

Coordination, Markets, and Protocols

RenCryptoFish

January 27, 2026

Every major coordination problem in history has been solved in a similar way: someone builds a central authority and takes a cut. Markets, platforms, brokers, exchanges. The pattern is so consistent it looks like a law of nature. Protocols are the rare exception. TCP/IP, HTTP, SMTP, Bitcoin. When they work, they become infrastructure that entire economies run on. But the path to getting there is brutal, because coordination has a natural gravity that pulls toward centralization.

This article builds a framework for understanding why. It starts with the coordination problem, introduces two dimensions that explain most market structures, and then shows why protocols occupy such a narrow and fragile corner of that space.

Start with the coordination problem. Any time N independent entities need to work together, there are 3 distinct forces working against them as N increases.

Complexity: the number of potential 1-on-1 negotiation channels explodes quadratically. By the time you reach 30 entities, you are managing over 400 potential interpersonal conflicts or communication links.

Incentive: the dilution of an individual's share. At $N=2$, you are 50% responsible. By $N=20$, your contribution feels mathematically negligible (5%), making "free riding" the rational choice for most participants.

Stability: assume everyone is 95% reliable. In a small group, the chance of everyone agreeing is high (~90%). By $N=30$, the mathematical probability of holding the group together without a single defection or veto drops to near 20%.

The "Coordination Trap": Why Difficulty Explodes as N Grows

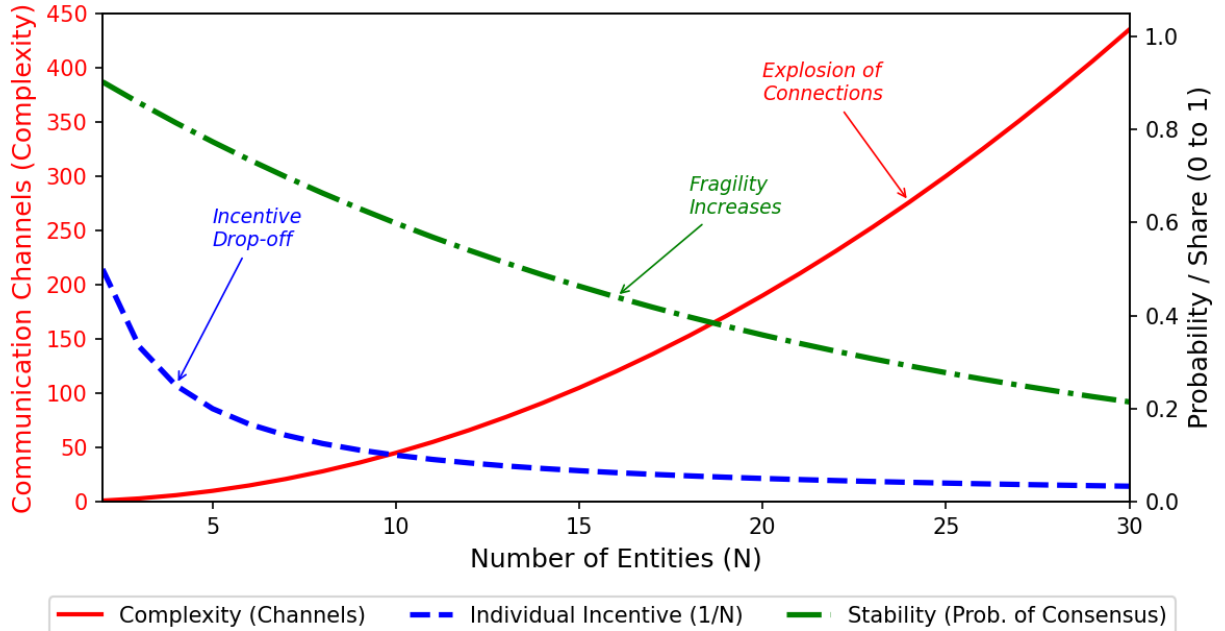


Figure 1: The Coordination Trap: Why Difficulty Explodes as N Grows

Markets are essentially "Coordination Technologies" to solve this problem, but they can take a variety of structures. There are a lot of ways to cut this, below I walk through one way to frame this.

1 H-V: Asset Heterogeneity and Verification Cost

Markets have many structures, terms commonly seen might be bazaar, broker, platform, and protocol. These structures are often a locally optimal solution based on attributes of the market, participants, and more.

Market Structure Matrix: H vs. V

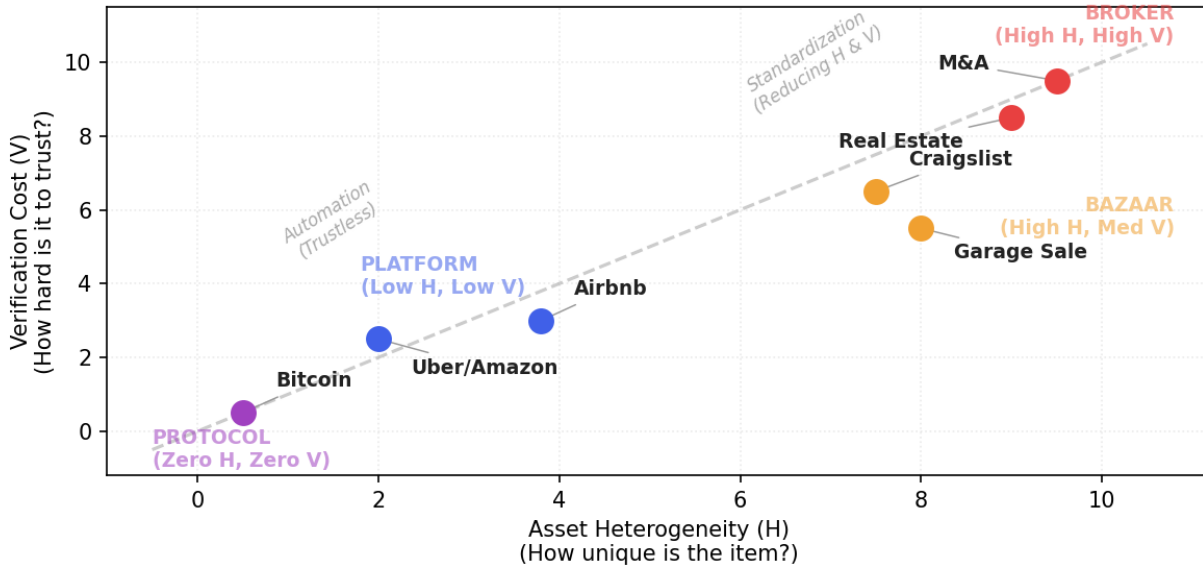


Figure 2: Market Structure Matrix: H vs. V

Two dimensions do most of the explanatory work:

Asset Heterogeneity (H): How unique is the item? (1 share of Apple stock = 0 heterogeneity, a vintage house = High H)

Verification Cost (V): How hard is it to prove the item is good? (checking a diamond = High V, checking a digital file = Low V)

Name	Asset Heterogeneity (H)	Verification Cost (V)
Broker	High	High
Bazaar	High	Medium-High
Platform	Low	Low
Protocol	Very Low	Very Low

Let's walk through various combinations of H and V:

1.1 Broker (High H, High V)

Why it stays here: Consider the market for M&A (buying companies) or High End Real Estate. Every deal is unique (H), and if you buy a "lemon" you lose billions (V is critical).

$$\text{Cost_Search} + \text{Cost_Verify} > \text{Commission}$$

1.2 Bazaar (High H, Medium-High V)

Why it stays here: In a Bazaar (e.g., a garage sale or early Craigslist), every item is unique (H is high). You cannot standardize the contract because selling a used toaster is different from selling a bike. Because H is high, the Information Asymmetry is massive. A central platform cannot efficiently price these items, only the two people staring at the object can agree on value.

1.3 Platform (Low H, Low V)

The Compression Strategy: Platforms (Uber, Amazon, Airbnb) win by forcing H to zero. You don't choose "Dave who drives a Ford"; you choose "X Tier Ride."

The Efficiency: By reducing Heterogeneity (H to 0), it's possible to reduce the transaction to a single variable: Price.

The Stability: This creates a "Thick Market." Because everyone is in the market for the exact same thing (a standard ride), liquidity is maximized.

1.4 Protocol (Very Low H, Very Low V)

Platforms tend to rely on reputational trust, social graphs, and other mechanisms to reduce verification costs. Protocols substitute this with standards and/or cryptographic proofs.

Protocols push Cost of Verification $\sim$ Cost of Compute

When this wins: In markets where "Trust" is the biggest cost. For example we pay banks huge fees and taxes to verify money moved from A to B, a protocol over time compresses this to the cost of compute $\sim$ cost of electricity.

The limit: Protocols struggle with high H (heterogeneity). A blockchain can prove you own a token, but it cannot prove that the physical banana represented by that token isn't rotten. This is the "Oracle Problem." Until machines can verify physical reality (High V) without humans.

2 What happens off the diagonal?

The off-diagonal corners are unstable. In the top left (Low H, High V), a product looks standard but is nearly impossible to verify. Think counterfeit pharmaceuticals or blood diamonds before certification. This is Akerlof's "Market for Lemons": buyers refuse to trust, honest sellers can't prove themselves, and the market collapses. To survive, it must move onto the diagonal by introducing a broker ("I only buy from Tony") or inventing a verification test (the Kimberley Process) that pulls it toward a platform. The bottom right (High H, Low V) is equally unstable but for the opposite reason: every item is unique, yet verification is instant. When you can perfectly measure uniqueness, you can categorize it, and categorization kills heterogeneity. The market standardizes and slides left toward the platform model. Both corners resolve the same way: back to the diagonal.

3 Technology and the Diagonal

Many of the biggest platforms that have emerged on the internet have taken an industry or market in the top right (broker or bazaar) and used technology to reduce verification costs (reputation based systems) and heterogeneity (categorization and machine learning).

The Diagonal Migration: Technology Pushes Markets from Broker → Platform → Protocol

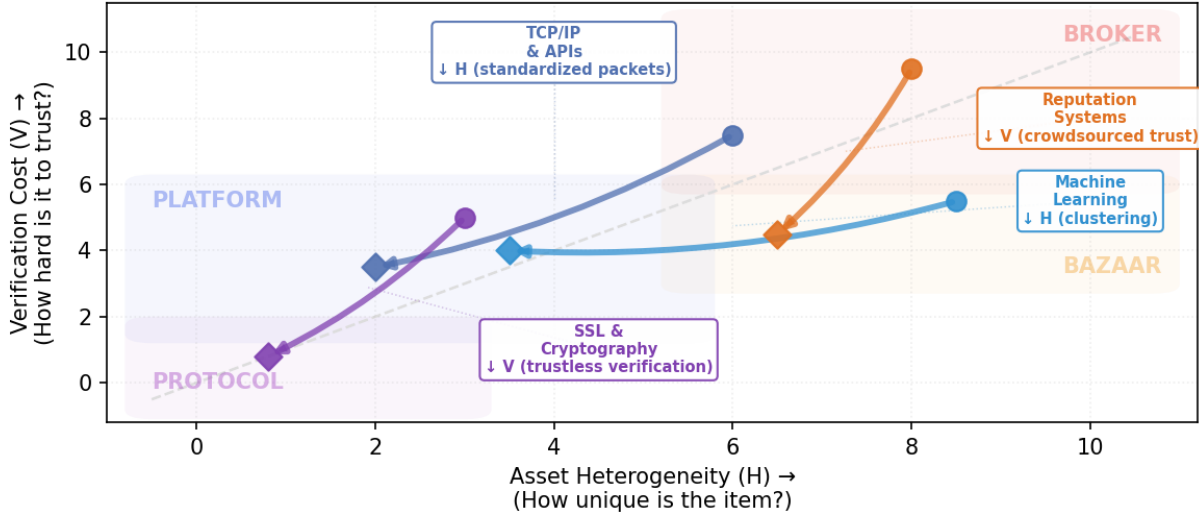


Figure 3: The Diagonal Migration: Technology Pushes Markets from Broker to Protocol

Technology	Variable Reduced	Mechanism
TCP/IP	H to 0	It didn't matter if you used a PC, Mac or Linux Server; the data "packet" became standardized. This created the Internet.
SSL / HTTPS	V to 0	The little lock icon. Cryptography verifies you are actually talking to your bank, not a man in the middle.
The API	H to 0	It turned messy internal databases into standardized "sockets" that other software can plug into.
Reputation Systems	V to 0	Google/eBay/Airbnb/Uber reviews. They crowdsourced verification, allowing us to trust strangers.
Machine Learning	H to 0	Clustering algorithms (K-means, DBSCAN, hierarchical clustering, Gaussian mixtures, etc.) can identify groupings among assets that appear diverse on the surface but share underlying similarities.

4 Friction and Frequency

H and V explain what a market trades and how it verifies value. But there is a third dimension that determines whether trust is cheap or expensive: how often participants interact. Markets can also be defined by the Frequency of Interaction. Frequency of Interaction can act as the "Memory" of a market. If you treat every interaction as the first time you've met (Low Frequency), you must pay the full cost of Verification and Setup (High Friction). If you interact frequently, you build "State" (Memory/Reputation), which allows you to bypass those costs.

Frequency can also override verification through "The Shadow of the Future," anticipated future interaction may sustain cooperation. If the frequency is high enough, you do not need verification

or even law. For example take the diamond market in New York where dealers trade high sums of diamonds with each other often over phone or chat based on a handshake agreement where settlement might be delayed. If counterparties trade with each other every day for decades, cheating once could risk being exiled forever. The same was also true for FTX, users logged on every day, traded, deposited, and withdrew and nothing went wrong. Even the initial concerns were often dismissed by those who had gotten used to the high frequency of interaction with the exchange.

The history of the web provides valuable lessons of increasing frequency to drive friction to zero.

Friction vs. Frequency: Internet and Bitcoin

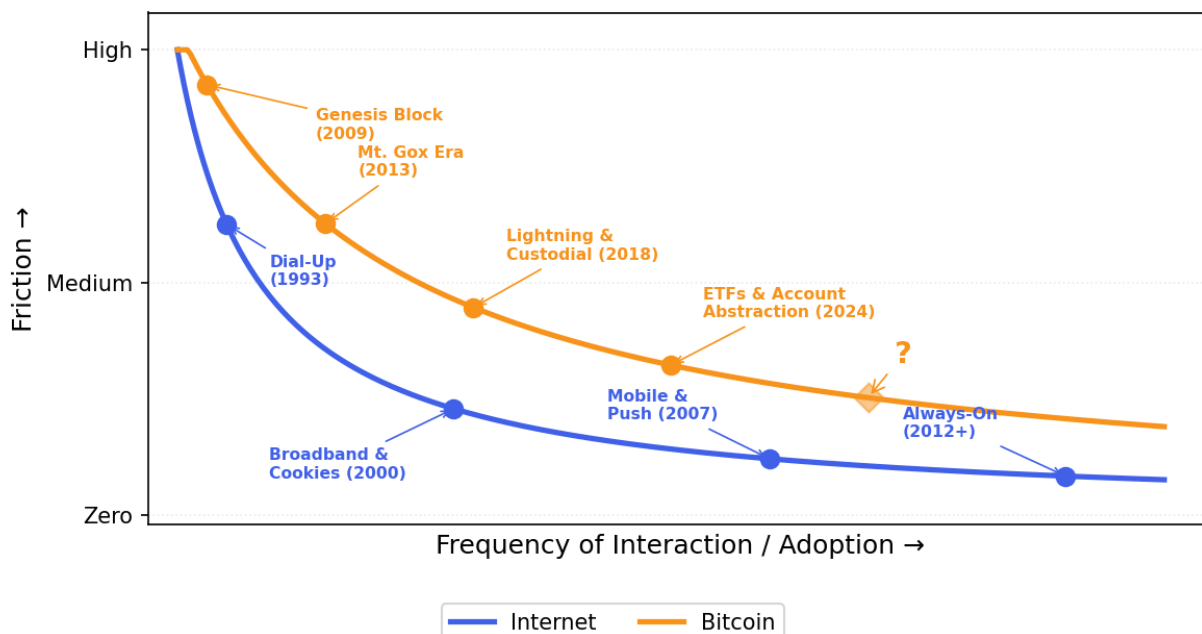


Figure 4: Friction vs. Frequency: Internet and Bitcoin

4.1 Phase 1: Dial Up (Low Frequency)

The State: You had to physically log on. The cost to initiate a session was high (noise, blocking the phone line).

The Result: You went online once a day for 30 minutes. You batched your emails. The friction of connecting kept the internet separate from real life.

4.2 Phase 2: Broadband & Cookies (Medium Frequency)

The Mechanism: The "Session Cookie" was the breakthrough. It meant the server remembered you. You didn't have to verify your password for every single click.

The Frequency Hack: The cookie simulates an infinite frequency connection. It tells the server, "I am the same person from 1 second ago."

The Result: Ecommerce. You could fill a shopping cart. Without the memory of high frequency, the cart would empty every time you clicked a new page.

4.3 Phase 3: Mobile & Push (Infinite Frequency)

The State: The iPhone made the connection permanent. You are never offline.

The Result: The Attention Economy. Because frequency is infinite, friction is zero. Apps can push notifications to you. The barrier between "Digital" and "Physical" collapsed.

Era / Tech	Low Frequency State (High Friction)	High Frequency Solution	The Resulting Friction Drop
Retail	Buying a song for \$0.99 (iTunes)	Subscription (Spotify)	Payment Friction to 0. You consume without deciding.
Finance	Wiring Money (SWIFT)	Netting / Clearing House	Liquidity Friction to 0. Counterparties only settle the difference at the end of day.
Computing	Loading a Program (CD-ROM)	SaaS / Cloud (Always on)	Update Friction to 0. Software updates happen while you sleep.
Crypto	Onchain Transaction	?	?

5 Crypto and Frequency

This is a dimension where crypto has a lot of room to develop. There have been a few examples of implementing a high frequency of interaction solution around protocols like Bitcoin such as:

Lightning state channels: 2 counterparties open a private channel and can transact back and forth millions of times almost instantaneously only touching the blockchain to settle the final balance. This has found some adoption in the B2B use case, predominantly between exchanges.

Custodial wrappers: Apps like Cash App use custodial lightning which improves user experience, higher frequency of interaction at the cost of verification. ETFs offer Bitcoin to a wide investor base who are familiar with the ETF instrument, allowing people to save in Bitcoin through an automatic 401K or other retirement plans.

Account Abstraction: This is the crypto equivalent of the Session Cookie. Instead of signing every single transaction (high friction, like entering your password on every click), a session key lets you authorize a dApp once and interact freely for a defined period. It simulates high frequency trust without repeated verification.

The pattern across all three is the same tradeoff the internet made. Lightning works but mostly between exchanges, not end users. Custodial wrappers reach mass adoption by reintroducing the trusted intermediary the protocol was designed to remove. Account abstraction delegates signing authority, which means trusting code you didn't write. Each solution increases frequency by giving back some verification. The question is whether crypto can find a frequency solution that doesn't quietly rebuild the platform.

6 Protocols are few and far between

6.1 The Friction Paradox

Protocols push Verification Cost toward zero by replacing trust with cryptographic proof. But every proof is friction the user must bear. Every seed phrase, every gas fee, every transaction signature is the cost of trustlessness made tangible. Platforms can raise venture capital precisely because they

capture value, and they spend that capital sanding down friction. Protocols, by design, have no central entity to do this.

This is the core tension: low V and low friction are at odds. Choosing trustlessness means choosing friction. A platform like Uber can spend billions making rides one-tap. A protocol has no one to write that check.

6.2 The Bootstrap Problem

Because no single entity captures value from a protocol, no single entity is incentivized to fund its development or polish its UX. The successful internet protocols sidestepped this: TCP/IP had DARPA, HTTP had CERN, SMTP had ARPANET. All funded by governments or academia before commercial platforms existed to compete. Bitcoin is the outlier: the token itself was the incentive mechanism. But even the "Fat Protocols" thesis, the idea that crypto tokens would solve the funding problem by letting protocols capture value directly, has largely bootstrapped speculation rather than usability.

6.3 The Protocol Graveyard

Protocols rarely die from technical inferiority. They die from friction, funding, and platform absorption.

Gopher was the dominant information protocol of the early '90s until the University of Minnesota announced licensing fees. HTTP was technically comparable but free. The web won.

XMPP powered Google Talk, Facebook Messenger, and others. Google embraced the open standard, gained users, then dropped federation for Hangouts in 2013. WhatsApp was built on a modified version of XMPP. The protocol community was hollowed out in the subsequent years. Embrace, extend, abandon.

RSS was beautifully simple: any site could publish a feed, any user could subscribe, no algorithms, no gatekeepers. But it couldn't be monetized. Google Reader's shutdown in 2013 killed the ecosystem, and social media absorbed the function.

OpenID offered self-sovereign identity: log in anywhere with one identity you control. It lost to "Sign in with Google" because managing your own identity is more friction than trusting a platform to do it for you.

The pattern: protocols that can't fund friction reduction get outcompeted by platforms that can.

6.4 The Gravitational Pull

The Gravitational Pull: Protocol → Platform

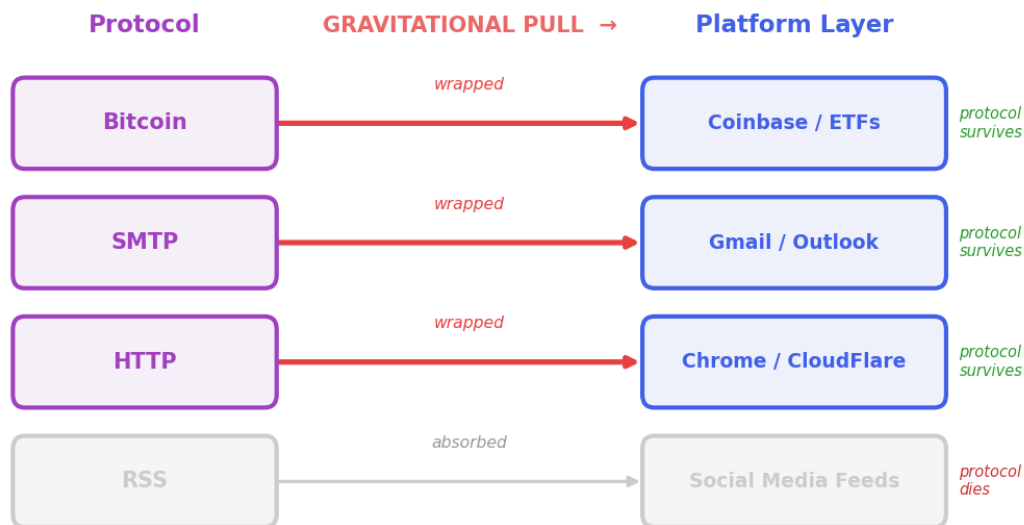


Figure 5: The Gravitational Pull: Protocol to Platform

Protocols that succeed at reducing V create a friction gap that platforms rush to fill. Coinbase custodies 85% of spot ETF Bitcoin. Passkeys replace seed phrases via Google and Apple. Account abstraction hides gas fees behind a familiar UX. Every one of these friction-reduction layers reintroduces trust and centralization, re-creating the platform structure the protocol was designed to eliminate.

The pull from protocol back toward platform is nearly gravitational. The rare protocols that resist it (TCP/IP, HTTP, SMTP, Bitcoin) share traits: they solved low- H , low- V problems, they were simple enough to ship fast, and they arrived before or alongside a unique window where platforms couldn't yet compete.