bot detection, captures localized content variations, and is structurally resistant to censorship. This results in a dataset that is not only "open" but technically superior—more representative, more robust, and harder to capture by any single political or corporate interest. The network itself becomes the source of truth, verified by the very edges that comprise it.

In this blueprint, three fundamental categories of datasets are identified that form the bedrock of the Internet Commons. First is the **Map of the Virtual World** (Web Index): the comprehensive, searchable graph of all public digital content. Second is the **Map of the Physical World**: the street-level imagery, geometry, and point-of-interest data required for navigation and logistics. Third are **User Datasets**: the privacy-preserving aggregates of human behavior in Public Internet, intent, and preference that power personalization and market efficiency.

6.1 Protocols for Data Commons

Workloads for the new edge devices are containerized "nodes" of data-commons protocols, some of which may already exist today as decentralized network attempts (e.g., Grass for web crawling, Hivemapper for mapping, except these are token-based protocols which implies all the disadvantages of token-based protocols and misalignment of incentives). Different protocols focus on different data commons. The protocols themselves are decentralized, networked systems designed to be ran permissionlessly.

Whether existing protocols such as Grass or newly designed ones, the ISP takes full management responsibility: deploying, monitoring, and optimizing nodes on the edge devices. Rewards—be it tokens, credits, or fiat equivalents — accrue entirely to the ISP, incentivizing adoption without requiring user involvement - the device runs workloads in the background, using spare cycles (e.g., during off-peak hours) without impacting performance.

6.1.1 Privacy-preserving user data collection

A key contribution of the Data Commons is the creation of anonymized user-centric data aggregates. The protocol achieves this through the **Request for Data (RfD)** mechanism, which establishes a privacy-preserving, server-side data collection model.

The RfDs integrate directly with applications hosted at Site Operators. When a data buyer signals demand via an RfD, it triggers a standardized collection workflow at the application layer. These applications emit anonymized telemetry such as search queries, media consumption patterns, and session quality metrics.

This server-side approach ensures data integrity and user privacy by design. The ISP, leveraging its verifiable relationship with the user (established via proof-of-ownership of the connectivity endpoint), aggregates this telemetry across its user base. It then applies rigorous privacy-preserving transformations before the data ever leaves the secure enclave. The result is a high-fidelity, privacy-compliant dataset contributed to the public commons, available to all ecosystem participants.

6.1.2 Indexing the web

The web index—a deduplicated dataset of the public web, its link graph, and metadata—is the bedrock of the Internet Data Commons. It underpins search, discovery, and ranking. More importantly, it fuels the training and operation of the AI models that increasingly mediate human access to information.

The future of the open web depends on ensuring this knowledge base remains a public good, rather than a proprietary asset of a few hyperscalers. As AI agents and LLMs become the primary interface for the web, the entity that controls the index controls the "truth" presented to users. Large-scale privatization of this index threatens to fragment the web into walled gardens.

Distributed indexing from the network edge offers a robust counter-strategy. By running indexing jobs on CPE endpoints (not only, but mainly), the collection technology is effectively "decoupled" from centralized platforms. This approach yields a decisive quality advantage: probes originate from residential IP addresses and traverse the same network paths as ordinary user traffic. This vantage point bypasses crawler-detection heuristics, surfaces geo-specific content, and captures the web as it actually appears to humans—including consent flows and localized variations—while strictly honoring `robots.txt`.

This dynamic creates a powerful leverage against information monopolies. A distributed, user-powered index provides broader coverage and higher freshness than centralized crawlers can achieve. It presents a clear alternative: an open, verifiable, and continuously updated commons that belongs to no single corporation. By participating in this network, users and ISPs actively prevent the enclosure of the open web, ensuring that the raw material of human knowledge remains accessible to all builders and innovators.

6.1.3 Mapping streets

Maps are a critical subset of Data Commons: up-to-date road geometry, place metadata, and street-level imagery underpin routing, navigation, logistics, and safety. Broad, continuously refreshed coverage improves all downstream services.

The future of transportation belongs to self-driving cars, which inherently rely on arrays of high-fidelity cameras and sensors to perceive their environment. This inevitable technological shift presents a pivotal choice: will the massive streams of visual data collected by these vehicles become the proprietary hoard of car manufacturers, or will it remain under the sovereign control of the car owners? It is imperative that the data belongs to the owner, not the manufacturer.

Any attempt at large-scale privatization of this data is actively counteracted by ensuring the camera collection technology remains "decoupled" from the vehicle's manufacturing vertical.

An example of this model is Hivemapper, which separates the data collection layer from the automotive OEM. This creates a binary choice for car manufacturers: either they submit to a global, neutral data collection framework where data ownership is preserved for the user, or they will face a market where third-party devices like Hivemapper are retrofitted into cars, bypassing the manufacturer's data silos entirely (users actively incentivized to place the dashcams to the self driving cars for extra earnings).

Steering the future in this direction requires utilizing powerful economic and ecosystem leverages. By integrating map-building rewards directly into connectivity plans, ISPs can subsidize the hardware cost of independent collection devices, achieving critical mass rapidly. This creates a competitive wedge: manufacturers are forced to open their data pipelines to neutral aggregators to compete with the coverage and freshness of the decentralized network. If they refuse, the "decoupled" ecosystem—incentivized by rewards and supported by ISP infrastructure—simply outpaces them, rendering their proprietary data silos inferior. The dynamic shifts power back to the user, who ultimately decides which network their vehicle contributes to, ensuring the street-level view of the world remains a common good.

7 Threat model and mitigations

ESP's security and economic integrity depend on mitigating threats from three actor classes: dishonest users, dishonest applications, and dishonest ISPs. The protocol's design prioritizes ease of mitigation for lower-tier threats while investing sophisticated mechanisms against the highest-risk actor—the dishonest ISP.

Dishonest users attempting to consume services without payment pose limited systemic risk. Because RfS events must exist on-chain for apps to claim payment, a user who bypasses RfS generation (e.g., by modifying browser code) will be caught: the app has no on-chain RfS to reference in its disbursement request, and the ISP's traffic logs reveal consumed bandwidth without corresponding consent events posted to the blockchain. If the pattern persists, the ISP can throttle or terminate service for that endpoint, a power ISPs already exercise for terms-of-service violations. Moreover, from the app's perspective, the user is irrelevant—the app is paid by ESP based on on-chain RfS records, not by the user directly. The ISP bears the residual risk of user fraud and can handle it through traditional enforcement: usage caps, bill adjustments, or service suspension.

Dishonest applications that claim payment for undelivered services are neutralized by the on-chain RfS requirement. An app can only submit disbursement requests referencing RfS events that actually exist on the blockchain. If an app claims it served phantom usage, ESP will find no matching on-chain RfS and will not settle. Apps cannot fabricate on-chain RfS events because those are posted exclusively by CPE devices with hardware-attested signatures. Persistent mismatches (claiming RfS events the app never actually fulfilled) degrade the app's reputation and eventually triggers from outgoing payments from the master clearing account.

Dishonest ISPs represent the most potent threat: an ISP could suppress RfS posts to avoid paying apps (stealing from apps) or inflate posts to overcharge users (stealing from users). ESP's defenses are multi-layered:

- **Pull-based scraping:** As described above, third-party ISPs scrape meters from each other's CPE devices, preventing any single ISP from suppressing RfS posts. If ISP A's users generate RfS events that ISP B's scraper observed but ISP A failed to post on-chain via its blockchain nodes, the protocol flags the discrepancy and other ISPs can submit the missing RfS events on behalf of the affected devices.
- **Device-side attestation:** CPE devices produce cryptographically signed RfS events that are tamper-evident and auditable. Even if an ISP controls the blockchain node receiving posts, it cannot forge device signatures or alter RfS events without invalidating attestations. All RfS events carry hardware-rooted signatures from the originating device.
- **Collateral and slashing:** ISPs must stake collateral (in the form of deposited stablecoins or protocol-native tokens) to participate. Detected dishonesty—suppressed RfS posts, fabricated events, failed challenges—results in collateral slashing proportional to the severity and recurrence of violations.

7.1 Empowering users to retain economic agency

A cornerstone of user trust is the ability for any user—without specialized expertise—to verify that their ISP and the apps they use are behaving honestly. Every CPE device maintains a local dashboard accessible via a simple web interface (e.g., at a reserved local IP address like `192.168.1.1/esp-dashboard`). This dashboard displays:

- A chronological log of all RfS, RfD, and RfJ events generated from the household, including app/service identifiers, timestamps, unit prices, and correlation keys.
- Summaries of scraped meter reports: which external ISP scrapers have pulled data from the device, when, and what aggregates were submitted.
- Links to on-chain transaction records showing RfS events posted to the blockchain and matched application disbursement requests, allowing the user to confirm that activity logged locally corresponds to activity recorded on-chain and settled by the protocol.
- Discrepancy alerts: if the device detects that an RfS was generated locally but does not appear on-chain within a reasonable time window, it flags the event for user review, indicating potential ISP suppression.

This local transparency empowers users to audit both their ISP (are they posting my RfS events to the blockchain honestly?) and applications (are they claiming payment only for services I actually received?). Because all RfS events are signed by the device's hardware enclave and correlation keys are deterministic, a user can independently verify end-to-end consistency: "I clicked button X, the device logged and posted RfS Y to the blockchain, the app submitted a disbursement request referencing RfS Y, ESP matched them on-chain, and payment settled."

8 Game theory of the ESP

ESP is a multi-sided protocol with several strategic actors: ISPs, users, site operators, founder teams (application builders), buyers of data/jobs etc. The protocol only works if these actors prefer honest participation over the most obvious deviations (overcharging, under-reporting, withholding payouts, or attempting to capture the customer relationship). This section explains why the equilibrium we want is plausible: everyone can make money *only* if the ESP itself keeps working, and most profitable strategies require keeping the other sides willing to stay.

8.1 Why ISPs rationally sell data via ESP rather than alone

A natural question arises: why would an ISP choose to sell user data or computational resources through the ESP rather than directly to buyers on its own? The answer lies in the limitations of fragmented data and the scaling power of aggregation.

Any single ISP, even a large one, only possesses a partial view of the global user base and Internet traffic. To a data buyer or a global service provider, this fragmented perspective has limited usefulness. Buyers typically seek comprehensive, cross-network insights or global computational reach. Consequently, an individual ISP struggling to sell its isolated data silo would face high customer acquisition costs, low perceived value from buyers, and the overhead of managing complex direct sales and compliance.

Naturally, ISPs resist any system where earnings are socialized. Their default preference is to own the entire customer relationship and capture 100% of the revenue generated by their network. However, the economic reality of data markets forces a different calculation.

By participating in the ESP, an ISP contributes to a globally aggregated pool of data and services. This aggregate product is vastly more valuable to buyers than the sum of its isolated parts, commanding a premium and attracting global demand. Even though participating means accepting that a portion of the revenue from data sales or job execution is socialized by being streered though the central Wealth Fund account and redistributed socially, it is still highly rational. The ISP's fractional slice of this highly-liquid, global pie is far greater than 100% of their own illiquid, fragmented data.

This redistribution mechanism compensates ISPs for their role in creating an aggregate value they could never capture independently. The redistributed funds can then be used by the ISPs to cover network expenses or passed back to their users as direct rewards or subscription discounts, ultimately strengthening their core consumer offering and user relationship.

8.2 Why users rationally configure a different ISP for scraping

While a user's primary internet connection and Customer Premises Equipment (CPE) are provided by their "parent" ISP, the ESP architecture allows users to authorize a different, third-party ISP to scrape their local metering logs (RfS events). Rationally, users are highly motivated to configure a different ISP for this scraping role rather than defaulting to their parent ISP.

If the parent ISP acts as the sole scraper and aggregator of a user's data, it creates a localized monopoly. The parent ISP might be incentivized to under-report activity, censor specific

application usage, or delay posting events to the blockchain to manipulate settlement or hoard valuable data insights.

By delegating the scraping rights to a competing ISP, the user introduces a powerful separation of powers. The external scraping ISP has a direct financial incentive to accurately pull, aggregate, and report the user's data to the network, as their own revenue depends on the volume and quality of the data they process. This competitive check ensures that the user's economic activity is reliably recorded on-chain, maximizing the user's potential rewards and preventing the parent ISP from silently dropping data or degrading the user's agency.

8.3 Why applications rationally choose ESP

For a founder team, the scarce resource is not code—it is distribution, billing customer acquisition, scale etc. Building a global application outside ESP means: acquiring users one-by-one, building a payments stack, handling fraud and chargebacks, negotiating app-store or platform taxes, and integrating with regulatory regimes etc. ESP collapses that complexity into a single global counterparty and a uniform monetization scheme.

This is why, in equilibrium, serious applications are "forced" to use ESP in the same way serious web services were "forced" to speak TCP/IP: not by coercion, but because the alternative has prohibitive friction. The founder team can still self-host and self-distribute, but doing so recreates the very hyperscale platform dynamics ESP is designed to unwind.

8.4 The franchise equilibrium: founders are rewarded, but replaceable

ESP intentionally separates **product authority** from **distribution authority**. The founder team controls the code, roadmap, and brand; the site operator controls deployment, local compliance, and access to subscribers. The protocol binds these roles together financially via the **Founder Account** (domain-bound) and its configured royalty share.

Two constraints matter:

- **Replacement friction must be low.** If a founder team becomes abusive (extractive pricing, dark patterns, degraded quality, refusal to patch security issues), site operators must be able to promote and run a replacement quickly. Because the site operator already owns the deployment and distribution channel, replacing the operating team is operationally feasible: they can surface an alternative service to their users, migrate workloads, and route traffic to a new domain/Founder Account.
- **Replacement friction must not be zero.** If founders could be replaced instantly without cost, no team would invest in long-term product development. Royalties, brand accumulation, and the time it takes to build user trust create the "bond" that makes investment rational. In other words: founders are *not* protected by lock-in; they are protected by *continued usefulness*.

The key game-theoretic outcome is that founder teams maximize long-run royalties by maximizing long-run user satisfaction and usage volume, not by maximizing short-run extraction.

8.5 Why site operators do not attack the founder team

A naive fear is that a powerful ISP could "steal" an application by forking it, rebranding it, and capturing the revenue. ESP makes this unattractive for three reasons:

- **Local reach is not global reach.** A single site operator can only push a takeover to its own subscriber base. That limited outreach makes capturing global applications strategically hard and economically smaller than it appears.
- **Expropriation damages the operator's core business.** Users primarily care about reliable, high-quality services. If an operator is seen as hijacking brands or breaking settlement norms, it reduces user trust and increases churn risk—a much larger loss than a one-off capture.
- **Royalties are cheaper than reinvention.** Paying the founder royalty is usually the dominant strategy: it buys a steady stream of updates, security patches, and competitive product iteration. A takeover forces the operator to become a product company, which is precisely the specialization ESP tries to avoid.

Thus, when an application is popular (high RfS volume and high disbursements), site operators have *no incentive* to go against the founder team. The profitable move is to keep the service healthy and keep usage high.

8.6 When and why replacement becomes the rational move

The incentives flip when users are unhappy. If usage declines, support tickets rise, or reliability drops, the operator suffers twice: subscriber satisfaction falls, and the operator's revenue tied to RfS activity falls. At that point, the operator is incentivized to:

- surface alternatives (competing services with better price/quality),
- pressure the founder team to improve (because the founder also loses royalties when usage declines),
- and, if needed, sponsor or adopt a replacement founder team and migrate users to a new domain.

Importantly, replacement is *doable* because operators control distribution. The threat of replacement is therefore credible, and credible threats are what make cooperative equilibria stable.

8.7 Honest metering is the best strategy

ESP's settlement rules are designed so that the most obvious fraud strategies either do not work or destroy the perpetrator's own payoff.

Applications cannot profitably claim phantom usage. Payments trigger only when a disbursement request matches an on-chain RfS. Persistent mismatch degrades reputation and can trigger rate limits or slashing (as described in the settlement engine). The dominant strategy for apps is therefore to serve real requests and claim only what can be matched.

ISPs cannot profitably fabricate usage. RfS events are initiated by the user's browser, signed at the edge, and merely *witnessed* by the CPE device; the CPE does not generate RfS on its own. Fabricating RfS at the ISP level therefore requires fabricating user-originated signatures at scale, which is the wrong side of the cost curve. Additionally, the pull-based scraping model (where other ISPs can pull witnessed events) reduces the ability of a single operator to selectively censor or rewrite its own logs.

ISPs cannot profitably under-report or withhold. Withholding RfS events or obstructing settlement harms the operator's own subscribers: applications will treat that operator as unreliable and will degrade or deny service to those users. Under-reporting therefore converts into a quality-of-service penalty that pushes users to switch providers or routes demand elsewhere. In equilibrium, honest reporting maximizes the operator's long-run revenue.

8.8 Smaller moats and weaker lock-in

In Web 2.0, the strongest moats come from controlling distribution (app stores, browser defaults), controlling identity (accounts), and controlling user data. ESP intentionally shrinks these moats:

- **Distribution shifts toward ISPs and away from hyperscalers.** Services are surfaced from the edge environment rather than from a centralized platform.
- **User identity is not a platform asset.** The user does not need to register with every app; the ISP is the consistent counterparty.
- **Data extraction is replaced by paid aggregates.** Value from data is expressed through RfD/OfD flows rather than silent siphoning.

This makes extreme lock-in (walled gardens) less likely to emerge, because applications cannot as easily hoard the inputs that create long-term monopolies.

8.9 Why ISPs rationally push ESP-native architectures

ESP changes the ISP's payoff function: operators earn more when valuable services, data products, and jobs run *at the edge* under the protocol's settlement rules. This creates a direct incentive for ISPs and site operators to:

- standardize and promote ESP-compatible application architectures (Of*/Rf* embedded in web delivery, edge attestations, etc.),
- resist "over-the-top" platforms that attempt to re-intermediate distribution and capture the billing relationship,
- and invest in local infrastructure (compute, storage, privacy-preserving aggregation) because the protocol monetizes it directly.

8.10 The "weak dictator" principle

At any point in time, some actor is effectively the "dictator" of a layer: founders dictate the product; site operators dictate deployment; ISPs dictate the subscriber relationship; the validator

set dictates the protocol code. ESP's goal is not to pretend dictatorship disappears, but to *weaken* it by ensuring switching costs are low and governance is contestable:

- founders can be replaced by better founders,
- operators compete for subscribers (and cannot afford to degrade service without churn risk),
- and protocol upgrades are "voting with code"—visible, forkable, and constrained by the need to keep the ecosystem participating.

The practical result is a more robust form of decentralization: not the absence of power, but the presence of constant, credible pressure that prevents any single power center from becoming unaccountable.

9 Capital formation in ESP environment

Traditional Internet capital formation is built around two hard constraints: (i) revenue is delayed and opaque (often hidden behind proprietary dashboards and platform terms), and (ii) distribution is gatekept (app stores, ads, partnerships), so builders raise money primarily to buy growth. ESP flips both constraints. Revenue becomes continuous, granular, and protocol-native (every service interaction settles), while distribution is structurally bundled with the ISP relationship (the user already has a paying counterparty). The result is a setup where a meaningful portion of funding can be securitized directly against *verifiable future usage*, rather than against speculative equity stories.

ESP introduces a capital instrument - the ESP Fund Raising Token - that looks economically like RBF - Revenue-Based Financing - but with two properties that traditional markets struggle to deliver at Internet scale:

- **Programmatic enforcement:** repayment is not a promise; it is a routing rule inside the settlement engine. If usage happens, repayment happens automatically.
- **Built-in sunset:** early backers can be rewarded without creating a permanent rent-extraction layer. Once the predefined cap is reached, the instrument dissolves and the application operates with clean economics.

Conceptually, the application sells a bounded claim in form of tokens on its future ESP-settled cashflows. In exchange, it receives upfront funds to build. When the service goes live, each disbursement is split into operator revenue (to run the franchise), founder royalty (to fund ongoing R&D and reward creation), and fund raise revenue (to repay early backers up to a maximum cap). When the fund raise cap is fully reached, the fund raise path is removed and all future value flows only to founders and operators.

We compare ESP fund raising to traditional equity financing. Traditional equity financing is optimized for high-growth, winner-take-most outcomes where ownership and control rights are central. It also creates structural pressures: raise larger rounds, grow at all costs, and optimize for exit events (acquisition/IPO) rather than for stable service quality. Equity still has a role in ESP environment — especially for deep R&D, acquisitions, or platform-scale bets—but ESP enables a much larger middle class of services to be financed without turning every product into a venture-scale company.

Debt based financing is efficient when cashflows are stable and collateral exists, early Internet services howeever often lack both. ESP makes something debt-like possible earlier because the "collateral" is not physical assets but *enforceable revenue routing*. However, unlike classic debt, the fund raise mechanism is naturally flexible: if usage is slow, repayment is slow, without forcing default dynamics that kill the product.

RBF (Revenue Based Financing) is the closest analog: investors get a share of revenue until a multiple or cap is reached. The problem in Web2 is enforcement and transparency: revenue is privately reported, contracts are off-chain, and the counterparty can withhold or reclassify revenue.

ESP essentially turns RBF into a protocol primitive:

- revenue is generated and settled inside the system;
- the split happens automatically;
- the repayment cap is enforced mechanically;
- performance metrics are auditable.

9.1 ESP Fund Raising Tokens vs Crypto Tokens

Many Web3 token markets are dominated by assets whose value is weakly anchored to cashflows. Prices are therefore often driven by reflexive expectations: narrative, listings, emissions schedules, influencer attention, and speculative momentum. Fund raise token markets should differ in several predictable ways:

From narrative to underwriting. Instead of "what story will the market buy next," traders will ask: *what is the expected remaining payout stream, how quickly will it arrive, and what risk discount applies?* This shifts practice toward underwriting, closer to credit and structured products.

From perpetual emissions to amortization. Where many tokens rely on ongoing emissions and incentives that dilute holders, fund raise tokens amortize as the cap is paid down. That creates clearer horizons, encourages time-based valuation (duration), and reduces the need for perpetual hype.

From governance premium to cashflow premium. Governance tokens often carry an implicit "control premium." Fund raise tokens primarily trade on a "cashflow premium" (a bounded stream of usage-derived payouts). Governance still matters, but mainly as a risk factor.

From reflexivity to bounded reflexivity. Reflexive loops will still exist (price attracts attention; attention attracts liquidity), but the cap bounds upside and repayment progress anchors expectations. Over time this encourages convergence toward fundamentals.

Toward a yield curve of Internet services. As many projects issue fund raise tokens with different caps, fund raise shares, and adoption trajectories, markets can compare them the way bond markets compare issuers: expected yield, duration, repayment momentum, jurisdictional footprint, and distribution quality. This is qualitatively different from "one governance token per protocol."

9.2 Behavior of retail Fund Raise Token holders

The highest-level change is that investors become a market for *measurable innovation.* The token is not a vague claim on future dominance; it is a bounded claim on a specific, auditable stream that exists only if users repeatedly choose the product. As a result, capital allocation and secondary trading become increasingly about forecasting adoption trajectories and operational quality rather than manufacturing scarcity narratives.

A useful way to understand fund raise token markets is to focus on the behavior of a typical **retail participant** (a non-institutional holder). Their behavior tends to evolve in two regimes:

- **Pre-launch (and very early launch):** the token trades primarily on beliefs about whether meaningful revenue will exist at all.
- **Post-launch (once usage is measurable):** the token trades primarily on repayment momentum, time-to-cap, and risk-adjusted yield.

This matters because the same instrument can look like a high-volatility "startup option" early, and like an amortizing cashflow claim later.

Before launch: speculation is primarily about the probability of revenue. Before an application generates observable RfS volume, the fund raise token's value is dominated by a single question: *What is the probability that this project will generate enough real usage to route meaningful funds into the fund raise account?*

Let:

- C be the fund raise cap (maximum total amount that can ever flow into the fund raise account),
- S be the total fund raise token supply,
- $X(t)$ be cumulative inflows into the fund raise account by time t (so $0 \le X(t) \le C$),
- and define $\Pi = \frac{C}{S}$ as the *maximum* lifetime payout per token if the cap is fully reached.

Pre-launch, $X(t) \approx 0$, so the maximum possible undiscounted payoff per token is simply Π. A retail buyer is therefore implicitly trading a probability-weighted version of Π.

A compact approximation is:

$$P_0 \approx \underbrace{\Pr(\text{cap reached})}_{p} \cdot \underbrace{\Pi}_{C/S} \cdot \underbrace{D(\tau)}_{\text{time+discount}} - \underbrace{\Lambda}_{\text{risk \& liquidity discount}}$$

where:

- p is the market-implied probability that the project eventually reaches the cap (or reaches a large fraction of it),
- $D(\tau)$ is a discount factor capturing time-to-repayment (often modeled as $e^{-k\tau}$ or $(1+k)^{-\tau}$ for some risk-adjusted rate k and horizon τ),
- Λ aggregates early-stage frictions: smart-contract risk, execution risk, operator adoption uncertainty, and illiquidity.

In this regime, price changes are mostly belief updates about p and τ. Retail participants typically infer these from coarse signals: operator partnerships, credible deployment paths, audits, early demos, known founders, comparable products, and the perceived competitiveness of the offering. Price action can therefore be discontinuous: a single credible distribution signal can move p materially even before any revenue exists.

After launch: trading shifts toward repayment momentum and time-to-cap. Once the service is live and RfS-driven settlement begins, the fund raise account is funded by a deterministic routing rule: a share of service disbursements flows into the fund raise account until C is reached. At that point, the instrument is economically complete: no additional value is routed to fund raise token holders.

Define the **remaining fund raise headroom** at time t:

$$R(t) = C - X(t).$$

For a holder with one token, the *remaining undiscounted* maximum they can still receive (if the project continues until the cap is reached) is:

$$\pi(t) = \frac{R(t)}{S}.$$

A concrete example. Assume:

$$C = \$10{,}000{,}000, \quad S = 100{,}000{,}000 \;\Rightarrow\; \Pi = C/S = \$0.10 \text{ per token (maximum lifetime).}$$

Suppose 40% of the cap has already been funded, so $X(t) = \$4{,}000{,}000$ and $R(t) = \$6{,}000{,}000$, implying:

$$\pi(t) = R(t)/S = \$0.06 \text{ remaining per token.}$$

If observed inflows suggest $\hat{\tau}(t) \approx 24$ months (implying an average duration of ~ 12 months for the amortizing cashflows) and a holder uses $k = 20\%$ annualized, then:

$$\mathrm{PV}(t) \approx 1.0 \cdot 0.06 \cdot (1/1.2) \approx \$0.05$$

(ignoring additional risk discounts). If the market price is $P(t) = \$0.055$, selling can be rational: the market is paying more than the holder's risk-adjusted present value, so the holder realizes value now and transfers remaining time/risk exposure to someone with a lower discount rate or higher conviction.

Conversely, if $P(t) = \$0.045$, holding is rational under the same assumptions: the holder expects to capture more value by waiting.

How retail behavior changes across the lifecycle. This framework predicts a typical behavioral arc:

- **Early stage:** trading is dominated by belief updates about p (will it work?) and by large liquidity premia; price can be highly volatile and narrative-sensitive.
- **Scaling stage:** as $X(t)$ and $m(t)$ become observable, retail behavior shifts toward monitoring repayment momentum and estimating $\hat{\tau}(t)$; price becomes increasingly "cashflow-anchored."
- **Late stage:** when $R(t)$ is small, the token behaves like a short-duration claim. Volatility tends to compress and valuation converges toward a near-term payout profile (minus

liquidity and residual risk).

The result is a market in which retail participation is less about infinite-horizon narrative dominance and more about continuously re-pricing a bounded, amortizing claim as real adoption data arrives.