# Extropy Codex

## Comprehensive Technical Specification

Randall Gossett

Standalone synthesis of the Extropy Engine protocol

Formal value functions, validation, DAG substrate, governance, identity, tokens, implementation, and roadmap

Built from the book, repository, technical specification, peer review, Randall's Feedback, and Signal paper

---

Assembled 2026-05-13

# Extropy Codex: Comprehensive Technical Specification

Randall Gossett

## Extropy Codex: Comprehensive Technical Specification

**Version:** 1.0 (Codex synthesis) **Status:** Sandbox protocol specification, prototype implementation, open research program **Author:** Randall Gossett **Repository:** github.com/00ranman/extropy-engine **Public site:** lladnaros.com **Companion book:** *Unfuck the World for a Dollar*

## Table of Contents



## 1. Abstract and Reading Guide

### 1.1 Abstract

The Extropy Engine is a protocol for measuring, validating, and rewarding **verified entropy reduction** across eight operational domains. It is not a cryptocurrency, not a reputation system, and not a behavioral scoring service. It is a coordination layer that mints accounting tokens against evidence of measurable disorder reduction, where the evidence is witnessed by a routed pool of validators, where the resulting receipts are anchored in a causally ordered event graph, and where every participant retains sovereignty over their own intelligence stack, identity material, and local decision context.

The protocol's core claim is narrow: when value is operationalized as validated entropy reduction (under a domain-specific instrument, with a falsifiability condition specified in advance, and with the actor's identity prevented from entering the value calculation), the resulting coordination layer resists the Goodhart pressure that destroys conventional metrics. Reputation does not enter the value formula. Reputation density is a separate quantity that lives in a separate token. Rarity is a property of the loop class, not of the actor. Frequency-of-decay is an anti-grind penalty, not a quality score. Settlement-time is a normalized factor with a floor, not raw seconds.

The Codex consolidates the canonical specification (v3.1.2), the architectural foundations (Digital Autarky, the contribution graph, the three-layer separation), the formal governance theory (Randall's Feedback V4), the representational fidelity decay theory (the Signal paper), the DAG substrate paper, the peer review response, and the public narrative companion (the book *Unfuck the World for a Dollar*) into a single technical reference. The Codex is the reading order. Other materials remain available but are not required to understand the system.

### 1.2 What This Document Is

This is the standalone technical specification of the Extropy Engine. A technical reader should be able to read this document end-to-end and come away with a working understanding of:

- The protocol primitive (validated entropy reduction)
- The formal value functions (XP, CT, EP)
- The canonical eight domains and their measurement instruments
- The validation lifecycle and what V_c (validation confidence) means relative to F (frequency-of-decay)
- The DAG substrate, the personal signed local log, and the identity layer
- The DFAO governance model and the provisional defaults
- The token economy (six canonical tokens: XP, CT, CAT, IT, DT, EP)
- The Goodhart resistance theory (representational fidelity decay) and how it informs formula design
- Randall's Feedback as the formal governance foundation
- The repository implementation status, including microservices, ports, tests, and known gaps
- The peer review corrections that were applied to reach v3.1.2

### 1.3 What This Document Is Not

This is not the companion book, not the website, not the repository, and not the raw academic papers. It does not reprint full appendices. Where the underlying material is dense, this Codex synthesizes it and points back to the source by name. Where two source documents disagree, this Codex states which reading is canonical and why.

### 1.4 Reading Paths

For an executive reader, read sections 1, 2, 3, 5 (formulas only), 9, 13, and 19.

For a technical reader integrating against the protocol, read sections 4, 5, 9, 10, 11, 12, 14, and 15.

For a researcher or peer reviewer, read sections 4, 6, 7, 17, and 18, then drop into the formal mathematics in 5 and 7.

For a builder evaluating whether to adopt or extend the protocol, read sections 2, 3, 9, 13, 15, and 19.

### 1.5 Status Notes

The Extropy Engine is a sandbox protocol. The math is canonical as of v3.1.2. The reference implementation is partial: Phase 1 (protocol kernel) is complete and passing its happy-path integration test suite (12 of 12); Phase 2 (full distributed DAG, full retroactive validation pipeline, production cryptography) is in progress. The Codex labels prototype components and production-deferred components explicitly. Nothing in this document should be read as a deployed production claim.

## 2. What Problem the Extropy Engine Solves

### 2.1 The Proxy Worship Problem

Human coordination at scale runs on proxies. Test scores stand in for learning. Followers stand in for cultural contribution. Credit scores stand in for trustworthiness. Engagement metrics stand in for satisfaction. Citations stand in for scientific value. Compliance checklists stand in for safety. Each of these proxies began as a useful, low-cost approximation of an underlying condition. Each of them, once attached to status, money, or punishment, drifted from the condition it was built to track.

The drift is not a moral failure. It is the predictable outcome of putting incentive weight on a representation. When a representation acquires economic, social, or punitive value, rational actors stop optimizing for the underlying condition and start optimizing for the representation itself. This is Goodhart's Law in its full generality. It is Campbell's Law in its specific institutional form. It is the Cobra Effect in its perverse-incentive form. It is, when modeled dynamically, a differential equation whose decay rate depends on social load and domain-specific feedback latency. The Signal paper (*When the Signal Eats the Source*) formalizes this dynamic and gives it the name **representational fidelity decay**.

### 2.2 Why Existing Remedies Do Not Work

Reputation systems were the first attempt to fix the proxy problem. They fail because reputation itself is a proxy for trustworthiness, and once reputation has rewards attached, it drifts. Reputation laundering becomes a service category. The very best actors are not the ones with the highest reputation; they are the ones who learned to game whichever signals reputation systems collect.

Blockchain tokens were the second attempt. They fail because tokens optimize for liquidity, not contribution. The token graph becomes a speculation surface long before it becomes a coordination surface. Governance tokens concentrate among whales. Voting devolves to plutocracy or to apathy. The signal-to-noise on contribution evidence is dominated by financial signal.

DAO governance was the third attempt. It fails because participation is voluntary and weight is bought. The legitimate coordination layer that DAOs were supposed to be becomes either a shell over a small founding cartel or a forum where decisions are nominal and execution is unilateral.

Centralized compliance regimes are the fourth attempt. They fail because compliance metrics drift faster than regulators can update them. The EU AI Act's "human-made" content provision is a current example: the label has social load, the verification feedback is institutional and slow, and the predictable outcome is rapid fidelity collapse. The Signal paper develops this case study in detail.

### 2.3 What the Extropy Engine Proposes

The Extropy Engine proposes that the unit of value should be entropy reduction itself, measured under a domain-specific instrument, with a falsifiability condition specified before the claim is opened. Value is minted only when:

1. A claim is opened with a specified domain, a specified instrument, a baseline measurement, and a falsifiability condition.
2. The claim closes through a validation lifecycle that surfaces the evidence to a routed pool of validators or volunteer micro-validators.
3. Consensus on the entropy delta is reached and the validation confidence score (V_c) clears the threshold.
4. The minting service writes XP using the canonical formula, with rarity, frequency-of-decay, entropy magnitude, domain weighting, and settlement-time factor.
5. A 30-day retroactive window allows new evidence to confirm or burn the provisional mint.

Reputation never enters this calculation. Reputation density (rho) accrues separately, lives in a separate token (CT), feeds validator routing, and conditions governance influence; it does not multiply XP. This architectural invariant is the most important fact about the Extropy Engine. Everything else follows from it.

### 2.4 The Three-Layer Separation

The system presents three layers to three different audiences:

- **Layer 1 (User-facing):** Discounts, gamified feedback, the visible "character sheet." The user holds their own keys. The math is hidden. The experience is loyalty-program-shaped.
- **Layer 2 (Merchant-facing):** Free point-of-sale, customer pipeline, operational signal richer than standard KPIs. Merchants do not pay SaaS. Revenue captured through merchant-services fees comparable to or lower than incumbents.
- **Layer 3 (Engine):** The protocol itself. Six canonical tokens, validator consensus, the contribution graph, the causal DAG, the governance substrate, the formal mathematics.

The separation is intentional and is itself part of the Goodhart defense. End users cannot game a number they cannot see. Merchants cannot extract a metric whose internal weights they do not control. The protocol math is invariant under actor swap: same entropy reduction yields the same XP regardless of who reports it.

### 2.5 What This Is Not, Stated Plainly

The Extropy Engine is not a credit score. R (rarity) is a property of the contribution class, not of the actor; reputation does not multiply XP. It is not a social credit system; users hold their own keys and the state cannot write to a user's record. It is not surveillance capitalism; disclosure is voluntary and over-sharing does not earn more reward. It is not a typical loyalty program; the reward is real value saved, not engagement farmed. It is not a SaaS company; merchants are not charged for the POS layer, and users are not paywalled. It is not a finished economy, not a central truth authority, and not a claim that all human value reduces to a single naive scalar. It is a protocol for measuring, validating, iterating, and correcting.

## 3. Core Thesis: Entropy Reduction as Value

### 3.1 The Claim, Stated Precisely

The Extropy Engine treats **validated entropy reduction** as the operational unit of contribution value. Entropy here is used in a broad but disciplined sense:

- In thermodynamics, entropy describes physical disorder and energy dispersal (Shannon-Boltzmann, with units of nats, bits, or J/K).
- In information theory, entropy describes uncertainty and signal ambiguity.

- In social systems, the relevant analogue is coordination disorder: conflict, mistrust, institutional opacity, incentive failure, degraded feedback.
- In code, the analogue is structural disorder: bug density, complexity, maintainability degradation.

These are not identical quantities, and the framework does not pretend they are. The framework claims only that each domain has measurable forms of disorder, that each domain admits a measurement instrument and a falsifiability condition, and that value can be operationalized as validated reduction of that disorder.

### 3.2 The Shannon-Landauer-Bennett Grounding

The mathematical lineage anchoring the framework runs through Shannon (1948) on information entropy, Landauer (1961) on the thermodynamic cost of logical operations, and Bennett (2003) on the energetic equivalence of bit erasure. These results establish that information entropy and thermodynamic entropy are physically connected, not metaphorically related. The Extropy Engine uses this connection as its theoretical floor: the cross-domain comparison of entropy reductions is not arbitrary; it is grounded in established physics and information theory.

This is the strongest part of the theoretical edifice and also the part most often misunderstood. The framework does not claim that social entropy reduction is literally thermodynamic entropy reduction. It claims that each domain has a measurement protocol grounded in either physical or informational disorder, and that the protocol can be operationalized, audited, and falsified independently per domain. Aggregation across domains requires explicit normalization, which is addressed in section 5.

### 3.3 The Seven Questions a Value Protocol Must Answer

A protocol measuring contribution must answer seven questions for every claim it accepts:

1. What changed?
2. In which domain?
3. Was the disorder actually reduced (according to which instrument, with which baseline)?
4. Who can validate?
5. What would falsify the claim?
6. Did the validation close (with what confidence)?
7. Can the result be audited (and reverted if necessary)?

A system that cannot answer these seven questions is not measuring contribution. It is measuring narrative, status, or control. The Extropy Engine's loop lifecycle is the implementation of these seven questions: OPEN encodes 1, 2, 3, 5; VALIDATING encodes 4 and 6; CLOSED encodes 6 and 7; SETTLED encodes 7; FAILED and ISOLATED encode the falsification paths.

### 3.4 Why Multiplication

The XP formula multiplies its five terms rather than summing them. This is deliberate. Multiplication enforces necessity: if any term is zero (no rarity differential, no entropy delta, no domain-weighted evidence, no settlement), XP is zero. A submitter cannot compensate for missing evidence by maxing out rarity. A trivial submission cannot reach high XP by speed alone. The formula is multiplicative because the underlying claim is conjunctive: value is created only when all five conditions are simultaneously satisfied.

### 3.5 Why Falsifiability

Every claim opened in the loop must specify a falsifiability condition. This is the protocol's way of refusing to accept ideology disguised as contribution. The model is allowed to lose. That phrase is not rhetorical. It means that the engine, the formulas, and the validators all submit to external test. If the operationalization fails to track real entropy reduction, the operationalization is wrong. The honest enumeration of open gaps (section 19) is the explicit acknowledgment that the model has not yet been proven; it has been formally constructed, instrumented for measurement, and exposed to peer review.

## 4. Canonical Terminology and Correction Ledger

This section pins terminology. Every symbol in the Codex carries the meaning defined here. Where prior drafts of the technical specification, the book, the one-pager, or the repository used a different meaning, the correction is noted explicitly and is non-negotiable going forward.

### 4.1 The v3.1.2 Architectural Invariant

In version 3.1.1 of the technical specification, **R** was conflated with reliability or reputation, and **F** was conflated with falsifiability. Version 3.1.2 corrected both. The correction is an architectural invariant: it cannot be softened, walked back, or qualified.

- **R is Rarity.** R is action-class scarcity, a property of the loop, not the actor.
- **F is Frequency-of-decay.** F is an anti-grind penalty keyed on (submitter_id, claim_type, primary_domain, DFAO_id). F is not falsifiability and not feedback closure.
- **Reputation (rho) never enters XP.** Reputation density is a separate quantity. It lives in the CT formula and in validator routing. It does not multiply XP.

### 4.2 Canonical Symbols

| Symbol | Canonical name | Range | Lives in | What it is not |
|---|---|---|---|---|
| R | Rarity | [0.1, 10.0] | XP formula | Not reputation, not reliability, not validator history |
| F | Frequency-of-decay | (0, 1] | XP formula, CT formula | Not falsifiability, not feedback closure |
| rho | Reputation density | (0, infinity) | CT formula, validator routing | Not R, does not enter XP |
| Delta S | Entropy reduction magnitude | (0, infinity) | XP, CT | Domain-native units pre-normalization |
| w | Domain weight vector (eight elements) | scalar elements | XP formula | DFAO-tunable |
| E | Evidence vector (eight elements) | scalar elements | XP formula | Per-claim measurement |
| T_s | Normalized settlement-time factor | (T_floor, 1.0] | XP formula | Not raw seconds, must be normalized, has a floor and a cap |
| V_c | Validation confidence | [0, 1] | Validation aggregation | Not F. Surfaced by SignalFlow and the epistemology engine |
| c_l | Causal closure speed (per-domain constant) | (0, infinity) | Irreducible-form XP analogy | Calibration target, not an operational protocol constant |

| Symbol | Canonical name | Range | Lives in | What it is not |
|---|---|---|---|---|
| lambda | XP/CT decay rate | (0, 1) | Temporal service | Per-domain, governance-tunable |
| L | Local merchant loyalty multiplier | (0, infinity) | EP formula | Merchant-context-specific |

### 4.3 Canonical Tokens (Six Only)

The six canonical tokens are **XP, CT, CAT, IT, DT, EP**. There are no others. **GT and RT are legacy or deprecated names** that have appeared in older drafts of the repository and the technical specification. The CONTRIBUTION_GRAPH.md hard rule states it without ambiguity: "Exactly six. No others. No legacy names (no GT, no RT)." The token-economy package may accept legacy field names during a rollout window, but the canonical set is fixed.

### 4.4 Canonical Domains (Eight Only)

The eight canonical entropy domains are:

1. Cognitive
2. Code
3. Social
4. Economic
5. Thermodynamic
6. Informational
7. Governance
8. Temporal

Some earlier and public-facing documents listed **ecological** and **spiritual** in place of **code** and **temporal**. Those listings are stale. The canonical eight are defined by the EntropyDomain enum in the contracts package and by the database enum in the initialization script. They are what the implementation operates on, and they are what the Codex names everywhere.

### 4.5 Why Ecological and Spiritual Are Not Formal Domains

Ecological disorder is not a separate formal domain because ecological harms decompose cleanly into the canonical eight. Habitat degradation is thermodynamic and informational. Species extinction is informational (the loss of an irrecoverable genome) and thermodynamic. Ecosystem disruption is social (in the human-coordination sense), thermodynamic, and informational. A claim addressing an ecological harm is therefore a multi-domain claim with a weight vector across the canonical eight, not a single-domain claim in a separate ninth bucket. This is not a denial that ecological harm matters; it is a statement that ecological measurement instruments map to existing canonical instruments.

Spiritual experience is not a formal domain because it cannot be operationalized under a falsifiable instrument without collapsing into one of the canonical eight. Meaning-making is cognitive. Community formation is social. Ritual is governance. Awe and presence are cognitive. The framework treats "spiritual" as a meaningful philosophical and cultural category that can be discussed (the *God as Emergent Entropy Reduction* appendix does so explicitly), but spiritual experience is not minted as XP and not measured by a spiritual-domain instrument because no such instrument is grounded in the Shannon-Landauer-Bennett lineage that the rest of the framework depends on.

## 4.6 The Correction Ledger

This is the running list of terminology and content corrections that have been applied across the source material to bring everything into alignment with the v3.1.2 canonical. The Codex incorporates these corrections; the source files may still carry uncorrected passages.

| ID | Source | Issue | Correction |
|---|---|---|---|
| C1 | Book chapter 8, CLOSED state passage | "R: from the contributor's history (reputation service)" | "R: from rarity assessment of the contribution pattern using domain baselines, local DFAO weighting, and validator review; R is a property of the loop, not the actor." |
| C2 | Book appendix A | "R rewards who you are: demonstrated reliability in this domain" | "R rewards the rarity of what was done: how scarce or hard-to-replicate this contribution pattern is in the current network context. R is a property of the loop class, not of the actor." |
| C3 | One-pager | Domain list "thermodynamic, informational, social, economic, ecological, governance, cognitive, spiritual" | Canonical eight: "cognitive, code, social, economic, thermodynamic, informational, governance, temporal." |
| C4 | Tech spec section 10.5 | "Aggregation produces F" (naming collision with frequency-of-decay) | Aggregation produces V_c (validation confidence). F in the XP formula is frequency-of-decay only. |
| C5 | Tech spec section 6.5 | "log(1/T_s) preserves boundedness" (mathematically false) | The log term grows logarithmically and is unbounded above as T_s approaches zero; a minimum T_s floor and an explicit cap are required at the protocol level. |
| C6 | God paper gloss | F described as "feedback closure strength" | F in the canonical XP formula is frequency-of-decay. Feedback closure strength is a separate concept and is captured in V_c or in C(t) from the Signal paper, not in F. |
| C7 | Repository README and token-economy package | Listing GT and RT alongside canonical tokens | GT and RT are legacy/deprecated names. Canonical six are XP, CT, CAT, IT, DT, EP. |
| C8 | Tech spec section 6.1 | R defined as "reliability coefficient earned through validated contributions" | R is rarity, looked up from per-domain RARITY_DEFAULTS, governance-tunable, a property of the loop class. |
| C9 | Tech spec section 6.2 | F defined as "falsifiability score, how testable the claim | F is frequency-of-decay. "Testability of a claim" is a |

| ID | Source | Issue | Correction |
|---|---|---|---|
| | | is" | separate, distinct concept that lives on claims and feeds into routing and governance dashboards; it is not the XP multiplier. |
| C10 | Tech spec, multiple sections | T_s referred to as raw seconds in xp-mint | T_s is a normalized settlement-time factor in (T_floor, 1.0]. It is unitless. Callers must normalize at the boundary. |

### 4.7 Naming Hygiene Going Forward

The Codex uses LaTeX-style names for symbols in math contexts (R, F, rho, c_l, V_c) and full English names in prose. When a passage refers to "the reputation density variable," the symbol is rho. When a passage refers to "the validation confidence aggregation," the symbol is V_c. When a passage refers to "the rarity multiplier," the symbol is R. When a passage refers to "the frequency-of-decay penalty," the symbol is F. The Codex will not use F as a stand-in for falsifiability, feedback closure, or any other concept, and will not use R as a stand-in for reputation.

## 5. Mathematical Foundation: XP, CT, EP, Domains, and Normalization

### 5.1 The Canonical XP Formula

The canonical experience-points formula, version 3.1.2, is:

[ XP = R F S (w E) (1 / T_s) ]

Where:

- R is rarity, in [0.1, 10.0], looked up from per-domain RARITY_DEFAULTS, governance-tunable. R is a property of the loop class, not of the actor.
- F is frequency-of-decay, in (0, 1], with F = 1.0 at first occurrence of the (submitter_id, claim_type, primary_domain, DFAO_id) fingerprint. F decays per submission per fingerprint under either a log-tail curve or a harmonic curve, with the curve selectable per governance configuration.
- Delta S is the verified entropy delta. Delta S must be strictly positive for the formula to mint. Domain-native units pre-normalization (J/K for thermodynamic, bits for informational, normalized scalars for cognitive, social, etc.). Normalization is required before cross-domain aggregation; see 5.6.
- w dot E is the dot product of the eight-element domain weight vector and the eight-element evidence vector. The weight vector is set per DFAO and is governance-tunable. The evidence vector is measured per claim.
- log(1/T_s) is the settlement-time term. T_s is a normalized settlement-time factor in the interval (T_floor, 1.0], where T_floor is a governance-tunable lower bound. The log term grows as T_s decreases; the floor prevents runaway values, and an explicit upper cap on the log term applies (default cap: log(1/T_floor)).

The formula is enforced in a single source file (packages/xp-formula/src/index.ts) that every minting service imports. There are no re-implementations. The minting service stamps each mint with FORMULA_VERSION = 'canonical-v3.1.2'.

### 5.2 R: Rarity in Detail

R is a per-domain table of rarity multipliers for canonical action classes. Calibration anchors:

- 0.1 to 0.5: routine, high-frequency actions (logging a meal, marking a chore done).
- 0.5 to 1.5: common contributions (a documented code commit, a community check-in).
- 1.5 to 3.0: substantive contributions (a new instrument calibration, a working mediation).
- 3.0 to 6.0: scarce contributions (a domain breakthrough, a novel governance protocol).
- 6.0 to 10.0: essentially irreplaceable (a foundational result, a critical bug fix without which the network would have failed).

R is looked up from a domain-keyed default table and is then weighted by the local DFAO. The DFAO weighting allows a small community to value local rarity (a working repair shop in a small town is rarer than the same shop in a city) without breaking the universal table. R is not actor-bound: the same loop class gets the same R regardless of who submits it. R can be revised through governance proposal, with the revision applying to future loops only.

The pre-v3.1.2 reading of R (as "reliability earned through validated contributions") is the reputation laundering definition that v3.1.2 explicitly rejected. Any text in the source documents that uses that older definition is stale and is corrected in the correction ledger.

### 5.3 F: Frequency-of-Decay in Detail

F penalizes repeated instances of the same action class. The fingerprint is the tuple (submitter_id, claim_type, primary_domain, DFAO_id). For each fingerprint, F starts at 1.0 and decays per occurrence.

Two decay curves are supported, selectable per governance configuration:

- **Log-tail decay:** F(n) = 1 / (1 + log(n)) where n is the count of prior occurrences. Smooth decay with a long tail.
- **Harmonic decay:** F(n) = 1 / (n + 1) where n is the count of prior occurrences. Faster early decay.

Reset rules: F resets to 1.0 when the (submitter_id, claim_type, primary_domain, DFAO_id) fingerprint changes. A submitter who legitimately moves into a new DFAO context resets F for that fingerprint, but cannot reset by trivially varying claim_type or primary_domain (the validation pool can reject the variation as non-substantive).

F is not falsifiability. Falsifiability is a per-claim quality criterion specified at OPEN; it conditions whether the claim can be accepted at all, but it does not appear as a multiplier in the XP formula. F is not feedback closure. Feedback closure is captured at the validation aggregation step as V_c.

### 5.4 Delta S: Entropy Reduction Magnitude

Delta S is the measurable disorder reduction effected by the loop. It is measured in domain-native units before normalization:

- Thermodynamic: joules per kelvin (J/K), or kilowatt-hours saved against a baseline, or material-flow reduction expressed in entropy-equivalent terms.
- Informational: bits of uncertainty removed, or signal-to-noise improvement expressed in bits.
- Cognitive: a normalized scalar derived from compression of explanation length, error rate reduction in a learning task, or a structured rubric for understanding gains.
- Code: cyclomatic complexity reduction, bug density reduction, test-coverage improvement, or formal verification coverage gain.

- Social: trust-network cohesion delta from validated network metrics, conflict-incident reduction over a windowed baseline.
- Economic: utilization improvement, throughput improvement, allocation efficiency gain.
- Governance: decision latency reduction, reversal-frequency reduction, participation-quality improvement.
- Temporal: cycle-time reduction, wait-time reduction, synchronization improvement.

Each measurement requires a baseline, an instrument, an error bound, and a falsification condition. The instrument is specified by the DFAO and is governance-tunable. The Delta S submitted with a claim is the raw domain-native value; the normalization step converts it to a comparable scalar before XP is computed.

### 5.5 w dot E: The Eight-Domain Weighting

The weight vector w is an eight-element governance-tunable vector that the DFAO sets to express its local priorities. A research DFAO might weight cognitive and informational heavily. A municipal DFAO might weight social and governance. A workshop DFAO might weight thermodynamic and code.

The evidence vector E is an eight-element vector measured per claim, expressing how much of the claim's reduction landed in each domain. Most real contributions are multi-domain. A working software project has code, cognitive, and economic components. A neighborhood mediation has social, governance, and informational components.

w dot E is the scalar dot product. The dot product is the protocol's way of saying "the value of this contribution depends on how well its evidence matches what this community cares about, but the canonical weights remain visible and auditable."

### 5.6 Cross-Domain Normalization

The peer review's strongest substantive criticism is that cross-domain Delta S aggregation requires normalization. The Codex resolves this as follows.

Each domain has a normalization function $\text{normalize\_d}(\text{delta\_s\_raw})$ that converts the domain-native unit to a unitless scalar in a comparable range (typically [0, infinity) with a calibration point of 1.0 representing a "standard" contribution magnitude). The normalization function is governance-tunable per domain and is published in the domain's operationalization document.

The normalization is not a claim that thermodynamic entropy and cognitive entropy are physically equivalent. It is a claim that, for the purpose of multi-domain XP aggregation, each domain's reduction can be expressed as a multiple of a domain-specific reference contribution. The reference contributions are calibrated against base-rate domain instruments and are revised through governance proposal.

Open problem: as of v3.1.2, the calibration of normalize_d across domains is preliminary. Operational data from a running deployment will refine the calibrations. This is acknowledged in section 19 as a P1 open engineering gap.

### 5.7 T_s: Normalized Settlement-Time Factor

T_s expresses how fast a claim moved through the lifecycle, normalized to the domain's typical settlement time. T_s is unitless, in the interval (T_floor, 1.0], with T_s = 1.0 representing the slowest acceptable settlement and T_s near T_floor representing the fastest acceptable settlement.

The log term log(1/T_s) is:

- Zero when T_s = 1.0 (no time bonus).
- Positive and growing as T_s decreases.
- Capped at log(1/T_floor) (the explicit cap that the peer review correctly identified as missing in the original spec).

T_floor is governance-tunable. Default: 0.01. With default T_floor, the maximum value of log(1/T_s) is approximately 4.6.

Important: T_s is not raw seconds. The repository's earlier code accepted a field named `settlementTimeSeconds`, which was misnamed. The canonical field is `settlementTimeFactor`, and callers must normalize before passing it to the formula. The convenience wrapper `computeTimestampDecay(deltaT, lambda)` may be used to derive T_s from raw elapsed seconds via T_s = exp(-lambda * deltaT), but lambda is a governance-tunable per-domain decay constant, not a silent default.

### 5.8 The Irreducible Form

The technical specification introduces an "irreducible form" of the XP relation:

[ XP = S / c_l^2 ]

Where c_l is a domain-specific causal closure speed. Default values per the contracts package:

| Domain | c_l |
|---|---|
| cognitive | 1e-6 |
| code | 1e-4 |
| social | 1e-3 |
| economic | 1e-2 |
| thermodynamic | 1e-4 |
| informational | 1e-5 |
| governance | 1e-3 |
| temporal | 1e-6 |

The irreducible form is explicitly framed as a structural analogy to E = mc^2, not as a physical law. It is a future calibration target. It is not the operational formula. The canonical operational formula is the five-term version in 5.1. The relationship between the two forms is not yet derived; the peer review correctly flagged this. The Codex resolves the ambiguity by stating: the operational formula is the five-term version; the irreducible form is a structural analogy whose practical use is to anchor the c_l calibration table for future revisions of the operational formula.

### 5.9 The CT Formula

The Contribution Token formula is:

[ CT = C F E ]

Where:

- C is the contributor's capability (a measured competence in the domain).
- F is the same frequency-of-decay term as in XP.
- rho is reputation density (per-domain, accrued through validated contributions; this is where reputation legitimately enters the system).

- Delta is the entropy delta in the CT context (analogous to but not necessarily identical to Delta S in the XP context; CT can be minted for sustained contribution patterns rather than per-loop).
- E is the eight-domain weighting / essentiality factor for the contribution pattern.

CT is identity-bearing. XP is not. CT can be locked up, used as collateral for governance influence, used to bridge across DFAOs, or (with transfer friction and a 14-day lockup default) transferred. XP is per-action and decays.

The token-economy endpoint POST /ct/mint accepts both canonical field names (capability, frequencyOfDecay, reputationDensity) and legacy field names (context, feedbackClosure, reputation) during the v3.1.2 rollout window. Legacy fields are deprecated and will be removed in v3.2.

### 5.10 The EP Formula

Emergence Points are the merchant-loyalty token at Layer 1:

[ EP = XP L ]

Where L is the local merchant loyalty multiplier, set per merchant and bounded by governance defaults. EP is local: it converts to merchant-side discounts and is not transferable across merchant networks. EP is the user-facing reward; it is what the customer experiences as a "loyalty discount" without seeing the underlying XP arithmetic.

### 5.11 Decay, Friction, and Lockup

The token economy has three time-evolving parameters:

- **XP decay rate (lambda_XP):** Default 0.01 per 30-cycle period (approximately 1% per month). Per-domain governance-tunable.
- **IT decay rate:** 5% per month. Reflects the principle that governance weight should not accrue indefinitely without continued contribution.
- **Transfer friction (delta):** 2% loss per CT transfer. Discourages CT churn while permitting legitimate cross-DFAO movement.
- **CT lockup window:** 14 days default. Prevents CT spammed in and instantly transferred out.

All four parameters are governance-tunable per DFAO within parent safety bounds.

### 5.12 Dimensional Analysis

The peer review correctly flagged the absence of dimensional analysis. The Codex resolves this as follows:

- After normalization, all five terms in the XP formula are unitless scalars.
- XP is therefore a unitless scalar, comparable across domains and DFAOs after normalization is applied.
- Pre-normalization Delta S carries domain-native units; the normalization step is the dimensional bridge.
- The irreducible form XP = Delta S / c_l^2 carries Delta S's units divided by (c_l units)^2. Because c_l is presented as a structural analogy rather than a physical constant, this dimensional concern is acknowledged but does not affect the operational formula.

# 6. Meaning Drift and Goodhart Resistance

## 6.1 Why This Section Is Load-Bearing

The Extropy Engine's most important design decision (R is rarity, not reputation; reputation does not enter XP) was not made on aesthetic grounds. It was made because the underlying theory of why metrics fail predicts exactly the failure mode that putting reputation in XP would produce. That theory is **representational fidelity decay**, developed in detail in the paper *When the Signal Eats the Source*. This section condenses the theory and shows how it informs the protocol's formula design.

## 6.2 The Central Dynamic

When a descriptive label acquires incentive value (economic, social, or punitive weight), rational actors stop tracking the underlying territory and start tracking the label boundary. The label becomes the optimization target. The representation drifts from the reality it was built to track. This is Goodhart's Law in its full generality.

The Signal paper formalizes the drift as a differential equation:

[  = -k S(t) F(t) + r C(t) (1 - F(t)) ]

Where:

- F(t) is the representational fidelity at time t (in [0, 1]; 1 means perfect map-to-territory correspondence; 0 means full semantic capture).
- S(t) is the social load on the label (the aggregate incentive, status, and punitive weight attached to it).
- C(t) is the corrective feedback strength (the rate at which external validation pulls fidelity back toward the territory).
- k is the domain's susceptibility coefficient (how easily this domain's labels drift).
- r is the restoration rate (the rate at which corrective feedback restores fidelity when applied).

The first term drives decay. The second drives recovery. Most existing treatments of Goodhart's Law address only decay. The Signal paper's contribution is the bidirectional model: fidelity can recover, but only when corrective feedback is structurally present and the domain admits enough feedback latency to apply it.

## 6.3 The Three-Domain k-Parameter Taxonomy

Domains differ systematically in their susceptibility to fidelity decay, primarily because they differ in feedback latency:

- **Physical domain (k ~= 0.02):** Strong causal correction; fast feedback; reality pushes back quickly. Engineering metrics, physical measurements, thermodynamic instruments.
- **Institutional domain (k ~= 0.15):** Slow, bureaucratically mediated feedback. Peer review, credentialing, regulatory metrics.
- **Social/moral domain (k ~= 0.45):** Weakest correction; near-zero C(t). Status, identity claims, moral labels, content authenticity claims.

The k values are empirical estimates calibrated against documented Goodhart episodes; they are not arbitrary. The taxonomy explains why some metrics (atmospheric CO2 measurements) remain stable while others (university rankings, AI-content authenticity) collapse rapidly: the underlying domain's k determines the decay rate, and the structural availability of C(t) determines whether recovery is possible.

## 6.4 The Four Observable Phases

Every label that acquires incentive value passes through up to four phases:

1. **Descriptive phase:** Low S(t), high F. The label is functional. People use it to track the underlying condition.
2. **Standardization phase:** Rising S(t); institutions invest in verification. F may improve as measurement gets formalized.
3. **Capture phase:** S(t) saturates; gaming proliferates; F decays. The actors with the most to gain figure out how to optimize for the label without optimizing for the condition.
4. **Collapse or correction phase:** Either semantic emptying (the label no longer tracks anything; "thought leader" is an example) or an external C(t) shock triggers partial restoration (a regulator updates the instrument; a credible peer-review process corrects the field).

The four-phase trajectory is not deterministic; some labels stabilize in phase 2 because the underlying domain supports continuous corrective feedback. Most labels in the social/moral domain are observed to pass through all four phases on a multi-decade timescale, with phase 3 dominating.

## 6.5 The Case Study: AI Content Detection

The Signal paper applies the model to the contemporary "human-made content" label, especially under the EU AI Act Article 50 enforcement mandate. The prediction is that:

- The label's S(t) is rising rapidly under regulatory enforcement.
- The domain is social/moral (k ~= 0.45) rather than physical (k ~= 0.02).
- C(t) is institutionally mediated and slow.
- The expected outcome is rapid phase-3 capture and phase-4 collapse, with the regulatory mandate accelerating rather than preventing the decay.

This prediction is testable. The model is allowed to lose: if "human-made" labels retain fidelity over a five-year horizon under sustained enforcement, the k = 0.45 estimate for the social/moral domain is wrong and the model needs revision.

## 6.6 Self-Application

The Signal paper applies its own model to itself. It claims k ~= 0.15 for itself (institutional domain). The corrective feedback C(t) is peer review, falsification attempts, and empirical challenge. The paper is allowed to lose. The model is not exempt from its own mechanism. This is the same epistemic posture the Extropy Engine adopts at the protocol level.

## 6.7 How This Informs the XP Formula

The representational fidelity decay theory is the theoretical foundation for the architectural invariant that reputation cannot enter XP. The chain of reasoning is:

1. Reputation is a representation (a label). It has social load. Its k value is high (institutional or social/moral domain).
2. If reputation enters the value formula, then optimizing for value forces optimization for reputation, which forces gaming of the reputation signal.
3. Reputation-gaming is not a hypothetical: every reputation system that has been deployed at scale has been gamed. The empirical evidence is overwhelming.

4. Therefore, the value formula must be insulated from the reputation representation. R is rarity (a property of the loop class) and rho is reputation density (a property of the actor); they are kept in separate variables, in separate tokens, with separate downstream effects.

This is not a moral choice. It is a mechanical consequence of the theory. If the theory is wrong, then putting reputation in XP would not cause harm and there would be no reason to keep it out. The framework predicts harm; the architectural invariant is the prediction's implementation.

### 6.8 How This Informs Governance

The Signal theory predicts that any metric the Engine uses to mint XP will become a target. The Engine's response is structural, not exhortative:

- **Falsifiability conditions in advance:** Every claim specifies what would falsify it at OPEN. Drift cannot be hidden because the falsification path is documented before the validation runs.
- **Retroactive burn:** A 30-day window after CLOSED allows new evidence to confirm or burn a provisional mint. Drift caught after the fact gets corrected.
- **Validator collusion penalties:** Validators whose consensus is later contradicted take reputation penalties. The validation pool has skin in the game.
- **Instrument update cycles:** Governance can revise the measurement instrument when drift is detected, with the revision applying to future loops only (preserving DAG immutability).
- **Goodhart pressure as diagnostic fuel:** Patterns of claim failure, validator disagreement, and domain drift are surfaced as governance signals, not hidden as failures. The peer review specifically commended this stance.

The Codex treats Goodhart pressure as a permanent feature of the operating environment, not a bug to be eliminated. The protocol does not claim immunity; it claims structural resistance and visible correction.

### 6.9 Relation to F in the Formula

The F term in the XP formula (frequency-of-decay) is one specific instance of Goodhart resistance at the formula level: it prevents grinding the same action class to inflate XP. F is not the only line of defense; it is one of several. The full set:

1. R as a loop-class property (not actor-bound) prevents reputation laundering through XP.
2. F as a per-fingerprint decay prevents action-class grinding.
3. Delta S requires a domain-instrument-baseline-falsifiability record; trivial submissions are filtered at OPEN.
4. w dot E requires evidence across the eight domains; single-domain trivia cannot reach high XP.
5. T_s with a floor and a cap prevents log-term inflation through arbitrarily fast settlement.
6. V_c thresholding at consensus prevents low-confidence claims from minting.
7. Retroactive burn corrects drift after the fact.

Each of these mechanisms targets a specific failure mode predicted by the representational fidelity decay theory. The formula's complexity is not aesthetic; it is the operational consequence of the theory.

# 7. Randall's Feedback: Formal Foundation for Emergence-First Governance

## 7.1 What Randall's Feedback Is

**Randall's Feedback (RF)** is the formal coordination and validation framework that grounds the Extropy Engine's governance layer in active inference, feasible reward sets, Byzantine-robust peer prediction, and dynamic Goodhart resistance. RF has gone through four major versions:

- V1 (2024): introduced the core axioms and the SIIR (Sense-Infer-Integrate-Respond) loop.
- V2 (2024): formalized the Complexity Thermostat.
- V3 (2025): added the bootstrap governance phasing.
- V4 (2026): the current version, delivering six theorems with formal proofs that ground RF in the Free Energy Principle and modern peer-prediction literature.

The full V4 paper is *Formal Foundations for Emergence-First Governance*. This Codex section synthesizes V4 down to the load-bearing claims and shows how they support the Extropy Engine specifically.

## 7.2 The Four Axioms

RF V4 rests on four axioms, each formalized in mathematical content rather than slogan form.

**Axiom 1: Feedback Primacy.** The governance state evolves according to a feedback dynamical system: S_{t+1} = S_t + alpha * F(S_t, E_t, G_t), where F is the feedback functional, alpha is the learning rate, S_t is the state at time t, E_t is the environmental signal, and G_t is the current goal representation. All governance transitions are mediated by F; no state change occurs outside this feedback loop.

**Axiom 2: Goal Evolution.** Goals evolve via G_{t+1} = G_t + beta * L(G_t, O_t), where L is the goal-update operator, beta is the goal learning rate, and O_t is an observation measured independently of the reputation state. L is required to be Lipschitz continuous (kappa in (0, 1)) and to contract toward a fixed point G* that is invariant under admissible perturbations of the reputation state.

**Axiom 3: Fractal Self-Similarity via Functors.** Governance patterns at the agent level compose into patterns at the sub-DAO level and the protocol level through functors in the FeedLoop category that preserve the SIIR loop structure. Formally: there exists a scale functor F: FeedLoop_micro -> FeedLoop_macro and a natural transformation eta: Id => F such that the SIIR loop commutes at every scale.

**Axiom 4: Cross-Domain Applicability.** The axioms and theorems of RF apply across governance domains via domain-specific instantiation. Each domain specifies the state space S, the environment E, the contribution metric m, and the goal representation G while satisfying the structural requirements of Axioms 1 to 3.

These axioms are not aspirational. The subsequent theorems treat them as the formal constraint on the class of systems to which RF applies.

## 7.3 The SIIR Loop as Dynamical Primitive

The SIIR loop (Sense, Infer, Integrate, Respond) is the fundamental dynamical primitive of RF governance. Each cycle has four phases:

- **Sense.** The agent observes the governance environment, collecting sensory data about other agents' contributions, validation outcomes, and governance state.
- **Infer.** The agent updates its internal belief model by minimizing variational free energy given the new observations. In Bayesian-mechanics terms, internal states flow down the free-energy gradient.

- **Integrate.** The updated beliefs are integrated with the agent's goal representation to form an action plan. In active-inference terms, policy selection via expected free energy minimization: P(u) = softmax(-G(u)).
- **Respond.** The agent executes the selected action (contribution, vote, validation), which modifies the environment and generates new sensory signals for other agents, completing the feedback loop.

By Axiom 3, the SIIR loop operates simultaneously at multiple governance scales: individual agents, sub-DAO contribution clusters, and protocol-level systems all execute the SIIR cycle on appropriately aggregated data.

### 7.4 The Six Theorems

RF V4 establishes six formal contributions, each with a complete proof in the full paper. The Codex states the claims; the formal proofs reside in the appendix-grade source document for readers who require them.

**Theorem V4-1 (RF Governance Convergence).** Agents satisfying the particular partition conditions (Friston et al. 2023) converge asymptotically to a governance attractor via Hamiltonian matching under the Free Energy Principle. The agents' internal generative model converges to the governance Hamiltonian H_G.

**Theorem V4-2 (Collective Goal-Update Admissibility).** Necessary and sufficient conditions on the goal-update operator L are established for the collective system to remain in a Banach fixed-point contraction on the product of feasible reward sets (Metelli et al. 2023). The result: governance can update goals over time without collapsing into chaos, provided the pairwise overlap condition holds.

**Theorem V4-3 (Dynamic Goodhart Resistance).** The discrepancy process between the contribution metric and the evolving goal remains light-tailed under bounded goal-update rates and Lipschitz metric continuity. The result: Goodhart pressure cannot accumulate unboundedly under RF governance.

**Theorem V4-4 (Unhackability Under Constrained Policy Classes).** RF's admissibility conditions restrict the strategy space to a finite (or effectively finite) set Pi_RF, enabling non-trivial unhackable metric-goal pairs in the sense of Skalse et al. 2022.

**Theorem V4-5 (Convergent Validation Without Ground Truth).** Byzantine-robust peer prediction is extended to n validators with linear collusion tolerance. The result: consensus can be reached on claim validity without a trusted oracle, provided the validator population is structured as RF specifies.

**Conjecture V4-6 (MeritRank Trilemma Resolution).** RF achieves approximate Generalizability, approximate Trustlessness, and epsilon-Sybil-Resistance simultaneously. This is stated as a conjecture in V4; the formal proof is open.

### 7.5 Connection to Extropy Engine

The Extropy Engine's governance layer is the operational instantiation of RF. Specifically:

- The contribution graph (the unified frame in which all submissions and validations are contributions) is the operational form of Axiom 1 (Feedback Primacy).
- The DFAO governance configuration (quorum, conviction voting, parameter tuning) is the operational form of Axiom 2 (Goal Evolution), with the Lipschitz constraint enforced by parameter-update rate limits.
- The fractal DFAO scales (NANO, MICRO, MESO, MACRO, PLANETARY) are the operational form of Axiom 3 (Fractal Self-Similarity).
- The eight-domain canonical taxonomy is the operational form of Axiom 4 (Cross-Domain Applicability).

The architectural invariant that reputation does not enter XP is the operational form of Theorem V4-3 (Dynamic Goodhart Resistance): keeping the value formula insulated from the reputation representation is exactly the structural condition that bounds the discrepancy process.

The 30-day retroactive burn window and the validator-consensus model are the operational form of Theorem V4-5 (Convergent Validation Without Ground Truth): validation proceeds without a trusted oracle, with Byzantine tolerance proportional to the validator pool.

### 7.6 The Two-Phase Bootstrap

RF V4 specifies a two-phase governance bootstrap that the Extropy Engine inherits:

- **Phase 1 (Contribution-Only):** All agents have equal standing. Contributions are validated purely on content, without reputation weighting. Phase 1 duration is determined by the sample complexity bound: at least Omega(H^3 * S * A / epsilon^2) observations before the first goal update is triggered.
- **Phase 2 (Reputation-Weighted):** After sufficient contribution data has been collected, the system transitions to reputation-weighted validation. The transition is triggered when the estimated feasible reward set intersection G* has sufficiently small Hausdorff diameter.

This phasing is the formal justification for why the Engine cannot launch with reputation weighting from day one: the sample-complexity bound must be satisfied first.

### 7.7 The Complexity Thermostat

The RF Complexity Thermostat (named CT in RF V4; not to be confused with Contribution Token CT in Extropy) regulates the unified entropy measure S_RF, which combines goal entropy, contribution entropy, and validation entropy. When S_RF exceeds an upper threshold (governance is too chaotic), the Thermostat tightens admissibility. When S_RF falls below a lower threshold (governance is too rigid), the Thermostat relaxes admissibility. This is the formal mechanism by which RF maintains the system in the productive region between collapse and disorder.

In the Extropy Engine, the Complexity Thermostat corresponds to the governance-tunable parameters in the provisional defaults table (quorum, deliberation period, reward escalation, retroactive validation window). These parameters can be adjusted by governance proposal as the system's entropy state shifts.

### 7.8 Honest Limitations

RF V4 is deliberately stratified by confidence level. Theorems V4-1 through V4-5 are stated with complete proofs. Conjecture V4-6 is stated as a conjecture; the formal proof is open. The framework distinguishes proven results from conjectured ones explicitly, refusing to elide the distinction.

The Extropy Engine inherits this posture. The governance layer rests on five proven theorems and one conjecture. The conjecture is the strongest claim (simultaneous achievement of generalizability, trustlessness, and Sybil resistance) and is therefore the most likely to require revision under adversarial pressure.

## 8. The Eight Domains and Their Instruments

### 8.1 The Canonical Eight

The Extropy Engine operates on eight canonical entropy domains, each defined by its measurement instrument, its measurement protocol, its known failure modes, and its falsification condition. The eight are:

1. Cognitive

2. Code
3. Social
4. Economic
5. Thermodynamic
6. Informational
7. Governance
8. Temporal

The list is enforced by the EntropyDomain enum in the contracts package and by the database enum in the initialization script. Anything outside the eight is not a domain.

### 8.2 Cognitive Entropy

Cognitive entropy is the disorder in understanding, explanation, learning, conceptual compression, and mental-model coherence. A reduction in cognitive entropy is a contribution that makes a previously opaque domain comprehensible, that compresses a long explanation into a shorter one without loss of fidelity, or that resolves a previously unclear conceptual question.

Instruments: explanation length compression (under a content-preserving constraint), error rate reduction in a structured learning task, time-to-mastery improvement under a standardized curriculum, structured rubric assessments scored by validators.

Known failure modes: “explainer slop” (long-form content that feels clarifying but adds no compression), pseudo-pedagogy (content optimized for engagement metrics rather than understanding), authoritative-tone substitution (cognitive entropy claims that rely on authorial reputation rather than measurement).

Falsification condition: a claim of cognitive entropy reduction is falsified if, after exposure to the content, independent learners show no improvement on a pre-specified comprehension instrument relative to a control group.

Maturity: medium. Cognitive measurement is harder than thermodynamic but easier than social.

### 8.3 Code Entropy

Code entropy is the disorder in software systems: architectural fragmentation, complexity accumulation, maintainability degradation, correctness failures, test-coverage gaps.

Instruments: cyclomatic complexity delta, lines-changed-to-functionality-added ratio, bug density (defects per KLOC) reduction, test coverage improvement, mean time to detect (MTTD) and mean time to recover (MTTR) improvements, formal-verification coverage gain.

Known failure modes: gaming via trivial refactors that improve a metric without improving substance, test coverage padding through tautological tests, complexity reduction by externalizing complexity to other modules.

Falsification condition: a claim of code entropy reduction is falsified if downstream metrics (defect rate, time-to-modify) do not improve in the following measurement window, or if a peer review identifies the refactor as superficial.

Maturity: high. The software-engineering literature provides decades of mature instruments.

### 8.4 Social Entropy

Social entropy is the disorder in trust networks, cooperation patterns, conflict dynamics, and community coherence.

Instruments: validated network-cohesion deltas (clustering coefficients, reciprocity scores), conflict-incident reduction over a windowed baseline, structured trust surveys with falsifiable questions, mediation outcomes confirmed by all parties.

Known failure modes: social-desirability bias in self-reports, the "harmony" trap where conflict is suppressed rather than resolved (apparent reduction with real-world increase), Goodhart pressure on trust metrics (this domain has the highest k value in the Signal taxonomy).

Falsification condition: a claim is falsified if independent observation in the following window shows reverted patterns (resumed conflict, broken cooperation, exit from the network).

Maturity: low. Social entropy is the least mature domain. The framework treats this honestly and weights social claims accordingly.

### 8.5 Economic Entropy

Economic entropy is the disorder in resource allocation, exchange efficiency, waste, scarcity, and matching.

Instruments: utilization improvement, throughput improvement, waste reduction (measured in domain-native units), allocation efficiency under a structured market-clearing or matching test.

Known failure modes: rent extraction disguised as efficiency, externalization of waste to other domains (the "local economic gain at thermodynamic cost" pattern), benchmark gaming through narrow optimization.

Falsification condition: a claim of economic entropy reduction is falsified if the system-wide accounting (including externalities into other canonical domains) shows no net reduction, or if the improvement reverts under a controlled measurement repeat.

Maturity: medium-high. The economics literature provides substantial instrumentation, but instrumentation often does not include externality accounting.

### 8.6 Thermodynamic Entropy

Thermodynamic entropy is the highest-precision reference domain. Physical disorder: energy efficiency, waste heat, material flow, environmental degradation.

Instruments: kilowatt-hours saved against a baseline, joules-per-kelvin reduction in industrial processes, material recovery rates, emissions reductions verified by third-party measurement, repair counts (a repaired object retains its low-entropy form longer).

Known failure modes: measurement boundary games (the "we don't count Scope 3" pattern), baseline manipulation (choosing a high-baseline year to make any reduction look favorable), substitution-with-hidden-costs (replacing one waste stream with another).

Falsification condition: a claim is falsified if third-party measurement (or extended-boundary accounting) contradicts the reported reduction.

Maturity: highest. Thermodynamic measurement is the most mature domain and serves as the reference for cross-domain calibration. The Shannon-Landauer-Bennett lineage gives this domain the strongest theoretical floor.

### 8.7 Informational Entropy

Informational entropy is the disorder in records, data quality, signal-to-noise, archival coherence, and retrieval latency.

Instruments: error rate reduction in records (deduplication, correction), completeness improvement (filled-in fields under a structured schema), retrieval latency improvement, signal-to-noise ratio improvement in datasets, bits-of-uncertainty removed against a baseline.

Known failure modes: noise reduction at the cost of legitimate variation, completeness padding with fabricated values, retrieval latency improvement through cache-only optimizations that mask underlying corruption.

Falsification condition: a claim is falsified if independent audit of the resulting records shows comparable or worse data quality than the baseline.

Maturity: high. The information-retrieval and data-quality literatures provide mature instruments.

### 8.8 Governance Entropy

Governance entropy is the disorder in decision systems: accountability, legitimacy, rule coherence, participation quality.

Instruments: decision-latency reduction (time from issue raised to decision rendered), reversal-frequency reduction (how often decisions get re-litigated), participation-quality improvement (under a structured measure), conflict-resolution outcomes confirmed by all parties, rule-coherence improvement (fewer contradictions in the rule set).

Known failure modes: speed at the cost of legitimacy (“we decided quickly, but nobody trusts the decision”), procedural compliance disguised as substantive participation, rule simplification that externalizes complexity to enforcement.

Falsification condition: a claim is falsified if participants exit the governance system, if the decision is reversed under subsequent appeal, or if downstream coordination breaks.

Maturity: low-medium. Governance instrumentation is improving but remains less mature than physical or informational.

### 8.9 Temporal Entropy

Temporal entropy is the disorder in time allocation, sequencing, synchronization, and operational cadence.

Instruments: cycle-time reduction, wait-time reduction, throughput per unit time, synchronization improvement (alignment of dependent processes), schedule-adherence improvement.

Known failure modes: speedup at the cost of quality, batching that improves throughput but degrades responsiveness, schedule compliance through pre-emption of legitimate work.

Falsification condition: a claim is falsified if the temporal improvement reverts under sustained measurement, or if the temporal gain has been achieved by externalizing temporal cost to a dependent process.

Maturity: medium. Operations-research and queueing-theory literatures provide instruments, but cross-domain temporal accounting is still preliminary.

### 8.10 Intersectionality and Domain Vectors

Most real contributions land across multiple domains simultaneously. A working software-project deployment reduces code, cognitive, and economic entropy. A neighborhood mediation reduces social, governance, and informational entropy. A factory retrofit reduces thermodynamic, economic, and temporal entropy.

The evidence vector E is an eight-element vector that expresses how much of the claim's reduction landed in each domain. The DFAO's weight vector w expresses how the local community values the eight domains. The dot product w dot E is the protocol's way of computing the contextual value of a multi-domain contribution without abandoning the universal eight-domain taxonomy.

### 8.11 Why Not More Domains

The case for adding a domain to the canonical eight requires:

1. Demonstrating that the proposed domain admits a measurement instrument grounded in either physical or informational disorder (the Shannon-Landauer-Bennett floor).
2. Demonstrating that the proposed domain's measurement cannot be expressed under the existing eight without loss.
3. Demonstrating that the proposed domain has a falsifiability condition definable in advance.

Ecological harm fails (2): it decomposes cleanly into thermodynamic, informational, social, and economic without loss. Spiritual experience fails (1) and (3): it lacks an operational instrument grounded in the theoretical floor, and its falsifiability conditions cannot be specified in advance without collapsing the category. Neither is dismissed; both are addressed elsewhere (ecological via multi-domain claims, spiritual via the philosophical appendix on emergent entropy reduction).

### 8.12 Operationalization Status

| Domain | Instrument maturity | Open work |
|---|---|---|
| Thermodynamic | High | Cross-boundary accounting for Scope 3-type externalities |
| Informational | High | Schema-level falsification protocols |
| Code | High | Refactor-substance validators |
| Economic | Medium-high | Externality accounting into other canonical domains |
| Cognitive | Medium | Standardized comprehension instruments per sub-domain |
| Temporal | Medium | Cross-process temporal accounting |
| Governance | Low-medium | Participation-quality instruments, reversal-rate baselines |
| Social | Low | The hardest domain; structured trust instruments under development |

These maturity ratings inform initial domain weights. A DFAO operating in a low-maturity domain is expected to weight other domains more heavily until the social-domain instruments mature.

# 9. The Contribution Graph and Loop Lifecycle

## 9.1 The Contribution Graph

The Extropy Engine has only one entity type at the architectural level: the **contribution**. Validation is a contribution. Governance voting is a contribution. Submitting a claim is a contribution. Mediating a dispute is a contribution. The framework refuses the artificial distinction between "tasks," "quests," "jobs," and "validations" because that distinction is itself a Goodhart vector: it lets actors game the category boundaries.

The contribution graph is the directed acyclic graph (DAG) of all contributions, with edges expressing causal relationships (this validation supported this claim; this governance vote applied to this proposal; this convergence vertex links these two person-DAGs).

## 9.2 The Loop as Unit of Work

A **loop** is the unit of work in the contribution graph. A loop has:

- A submitter (the identity that opened the loop).
- A claim (what is being asserted to have reduced disorder).
- A domain (which of the canonical eight, plus the evidence vector E).
- An instrument (the measurement protocol).
- A baseline (the pre-loop measurement).
- A falsification condition (what would prove the claim wrong).
- A target settlement time (the expected duration).
- A validation pool routing record (which validators were assigned, by what criteria).
- A consensus record (the validators' independent judgments and the aggregation V_c).
- A settlement record (the final XP minted, the final state, and the retroactive-burn window).

Every loop produces a sequence of DAG vertices that capture its causal history. The DAG is event-sourced: state is derivable from replay; history is permanent; reinterpretation occurs by appending future vertices, not by mutating past ones.

## 9.3 The Five States and Two Failure Branches

The loop lifecycle has five canonical states and two failure branches:

```
[   ][   ]
```

**OPEN.** The loop is created. The submitter specifies the claim, domain, instrument, baseline, falsification condition, and target settlement time. The DAG vertex LOOP_OPEN is written. The validation pool is identified by SignalFlow routing.

**VALIDATING.** Validators (or volunteer micro-validators) examine the claim, possibly against blind slices (one-tenth of the full context, as specified in 10.4). Each validator records an independent judgment. The epistemology engine aggregates the judgments under a Bayesian model.

**CONSENSUS.** The aggregation has produced a V_c value above the threshold. Consensus has been reached. The loop is ready to close.

**CLOSED.** XP is computed via the canonical formula and minted as **provisional**. The DAG vertex LOOP_CLOSE is written, followed by XP_MINT.PROVISIONAL. The 30-day retroactive-burn window opens.

**SETTLED.** The retroactive-burn window has elapsed without contradicting evidence, or the contradicting evidence has been resolved in favor of the claim. The provisional XP is upgraded to confirmed. The DAG vertex XP_MINT.CONFIRMED is written.

**FAILED.** The retroactive-burn window has surfaced contradicting evidence sufficient to falsify the claim. The provisional XP is burned. The DAG vertex XP_BURN is written. The validators whose consensus was contradicted take reputation penalties (rho_validator decreases).

**ISOLATED.** The loop has been identified as compromised (Sybil cluster, fingerprint-collision, fraud cluster). The XP minted at CLOSED is frozen pending governance review. The DAG vertex LOOP_ISOLATED is written.

## 9.4 Provisional and Confirmed Mints

The two-phase mint is one of the protocol's most important Goodhart-defense mechanisms. At CLOSED, XP is minted as provisional. The submitter sees the XP in their character sheet but cannot derive economic value from it (specifically, EP conversion is locked) until SETTLED.

During the 30-day window:

- New evidence can be submitted by any validator or external observer.
- If the new evidence falsifies the claim, the loop transitions to FAILED and the XP is burned.
- If the new evidence corroborates the claim, the loop's confidence is strengthened (V_c is updated).
- At the end of the window, if no falsifying evidence has been admitted, the loop transitions to SETTLED.

The retroactive-burn window is governance-tunable (default 30 days). Some domains may use shorter windows (thermodynamic with strong instruments) or longer (social with high k value and slow corrective feedback).

## 9.5 DAG Event Vocabulary

The DAG's event ontology is the canonical vocabulary of vertex types:

| Vertex type | Meaning |
|---|---|
| LOOP_OPEN | A new loop has been created |
| LOOP_VALIDATING | The loop has entered validation |
| VALIDATION_RECORDED | A validator has recorded a judgment |
| LOOP_CONSENSUS | Consensus aggregation completed |
| LOOP_CLOSE | The loop has closed with provisional XP |
| XP_MINT_PROVISIONAL | Provisional XP has been minted |
| XP_MINT_CONFIRMED | Provisional XP upgraded to confirmed |
| XP_BURN | Provisional XP burned due to falsification |
| LOOP_ISOLATED | Loop quarantined for integrity review |
| LOOP_SETTLED | Loop fully settled |
| LOOP_FAILED | Loop falsified during retroactive window |
| CONVERGENCE | Multiple person-DAGs share a vertex |
| GOVERNANCE_PROPOSAL | A governance proposal has been opened |
| GOVERNANCE_VOTE | A governance vote has been cast |
| REPUTATION_UPDATE | A rho update for an actor |
| REVEAL_CONSENT | A user has consented to selective reveal |
| CT_MINT | A CT mint has occurred |

| Vertex type | Meaning |
|---|---|
| IT_MINT | An IT mint has occurred |
| EP_CONVERT | An EP conversion has occurred |

The DAG vertex schema is defined in the contracts package and is the event-bus contract between services. Twenty-two typed inter-service events are defined; the table above lists the load-bearing ones.

### 9.6 Event-Sourced Reinterpretation

The DAG is append-only. Past vertices are not mutated. Reinterpretation (such as: "we now understand that this loop's Delta S was overestimated; we should burn portion of the XP") happens by appending a corrective vertex (e.g., XP_BURN) that links causally to the original vertex. The original record stays. The corrective record changes the effective state.

This is the formal mechanism by which the Engine learns without rewriting history. The DAG preserves the lineage of every revision. Auditors can replay the DAG to reconstruct what was true at any point in time, and can identify exactly when and how interpretation changed.

### 9.7 The Happy Path

For reference, the canonical event flow for a successful loop (verified by the 12-of-12 integration test suite):

```
POST /claims
  -> claim.submitted
  -> loop.opened
  -> claim.decomposed (if needed; legacy v3.0 path)
  -> task.created xN -> task.assigned xN -> task.completed xN
  -> subclaim.updated xN -> claim.evaluated
  -> loop.consensus_started
  -> loop.closed
  -> xp.minted.provisional
  -> reputation.accrued
  -> loop.settled
```

This is the happy path. The decomposition step is the legacy v3.0 path that exists for backward compatibility with the test harness; the v3.1 redefinition pushes decomposition to the edge (personal AI), and the epistemology engine no longer performs decomposition on behalf of users.

## 10. Validation Model, V_c, SignalFlow, and Neighborhoods

### 10.1 What Validation Does

Validation is the protocol's way of converting a claim into a witnessed fact. Without validation, the claim is just an assertion. With validation, the claim is a witnessed event with an aggregated confidence score. The aggregated confidence is V_c (validation confidence), not F (frequency-of-decay). These are entirely different quantities; the naming collision in earlier drafts was corrected in v3.1.2.

V_c is in [0, 1]. V_c is the output of the Bayesian aggregation of validator judgments over a claim. V_c is what gates the transition from VALIDATING to CONSENSUS (the threshold is governance-tunable; default 0.66).

### 10.2 SignalFlow: Validator Routing

SignalFlow is the routing service that selects validators for a claim. Routing is multi-factor:

[ = w_d + w_r + w_l ^{-1} + w_a ]

Default weights: (0.35, 0.30, 0.15, 0.20). Governance-tunable.

- **domain_match:** how well the validator's per-domain expertise matches the claim's primary domain.
- **reputation:** rho_validator, the validator's accumulated reputation density. This is the legitimate use of reputation: it conditions which validators see a claim, not which claims mint XP.
- **load:** how busy the validator is; inverse-weighted so under-utilized validators are preferred.
- **historical_accuracy:** the validator's track record on prior claims, conditional on the retroactive-burn outcomes.

The routing produces a pool of validators (size governance-tunable, typically 5 to 12 for routed validation, larger for high-stakes claims). Validators receive the claim with as much context as required for their judgment.

### 10.3 The Cross-Layer Coupling

The peer review correctly flagged that SignalFlow's use of reputation creates a cross-layer coupling: reputation -> routing -> XP-earning opportunity -> CT balance over time. This is intentional. The Codex acknowledges and documents it.

The reasoning: reputation is not allowed to multiply XP, but reputation is allowed to condition which validators see which claims. A high-reputation validator sees more claims and earns more validation XP and accrues more rho. This is a feedback loop, but a deliberate one. The architectural invariant (reputation does not enter the XP formula) is preserved: the value of a successfully validated claim is the same regardless of which validator validated it. The validators earn validation XP for their work, weighted by the same formula as any other contribution.

The cross-layer coupling is documented, not hidden, and is subject to governance review if it produces drift.

### 10.4 Volunteer Micro-Validation: One-Tenth Blind Slices

For routine, high-volume claims (the bulk of the contribution graph), the protocol supports **volunteer micro-validation via one-tenth blind slices**. A blind slice is a fragment of the full claim context (one-tenth of the evidence, no submitter identity, no DFAO context). Volunteer validators see only the slice and record a binary judgment (consistent / inconsistent with the claim).

Each claim is sliced into ten blind fragments. Each fragment is validated by a different volunteer. The aggregation across the ten fragments produces V_c. The protocol does not allow a single validator to see all ten slices; this is the Sybil and collusion defense.

The blind-slice mechanism is the operational primitive for high-volume contribution validation. It is cheap, parallelizable, and Sybil-resistant. It produces V_c, not F. The naming collision in earlier drafts (validation aggregation produces F) was the P0 error corrected in v3.1.2. F is frequency-of-decay; V_c is validation confidence. They are different.

### 10.5 The Bayesian Aggregation Model

The aggregation across validators uses a Beta(alpha, beta) conjugate model. Each validator judgment updates the alpha, beta posterior over the claim's truth. The validator's historical accuracy weighs the update; a more accurate validator updates the posterior more strongly.

Two aggregators are supported, selectable per governance:

- **logodds aggregator (v3.1 default):** log-odds combination across validator judgments. Robust to extreme judgments.
- **geometric aggregator (legacy):** geometric mean. Preserved for backward compatibility.

The verdict mapping (corrected on 2026-05-06 as a Bayesian update bug fix): both `confirmed` and `supported` validator judgments are treated as affirmative. The earlier behavior treated `supported` differently, which was an arithmetic bug.

### 10.6 Validation Neighborhoods

Validation neighborhoods are the sharded routing structures for the volunteer micro-validation layer. Each domain has its neighborhoods; each DFAO can configure its neighborhood routing. The neighborhood structure prevents a single validator from concentrating too much routing weight in any one domain.

The `validation-neighborhoods` package is currently a skeleton (the interface is specified; the operational sharding is in progress). The Codex documents the design; the implementation is partial.

### 10.7 Validator Reputation: rho_validator

A validator's reputation density rho_validator increases when their consensus judgments survive retroactive-burn windows and decreases when their consensus is later contradicted (loop transitions to FAILED). The reputation update happens at SETTLED for confirmed loops and at FAILED for falsified loops.

The penalty for a falsified consensus is larger than the reward for a confirmed consensus. This asymmetry is the formal incentive for validators to be cautious rather than rubber-stamp claims.

### 10.8 The Epistemology Engine as Witness Layer

In v3.0 of the specification, the epistemology engine was a central decomposition service that took submitted claims and atomized them into sub-claims using a Bayesian decomposition algorithm. This design was rejected in v3.1 because it violated Digital Autarky: the central service was thinking on behalf of the user.

In v3.1, the epistemology engine is **redefined as a mesh observability and emergent peer-review witness layer**. Specifically, the engine:

- Aggregates validation outcomes across the network.
- Surfaces consensus drift (claims where validator judgments are diverging).
- Surfaces Sybil clusters (validator groups whose judgments correlate suspiciously).
- Surfaces falsifiability statistics (which claim types are being falsified at unusual rates).
- Does **not** decide what is true. The validators decide; the engine reports.
- Is read-mostly. Stateless under restart. Rebuildable from DAG replay.

The legacy v3.0 decomposition endpoints are still present in the codebase, marked deprecated, scheduled for removal in v3.2. The current integration test harness exercises the legacy path, which is acknowledged as a transitional state.

### 10.9 Anti-Collusion: Retroactive Slashing

The 30-day retroactive window is the structural defense against validator collusion. A validator pool that conspires to confirm a fraudulent claim has 30 days during which independent observers can submit falsifying evidence. If the evidence stands, the colluding validators take a rho_validator penalty proportional to the XP burned.

The penalty is structured so that the expected value of collusion is negative: the cost of being caught (rho penalty plus reputation loss) exceeds the expected gain (a share of the colluding submitter's payoff, discounted by the probability of detection).

The cartel-threshold analysis in the repository documents the game-theoretic result: at validator population N >= 10 and detection probability p >= 0.3, no cartel strategy is profitable in expectation. This is the formal basis for the validator-pool sizing defaults.

### 10.10 Verdict Vocabulary

The validator's allowed verdicts are:

- **confirmed:** strong support; evidence matches the claim under the instrument.
- **supported:** moderate support; evidence is consistent but not definitive.
- **inconclusive:** the evidence available is insufficient to render judgment.
- **contradicted:** the evidence available contradicts the claim under the instrument.
- **refused:** the validator declines to render judgment (conflict of interest, out of domain).

Both confirmed and supported count as affirmative in the Bayesian aggregation. Inconclusive does not update the posterior. Contradicted is a negative update. Refused removes the validator from the pool for this claim.

## 11. The DAG Substrate and Person as DAG

### 11.1 The Three Axioms of the DAG Architecture

The DAG architecture paper formalizes the substrate under three axioms:

1. **Events, not state.** The DAG stores events. State is derived by replay. There is no row to mutate.
2. **Convergence is value.** When person-DAGs share a vertex (a co-authored mediation, a joint repair, a co-witnessed validation), the convergence vertex is where multi-party minting happens.
3. **Graphs nest fractally.** A person is a DAG. A DFAO is a DAG of person-DAGs. The protocol is a DAG of DFAO-DAGs. The same vertex/edge primitives operate at every scale.

These axioms are the operational form of RF Axiom 3 (Fractal Self-Similarity via Functors).

### 11.2 Person as DAG

The most distinctive architectural choice in the Extropy Engine is that **identity is a DAG**, not a database row. Each person is a signed causal history: an ordered set of vertices (the events the person has participated in) and edges (the causal links between events).

This has several immediate consequences:

- A person cannot be summarized to a single number. There is no "credit score" because the person's record is a graph, not a scalar.
- Retroactive recomputation works because the graph is event-sourced. New information appends a vertex; old vertices remain.
- Identity proofs are graph-shaped: a person can prove participation in a specific event without exposing the rest of their graph (selective disclosure via ZKP over the graph structure).
- Multi-party events naturally form convergence vertices where multiple person-DAGs share a node.

### 11.3 The DAG Substrate Service

The dag-substrate service (port 4008, ~1573 lines of TypeScript) provides:

- **Vertex Store:** PostgreSQL-backed storage of vertices, indexed by content hash.
- **Edge Store:** Causal links between vertices, with edge-type semantics.
- **Tip Selection:** IOTA Tangle-inspired weighted random walk to identify recent vertices for new submissions to attach to.
- **Confirmation Propagation:** Recursive confirmation flow up to depth 20, with each confirmed vertex strengthening the confirmation weight of its ancestors.
- **Auto-Vertex Handlers:** Redis pub/sub triggers that automatically create vertices in response to event-bus events.
- **Content Hashing:** SHA-256 content hashes ensure tamper-evidence.
- **Signature Verification:** Ed25519 signatures verify the submitting identity.

The schema is in scripts/init-db.sql under the dag schema with tables vertices, edges, and confirmations.

### 11.4 IOTA-Inspired Tip Selection

Tip selection (which previous vertices a new submission attaches to) uses a weighted random walk inspired by IOTA Tangle. The walk starts at the genesis vertex and steps forward through the graph, with each step choosing among the unconfirmed children weighted by their content quality and validator support. The walk terminates at a tip (an unconfirmed vertex).

A new submission attaches to two tips (the standard IOTA pattern), which has the effect of confirming those two tips. This creates the "everyone validates everyone" property of the Tangle: every new submission contributes to the confirmation weight of two prior submissions.

The Extropy Engine modifies the IOTA pattern in one important way: the tip selection is biased toward tips within the same DFAO or within converging DFAOs. This produces locality in the graph that matches the locality of human contribution.

### 11.5 Convergence Vertices

When two or more person-DAGs share a vertex (because they participated in the same event), the vertex is a convergence vertex. Examples:

- Two people complete a joint repair. The repair is one event with two participants. The vertex has two incoming edges (one from each person's DAG).

- A mediation with three parties produces a convergence vertex with three incoming edges.
- A validation pool that all confirm the same loop creates convergence at the validation event.

Convergence vertices are where multi-party minting happens. The XP earned at a convergence vertex is split among the participants according to the convergence-split rules specified at OPEN. The split rules are governance-tunable; defaults include equal split, contribution-weighted split, and DFAO-defined split functions.

### 11.6 The DAG Nests: From Person to Civilization

The fractal property of the DAG architecture is that the same vertex/edge primitives operate at every organizational scale:

- An individual's PSLL is a DAG of their personal events.
- A NANO DFAO (an individual project, 1 person) is a DAG of that person's project events.
- A MICRO DFAO (a team, 2 to 7 people) is a DAG of cross-person events within the team.
- A MESO DFAO (a department or community, 8 to 200 people) is a DAG with sub-DFAO nesting.
- A MACRO DFAO (an organization or city, 200 to 100,000 people) is a DAG of MESO DAGs.
- A PLANETARY DFAO (civilization-scale, 100,000+ people) is a DAG of MACRO DAGs.

The same SIIR loop (sense, infer, integrate, respond) operates at every scale. The same convergence-minting primitive operates at every scale. The same retroactive-burn window operates at every scale (with governance-tunable timing per scale).

### 11.7 Causal History Walks

Querying the DAG is done through causal-history walks. A walk traverses from a target vertex backward through its causal ancestry. The walk can be filtered (only vertices of a specific type, only vertices from a specific actor, only vertices within a time window).

The walk is the substrate for several first-class queries:

- **Provenance:** what events led to this XP mint?
- **Validator track record:** what loops did this validator validate, and how many were later falsified?
- **Convergence history:** which person-DAGs have converged with this one, and at what events?
- **Domain trajectory:** how has this DFAO's domain-weighting changed over time?

### 11.8 Production Migration

The current DAG substrate runs on a single-node PostgreSQL backend. Production deployment requires:

- **libp2p gossip layer** for inter-node DAG replication. Currently tracked as a P2 gap.
- **Noise protocol** for inter-node transport security. Currently tracked as a P2 gap.
- **Sandbox node-handshake** for VPS-to-laptop bootstrap. Implemented (`node-handshake` service, port 4200) but the actual VPS-to-laptop test has not yet been run.

The Codex distinguishes the implemented sandbox (which proves the architecture) from the production-required components (which remain on the roadmap).

# 12. Identity, PSLL, Privacy, and Digital Autarky

## 12.1 Digital Autarky as Constitutional Principle

**Digital Autarky** is the constitutional principle of the Extropy Engine. It states: every participant retains sovereignty over their own intelligence stack, identity material, decision context, and local event history. The network is a coordination layer, not a supermind. There is no central AI that thinks for the user. There is no central service that decomposes claims on behalf of submitters. There is no place in the protocol where someone else's reasoning is allowed to overwrite the user's reasoning.

The principle is enforced architecturally, not merely stated as policy. Specifically:

- Personal decomposition (breaking a complex claim into sub-claims) runs on the user's device through their personal AI, not on a central service.
- The PSLL (Personal Signed Local Log) lives on the user's device; the network only sees periodic Merkle roots anchored as DAG vertices.
- Identity material (private keys, biometric templates, KYC inputs) lives on the user's device; the network sees only the public DID and the ZKP-disclosed credentials.
- Validators see what the user chooses to disclose, in the form the user chooses to disclose it. The protocol's selective-disclosure primitives are the user's controls.

The autarky principle reverses the assumption common in centralized systems: that "the network knows best" and the user is a leaf node. In the Extropy Engine, the user is sovereign, and the network is the coordination layer the user opts into.

## 12.2 The Hybrid Identity Layer

Identity is hybrid: OAuth + on-device KYC + DID + ZKP. Each layer plays a specific role:

- **OAuth** provides convenient login from existing identity providers (Google, GitHub). This is the entry point. OAuth does not authenticate the protocol-level identity; it just bootstraps the user into the onboarding flow.
- **On-device KYC** binds the user's identity to a unique person. Three options are supported: an ID scan (driver's license, passport), a biometric bind (face or fingerprint, processed locally), or a trusted-issuer handoff (the user comes from an existing verified identity provider). The peer review correctly flagged that these three options have radically different Sybil-resistance guarantees; the threat model per option is documented separately.
- **DID** (Decentralized Identifier) is the protocol-level identity. Format: did:extropy:<64-hex> where the hex is the user's 32-byte Ed25519 public key.
- **ZKP** (Zero-Knowledge Proof) is the disclosure layer. The user can prove statements about their credentials without revealing the credentials themselves (e.g., "I am a verified member of this DFAO" without revealing which other DFAOs they belong to).

## 12.3 BBS+ Default, zk-SNARKs Supported

The default ZKP scheme is BBS+. BBS+ is well-suited to the credential-presentation model: it supports selective disclosure of individual attributes within a multi-attribute credential, with efficient verification. zk-SNARKs are supported as an alternative for proof statements requiring more complex arithmetic circuits.

The scheme choice is governance-tunable. A DFAO can require zk-SNARKs for high-stakes claims while accepting BBS+ for routine validation.

## 12.4 Verifiable Credentials

The Verifiable Credential (VC) envelope is compact JWS with EdDSA signatures. The kid (key ID) takes the form <issuerDid>#keys-1. Default validity is one year. Self-issued credentials are the canonical case: a user issues credentials about themselves and presents them with ZKP backing. Third-party-issued credentials (from a DFAO, an institution, a verified registrar) are also supported.

## 12.5 Nullifiers

For the protocol to prevent double-spending of one-time actions (one-vote-per-person-per-proposal, one-claim-per-person-per-loop, etc.) without exposing the user's identity, nullifiers are used. The nullifier construction:

[ = ((, || '|' || )) ]

The same user computing the same nullifier for the same context tag produces the same nullifier, allowing the protocol to detect double-action. Different context tags produce uncorrelated nullifiers, preserving privacy across contexts.

The current implementation uses HMAC-SHA256 as a sandbox stand-in. Production swap to Pedersen-based commitments with BBS+ is on the roadmap and is documented in IDENTITY_IMPL.md. The swap goes through the same ZKP envelope, so application code is unchanged.

## 12.6 The Onboarding State Machine

oauth-pending -> oauth-verified -> kyc-attested -> did-issued -> closed

Each transition has a documented input contract. Replay protection is enforced at two points: challenge-consume at the OAuth handoff (single-use, time-bounded), and per-context nullifier at the DID issuance (single-use per context tag). Both return HTTP 409 on collision.

## 12.7 The 7-of-12 Reveal Threshold

When the network must compel disclosure (a high-stakes legal subpoena, a confirmed fraud investigation, a governance-approved audit), the user's identity can be revealed through a threshold scheme. The default threshold is **7-of-12**: any seven of twelve ecosystem-validator shareholders, acting together, can reconstruct the reveal key.

The construction: Shamir Secret Sharing over GF(256), with an AES-256-GCM wrapper for the actual identity material. The server returns the shares to the caller at issuance time and does not retain them. This means even a fully compromised server cannot reveal identities without the cooperation of seven of twelve external shareholders.

The peer review flagged that the 7-of-12 ratio lacks formal adversarial derivation. The Codex provides the derivation: under threshold-signature semantics, an adversary controlling fewer than 7 shareholders cannot reconstruct the secret. The 7-of-12 ratio provides a margin above the 50%+1 majority threshold (which would be 7 out of 12 anyway), while remaining achievable without coordinating a near-unanimous group. The ratio is governance-tunable.

## 12.8 The Personal Signed Local Log (PSLL)

The PSLL is the user's local, append-only, hash-chained, cryptographically signed event log. The PSLL is the substrate of the user's autarky: everything the user does in the Extropy Engine is recorded first in the PSLL, then summarized to the network through periodic Merkle-root anchors.

The PSLL is borrowed from the Holochain source-chain pattern, renamed, and reimplemented natively for the Extropy Engine's substrate decisions.

## 12.9 PSLL Entry Schema

A PSLL entry has the following fields:

- entryType: one of claim_submitted, validation_performed, quest_accepted, quest_completed, decomposition, governance_vote, reputation_update, reveal_consent, custom.
- timestamp: the entry's creation time.
- previousHash: the hash of the previous entry in the chain.
- payload: the entry's content (typed by entryType).
- signature: an Ed25519 signature over the entry by the user's identity key.

The hash chain is enforced: every entry references the previous entry's hash. The chain is tamper-evident: any modification to a past entry invalidates every subsequent entry's chain reference.

## 12.10 Merkle-Root Anchoring

Periodically (default: one anchor per loop close), the user's PSLL produces a Merkle root over the recent entries, signs it, and submits the signed root to the DAG as an anchor vertex. The network now has a tamper-evident commitment to the user's local log without seeing the log contents.

If the user later wishes to prove a specific entry (in a dispute, for an audit), they can produce a Merkle proof relative to the anchored root. The network can verify the proof without seeing the rest of the log.

## 12.11 PSLL-DAG Conflict Resolution

The peer review correctly flagged that the spec did not address what happens when a node's PSLL and its DAG anchors diverge (network partition, malicious edit attempt, bug). The Codex resolves this:

- A divergence is detected when a Merkle proof against a DAG anchor fails to verify.
- A detected divergence triggers a dispute review, not a silent overwrite.
- The DAG anchor is canonical for the network's view; the PSLL is canonical for the user's view.
- If the divergence is the user's PSLL being corrupted (legitimate case), the user replaces the PSLL from a backup or re-syncs.
- If the divergence is the user attempting to mutate their PSLL retroactively (illegitimate case), the DAG anchor catches the mutation and the divergence is recorded against the user's reputation.

## 12.12 PSLL Cryptographic Defaults

The peer review correctly flagged that the PSLL's hash function and signature scheme were unspecified in earlier drafts. The Codex pins them:

- **Hash function:** BLAKE3 for commitments (faster than SHA-256, equivalently secure). SHA-256 is acceptable as a fallback. Poseidon is used in circuits requiring ZKP compatibility.
- **Signature scheme:** Ed25519 for entry signatures. BBS+ for selective disclosure of PSLL-derived claims.

These are governance-tunable defaults, but the defaults are explicit. The hash function and signature scheme can be revised through governance proposal; the revision applies to future entries only (preserving the chain integrity of existing entries).

### 12.13 What the Network Does Not Get

The network does not get:

- Raw PSLL entries. The network sees only the Merkle root anchors.
- Identity inputs. Private keys, biometric templates, KYC raw documents stay on the user's device.
- The user's claim-decomposition reasoning. Decomposition runs on the user's personal AI.
- Cross-context correlation. Nullifiers prevent the network from correlating the user's actions across DFAOs or claim types.

What the network does get:

- The user's DID (public).
- Per-claim selective disclosures, via ZKP, of the specific attributes required.
- Anchored Merkle roots of the PSLL.
- The user's actions in the contribution graph (the events the user has participated in).
- The user's accumulated XP, CT, CAT, IT, DT, EP balances (the public tokens; XP and IT are non-transferable by construction).

This separation is what makes Digital Autarky operational: the network has exactly what coordination requires, and nothing more.

## 13. Token Economy and the Market Layer

### 13.1 The Six Canonical Tokens

The token economy has exactly six canonical tokens. They are:

| Token | Full Name | Transferability | Decay | Purpose |
|---|---|---|---|---|
| XP | Experience Points | Non-transferable | Yes (~1%/month default) | Per-action entropy-reduction score |
| CT | Contribution Token | Transferable with 14-day lockup, 2% transfer friction | Slow | Identity-bearing contribution capacity |
| CAT | Capability Attestation Token | Portable | No | Skill credential |
| IT | Influence Token | Non-transferable | 5%/month | Governance voting weight |
| DT | Domain Token | Domain-specific | Variable | Subject-matter expertise marker |
| EP | Emergence Points | Local to merchant network | Variable | Merchant loyalty / discount value |

These are the only tokens. **GT and RT** are legacy names from earlier drafts and from a transitional state in the token-economy package. They are not canonical. The CONTRIBUTION_GRAPH.md rule states it without ambiguity: "Exactly six. No others."

### 13.2 XP: Experience Points

XP is the per-action value mint. XP is non-transferable: it cannot be sold, gifted, or transferred. XP decays slowly (default 1% per month) to ensure that recent contribution dominates over distant historical contribution.

XP is what the formula in section 5.1 computes. XP is what gets minted at LOOP_CLOSE as provisional and confirmed at LOOP_SETTLED.

XP feeds EP (the merchant-side discount currency) via the EP = XP * L formula. The conversion is at the user's discretion within governance-tuned rate limits.

### 13.3 CT: Contribution Token

CT is the identity-bearing contribution capacity. Where XP measures the loop, CT measures the contributor. CT can be transferred (with a 14-day lockup and 2% transfer friction). CT is what other DFAOs see when they consider a new participant; it is the contributor's portable evidence of sustained contribution.

The CT formula is in section 5.9. Reputation density rho is the term that legitimately makes CT identity-bearing: a contributor's rho enters their CT calculation, weighting their CT against their accumulated reputation. This is where reputation legitimately lives; this is what the architectural invariant "reputation does not enter XP" preserves by isolating reputation to the CT pathway.

### 13.4 CAT: Capability Attestation Token

CAT is a portable skill credential. A CAT does not decay (skill, once proven, does not erode within the protocol's accounting; the actor's exercise of the skill matters but the underlying credential does not).

A CAT is issued by a DFAO when a contributor has demonstrated sustained competence in a specific capability over a specified number of loops. The CAT is then portable: another DFAO can verify the CAT and admit the contributor at a higher initial standing.

CATs are how the protocol implements "qualifications" without binding qualifications to a central authority. Each DFAO is free to issue or not issue CATs against its own standards; other DFAOs are free to recognize or not recognize specific CATs.

### 13.5 IT: Influence Token

IT is the governance weight token. IT is non-transferable and decays at 5% per month, ensuring that governance influence requires continued contribution.

IT is earned through participation in governance (proposing, voting, mediating, serving on conviction-voting cohorts). IT cannot be purchased.

The 5% monthly decay is the formal mechanism by which the protocol prevents permanent influence accumulation. A participant who stops contributing loses governance weight quickly; the system biases toward current participants.

### 13.6 DT: Domain Token

DT is a domain-specific expertise or access marker. Each of the eight canonical domains has its DT. Holding a domain's DT signals expertise in that domain (the contributor has accumulated significant XP and CT in that domain) and confers domain-specific access (validator routing eligibility, governance-vote weight on domain-specific proposals).

DT semantics vary by domain: thermodynamic-domain DT requires demonstrated thermodynamic-instrument competence; social-domain DT requires demonstrated mediation outcomes.

### 13.7 EP: Emergence Points

EP is the user-facing merchant-loyalty token. EP = XP * L where L is the local merchant loyalty multiplier. EP is what the customer experiences at the point of sale: a discount, a loyalty reward, a credit applied to a purchase.

EP is local: it converts to a specific merchant or merchant network. EP is not transferable across merchant networks (a customer cannot accumulate EP at one shop and spend it at an unrelated shop unless both shops are in the same merchant DFAO).

The merchant-services fee charged by the network (default ~1.8%, vs 2.5-3% for incumbent payment processors) is the network's primary revenue source. The fee captures value at the moment of EP-to-fiat or EP-to-goods conversion.

### 13.8 The Market Layer

The market layer is the Layer 2 architectural surface: the merchant-facing presentation of the protocol. Merchants see:

- A free point-of-sale interface. No SaaS fee.
- A customer pipeline (customers walking in asking which businesses are on the network).
- Operational signal richer than standard KPI dashboards (the merchant gets visibility into customer-level entropy patterns, inventory entropy, supply-chain entropy, with the customer's selective-disclosure consent).
- Standard merchant-services fees, comparable to or lower than incumbents.

Merchants do not need to understand the protocol's mathematics. The protocol's complexity is hidden behind the POS interface. The merchant's experience is "loyalty program with better economics."

### 13.9 The Two-Sided Market

The Extropy Engine is, at the market layer, a two-sided market:

- **Customer side:** discounts, gamified feedback, real money saved through entropy-reducing habits.
- **Merchant side:** free POS, customer pipeline, deeper operational signal.

The two sides scale each other: more customers create demand for more merchants; more merchants create demand for more customers. The protocol's job at this layer is to make the matching efficient and the value capture appropriate.

Two market rules are non-negotiable:

1. **No SaaS charges to merchants.** Revenue is captured at transaction time, not subscription time.
2. **No paywalls to users.** Participation is free.

Everything else is negotiable per merchant contract.

### 13.10 The Business Model

| Source | What we capture | What we don't charge for |
|---|---|---|
| Merchant-services fee | ~1.8% per transaction | The POS itself |
| DFAO node registration | Tiered fees for deeper integration | The customer app |
| Premium merchant analytics | Subscription for deeper signal beyond dashboards | Basic operational data |
| Treasury yield on EP float | Standard mechanism (the EP not yet converted earns yield) | User accounts |
| Multinational integrations | Specialized contracts | Network participation |

The business model is structurally aligned with the protocol's incentives: the network earns revenue only when value is created (entropy reduction validated, EP converted to merchant goods, merchants growing through customer flow). The network does not earn revenue from user attention, user data resale, or user lock-in.

### 13.11 Token Lifecycle Defaults

| Parameter | Default | Governance-Tunable |
|---|---|---|
| XP decay rate | 0.01 per 30-cycle period (~1%/month) | Yes, per-domain |
| CT lockup | 14 days | Yes |
| CT transfer friction | 2% per transfer | Yes |
| IT decay | 5%/month | Yes |
| Reward escalation early | linear 1.0x to 3.0x over 7 days | Yes |
| Reward escalation late | log to cap 10.0x | Yes |
| EP conversion rate limits | governance-set per merchant network | Yes |

These defaults are the starting point. A mature deployment is expected to revise them based on observed market behavior.

## 14. DFAO Governance and Parameter Evolution

### 14.1 What a DFAO Is

A **DFAO** is a Decentralized Fractal Autonomous Organization. The "fractal" name is structural, not metaphorical: DFAOs nest within DFAOs, with the same governance primitives operating at every scale.

A DFAO has:

- A scale (NANO, MICRO, MESO, MACRO, or PLANETARY).
- A membership (the set of identities that participate).
- A governance configuration (quorum, deliberation period, voting method, intervention thresholds).
- A weight vector w over the eight domains (its local priority weighting).
- A rarity table override (its local RARITY_DEFAULTS adjustments).
- A set of measurement instruments (the specific instruments it accepts per domain).
- An anchor in the DAG (the DFAO's own genesis vertex and subsequent governance events).

DFAOs can be nested: a MICRO DFAO inside a MESO DFAO inside a MACRO DFAO. Each level inherits defaults from its parent but can override within parent-defined safety bounds.

## 14.2 The Five DFAO Scales

| Scale | Member Count | Typical Use |
|---|---|---|
| NANO | 1 | An individual's personal project or solo contribution flow |
| MICRO | 2 to 7 | A small team, a family pilot, a working group |
| MESO | 8 to 200 | A department, a community group, a small organization |
| MACRO | 200 to 100,000 | An organization, a city, a large institution |
| PLANETARY | 100,000+ | Civilization-scale coordination |

The scale informs the appropriate governance configuration. A NANO DFAO has trivial governance (one participant). A MICRO DFAO can use simple-majority voting on most decisions. A MESO DFAO benefits from conviction voting and structured deliberation. MACRO and PLANETARY DFAOs require sub-DFAO structuring and delegated voting.

## 14.3 Voting Methods

Three voting methods are supported:

- **Linear reputation:** vote weight is proportional to the voter's reputation density rho. Simple but susceptible to influence concentration.
- **Quadratic:** vote weight is the square root of the voter's IT held, with a cost in IT for each vote cast. Reduces influence concentration.
- **Conviction:** vote weight grows over time as the voter maintains their position. Favors patient, sustained positions over impulsive ones. The Extropy Engine's default for non-trivial proposals.

The method is chosen per DFAO per proposal type, governance-tunable.

## 14.4 Conviction Voting in Detail

Conviction voting is the protocol's default for non-trivial governance decisions. The mechanism:

- A voter stakes a position (for or against a proposal) and a quantity of IT.
- The voter's effective vote weight grows over time as they maintain the position: [ w(t) = w_0 (1 - e^{-t/}) ] where w_0 is the staked weight and tau is a half-life parameter.
- The voter can withdraw at any time, but doing so resets their accumulated weight.
- The proposal passes when the aggregate weight in favor exceeds the proposal's threshold.

Conviction voting rewards patience and penalizes impulsive shifts. It is well-suited to decisions where the "right answer" is not immediately obvious and benefits from extended consideration.

## 14.5 The Provisional Defaults Table

The Extropy Engine ships with a provisional defaults table. Every parameter is governance-tunable, but the defaults provide a working starting point:

| Parameter | v3.1 Default |
|---|---|
| ZKP scheme | BBS+ |
| Identity reveal threshold | 7-of-12 + cause-shown |

| Parameter | v3.1 Default |
|---|---|
| Retroactive validation window | 30 days |
| IT decay | 5%/month |
| CT lockup | 14 days |
| Reward escalation early | linear 1.0x to 3.0x over 7 days |
| Reward escalation late | log to cap 10.0x |
| Validator weight factors | (0.35, 0.30, 0.15, 0.20) |
| PSLL anchor cadence | 1 per loop close |
| V_c threshold | 0.66 |
| Validator pool size | 5 to 12 (routed); 10 (volunteer micro-validation) |
| Domain weight defaults | 1.0 per domain (DFAO can override) |
| Essentiality factor E default | 0.8 |
| Task grain | 2 to 5 minutes |
| Validation slice | 1/10th blind |
| XP decay rate | 0.01 per 30-cycle period |
| CT transfer friction | 2% per transfer |
| Conviction half-life tau | 7 days |

## 14.6 Parameter Update Process

A governance parameter update follows the canonical proposal flow:

1. A member opens a GOVERNANCE_PROPOSAL vertex specifying the parameter, the proposed value, the rationale, and a target effective date.
2. The DFAO's governance configuration determines the deliberation period and voting method.
3. Votes are recorded as GOVERNANCE_VOTE vertices linked causally to the proposal.
4. At the end of the deliberation period, the aggregation determines whether the proposal passes.
5. If the proposal passes and the effective date arrives, the parameter is updated. The update applies to future loops only; existing loops continue under the prior parameter value (preserving DAG immutability).

The update is itself recorded as a DAG vertex (GOVERNANCE_PARAMETER_UPDATE) linked to the originating proposal.

## 14.7 Parent-Child DFAO Conflict Resolution

The peer review correctly flagged that earlier drafts did not specify how parent-child DFAO conflicts are resolved. The Codex resolves this:

- **Inheritance principle:** child DFAOs inherit defaults from their parent DFAO.
- **Override scope:** child DFAOs can override parameters within parent-defined safety bounds (e.g., a parent might allow children to set V_c thresholds anywhere in [0.5, 0.9]; values outside that range require parent approval).
- **Conflict resolution:** if a child wishes to override beyond the parent's safety bounds, the override requires parent governance approval (a proposal in the parent DFAO).
- **Emergency intervention:** parent DFAOs can suspend a child DFAO's overrides under specified emergency conditions (e.g., a child DFAO's parameters are producing high failure rates that threaten the parent's integrity).

The resolution preserves child autonomy within safe bounds while preventing parameter divergence that would compromise the protocol-level invariants.

### 14.8 Goodhart Governance: Diagnostic Use of Failure Rates

The protocol uses several governance-visible signals as diagnostic fuel:

- **Claim failure rate** by domain, by DFAO: indicates instrument drift or rising Goodhart pressure.
- **Validator disagreement rate**: indicates ambiguous claims or domain instability.
- **Domain drift**: changing patterns of which domains a DFAO is minting in.
- **Reversal frequency**: claims initially confirmed and subsequently falsified.
- **Fraud cluster detection**: Sybil patterns, fingerprint-collision attempts, validator collusion patterns.

These signals are not treated as failures of the protocol; they are treated as inputs to governance. Rising failure rates trigger governance review of the instrument. Rising validator disagreement triggers expansion of the validator pool or revision of the routing weights. Domain drift triggers review of the domain weight vector.

The Codex commends the protocol's posture on this: Goodhart pressure as diagnostic fuel, not as failure. This is the operational implementation of Theorem V4-3 (Dynamic Goodhart Resistance) from Randall's Feedback V4.

### 14.9 Complexity Thermostat Operationalization

In RF V4, the Complexity Thermostat regulates the unified entropy measure S_RF. In the Extropy Engine, the operationalization is the governance-tunable parameter table itself. When S_RF rises above an upper threshold (the protocol's contribution graph is too chaotic, too fragmented, too noisy), governance can tighten admissibility:

- Raise V_c thresholds.
- Tighten reward escalation curves.
- Expand validator pool sizes.
- Lower decay rates.

When S_RF falls below a lower threshold (the protocol is too rigid, too monoculture, too constrained), governance can relax admissibility:

- Lower V_c thresholds.
- Loosen reward escalation.
- Reduce validator pool overhead.
- Allow more domain experimentation.

The Complexity Thermostat is not an automated control loop in the current implementation; it is a manual governance instrument. Automation is a P3 roadmap item.

### 14.10 The DFAO Registry

The dfao-registry service (port 4009) is the system of record for DFAO lifecycle. It tracks:

- DFAO creation events (the founding members, the scale, the initial configuration).
- DFAO membership changes (joins, exits, merges, splits).
- DFAO governance configuration history.

- DFAO parameter override history.
- DFAO parent-child relationships.
- DFAO suspension and resumption events.

The registry is itself a DAG-anchored service: every DFAO event is a vertex in the contribution graph, and DFAO state is derived by replaying the relevant subgraph.

## 15. Implementation Architecture from the Repository

### 15.1 Repository Overview

The Extropy Engine reference implementation lives at github.com/00ranman/extropy-engine. The repository is a TypeScript monorepo managed by pnpm (>= 9). It contains 29 workspace packages under packages/ and three frontends under frontends/. The current canonical snapshot is main at v3.1.2 (commit-tagged "R is Rarity, not reputation").

The repository is mid-migration. The math is canonical as of v3.1.2 (2026-05-08). The specification text (docs/SPEC_v3.1.md) was written before that migration and has not been fully re-edited; some passages still carry the older R-and-F definitions. The Codex documents this explicitly and treats the canonical reading as: the v3.1.2 changelog plus the xp-formula/src/index.ts source code plus the post-rename portions of contracts/src/types.ts plus the CONTRIBUTION_GRAPH.md invariant.

### 15.2 Source-of-Truth Hierarchy

When two repository files disagree, the priority order to break the tie is:

1. packages/xp-formula/src/index.ts. The README states: "The formula lives in one place. Every service that mints XP imports from there. No reimplementations."
2. packages/contracts/src/types.ts (post-v3.1.2 portions). Single source of truth for inter-service types, branded IDs, formulas, event payloads, token enum.
3. docs/CHANGELOG.md v3.1.2 entry. Most recent authoritative reading.
4. docs/CONTRIBUTION_GRAPH.md. Explicitly marked "Architectural invariant. Hard rule."
5. docs/SPEC_v3.1.md. Canonical spec body, but written before v3.1.2 and not re-edited. Authoritative on architecture and topology, not on R/F definitions.
6. architecture/AUTARKY.md, architecture/SUBSTRATE.md. Authoritative on vision and substrate decision.
7. Companion specs (GOVERNANCE_DEFAULTS.md, IDENTITY.md, PSLL.md, QUEST_MARKET.md).
8. README.md. Audience-facing; reflects v3.1.2 on R and F.
9. docs/ONE_PAGER.md, docs/THREE_LAYER_SEPARATION.md, docs/BOOK_* files. Narrative framing; useful as Codex prose; not load-bearing on protocol details.
10. Historical docs (docs/SESSION_NOTES.md, docs/SPEC_v3.0_DEPRECATED.md, the Randall's Feedback companion-book drafts). Context only; do not implement against.

### 15.3 The Twelve Core Microservices

The deployed core protocol services, with port assignments and primary code locations:

| Package | Port | Status | Purpose |
|---|---|---|---|
| contracts | n/a | Active | Shared types, branded IDs, event-bus, db connection factory. Single source of truth for inter-service contracts. |
| xp-formula | n/a | Active | Pure-function canonical formula. Imported by every minting service. |
| epistemology-engine | 4001 | Active (redefined) | v3.1 mesh observability. Legacy v3.0 decomposition endpoints retained, marked deprecated. Bayesian aggregation lives here. |
| signalflow | 4002 | Active | Validator routing on four factors (domain x reputation x load x accuracy). |
| loop-ledger | 4003 | Active | Loop state machine: OPEN through SETTLED (or FAILED or ISOLATED). Records measurements and consensus votes. |
| reputation | 4004 | Active | Per-domain reputation accrual, decay, streak bonuses. |
| xp-mint | 4005 | Active | Two-phase mint: provisional on close, confirmed/burned on settle. Stamps the v3.1.2 FORMULA_VERSION. |
| dag-substrate | 4008 | Active | Vertex/edge primitives, causal DAG ledger. Single-file implementation. |
| dfao-registry | 4009 | Active | Fractal organization lifecycle (NANO to PLANETARY scales). |
| governance | 4010 | Active | Proposals, conviction voting, quorum, threshold execution. |
| temporal | 4011 | Active | Seasons, decay scheduling, loop timeouts. The “heartbeat.” |
| token-economy | 4012 | Active | 6-token balance management. Accepts canonical and legacy field names during rollout. |
| credentials | 4013 | Active | Verifiable credentials, badges, CAT certifications. |
| ecosystem | 4014 | Active | Cross-service aggregator. |

These are the load-bearing services. They are deployed via `docker compose up --build -d` against a Postgres 16 and Redis 7 base.

### 15.4 The v3.1 Skeleton Services

The v3.1 services whose interfaces are the source of truth but whose bodies are partial:

| Package | Port | Purpose |
|---|---|---|
| identity | 4101 | OAuth + on-device KYC + DID + ZKP (BBS+ default). HMAC nullifier sandbox; production swap to Pedersen+BBS+ is roadmap. All canonical-flow endpoints live, in-memory storage. |
| psll-sync | (TBD) | Personal Signed Local Log maintenance, Merkle-root anchoring to DAG, optional device-to-device sync. |
| quest-market | (TBD) | Micro-quest marketplace, dynamic reward escalation. |
| validation-neighborhoods | (TBD) | Sharded one-tenth blind-slice routing. |
| node-handshake | 4200 | Sandbox VPS-laptop handshake. Endpoints /hello, /capabilities, /dag/replay, /heartbeat. Ed25519 body signing over HTTPS. libp2p migration tracked in gaps. |

### 15.5 Application-Layer Verticals

The repository also contains application-layer packages that build on the core protocol:

| Package | Port | Purpose |
|---|---|---|
| homeflow | 4015 | Family pilot vertical. Setup wizard, household and member management, chores, recipes, pantry, shopping list, XP dashboard. File-backed default, Postgres optional. Google OAuth wired. |
| grantflow-discovery | 4020 | Grant opportunity matching. |
| grantflow-proposer | 4021 | Grant proposal generation. |
| extropialingo | (TBD) | Language-learning vertical with corresponding frontend. |
| levelup-academy | (TBD) | Adaptive learning vertical (standalone Python repo, not yet ported to the monorepo). |
| academia-bridge | (TBD) | Research-side integration. |
| decomposition-kit | (TBD) | Personal-AI-side decomposition primitives. Has tests. |
| bayesian | (TBD) | Bayesian aggregation primitives. Has tests. |
| ethics | (TBD) | Ethical constraint evaluation middleware. Has tests. |
| temporal-service | (default 4002, port collision with signalflow) | Universal Times service: Eon/Age/Era/Epoch/Cycle/Season and others. Distinct from temporal/. |
| api-gateway | (TBD) | Single-file gateway. |

The temporal-service package's default port (4002) collides with signalflow's port. The conflict is acknowledged. The Codex recommends renaming and re-assigning ports as a P2 cleanup item: temporal-heartbeat (4011 for seasons and decay) and temporal-universal-times (a non-colliding port for the base-10 calendar service).

### 15.6 Frontends

Three frontends are present in the monorepo:

| Frontend | Dev Port | Purpose |
|---|---|---|
| character-sheet | 3000 | Validator/contributor profile UI. The user-facing Layer 1 surface. |
| grantflow-ui | 3001 | Grant discovery and proposal UI. |
| homeflow-ui | 3002 | Household management dashboard. |

Frontend status varies. The character-sheet UI is the most developed.

### 15.7 Build Order and Dependency Graph

From docs/DEPENDENCY_GRAPH.md, the build order:

```
contracts -> event-bus
  -> epistemology-engine, signalflow, loop-ledger, reputation
  -> xp-mint (also needs loop-ledger)
  -> dag-substrate -> dfao-registry -> governance
  -> temporal
  -> token-economy (needs xp-mint)
  -> credentials (needs reputation)
  -> ecosystem (aggregates)
  -> homeflow, grantflow-*, identity, psll-sync, quest-market,
     validation-neighborhoods, node-handshake
```

The dependency graph is acyclic. The contracts package is the root (every other package imports from contracts). The event-bus is the second-tier dependency (every service that emits or consumes events depends on event-bus through contracts).

### 15.8 Event Flow (Happy Path)

The verified integration test exercises the canonical event flow:

```
POST /claims
  -> claim.submitted -> loop.opened
  -> claim.decomposed [legacy v3.0; v3.1 pushes this to edge]
  -> task.created xN -> task.assigned xN -> task.completed xN
  -> subclaim.updated xN -> claim.evaluated
  -> loop.consensus_started -> loop.closed
  -> xp.minted.provisional -> reputation.accrued
  -> loop.settled
```

The 12-of-12 integration test passes against this flow as of 2026-05-06.

### 15.9 Event Catalog

The repository defines 22 typed inter-service events. The load-bearing ones:

| Event | Emitter | Consumers |
|---|---|---|
| claim.submitted | API gateway | loop-ledger, epistemology-engine |
| loop.opened | loop-ledger | signalflow, dag-substrate |
| claim.decomposed | epistemology-engine (legacy) | loop-ledger |
| task.created | epistemology-engine (legacy) | quest-market |
| task.assigned | signalflow | quest-market |
| task.completed | quest-market | loop-ledger |
| subclaim.updated | quest-market | epistemology-engine, loop-ledger |
| claim.evaluated | epistemology-engine | loop-ledger |
| loop.consensus_started | loop-ledger | epistemology-engine |
| loop.closed | loop-ledger | xp-mint |
| xp.minted.provisional | xp-mint | dag-substrate, token-economy |
| reputation.accrued | reputation | dag-substrate, token-economy |
| loop.settled | loop-ledger | xp-mint, dag-substrate |
| xp.minted.confirmed | xp-mint | dag-substrate, token-economy |
| xp.burned | xp-mint | dag-substrate, token-economy, reputation |

The full event catalog lives in contracts/types.ts and is not duplicated here.

## 15.10 Verified Implementation Status

| Layer | Status |
|---|---|
| Type system and contracts | Complete (post v3.1.2 rename) |
| Event architecture | Complete; 22 typed inter-service events |
| Core loop lifecycle (5-service happy path) | Implemented; 12/12 integration tests passing |
| XP formula (canonical) | Pure function complete in xp-formula; stamped FORMULA_VERSION = 'canonical-v3.1.2' on every new mint |
| Reputation accrual | Implemented; not bidirectional into XP (correct per architectural invariant) |
| DAG substrate | Single-file implementation; vertex/edge primitives; production gossip layer (libp2p) not yet wired |
| DFAO governance | Implemented; cross-tier escalation rules are open gaps |
| Token economy | 6-token enum and balances; CT mint accepts canonical and legacy fields during rollout |
| Identity layer | All canonical-flow endpoints live; in-memory storage; HMAC nullifier sandbox |
| PSLL | Skeleton; spec frozen |
| Quest marketplace | Skeleton |
| Validation neighborhoods | Skeleton |
| Node handshake (VPS to laptop) | Implemented; VPS-laptop test not yet run |
| HomeFlow family pilot | Live; Google OAuth wired; file-backed default, Postgres optional |
| Frontends | Present; UI maturity varies |

### 15.11 Test Footprint

The repository contains 18 test files distributed as follows:

| Area | Test Files |
|---|---|
| temporal-service | 4 (universaltimes, store, clock, api) |
| identity | 4 (http-flow, escrow, did-vc-zkp, crypto) |
| homeflow | 5 (static, family-pilot subset: psll-append, identity-register, file-db, auth-middleware) |
| ethics | 1 (validator) |
| epistemology-engine | 2 (sybil, observability) |
| decomposition-kit | 1 |
| bayesian | 1 |

The top-level "12/12 passing" claim refers to the integration happy-path event cascade, not the unit tests. The Codex acknowledges that there is no contract test suite validating service APIs against the OpenAPI specs in architecture/; this is a P2 gap and the most likely source of future API drift.

### 15.12 Build and Run

The reference invocation:

```
pnpm install
pnpm run build
docker compose up --build -d
./scripts/test-happy-path.sh
```

The non-docker local run uses scripts/run-integration-test.py, a Python subprocess orchestrator that brings up the five core services and exercises the happy-path event cascade. The orchestrator is convenient for development but is not the deployment path; production deployment uses docker-compose.

Postgres 16 and Redis 7 are wired in docker-compose.yml. Per-service database schemas are defined in scripts/init-db.sql. The schemas include dag (for the substrate), loop_ledger, reputation, tokens, governance, identity, and supporting domain schemas.

### 15.13 Repository-Documented Drifts

The repository documents several known drifts that the Codex tracks but does not propagate:

- **R contradiction:** docs/SPEC_v3.1.md section 6.1 still says R is the reliability coefficient. Everything else says R is rarity. The Codex resolution is clear: R is rarity.
- **F contradiction:** docs/SPEC_v3.1.md section 6.2 still says F is the falsifiability score. Everything else says F is frequency-of-decay. The Codex resolution: F is frequency-of-decay; falsifiability is a separate concept handled at claim opening.
- **T_s definition contradiction:** three competing definitions coexist (settlement-time fraction, timestamp decay factor via exp(-lambda * deltaT), raw seconds in xp-mint). The Codex resolution: T_s is a normalized unitless factor in (T_floor, 1.0]; raw seconds is wrong and settlementTimeSeconds should be renamed.

- **lambda status:** the convenience wrapper uses lambda = 0.001 as a silent default. The Codex resolution: lambda is governance-tunable per domain and should be documented in GOVERNANCE_DEFAULTS.md.
- **Token drift:** the README's architecture section lists GT and RT; the canonical six-token enum and the CONTRIBUTION_GRAPH.md hard rule exclude them. The Codex resolution: six tokens (XP, CT, CAT, IT, DT, EP); GT and RT are not canonical.
- **Domain drift:** the one-pager and the xp-mint RARITY_DEFAULTS table list ecological and spiritual instead of code and temporal. The Codex resolution: canonical eight (cognitive, code, social, economic, thermodynamic, informational, governance, temporal).
- **DFAO scale drift:** the README mentions ECOSYSTEM as a DFAO scale; the enum has NANO/MICRO/MESO/MACRO/PLANETARY. The Codex resolution: five scales per the enum.
- **Epistemology-engine role drift:** the v3.0 decomposition endpoints are still present, marked deprecated, scheduled for v3.2 removal. The Codex resolution: epistemology engine is a mesh observability witness layer; decomposition moves to the edge.

### 15.14 Test Coverage Gaps

The Codex acknowledges the following coverage gaps in the test suite:

- **Contract test suite:** no automated validation of service APIs against OpenAPI specs.
- **VPS-to-laptop node handshake:** implemented but not exercised in an integration test.
- **Validator-collusion scenarios:** game-theoretic analysis documented; no adversarial integration test simulating collusion.
- **Sybil-cluster detection:** epistemology-engine has Sybil unit tests but no end-to-end Sybil cluster scenario.
- **Retroactive burn:** the 30-day window is structurally present in loop-ledger; no integration test exercises the full burn path.
- **DFAO parent-child conflict resolution:** the conflict rules are specified; no integration test exercises a cross-tier conflict.

Each gap is on the roadmap. The Codex documents the gaps so that adopters can supplement the test surface in their own deployments.

### 15.15 Production Deferral Catalog

The following are sandbox implementations deferred for production hardening:

- **HMAC nullifier construction** (production: Pedersen-based commitments with BBS+).
- **In-memory identity storage** (production: persistent store with encryption at rest and access auditing).
- **Single-node DAG substrate** (production: libp2p gossip with multi-node replication and Noise transport).
- **Single-node Postgres** (production: replicated cluster with read replicas).
- **Local docker-compose deployment** (production: orchestrated cluster, monitored, with secret management).
- **HTTPS-only node-handshake** (production: libp2p with multiple transports and DHT for peer discovery).

Each item has a corresponding gap entry in section 19.

# 16. Website and Public Narrative Layer

## 16.1 The Cultural Surface

The Extropy Engine project has a public site at lladnaros.com. The site is primarily a music-and-branding surface for the author's broader work ("glitch-rap meets extropy engine"). It is not, as of the current snapshot, a structured technical documentation hub.

This is intentional. The cultural surface is the entry point for non-technical readers. It frames the project in accessible terms (an alternative to extractive reward systems, an approach to entropy reduction as daily practice). It does not attempt to explain the protocol's mathematics; that is the role of the book, the Codex, and the repository.

## 16.2 Routing from the Public Site

The site is the public routing layer. A visitor lands on lladnaros.com and finds:

- Cultural content (music, art, the author's broader presentation).
- A link to the book (Unfuck the World for a Dollar).
- A link to the GitHub repository (the canonical implementation).
- A link to the technical documentation (this Codex, the v3.1.2 specification, the supporting academic papers).
- Contact information.

The routing is deliberate: the public encounters the cultural framing first, then is invited into deeper technical material as their interest develops. This routing matches the three-layer separation: Layer 1 (user-facing) is the public site; Layer 2 (merchant-facing) is the application materials; Layer 3 (engine) is the repository and the Codex.

## 16.3 The Book as Narrative Companion

The companion book, Unfuck the World for a Dollar, is the narrative form of the Extropy Engine thesis. Its 24 chapters are organized into six parts:

- Part I (chapters 1 to 3): the systems-level autopsy of why existing reward, reputation, and governance systems fail.
- Part II (chapters 4 to 7): the thesis that value is entropy reduction.
- Part III (chapters 8 to 14): the engine itself (XP formula, eight domains, core loop, DAG, DFAOs, tokens, temporal mechanics).
- Part IV (chapters 15 to 18): the defense (antifragility, psychology, game theory, Sybil resistance).
- Part V (chapters 19 to 22): the ecosystem (parallel economy, ecosystem maturity, emergent sovereignty, universal time).
- Part VI (chapters 23 to 24): the call to action.

Plus appendices A-D and a glossary.

The book and the Codex are complementary, not redundant. The book is the narrative case for why the Engine matters. The Codex is the operational specification of what the Engine is. A reader can begin with either; the book is more accessible, the Codex is more precise.

## 16.4 Known Stale Passages in the Book

The book is mostly aligned with v3.1.2, with one confirmed stale passage in chapter 8 (the CLOSED state description still says "R: from the contributor's history (reputation service)" rather than the canonical "R: from rarity assessment"). The correction is in the correction ledger (C1) and applies to future printings or revisions.

The book's appendix A is canonical-correct on the symbol definitions but contains one phrase ("R rewards who you are: demonstrated reliability") that is ambiguous and should be clarified in future revisions (C2 in the correction ledger).

## 16.5 The Academic Papers

The Extropy Engine project is supported by three formal academic-style papers:

- **Formal Foundations for Emergence-First Governance** (Randall's Feedback V4): the formal governance theory, six theorems with proofs.
- **When the Signal Eats the Source**: representational fidelity decay, the k-parameter taxonomy, the four observable phases.
- **God as Emergent Entropy Reduction**: the philosophical extension treating divinity as a functional name for recurring entropy-reducing patterns.

The first two are load-bearing for the protocol's theoretical foundation. The third is a philosophical companion that the Codex acknowledges as relevant but optional.

## 16.6 The God Paper as Philosophical Companion

The God paper proposes that "God" can be reconstructed as a falsifiable functional concept: the emergent function by which feedback-driven systems reduce entropy across domains. The proposal is explicit that it does not prove the existence of a personal deity, does not eliminate metaphysical mystery, and does not validate or invalidate religious experience as a category. It offers an operational reconstruction: a process participates in divine function to the extent that it produces validated, recursive, beneficial entropy reduction across domains without merely exporting greater disorder elsewhere.

The relevance to the Extropy Engine is that the same engineering posture (validated entropy reduction, falsifiability, audit, Goodhart resistance) is presented as the testable form of what religious language has historically pointed at imprecisely. The Codex treats this as a philosophical extension and not as a load-bearing part of the protocol specification. Readers can engage with or skip the God paper without affecting their understanding of the protocol.

One terminology note: the God paper glosses F as "feedback closure strength" rather than as the canonical "frequency-of-decay." This is a terminology slip relative to the v3.1.2 specification. The correction ledger entry C6 documents it. Feedback closure strength is a real concept (it relates to V_c and to C(t) in the Signal paper), but it is not what F means in the XP formula. Readers of the God paper should note this when cross-referencing the formal notation.

## 16.7 The One-Pager

The one-pager (ONE_PAGER.md and the PDF derived from it) is the business-pitch document. It targets investors, partners, and merchant-side decision-makers. It frames the protocol in terms of a two-sided market with a flywheel, lists the canonical formulas, and presents the business model table.

The one-pager has a known stale passage: its domain list reads "thermodynamic, informational, social, economic, ecological, governance, cognitive, spiritual" rather than the canonical eight (cognitive, code, social, economic, thermodynamic, informational, governance, temporal). This is correction C3 in the correction ledger.

The one-pager's terminology on R is correct ("R is rarity, a property of the loop, not the actor"). The terminology on F is correct ("F is frequency-of-decay"). The token list is correct (XP, CT, CAT, IT, DT, EP, no GT, no RT). The domain list is the only known stale element.

### 16.8 How These Materials Connect

The relationship among the public-facing materials:

- **Public site (lladnaros.com):** the cultural entry point, primarily artistic and brand-oriented.
- **Book (Unfuck the World for a Dollar):** the narrative case for the protocol, accessible to non-technical readers.
- **One-pager:** the business pitch, targeting investors and merchants.
- **Codex (this document):** the comprehensive technical specification, targeting technical readers and implementers.
- **Repository (github.com/00ranman/extropy-engine):** the canonical implementation; the source of truth at the code level.
- **Academic papers:** the formal foundations (RF V4, the Signal paper, the God paper).

A reader's path through these materials depends on their role:

- A curious citizen reads the public site and the book.
- An investor reads the one-pager and the relevant sections of the book.
- A merchant evaluating adoption reads the one-pager and sections 13 and 14 of the Codex.
- A technical reader integrating against the protocol reads sections 4, 5, 9, 10, 11, 12, 14, and 15 of the Codex, then drops into the repository.
- A peer reviewer reads sections 4, 5, 6, 7, 17, and 18 of the Codex, then drops into the relevant academic papers.

The Codex serves as the structural index across these materials.

## 17. Security, Attack Surfaces, and Failure Modes

### 17.1 The Adversarial Model

The Extropy Engine is not designed for a benign world. Its threat model assumes:

- A non-trivial fraction of participants will attempt to game any visible metric.
- Some fraction of validators will collude if collusion is profitable.
- Sybil attackers will create multiple identities if identity is cheap.
- Reputation laundering services will emerge if reputation is valuable and gameable.
- Regulators may attempt to compel disclosure beyond what the protocol's selective-disclosure layer would otherwise permit.
- Cryptographic primitives may be broken or deprecated on a multi-decade horizon.

The protocol's security posture is structural, not exhortative: it does not ask actors to behave well; it makes misbehavior expensive or impossible.

## 17.2 The Six Primary Attack Surfaces

**1. XP Inflation via Formula Gaming.** An attacker attempts to maximize XP through manipulation of one of the five formula terms.

- Defense against R inflation: R is a per-domain table, not actor-bound. Inflating R requires governance proposal to change the table, which is visible and contestable.
- Defense against F evasion: the fingerprint (submitter_id, claim_type, primary_domain, DFAO_id) prevents trivial relabeling. Cross-DFAO arbitrage (a known P2 issue, see 17.7) is the open vector.
- Defense against Delta S inflation: validators verify the claim against the instrument and the baseline. The retroactive-burn window catches false claims that initially pass validation.
- Defense against w dot E gaming: w is governance-tunable but governance proposals are visible. E is per-claim and must be supported by evidence.
- Defense against log(1/T_s) inflation: the T_floor and the explicit cap on the log term bound the maximum value of this multiplier.

**2. Reputation Laundering via XP.** This attack is structurally impossible by the architectural invariant. Reputation does not enter XP. A high-reputation actor's claim earns the same XP as a low-reputation actor's claim for the same loop. This is the most important architectural choice in the protocol.

**3. Validator Collusion.** A pool of validators conspires to confirm fraudulent claims for a share of the payoff.

- Defense: the 30-day retroactive-burn window allows independent observers to submit falsifying evidence. Validators whose consensus is later contradicted take a rho_validator penalty exceeding the expected collusion gain.
- Defense: blind-slice validation (one-tenth of context per validator) prevents a single validator from understanding the full claim, raising the coordination cost for collusion.
- Defense: SignalFlow routing prevents validators from selecting which claims they validate. Routing is determined by the four-factor scoring.
- Game-theoretic floor: at validator population N >= 10 and detection probability p >= 0.3, no cartel strategy is profitable in expectation. See the cartel-threshold-analysis.md for the full derivation.

**4. Sybil Attack.** An attacker creates multiple identities to manipulate consensus or to multiply rewards.

- Defense: the hybrid identity layer (OAuth + on-device KYC + DID + ZKP) raises the cost of identity creation.
- Defense: the per-context nullifier prevents an attacker from acting as multiple identities in a single context (single proposal, single loop).
- Open issue: the three KYC options (ID scan, biometric, trusted-issuer handoff) have different Sybil-resistance properties. The threat model per method is documented; some methods are weaker than others.
- Defense at scale: as the network grows, Sybil attack cost scales with the number of honest loops required to compromise consensus.

**5. Identity Compromise.** An attacker steals a user's private key and acts as that user.

- Defense: the PSLL hash chain makes mutation of past entries detectable.

- Defense: the user can rotate their DID (the new DID is bound to the prior identity through a documented rotation event in the DAG).
- Defense: the threshold reveal scheme (7-of-12) does not depend on the user's private key; even a fully compromised user cannot be re-identified without 7 of 12 external shareholders cooperating.

**6. Regulatory Compulsion.** A state actor compels the protocol to reveal user identity or to restrict user actions.

- Defense: the threshold reveal scheme ensures that no single actor (including the protocol's hosts) can comply unilaterally with a compulsion order. Compliance requires 7 of 12 ecosystem-validator shareholders to cooperate, which itself requires legitimate cause-shown.
- Defense: the user's identity material lives on the user's device. The protocol cannot reveal what it does not have.
- Open issue: the regulatory exposure of CT-to-fiat bridges (the Howey test in U.S. securities law) remains a P1 open problem. See section 19.

### 17.3 Goodhart Pressure as a Permanent Feature

The Signal paper's representational fidelity decay theory predicts that any metric the Engine uses will become a target. The protocol does not claim immunity. The protocol claims structural resistance, visible correction, and the diagnostic use of failure rates.

Goodhart pressure manifests across the protocol's surfaces:

- Rarity table gaming: actors attempt to convince the network to set high R values for action classes they specialize in. Governance proposals to revise the rarity table are visible and contestable.
- Domain weight gaming: actors attempt to concentrate the weight vector w on domains they excel in. DFAO-level governance prevents protocol-wide weight gaming; cross-DFAO arbitrage remains a vector.
- Validator-routing gaming: actors attempt to game the SignalFlow scoring to route preferred validators to their claims. The routing is deterministic given the inputs, which limits this attack to manipulating the inputs (which is what the cross-layer coupling discussion addresses).
- Instrument gaming: actors attempt to convince the network to adopt instruments they can manipulate. Instrument adoption is governance-tunable and visible; manipulated instruments produce visible drift.

The protocol's response in every case is the same: surface the pattern, route it through governance, revise the parameter or the instrument or the routing weight, and preserve the DAG record so that the lineage of revisions is auditable.

### 17.4 Cross-DFAO F Fingerprint Arbitrage

The peer review correctly flagged a specific attack vector: a contributor who submits the same type of claim across many DFAOs gets full F = 1 for each new fingerprint combination (because the DFAO_id is part of the fingerprint tuple). In a sufficiently large ecosystem with many DFAOs, this could functionally neutralize the anti-grind intent of F for prolific contributors.

The Codex acknowledges this as an open P2 issue. The proposed mitigations:

- Cross-DFAO frequency tracking: introduce a global frequency counter that decays F across DFAOs, weighted by claim similarity.

- Substantive variation requirement: require the validator pool to confirm that a claim's cross-DFAO presentation involves substantive variation, not trivial relabeling.
- Reputation-weighted F: in v3.2 or later, F can be modulated by the contributor's cross-DFAO contribution history.

These mitigations are not yet implemented. The Codex documents the gap.

### 17.5 Loop Lifecycle Race Conditions

The loop lifecycle state machine is implemented in loop-ledger. Race conditions to consider:

- Concurrent validation submissions: two validators submit their judgments at the same instant. The event-bus serializes the submissions; both are recorded; the aggregator uses the time-ordered set.
- Simultaneous consensus and falsification: a loop's consensus aggregation reaches V_c above threshold at the same instant as an independent observer submits falsifying evidence. The state machine prioritizes the latest event by timestamp; if the falsification arrives during VALIDATING, the loop transitions to FAILED; if it arrives after CLOSED but within the retroactive window, it triggers a re-evaluation that can move the loop to FAILED.
- Cross-service ordering: the event-bus guarantees in-order delivery within a topic. Cross-topic ordering is not guaranteed; services that depend on cross-topic ordering must use the DAG vertex sequence as the canonical ordering.

The state machine's formal transition table has not yet been fully published; this is a P2 gap.

### 17.6 PSLL-DAG Divergence

A subtle attack: the user mutates their PSLL retroactively, then anchors a new Merkle root that does not reflect the original entries. The attack is detected because:

- Each PSLL entry is hash-chained to the previous entry. Mutation of a past entry invalidates every subsequent hash.
- The Merkle root anchored to the DAG commits to the entire PSLL up to the anchor point.
- An anchor reference to a non-existent (mutated) entry fails to verify on Merkle proof.

If a divergence is detected, the protocol does not silently overwrite. A dispute review is opened. The DAG anchor is canonical for the network; the user's PSLL is canonical for the user. Divergence triggers a process, not a silent loss of integrity.

### 17.7 The Godel Boundary

A self-referential claim in the contribution graph can produce a paradox: a claim that asserts the falsity of its own validation, or a governance proposal that nullifies the protocol's ability to govern itself. The Godel Boundary Watchdog is a roadmap component intended to detect and quarantine such claims.

The current implementation does not include the Watchdog. Self-referential claims are an open P3 vector. The Codex documents this honestly: the protocol can handle most contribution claims without paradox; certain edge cases require formal handling that is not yet specified.

### 17.8 Cryptographic Agility

The protocol's hash and signature defaults (BLAKE3 / SHA-256 for hashes, Ed25519 for signatures, BBS+ for selective disclosure) are governance-tunable. If a primitive is deprecated, governance can adopt a new default.

The adoption is forward-only: existing PSLL entries and DAG vertices retain their original cryptographic commitments, but new entries use the updated defaults.

This is the operational form of cryptographic agility. The protocol does not lock itself into any specific primitive; it provides the mechanism for revising primitives as the cryptographic landscape evolves.

### 17.9 The Cultural Friction Vector

A non-technical attack surface: cultural resistance to the base-10 temporal layer or to other protocol-level conventions. If the protocol's conventions conflict with established institutional rhythms (the seven-day week, the fiscal-year calendar), adoption suffers regardless of the protocol's technical merit. The Codex acknowledges this as a P3 cultural friction vector and notes that the temporal layer is governance-tunable per DFAO.

### 17.10 What the Threat Model Does Not Cover

The Codex does not claim coverage of:

- Side-channel attacks on user devices (the protocol's autarky assumes the user's device is reasonably secure; if the device is compromised, the protocol cannot defend the user).
- Supply-chain attacks on the reference implementation (the code is open-source; users compile and run at their own risk).
- Catastrophic cryptographic breaks (quantum attacks against Ed25519, etc.; cryptographic agility addresses this on a normal-progression timeline but not on a sudden-break timeline).
- Adversarial cultural narratives that delegitimize the protocol without engaging with its mathematics.

These are real risks. The protocol does not pretend to address them within its specification.

### 17.11 The Honesty Clause

The original technical specification's section 2 contains what the peer review called "an honesty clause" that distinguishes this protocol from typical crypto white papers. The clause states that the protocol is a sandbox, that significant gaps remain, and that the model is allowed to lose. The Codex preserves and extends this clause: the honest enumeration of failure modes (here in section 17), the explicit gap catalog (section 19), and the documented correction ledger (section 4) collectively communicate the protocol's posture of falsifiability. A protocol that claims to be unbreakable is either lying or has not been examined carefully. The Extropy Engine claims to be examinable.

## 18. Peer Review Response and Applied Fixes

### 18.1 The Peer Review

The Extropy Engine v3.1.2 specification was submitted to external peer review. The review is preserved in Peer-Review-Extropy-Engine-v3.1.2-Technical-Specification.docx. The reviewer's overall verdict was:

> "Strong architecture with credible philosophical grounding. Several formula-level issues require resolution before the specification can be considered fully rigorous."

The reviewer commended several aspects:

- The Digital Autarky principle is principled and architecturally enforced.
- The single-source-of-truth pattern for the XP formula is correct engineering.

- The v3.1.2 R/F correction is important and clearly stated.
- The epistemology-engine redefinition (from central decomposition to mesh observability) is the right architectural correction.
- The provisional defaults table is the right pattern.
- The honesty clause in section 2 distinguishes this from typical crypto white papers.
- Goodhart pressure as diagnostic fuel is the correct epistemic stance.

The reviewer identified specific issues at three priority levels: P0 (must fix before claiming rigor), P1 (resolve in short order), P2 and P3 (improve over time). Each issue is addressed below.

## 18.2 P0 Issues and Resolutions

**P0-1: F naming collision in section 10.5.** The original text stated that "Aggregation of one-tenth blind slice validation produces F." This is a naming collision: F is already defined as the frequency-of-decay penalty in the XP formula. The validation aggregation output is a different quantity.

*Resolution:* The Codex (and the v3.1.2 spec correction) renames the validation aggregation output to **V_c (validation confidence)**. F in the XP formula is frequency-of-decay only. V_c lives in the validation lifecycle and gates the transition from VALIDATING to CONSENSUS. The two symbols are kept distinct in all documentation going forward. See section 4.6 (correction ledger, C4) and section 10.1.

**P0-2: log(1/T_s) boundedness claim in section 6.5.** The original text stated that "log(1/T_s) preserves boundedness." This is mathematically false: the function is unbounded above as T_s approaches 0. A claim settled in an arbitrarily short time window yields arbitrarily large XP from this term alone.

*Resolution:* The Codex (and the v3.1.2 spec correction) introduces a minimum T_s floor (T_floor, governance-tunable, default 0.01) and an explicit cap on the log term (log(1/T_floor)). The corrected text reads: "The time term grows logarithmically and approaches infinity as T_s approaches 0; the protocol requires a minimum T_s floor and applies an explicit cap to prevent unbounded values." See sections 4.6 (correction ledger, C5) and 5.7.

## 18.3 P1 Issues and Resolutions

**P1-1: Cross-domain Delta S normalization not formally addressed.** The original text asserts that cognitive entropy, social entropy, and so on are domain-specific instances of the same phenomenon, but does not show how domain-specific Delta S values are normalized before aggregation.

*Resolution:* The Codex documents a per-domain normalization function $\text{normalize\_d}(\text{delta\_s\_raw})$ that converts domain-native units to a unitless scalar in a comparable range. The normalization function is governance-tunable per domain and is published in the operationalization documentation per domain. Section 5.6 documents this. Open work: the calibration of normalize_d across domains is preliminary and is a P1 open engineering gap (see section 19).

**P1-2: Irreducible form derivation absent.** The original text introduces XP = Delta S / c_l^2 as a structural analogy but does not derive the relationship to the canonical five-term formula.

*Resolution:* The Codex states explicitly that the operational formula is the five-term version; the irreducible form is a structural analogy whose practical role is to anchor the c_l calibration table for future revisions of the operational formula. The derivation of a formal limit relating the two forms is not provided; this is acknowledged as an open theoretical question. Section 5.8 documents the resolution.

**P1-3: PSLL hash function and signature scheme unspecified.** The original text describes the PSLL as hash-chained and cryptographically signed but does not specify the primitives.

*Resolution:* The Codex pins the defaults: BLAKE3 (or SHA-256 fallback) for content hashes, Poseidon where ZKP circuits require, Ed25519 for entry signatures, BBS+ for selective disclosure. These are governance-tunable. Section 12.12 documents this.

**P1-4: Token economy enum members unexplained in spec body.** The original spec's section 13.1 lists "XP, CT, EP, IT, GT, RT" but does not explain EP, IT, GT, or RT.

*Resolution:* The Codex specifies all six canonical tokens (XP, CT, CAT, IT, DT, EP) with full definitions in section 13. GT and RT are explicitly marked as not canonical: they appeared in older drafts and in transitional code, but the canonical six-token set is fixed by the CONTRIBUTION_GRAPH.md hard rule. Section 4.6 (correction ledger, C7) documents this.

### 18.4 P2 and P3 Issues and Resolutions

**P2-1: Cross-DFAO F fingerprint arbitrage.** Acknowledged in section 17.4. Mitigations identified; implementation deferred to v3.2 or later.

**P2-2: SignalFlow reputation routing creates undocumented cross-layer coupling.** Documented in section 10.3. The coupling is intentional (reputation conditions routing, not value minting) but is now explicit in the specification.

**P2-3: On-device KYC Sybil resistance not threat-modeled per method.** Documented in section 17.2. The three KYC options have different threat properties; the per-method threat model is now part of the specification.

**P2-4: DFAO inheritance conflict resolution undefined.** Resolved in section 14.7. Children inherit defaults from parents; overrides require parent-defined safety bounds; conflicts beyond those bounds require parent governance approval.

**P3-1: 7-of-12 reveal threshold lacks adversarial derivation.** Resolved in section 12.7. The derivation: under threshold-signature semantics, an adversary controlling fewer than 7 shareholders cannot reconstruct the secret. The 7-of-12 ratio provides margin above 50%+1 while remaining practically achievable. The ratio is governance-tunable.

**P3-2: Loop lifecycle state machine lacks formal transition table.** Acknowledged in section 17.5. The full transition table publication is on the roadmap.

**P3-3: c_l bootstrap phase not specified.** Acknowledged as a P3 calibration gap. Per-domain c_l values are seed estimates; operational calibration data from a running deployment will refine them.

**P3-4: References section incomplete.** Acknowledged. The Codex does not include a formal bibliography; citations are inline by name (Shannon 1948, Landauer 1961, Bennett 2003, Friston et al. 2023, Metelli et al. 2023, Skalse et al. 2022, Karwowski et al. 2023, and others). A formal references appendix is a documentation roadmap item.

### 18.5 What the Peer Review Did Not Catch

The peer review is exhaustive but not omniscient. The Codex acknowledges several issues that emerged during the consolidation process but were not explicitly flagged in the peer review:

- The R contradiction inside contracts/types.ts line 9 (the file header still has the old R definition even though the field name below was renamed). Resolved by updating the file header in lockstep with future code changes.

- The book chapter 8 stale passage (the CLOSED state's reference to "reputation service" for R). Resolved by correction C1 applied to the Codex; future book printings should incorporate the correction.
- The one-pager domain list error (ecological and spiritual instead of code and temporal). Resolved by correction C3.
- The God paper's gloss on F as "feedback closure strength." Resolved by correction C6, flagging the slip and noting that F in the canonical formula is frequency-of-decay.
- The double `temporal-service` and `temporal` packages with their port collision. Documented in section 15.5 as a P2 cleanup item.
- The DFAO scale drift (ECOSYSTEM in the README versus the five-scale enum). Documented in section 15.13.

### 18.6 The Peer Review's Verdict, Updated

The peer review's verdict was: "Strong architecture with credible philosophical grounding. Several formula-level issues require resolution before the specification can be considered fully rigorous."

The Codex represents the post-peer-review state: the P0 issues are corrected, the P1 issues are addressed (with some open calibration work acknowledged), the P2 and P3 issues are documented with mitigation plans. The Codex is the consolidated specification that the peer review's corrections produce when fully applied.

The protocol's claim of rigor is now stronger than at the time of peer review, but it is not absolute. The model is allowed to lose. The Codex's open gap catalog (section 19) is the explicit acknowledgment of what remains unproven.

### 18.7 The Honest State of the Specification

A short statement of where the specification stands:

- The XP formula is well-defined and single-sourced in code.
- The R-F-rho separation is an architectural invariant and is enforced.
- The eight canonical domains and the six canonical tokens are fixed.
- The validation model is specified at the architectural level, with V_c (not F) as the aggregation output.
- The DAG substrate, the loop lifecycle, the identity layer, the PSLL, and the governance layer are specified.
- The peer-review-identified P0 errors are corrected.
- The cross-domain Delta S normalization is documented; the calibration is preliminary.
- The c_l values are seed estimates; calibration is open.
- Several skeleton services (PSLL-sync, quest-market, validation-neighborhoods) have specified interfaces but partial implementations.
- The production cryptographic primitives (Pedersen-based nullifiers, libp2p gossip, Noise transport) are deferred to operational hardening.

This is the honest state. The protocol is implementable, partially implemented, peer-reviewed, and open to falsification.

# 19. Open Engineering Gaps and Roadmap

## 19.1 The Honest Gap Catalog

The Extropy Engine specification carries an explicit catalog of open engineering gaps. As of v3.1.2, the catalog contains 65 items across 13 categories. The honesty of this catalog is a deliberate choice: a specification that claims completeness is either lying or has not been examined.

The gaps are organized by priority. P1 items are critical-path engineering work that must be addressed before significant deployment beyond sandbox. P2 items are robustness and security work that should be addressed before adversarial production exposure. P3 items are forward-looking research and platform work that will mature the system over a multi-year horizon.

## 19.2 P1 Gaps (Critical Path)

**P1.1 Consensus mechanism details (7 items).** The validation consensus uses Bayesian aggregation with logodds combination. Specific open details:

- The exact priors for the Beta(alpha, beta) conjugate in each domain.
- The handling of `inconclusive` and `refused` verdicts in the aggregation.
- The mathematical handling of validator-judgment correlations (when validators see overlapping context, their judgments are not independent).
- The convergence proof for the aggregation under adversarial conditions.
- The handling of late-arriving validator judgments (after the consensus threshold has been reached).
- The aggregation behavior when the validator pool is smaller than the typical default (5 to 12).
- The cross-pool aggregation for very-large-scope claims that require multiple validation pools.

**P1.2 Economic attack resistance (6 items).** The structural defenses against economic attack are documented, but several specifics remain open:

- The exact relationship between CT lockup duration and cross-DFAO transfer attack resistance.
- The economic equilibrium under high CT-to-fiat conversion volume.
- The handling of merchant-side concentration (when a small number of merchants accumulate disproportionate EP float).
- The treatment of EP-denominated debt or credit instruments.
- The protocol's response to coordinated short-selling of CT (if a secondary market develops).
- The treasury's optimal yield strategy on EP float.

**P1.3 Validator selection (5 items).** SignalFlow routing is specified at the architectural level, but several details are open:

- The exact rate-limiting on validator selection to prevent over-routing to specific validators.
- The handling of validator availability changes mid-loop.
- The treatment of newly-onboarded validators with no historical accuracy data.
- The cross-domain routing for multi-domain claims.
- The fallback routing when the primary pool cannot reach consensus.

**P1.4 Cross-domain calibration (6 items).** The normalize_d function exists per domain, but the calibration is preliminary:

- The reference contributions per domain that anchor normalize_d.
- The cross-domain comparability of the resulting normalized values.
- The handling of multi-domain claims where domains have different calibration maturity.
- The revision process for normalize_d when domain instruments mature.
- The audit trail for normalize_d revisions.
- The handling of historical XP values when normalize_d is revised.

**P1.5 Verdict vocabulary (2 items).** The five-verdict vocabulary (confirmed, supported, inconclusive, contradicted, refused) is fixed, but the aggregation treatment is incomplete:

- The exact weighting of supported relative to confirmed in the Bayesian update.
- The treatment of inconclusive for the purposes of validator reputation accrual.

### 19.3 P2 Gaps (Robustness and Security)

**P2.1 DAG distributed consensus (5 items).** The current DAG substrate is single-node. Distributed deployment requires:

- libp2p gossip layer for inter-node DAG replication.
- Noise protocol for inter-node transport security.
- DHT-based peer discovery.
- Conflict resolution when two nodes propose different vertices in the same time window.
- Eventual-consistency guarantees and bounds.

**P2.2 Retroactive validation specifics (4 items).** The 30-day retroactive-burn window is specified. Specific open issues:

- The exact arithmetic of partial burns (when new evidence contradicts only part of a claim).
- The handling of cascading burns (when burning a claim invalidates downstream claims that depended on it).
- The treatment of the validator-reputation penalty when consensus was reached on a partial set of validators.
- The handling of claims that are extended into multiple retroactive windows.

**P2.3 DFAO governance edge cases (5 items).** Cross-tier and edge cases:

- Detailed cross-tier escalation rules when child DFAO behavior threatens parent DFAO integrity.
- DFAO merger and split mechanics.
- DFAO inactivity handling (when a DFAO has no active participants).
- DFAO governance attacks (when a single participant accumulates disproportionate voting weight).
- The handling of contested DFAO leadership transitions.

**P2.4 Token economy equilibrium (4 items).** The six-token economy has open equilibrium questions:

- The long-run distribution of CT under various transfer-friction settings.
- The IT decay rate's interaction with governance participation patterns.
- The cross-DFAO portability of CAT and the handling of CAT issuer reputation.
- The EP-to-fiat conversion rate stability under merchant-network growth.

**P2.5 Privacy and access control (5 items).** Several privacy edge cases:

- The handling of ZKP-proven statements that turn out to be false (the proof was correct but the underlying statement was incorrect due to issuer error).
- The revocation of compromised credentials.
- The cross-DFAO context separation when a user participates in multiple DFAOs.
- The handling of inferred-identity attacks (correlation across multiple selective-disclosure events).
- The treatment of identity recovery when the user loses their private key.

### 19.4 P3 Gaps (Future Work)

**P3.1 Skill DAG (3 items).** A more elaborate skill-tracking substrate:

- A skill ontology with cross-domain skill mappings.
- Skill decay and refresh mechanics.
- Skill-based DFAO matching for project formation.

**P3.2 Oracle integration (4 items).** Integration with off-chain data sources:

- Trusted-oracle attestation for thermodynamic measurements (energy meters, IoT sensors).
- Multi-oracle aggregation for high-stakes claims.
- Oracle compensation mechanics.
- Oracle dispute resolution.

**P3.3 Performance and scalability (5 items).** As the network grows:

- Sharding strategy for the DAG substrate.
- Caching and indexing for high-frequency queries.
- Bandwidth optimization for the gossip layer.
- Storage compaction (event-sourced systems grow without bound; compaction with cryptographic proofs of the elided history is open work).
- Latency targets for the loop lifecycle under load.

**P3.4 Migration and upgrade paths (4 items).** Long-term protocol evolution:

- The mechanism for upgrading the XP formula without invalidating historical XP values.
- The mechanism for adding a new canonical domain (if ever needed).
- The mechanism for retiring a deprecated token type.
- The cross-version compatibility guarantees during protocol upgrades.

### 19.5 Most-Cited Specific Gaps

A short list of the most-cited specific open problems:

- **Godel Boundary Watchdog:** detection and quarantine of paradox-inducing self-referential claims. Currently undefined; on the P3 roadmap.
- **c_l calibration bootstrap:** the per-domain causal closure speeds are seed estimates pending operational calibration data. P1 calibration item.
- **Network-density threshold:** at what level of participant density does voluntary adoption tip into network-effect-dominant behavior? Empirical question, awaits deployment data.
- **Regulatory exposure (Howey test):** the U.S. securities-law analysis of CT-to-fiat bridges. P1 legal item; not a code change but a structural risk to economic deployment.

- **Cultural friction with base-10 temporal layer:** the optional base-10 calendar (Eon/Age/Era/Epoch/Cycle/Season) conflicts with seven-day-week institutions if activated. P3 cultural item; the layer is currently optional and DFAO-gated.

### 19.6 Roadmap to v3.2

The next minor version (v3.2) targets:

- Removal of the legacy v3.0 decomposition endpoints from the epistemology-engine.
- Removal of the GT and RT legacy field acceptance in the token-economy package.
- Production swap from HMAC nullifiers to Pedersen-based commitments.
- libp2p gossip layer initial implementation.
- Cross-DFAO frequency-tracking mitigations for the F arbitrage attack.
- Formal publication of the loop lifecycle transition table.
- First operational calibration data for normalize_d per domain.

### 19.7 Roadmap to v4.0

The next major version (v4.0) targets:

- Distributed DAG substrate with multi-node consensus.
- Production-grade identity layer with persistent storage and access auditing.
- Godel Boundary Watchdog initial implementation.
- Automated Complexity Thermostat (replacing the manual governance-tuning process).
- Cross-domain calibration matured against multi-year operational data.
- Comprehensive contract test suite against OpenAPI specs.
- Adversarial integration test suite simulating Sybil, collusion, and identity-compromise attacks.

### 19.8 The Sandbox Posture

The Codex maintains a clear distinction throughout: the Extropy Engine is a sandbox protocol. The reference implementation proves the architecture. The specification documents the design. The peer review and the correction ledger document what has been examined and addressed. The gap catalog documents what remains.

Nothing in the Codex should be read as a production-readiness claim. The protocol is mature enough to deploy in controlled pilots (the HomeFlow family pilot is an example). It is not mature enough for adversarial production at scale. The honest enumeration of gaps is the explicit acknowledgment.

### 19.9 Falsification Conditions for the Whole Project

Beyond per-component gaps, several conditions would falsify the project as a whole:

- **Domain instrument failure:** if all eight canonical domain instruments turn out to be Goodhart-vulnerable at the same rate, the protocol's domain-separated structure has no benefit over a single naive metric.
- **XP dominated by reputation:** if reputation correlates strongly with XP in practice (despite the architectural invariant), the architectural invariant is being violated somewhere in the deployment.

- **Validator collusion without graph signature:** if collusion can be sustained without producing the predicted graph-structural signature (which the cartel-threshold analysis identifies), the game-theoretic floor is wrong.
- **Cross-domain normalization instability:** if normalize_d cannot be calibrated to produce stable cross-domain XP comparisons, the cross-domain claim aggregation has no operational meaning.
- **Users farming metrics more efficiently than producing real entropy reduction:** if the cost of gaming exceeds the cost of legitimate contribution by less than the protocol's defenses anticipate, the protocol is gameable in practice even though it is not gameable in theory.

The model is allowed to lose. The Codex documents these conditions explicitly so that adopters and adversaries alike know what would constitute project-level failure.

## 20. Glossary and Formula Reference

### 20.1 Canonical Symbol Glossary

| Symbol | Name | Description |
|---|---|---|
| R | Rarity | Action-class scarcity per domain. Property of the loop, not the actor. Range [0.1, 10.0]. Lives in XP formula. |
| F | Frequency-of-decay | Anti-grind penalty. Keyed on (submitter_id, claim_type, primary_domain, DFAO_id). Range (0, 1]. Lives in XP and CT formulas. |
| rho | Reputation density | Per-domain reputation accrual. Lives in CT formula and validator routing. Does not enter XP. |
| Delta S | Entropy reduction magnitude | Verified disorder reduction in domain-native units pre-normalization. Range (0, infinity). |
| w | Domain weight vector | Eight-element vector expressing DFAO priorities across the canonical domains. |
| E | Evidence vector | Eight-element vector expressing claim evidence across the canonical domains. |
| T_s | Normalized settlement-time factor | Unitless factor in (T_floor, 1.0]. Not raw seconds. |
| T_floor | Settlement-time floor | Governance-tunable lower bound on T_s. Default 0.01. |
| V_c | Validation confidence | Aggregation output from validation. In [0, 1]. Not F. |
| c_l | Causal closure speed | Per-domain constant in the irreducible-form analogy. Calibration target. |
| lambda | Decay rate | Governance-tunable decay constant for XP and IT. Per-domain. |
| L | Local merchant loyalty multiplier | Multiplier in EP formula. Merchant-context-specific. |
| C | Capability | Contributor's measured competence in a domain. Lives in CT formula. |
| Delta | Entropy delta (CT context) | Analogous to Delta S in XP context. Lives in CT formula. |

## 20.2 Formula Reference

**XP formula:**

[ XP = R F S (w E) (1 / T_s) ]

Constraints: all five terms must be strictly positive for XP to mint. T_s must be normalized; raw seconds is not accepted. The log term is capped at log(1/T_floor).

**CT formula:**

[ CT = C F E ]

Where C is capability, F is frequency-of-decay (same semantics as XP), rho is reputation density (where reputation legitimately enters), Delta is the CT-context entropy delta, and E is the eight-domain weighting / essentiality factor.

**EP formula:**

[ EP = XP L ]

Where L is the local merchant loyalty multiplier.

**Irreducible form (structural analogy, not operational):**

[ XP = S / c_l^2 ]

Where c_l is the per-domain causal closure speed. Presented as a structural analogy to E = mc^2; not a physical law; not the operational formula.

**Representational fidelity decay (Signal paper):**

[  = -k S(t) F(t) + r C(t) (1 - F(t)) ]

Where F(t) is representational fidelity, S(t) is social load, C(t) is corrective feedback strength, k is the domain susceptibility coefficient, and r is the restoration rate.

**RF feedback primacy (Axiom 1):**

[ S_{t+1} = S_t + F(S_t, E_t, G_t) ]

**RF goal evolution (Axiom 2):**

[ G_{t+1} = G_t + L(G_t, O_t) ]

## 20.3 Canonical Domain Glossary

The eight canonical entropy domains, with their primary instruments:

- **Cognitive:** disorder in understanding, learning, conceptual compression. Instruments: explanation length, error rate reduction, rubric assessments.
- **Code:** disorder in software systems. Instruments: cyclomatic complexity, bug density, test coverage.
- **Social:** disorder in trust networks, cooperation, conflict. Instruments: network cohesion, conflict-incident reduction, validated trust surveys.
- **Economic:** disorder in allocation, waste, scarcity. Instruments: utilization, throughput, allocation efficiency.

- **Thermodynamic:** physical disorder, energy efficiency. Instruments: kilowatt-hours, joules per kelvin, material recovery, emissions.
- **Informational:** disorder in records, signal-to-noise, archival coherence. Instruments: error rate, completeness, retrieval latency.
- **Governance:** disorder in decision systems. Instruments: decision latency, reversal frequency, participation quality.
- **Temporal:** disorder in time allocation, sequencing, synchronization. Instruments: cycle time, wait time, throughput per unit time.

## 20.4 Canonical Token Glossary

| Token | Name | Description |
|---|---|---|
| XP | Experience Points | Per-action entropy-reduction score. Non-transferable, decays. |
| CT | Contribution Token | Identity-bearing contribution capacity. Transferable with lockup and friction. |
| CAT | Capability Attestation Token | Portable skill credential. No decay. |
| IT | Influence Token | Governance weight. Non-transferable, decays 5%/month. |
| DT | Domain Token | Subject-matter expertise marker. Domain-specific. |
| EP | Emergence Points | Local merchant loyalty. EP = XP * L. |

GT and RT are not canonical. They appeared in older drafts and in transitional code. The canonical six-token set is fixed.

## 20.5 Loop Lifecycle States

| State | Meaning |
|---|---|
| OPEN | The loop is created and validation routing is in progress. |
| VALIDATING | Validators are recording independent judgments. |
| CONSENSUS | The aggregation has produced V_c above threshold. |
| CLOSED | XP has been computed and minted as provisional. The retroactive-burn window is open. |
| SETTLED | The retroactive window has elapsed; provisional XP is confirmed. |
| FAILED | The retroactive window surfaced falsifying evidence; XP is burned. |
| ISOLATED | The loop has been quarantined for integrity review. |

## 20.6 Acronyms and Initialisms

| Acronym | Expansion |
|---|---|
| DAG | Directed Acyclic Graph |
| DFAO | Decentralized Fractal Autonomous Organization |
| DID | Decentralized Identifier |
| KYC | Know Your Customer (identity verification) |
| OAuth | Open Authorization |
| PSLL | Personal Signed Local Log |
| RF | Randall's Feedback (the formal governance framework) |
| SIIR | Sense-Infer-Integrate-Respond (the RF loop primitive) |

| Acronym | Expansion |
|---|---|
| VC | Verifiable Credential |
| ZKP | Zero-Knowledge Proof |
| FEP | Free Energy Principle |
| BBS+ | A signature scheme supporting selective disclosure |
| zk-SNARK | Zero-Knowledge Succinct Non-interactive Argument of Knowledge |
| HMAC | Hash-based Message Authentication Code |
| MTTR | Mean Time To Recovery |
| MTTD | Mean Time To Detection |
| CTF-PP | Convergent Truth-Finding Peer Prediction (in RF V4 Theorem V4-5) |

## 20.7 Service and Port Reference

| Service | Port | Status |
|---|---|---|
| epistemology-engine | 4001 | Active (redefined as mesh observability) |
| signalflow | 4002 | Active |
| loop-ledger | 4003 | Active |
| reputation | 4004 | Active |
| xp-mint | 4005 | Active |
| dag-substrate | 4008 | Active |
| dfao-registry | 4009 | Active |
| governance | 4010 | Active |
| temporal | 4011 | Active |
| token-economy | 4012 | Active |
| credentials | 4013 | Active |
| ecosystem | 4014 | Active |
| homeflow | 4015 | Active |
| grantflow-discovery | 4020 | Active |
| grantflow-proposer | 4021 | Active |
| identity | 4101 | Active (in-memory) |
| node-handshake | 4200 | Implemented (untested at full deployment) |

## 20.8 Governance Default Reference

| Parameter | Default |
|---|---|
| ZKP scheme | BBS+ |
| Identity reveal threshold | 7-of-12 + cause-shown |
| Retroactive validation window | 30 days |
| IT decay | 5%/month |
| XP decay | ~1%/month |
| CT lockup | 14 days |
| CT transfer friction | 2%/transfer |
| Reward escalation (early) | linear 1.0x to 3.0x over 7 days |

| Parameter | Default |
|---|---|
| Reward escalation (late) | log to cap 10.0x |
| Validator weight factors | (0.35, 0.30, 0.15, 0.20) for (domain, reputation, load, accuracy) |
| Validator pool size (routed) | 5 to 12 |
| Validator pool size (volunteer micro) | 10 |
| Blind slice fraction | 1/10 |
| V_c threshold | 0.66 |
| PSLL anchor cadence | 1 per loop close |
| Task grain | 2 to 5 minutes |
| Domain weight default | 1.0 per domain |
| Essentiality factor E default | 0.8 |
| Conviction half-life tau | 7 days |
| T_floor (settlement time minimum) | 0.01 |

### 20.9 The Three Architectural Invariants

The Codex names three architectural invariants that the protocol does not compromise:

1. **Reputation does not enter XP.** R is rarity (loop-class). rho is reputation density (lives in CT and routing).
2. **The user is sovereign.** Personal AI runs at the edge. The PSLL stays on the user's device. The network sees only what the user discloses.
3. **The DAG is event-sourced and append-only.** History is permanent. Reinterpretation appends; it does not mutate.

These are not preferences. They are the constitutional commitments. Every other parameter, every other instrument, every other weight is governance-tunable. These three are not.

### 20.10 The Closing Note

The Extropy Engine is a protocol for measuring validated entropy reduction. It is not a finished economy. It is not a central truth authority. It is not a claim that all human value reduces to a single naive number. It is a protocol that pays people to be the kind of people they already wanted to be, that measures contribution against falsifiable instruments, and that submits itself to peer review and correction.

The Codex is the consolidated reading. The book is the narrative case. The repository is the implementation. The papers are the formal foundations.

The model is allowed to lose. That is the entire epistemic posture. Build, measure, validate, correct. Lose carefully, and learn from losing. That is the protocol.

*End of Extropy Codex: Comprehensive Technical Specification*

## Appendix Material and Extended Treatments

The remainder of this Codex provides extended treatments of selected topics that support the main specification. These are organized as appendix-style additions and are intended for readers who want more depth on specific aspects without departing from the Codex's single-document form. Section numbering continues for cross-reference consistency.

## A. Extended Treatment of the XP Formula Terms

### *A.1 Why Five Terms and Not Four or Six*

The XP formula has exactly five terms, and the choice is not aesthetic. Each term answers a distinct question that the protocol requires for value to mint:

- R asks: was this contribution class scarce in the network?
- F asks: is this submitter repeating the same action class, or doing something fresh?
- Delta S asks: did disorder actually decrease, by how much, against which baseline?
- w dot E asks: how well do the contribution's measured effects match what this community values?
- log(1/T_s) asks: did the loop close in a reasonable timeframe relative to the domain's typical cadence?

Removing any term opens a gaming surface. Removing R (rarity) would mean trivial contributions earn the same as scarce ones, encouraging volume over substance. Removing F (frequency-of-decay) would mean repeating the same action class indefinitely yields full XP, encouraging grinding over genuine contribution. Removing Delta S would mean unfalsifiable claims earn XP, which is just a reputation system. Removing w dot E would mean single-domain trivia outweighs multi-domain substance. Removing the time term would mean abandoned or perpetually-stalled loops accrue value as readily as completed ones.

Adding a sixth term would have to answer a sixth distinct question that the existing five do not capture. No such question has been identified that does not collapse into one of the existing five. A reputation-flavored sixth term has been proposed and rejected: it would violate the architectural invariant.

### *A.2 The Multiplicative Structure*

The five terms are multiplied, not summed. The choice between multiplication and addition is a substantive design question.

Addition treats the terms as independent sources of value that can compensate for each other. A submitter with high rarity but zero entropy delta could still earn XP. This is incoherent: the system would mint value against claims that demonstrably reduced no disorder. Addition fails the conjunctive test that the protocol requires.

Multiplication enforces necessity: every term must be non-zero for XP to mint. A submitter cannot compensate for missing evidence with high rarity. A trivial submission cannot reach high XP through speed alone. The multiplicative structure embodies the conjunctive logic of validated entropy reduction: rarity AND frequency AND magnitude AND domain match AND timely settlement.

There is one consequence of the multiplicative structure that warrants attention: when any term is small, the XP value is small, regardless of the other terms. A high-magnitude entropy reduction in a low-rarity action class still mints low XP because R is small. This is intended. The protocol rewards the combination of substance and scarcity, not substance alone.

### *A.3 R Calibration in Practice*

The rarity table is the most contested governance object in the protocol. Setting R values is where the network's judgment about what counts as scarce gets encoded.

The protocol provides default rarity values for canonical action classes per domain. These defaults are deliberately conservative: most common action classes are set in the 0.5 to 1.5 range, with values above 3.0

reserved for genuinely scarce contributions (a working novel governance protocol, a domain-foundational instrument calibration).

DFAOs can adjust R values for their local context, within bounds set by the parent DFAO (and ultimately within bounds set by the PLANETARY-scale defaults). A small-town DFAO might set repair-shop-operation at a higher R than a megacity DFAO would, reflecting genuine local scarcity.

The bounds prevent runaway R inflation: a child DFAO cannot set R = 10 for a routine action class, because that exceeds the parent's safety bound. The bounds are enforced through governance proposals: any attempted override outside the safety range triggers a parent-DFAO governance review.

Governance proposals to revise the rarity table are visible to all DFAO members. The deliberation period and voting method are determined by the DFAO's governance configuration. Approved revisions apply to future loops only; existing loops continue under the prior R value (preserving DAG immutability).

### *A.4 F Decay Curves in Detail*

The two supported F decay curves have different practical behaviors:

**Log-tail decay:** F(n) = 1 / (1 + log(n+1)), where n is the count of prior occurrences of the fingerprint. Behavior:

- F(0) = 1.0 (first occurrence)
- F(1) ~= 0.59
- F(10) ~= 0.30
- F(100) ~= 0.18
- F(1000) ~= 0.13
- Long tail: F decreases but never reaches zero.

**Harmonic decay:** F(n) = 1 / (n+1). Behavior:

- F(0) = 1.0 (first occurrence)
- F(1) = 0.5
- F(10) ~= 0.09
- F(100) ~= 0.01
- F(1000) ~= 0.001
- Sharp decay: F approaches zero quickly.

Log-tail decay is appropriate for action classes where repeated contribution still has marginal value (a sustained habit, a recurring repair). Harmonic decay is appropriate for action classes where novelty is essential (a new architectural design, a new instrument).

The choice is per-DFAO and per-claim-type. The DFAO's rarity table includes the recommended decay curve for each canonical action class.

### *A.5 The Settlement-Time Term in Practice*

The settlement-time term log(1/T_s) creates an incentive for fast, honest settlement. A claim that closes in 0.1 of its target window yields log(10) ~= 2.3 from the time term. A claim that closes exactly at its target window yields log(1) = 0 from the time term.

The T_floor (default 0.01) caps the maximum value of the term at log(100) ~= 4.6. This prevents arbitrarily fast settlement from producing arbitrarily large XP.

The term's behavior at T_s = 1.0 is log(1) = 0, which means the formula multiplies by zero, producing zero XP. This is intentional: a claim that takes its full target window (or longer) yields no time bonus, which is appropriate. The minimum XP value (when log(1/T_s) is near zero but positive) is set by the product of the other four terms.

The term's behavior near T_floor is the cap: any T_s value below T_floor is clamped to T_floor before the log is computed. This handles edge cases where a claim is reported as having settled in negligible time (sub-second), which usually indicates either a measurement error or an attempted attack.

## B. The Loop Lifecycle: A Worked Example

### *B.1 Setup*

Consider a worked example: a contributor who runs a community repair workshop wants to mint XP for repairing a broken bicycle in their MICRO DFAO (a 4-person workshop collective).

The repair has thermodynamic content (the bicycle returns to low-entropy operational state), code-adjacent content (the contributor documented the repair process), economic content (the repair extends the bicycle's useful life, reducing replacement demand), and informational content (the documentation contributes to the workshop's knowledge base).

### *B.2 OPEN*

The contributor opens a loop with the following parameters:

- **Submitter:** the contributor's DID.
- **Claim:** "Repaired bicycle BR-2024-117; restored functional state from non-functional state; documented repair process for the workshop knowledge base."
- **Domain:** primary = thermodynamic; evidence vector E = (0.1 cognitive, 0.0 code, 0.0 social, 0.3 economic, 0.5 thermodynamic, 0.1 informational, 0.0 governance, 0.0 temporal).
- **Instrument:** workshop's repair instrument (a structured photographic and tested-functionality record, with the bicycle's working-state validated by a second workshop member).
- **Baseline:** the bicycle's non-functional state at intake (documented photographically).
- **Falsification condition:** if the second workshop member inspects the bicycle within 14 days and finds it does not perform its function under specified test conditions.
- **Target settlement time:** 24 hours from OPEN to CLOSED.

The DAG vertex LOOP_OPEN is written with the above parameters and signed by the contributor's identity key.

### *B.3 VALIDATING*

SignalFlow routes the validation request. For a workshop-level repair claim, the routing produces a pool of two workshop validators (the workshop is small). The validators receive the photographic record, the repair documentation, and the request to inspect the repaired bicycle.

Validator 1 (another workshop member) inspects the bicycle, tests it under the workshop's standard tests, confirms functionality, records `confirmed`. The DAG vertex VALIDATION_RECORDED is written.

Validator 2 (a workshop guest validator visiting from another DFAO) reviews the documentation, asks one clarifying question via the validation interface, then records `supported`. The DAG vertex VALIDATION_RECORDED is written.

The epistemology engine aggregates the two judgments via the Beta-conjugate Bayesian update. Both confirmed and supported count as affirmative. The posterior mean exceeds the V_c threshold of 0.66.

### *B.4 CONSENSUS*

The loop transitions to CONSENSUS. The DAG vertex LOOP_CONSENSUS is written.

### *B.5 CLOSED*

The XP formula computes:

- R = 1.2 (per the workshop DFAO's rarity table for the action class simple_repair_documented)
- F = 0.95 (the contributor has documented one prior repair in this fingerprint context; F has decayed slightly)
- Delta S = 0.7 (normalized; the thermodynamic content is well-bounded, the economic content adds marginal Delta S, the cognitive and informational contents add marginal Delta S)
- w dot E: the workshop DFAO's weight vector is (1.0, 1.0, 1.2, 1.0, 1.5, 1.1, 1.0, 1.0). Computing w dot E = 0.1*1.0* + *0.0*1.0 + 0.0*1.2* + *0.3*1.0 + 0.5*1.5* + *0.1*1.1 + 0.0*1.0* + *0.0*1.0 = 0.1 + 0.0 + 0.0 + 0.3 + 0.75 + 0.11 = 1.26
- T_s = 0.45 (the loop closed in roughly half its target window)
- log(1/T_s) = log(1/0.45) = log(2.22) ~= 0.80

XP = 1.2 * 0.95 * 0.7 * 1.26 * 0.80 = 0.804

XP is minted as provisional. The DAG vertex XP_MINT_PROVISIONAL is written. The contributor's character sheet shows 0.804 XP added (provisional). The 30-day retroactive-burn window opens.

### *B.6 SETTLED*

For 30 days, no falsifying evidence is submitted. The bicycle continues to function. The repair documentation is reviewed by two other workshop members during the window and they record supported post-hoc.

At the end of 30 days, the loop transitions to SETTLED. The provisional XP is upgraded to confirmed. The DAG vertex XP_MINT_CONFIRMED is written. The contributor's character sheet now shows 0.804 confirmed XP.

The reputation service updates the contributor's rho in the thermodynamic and economic domains (the primary and significant evidence domains for this claim). The DAG vertex REPUTATION_UPDATE is written.

### *B.7 The Conversion to EP*

The contributor wants to use their accumulated XP for a discount at a participating local cafe. The cafe is a merchant in the workshop's local merchant network. The cafe's loyalty multiplier L is 1.5.

The contributor initiates an EP conversion:

- EP = XP * L = 0.804 * 1.5 = 1.206

The 1.206 EP is now spendable at the cafe. The contributor's XP balance decreases by 0.804 (the converted amount). The DAG vertex EP_CONVERT is written.

The cafe sees the conversion through its merchant interface: a customer is presenting 1.206 EP for a discount. The cafe applies the discount per its loyalty terms and serves the customer.

### *B.8 What the Network Saw*

Throughout this entire example, the network saw:

- The contributor's DID (public).
- The LOOP_OPEN vertex with the claim metadata (no personal information beyond the DID).
- The two VALIDATION_RECORDED vertices.
- The LOOP_CLOSE, XP_MINT_PROVISIONAL, XP_MINT_CONFIRMED, and EP_CONVERT vertices.
- The contributor's updated XP and EP balances (the public part of their character sheet).
- The contributor's updated rho values in the relevant domains.

The network did not see:

- The contributor's name, address, or any personal identifier beyond the DID.
- The repair documentation in full (it stays on the contributor's PSLL; only the Merkle root anchored to the DAG is seen).
- The contributor's other activities in other DFAOs (cross-DFAO context is separated by nullifier).
- The validators' personal identifiers (only their DIDs and their validator-specific reputation).

This is Digital Autarky in action. The network gets exactly what coordination requires, and nothing more.

## C. The DAG Substrate in Practical Detail

### *C.1 Vertex Schema*

A DAG vertex has the following canonical fields:

- vertex_id: a unique identifier (typically the content hash).
- vertex_type: one of the canonical types (LOOP_OPEN, VALIDATION_RECORDED, XP_MINT_PROVISIONAL, etc.).
- submitter_did: the DID of the submitter.
- timestamp: the vertex creation time.
- payload: the vertex content (typed by vertex_type).
- content_hash: SHA-256 hash of the canonical serialization of the payload (or BLAKE3, governance-tunable).
- signature: Ed25519 signature over the content hash by the submitter's identity key.
- parent_vertices: the set of vertices this vertex causally depends on (the edges in the DAG).
- confirmation_weight: a derived field expressing the accumulated weight of subsequent vertices that confirm this vertex.

### *C.2 Edge Semantics*

An edge in the DAG expresses a causal dependency. The canonical edge types include:

- validates: a VALIDATION_RECORDED vertex depends on a LOOP_OPEN vertex.
- aggregates: a LOOP_CONSENSUS vertex depends on multiple VALIDATION_RECORDED vertices.
- mints: an XP_MINT_PROVISIONAL vertex depends on a LOOP_CLOSE vertex.
- confirms: an XP_MINT_CONFIRMED vertex depends on an XP_MINT_PROVISIONAL vertex.

- burns: an XP_BURN vertex depends on an XP_MINT_PROVISIONAL vertex and on falsifying-evidence vertices.
- converges: a CONVERGENCE vertex depends on multiple participating person-DAG vertices.
- anchors: a PSLL_ANCHOR vertex depends on the user's prior PSLL anchor and on the new Merkle root.
- governance: a GOVERNANCE_VOTE vertex depends on a GOVERNANCE_PROPOSAL vertex.

The edge types are not exhaustive; new edge types are added as new event types are introduced.

### *C.3 Tip Selection Algorithm*

The tip selection algorithm for new vertex attachment uses a weighted random walk:

1. Start at the genesis vertex.
2. At each step, examine the children of the current vertex (vertices that depend on the current vertex).
3. Filter children to those that are still tips (have no children of their own) or that have unsaturated tip-selection eligibility.
4. Weight the children by a score combining: validator support, time recency, DFAO locality, and content type compatibility.
5. Choose a child probabilistically by the weights.
6. Step to the chosen child.
7. Repeat until reaching a tip.

The walk terminates at a tip. A new vertex attaches to two tips (the IOTA Tangle convention). The dual-attachment creates the property that every new vertex confirms two prior tips, propagating confirmation weight up the DAG.

### *C.4 Confirmation Propagation*

When a new vertex is added with parent edges to two prior tips, those tips' confirmation weights increase. The confirmation flow propagates recursively up to a configurable depth (default 20). Each ancestor's confirmation weight increases by a diminishing fraction proportional to the distance.

The confirmation weight is not a binary "confirmed / unconfirmed" but a continuous value. A vertex with low confirmation weight is provisional; a vertex with high confirmation weight has been transitively endorsed by many subsequent contributions.

The continuous confirmation model is what makes the IOTA-inspired structure work without proof-of-work or proof-of-stake: confirmation emerges from the network's contribution flow, not from external resource expenditure.

### *C.5 PostgreSQL Backing*

The DAG substrate uses PostgreSQL 16 for persistence. The schema includes:

- dag.vertices: the vertex table, indexed by vertex_id and by submitter_did.
- dag.edges: the edge table, indexed by source and target vertex_id.
- dag.confirmations: the confirmation-weight table, indexed by vertex_id.

Queries common to the protocol:

- "Get the causal ancestry of this vertex up to depth N." (recursive CTE)
- "Get all VALIDATION_RECORDED vertices that depend on this LOOP_OPEN." (simple join)
- "Get the current tips for tip selection." (children-count query)
- "Get the confirmation weight of this vertex." (single row lookup)
- "Replay the DAG from genesis to this vertex." (BFS or DFS traversal)

The single-file implementation (packages/dag-substrate/src/index.ts, ~1573 lines) covers the production-equivalent query surface but uses single-node PostgreSQL. Production deployment requires libp2p gossip and multi-node consensus, which is on the v3.2 roadmap.

## D. Identity Layer: Concrete Flow

### *D.1 New User Onboarding*

A new user goes through the following flow:

1. The user visits the application (HomeFlow, the character-sheet UI, or another vertical) and clicks "Sign In."
2. The application redirects to the OAuth provider (Google by default; GitHub also supported). The user authenticates with the provider. The application receives an OAuth token. The onboarding state transitions to oauth-pending -> oauth-verified.
3. The application prompts the user to complete KYC on-device. The user selects a method (ID scan, biometric bind, or trusted-issuer handoff). The KYC payload is processed locally; the result is a verified-identity attestation. The onboarding state transitions to kyc-attested.
4. The application generates an Ed25519 key pair. The public key becomes the user's DID (did:extropy:<64-hex>). The private key is stored in the user's device secure storage (Keychain on macOS, Keystore on Android, etc.). The user's DID is registered with the network. The onboarding state transitions to did-issued.
5. The application creates the user's initial PSLL (the user's local event log) and generates the genesis entry. The genesis entry is hash-chained to itself; subsequent entries chain to the genesis. The PSLL's first Merkle root is anchored to the DAG. The onboarding state transitions to closed.

The entire flow takes 2 to 5 minutes for an experienced user. The flow can be interrupted and resumed; the state machine preserves progress.

### *D.2 Selective Disclosure*

When a user presents themselves to a validator pool for a specific claim, the disclosure is selective. The user can prove statements like:

- "I am a verified member of DFAO X." (without revealing other DFAO memberships)
- "I hold a CAT for thermodynamic-instrument-operation." (without revealing other CATs)
- "I have validated more than 50 loops in the cognitive domain." (without revealing specific loops)
- "My XP balance in this DFAO exceeds threshold T." (without revealing the exact balance)

The proofs are produced using BBS+ over the user's credential bundle. The verifier sees only the proven statement, not the underlying credentials. The verifier sees that the proof verifies, which is sufficient for the protocol's routing or admission decision.

### *D.3 Nullifier Use*

When the user takes a one-time action (a vote, a single claim per loop, a reveal-consent), the protocol requires a nullifier. The nullifier construction is deterministic for the user and the context:

nullifier = b64url(HMAC_SHA256(holderSecret, holderDid || '|' || contextTag))

The user's holderSecret is stored locally and is not derivable from the public DID. The contextTag is provided by the protocol for each one-time-action context (e.g., vote:proposal-12345, claim:loop-67890).

The protocol sees the nullifier in the action submission and checks whether the nullifier has been seen before in the same context. If yes, the action is rejected (HTTP 409). If no, the action is accepted and the nullifier is recorded for that context.

The same user computing the nullifier for the same context produces the same value (preventing double-action). Different users (different holderSecrets) produce different values (preventing impersonation). Different contexts (different contextTags) produce uncorrelated nullifiers (preventing cross-context tracking).

The current implementation uses HMAC-SHA256 as a sandbox stand-in. Production swap to Pedersen-based commitments goes through the same ZKP envelope and changes no application code.

### *D.4 Key Rotation*

If the user's private key is compromised or if the user wants to rotate keys for hygiene, the protocol supports a documented rotation flow:

1. The user generates a new Ed25519 key pair locally.
2. The user creates a rotation event (a DAG vertex KEY_ROTATION) signed by both the old key and the new key.
3. The rotation event is anchored to the DAG.
4. Subsequent PSLL entries and DAG submissions use the new key.
5. The old DID is marked as rotated (not deleted; the historical record remains).
6. The user's accumulated XP, CT, CAT, IT, DT, EP balances transfer to the new DID through the rotation event's signature chain.

The rotation is observable: anyone querying the user's DID history can see the rotation event and verify the dual-signature. The rotation does not break the causal history of the user's prior contributions; it just changes which key signs new contributions going forward.

## E. Worked Example: A DFAO Governance Proposal

### *E.1 The Proposal*

A MESO DFAO (a 40-person community focused on neighborhood-scale entropy reduction) decides that its current V_c threshold (0.66) is too low for thermodynamic-domain claims, which it considers high-stakes. A member opens a governance proposal:

- **Proposal:** "Raise the V_c threshold for thermodynamic-domain claims from 0.66 to 0.75 within this DFAO."
- **Rationale:** "Three thermodynamic claims in the past quarter were initially confirmed and subsequently falsified during the retroactive window. The 0.66 threshold appears to be admitting low-confidence consensus. A higher threshold should reduce the false-positive rate."

- **Effective date:** 14 days from proposal acceptance.
- **Voting method:** conviction voting with 7-day half-life.

The DAG vertex GOVERNANCE_PROPOSAL is written.

### *E.2 Deliberation*

For 14 days, members of the DFAO consider the proposal. Discussion happens through the DFAO's communication channels (which are not part of the protocol but feed back into the protocol through the proposal's discussion record).

Several members raise concerns: raising the threshold may slow legitimate claims; the three falsified claims may have shared a specific instrument flaw rather than reflecting threshold inadequacy. The proposal sponsor responds with additional analysis: the three falsified claims used three different instruments; the pattern suggests threshold-level rather than instrument-level inadequacy.

### *E.3 Voting*

Members cast conviction votes. Each vote consists of: a position (for or against), a stake of IT, and the timestamp at which the position was taken. The conviction weight grows over time as the position is maintained.

After 14 days, the aggregate conviction weight in favor exceeds the proposal's pass threshold. The proposal passes.

### *E.4 Execution*

On the effective date, the DFAO's V_c threshold for thermodynamic-domain claims is updated to 0.75. The update is recorded as a GOVERNANCE_PARAMETER_UPDATE vertex linked causally to the GOVERNANCE_PROPOSAL vertex.

Existing loops in the thermodynamic domain that were opened before the effective date continue under the prior threshold. Loops opened on or after the effective date use the new threshold.

### *E.5 Subsequent Observations*

Over the following quarter, the DFAO's thermodynamic claims show a lower false-positive rate (claims initially confirmed and subsequently falsified drop from 12% to 4%). The DFAO's thermodynamic claim throughput also drops slightly (legitimate claims that were borderline now fall short of the higher threshold). The trade-off appears to be acceptable to the membership; no proposal to revert the change is opened.

### *E.6 What the Example Shows*

The example illustrates several aspects of the protocol in operation:

- Governance is parameter-based, not procedural. Specific numbers can be adjusted; the underlying protocol does not change.
- Governance proposals are visible and discussable.
- Conviction voting rewards patient positions and penalizes impulsive shifts.
- Updates apply forward-only, preserving DAG immutability.
- Observed outcomes inform future governance.

## F. Multi-Party Convergence

### *F.1 The Setup*

Three contributors collaborate on a substantial entropy reduction: they design, build, and commission a community-scale composting system for their neighborhood. The project takes 4 months and reduces the neighborhood's waste flow significantly.

### *F.2 The Convergence*

At project completion, the three contributors jointly open a loop:

- **Submitters:** the three contributors' DIDs.
- **Claim:** "Designed and commissioned community composting system; reduced neighborhood waste flow by 60% measured against baseline; system in continuous operation for 30 days with maintenance plan documented."
- **Domain:** primary = thermodynamic; significant in informational, economic, social, governance.
- **Evidence vector E:** (0.05 cognitive, 0.0 code, 0.20 social, 0.15 economic, 0.40 thermodynamic, 0.10 informational, 0.10 governance, 0.0 temporal).
- **Instrument:** community-validated waste-flow measurement plus inspection by a third-party auditor.
- **Baseline:** the neighborhood's waste-flow records for the prior year.
- **Falsification condition:** if the system fails to maintain at least 50% waste-flow reduction over the following 90 days.
- **Convergence-split rules:** each contributor's share of the resulting XP is documented at OPEN; in this case, 35% / 35% / 30% reflecting differential effort contribution.

The DAG vertex LOOP_OPEN is created. It is a convergence vertex with three incoming person-DAG edges.

### *F.3 Validation and Settlement*

The validation routing identifies a validator pool spanning the relevant domains. The third-party auditor's measurement is incorporated into the validation. Consensus reaches $V_c = 0.81$. The loop transitions through CONSENSUS and CLOSED.

XP is computed via the formula. Suppose the result is XP = 12.5 for the joint loop. The convergence-split distributes:

- Contributor A: 35% of 12.5 = 4.375 XP
- Contributor B: 35% of 12.5 = 4.375 XP
- Contributor C: 30% of 12.5 = 3.75 XP

Each contributor's individual XP balance increases by their share. The DAG records the split in the XP_MINT_PROVISIONAL vertex.

The 30-day retroactive window opens. After 30 days with no falsifying evidence, the loop transitions to SETTLED and the provisional XP is confirmed.

### *F.4 The Convergence Vertex as Identity Substrate*

The convergence vertex is now part of each contributor's person-DAG. Future queries that examine any of the three contributors' histories will surface this convergence. Reputation accrual flows to all three contributors in the relevant domains (thermodynamic, social, economic, governance).

If the system fails over the longer term (well beyond the 30-day retroactive window), governance can review whether to issue a corrective vertex against the convergence (a partial retroactive correction). The DAG preserves the lineage of any corrections.

### *F.5 Why Convergence Matters*

Multi-party collaboration is the dominant mode of real-world entropy reduction. The convergence vertex is the protocol's first-class representation of this fact. The protocol does not force collaborators to submit separate claims and somehow split the work artificially. The protocol allows joint claims with documented splits, validated as a single unit.

The convergence-split rules are governance-tunable. DFAOs can establish defaults for common multi-party patterns (equal split for symmetric collaborations, contribution-weighted splits for asymmetric ones, DFAO-defined split functions for complex cases).

## G. Family Pilot: HomeFlow

### *G.1 What HomeFlow Is*

HomeFlow is the application-layer vertical that brings the Extropy Engine to the household scale. It is the protocol's first deployment vertical and the most mature application example.

A household enrolls as a MICRO DFAO (2 to 7 members). Each member onboards through the identity flow and creates their character sheet. The household configures its governance (typically simple-majority voting on most decisions; conviction voting on substantial changes). The household sets its weight vector w over the eight domains (a household might weight social and temporal heavily).

### *G.2 What HomeFlow Tracks*

The household tracks contributions in several routine categories:

- **Chores:** routine household maintenance (cleaning, cooking, laundry, errands). Mostly thermodynamic and temporal.
- **Repairs:** fixing broken household items. Thermodynamic and economic.
- **Childcare and care:** social and temporal.
- **Cooking and food planning:** thermodynamic, informational (recipe documentation), economic (food budgeting).
- **Shopping list and pantry management:** informational and economic.
- **Family meetings and household governance:** governance and social.

Each contribution is opened as a loop in the household DFAO. The household's validator pool is the household itself: one member opens the loop, another validates. The validation is informal (in conversation, often) but is recorded in the protocol as a VALIDATION_RECORDED event.

### *G.3 The XP Flow*

A child takes out the trash. The child opens a loop: claim "took out trash," domain = thermodynamic / temporal, baseline = trash bag full, falsification condition = trash not actually removed within 30 minutes. A parent confirms the claim. XP mints. The child's character sheet updates.

Over time, the child accumulates XP in their character sheet. The household has configured an EP conversion that converts XP to small treats or screen time. The child sees a tangible reward for habitual contribution.

The household's collective XP feeds into household-level governance: who has accrued the most contribution in the past month informs the household's decisions about specific allocations (which family member gets to choose the weekend activity, who gets the new device first, etc.).

### *G.4 Why HomeFlow Is the Pilot*

HomeFlow is the pilot vertical because:

- The household is small enough that the protocol's complexity does not become overwhelming.
- The validators are mutually trusted (family members), simplifying the validator-collusion threat model.
- The contribution categories are routine and well-understood, making the rarity table simple.
- The merchant side is intra-household (parents acting as merchants for children, or vice versa).
- The protocol's failure modes (a stale R passage, a confused F decay, a missing V_c threshold) are observable but not catastrophic.

The HomeFlow pilot is the protocol's empirical bed. Lessons from HomeFlow inform protocol revisions at the protocol level.

### *G.5 What HomeFlow Has Shown*

The HomeFlow pilot (running on consumer hardware, with file-backed default storage) has demonstrated:

- The onboarding flow works for non-technical users (family members who have no interest in cryptography can complete the flow).
- The character-sheet UI is intuitive (XP accrual is visible and motivating).
- The validation flow can be informal in a high-trust DFAO (informal conversations recorded as VALIDATION events).
- The XP-to-EP conversion works for intra-household trade (treats, privileges, allocations).
- The household-level governance can adapt over time (parents and children renegotiate weight vectors as family priorities evolve).

The pilot has also surfaced limits:

- The protocol's terminology can be opaque ("validation confidence," "frequency-of-decay") and benefits from translation into household-language ("did the parent agree this happened," "how often the child does this chore").
- The DAG's permanence creates some friction for trivial loops (a 2-minute chore creates a permanent DAG record).
- The retroactive-burn window is too long for household scale (30 days is appropriate for high-stakes claims but feels excessive for routine chores).

The lessons feed back into the protocol's governance defaults and into the UI design.

## H. The Book's Argument in Five Pages

This section summarizes the companion book's argument in compressed form, for readers who want the book's case without reading 400 pages.

### *H.1 The Diagnosis*

Existing systems for measuring and rewarding contribution are broken in predictable ways. Test scores stand in for learning and get gamed. Credit scores stand in for trustworthiness and become tools of social control. Likes and followers stand in for cultural contribution and become advertising surfaces. Market prices stand in for value and concentrate wealth in those who can manipulate prices. Compliance metrics stand in for safety and slowly drift from the safety they were built to track.

The pattern is mechanical, not moral. When a representation acquires incentive weight, rational actors optimize for the representation, not for the underlying condition. The label becomes the target. The fidelity decays. Goodhart's Law applies generally.

The book's argument is that this pattern is not an accidental feature of badly-designed systems. It is the default behavior of any system that puts incentive weight on a representation. Fixing one specific case (improving a test, reforming a credit score) does not change the dynamic; the new metric becomes the new target.

### *H.2 The Thesis*

The book proposes that value should be operationalized as validated entropy reduction. Entropy is used in a broad but disciplined sense: physical disorder, informational uncertainty, coordination dysfunction, the loss of structure to noise. Reduction is measurable per domain, validated by a routed pool of witnesses, anchored in a causally ordered event graph, and audited continuously.

The thesis is not that "all value is entropy reduction" in some grand metaphysical sense. It is more limited: that the existing categories of value (utility, virtue, capital, status) all admit operationalization through some measurable disorder reduction in some domain, and that the operationalization can be insulated from reputation laundering and Goodhart drift through specific architectural choices.

The protocol the book proposes is the Extropy Engine. The book walks the reader through the engine's components: the XP formula, the eight canonical domains, the loop lifecycle, the contribution graph, the validation model, the DAG substrate, the DFAO governance layer, the token economy.

### *H.3 The Engine*

The book's middle chapters (chapters 8 to 14) walk through the engine in detail. These chapters are the narrative form of what the Codex now specifies precisely:

- Chapter 8: the XP formula and what its terms mean.
- Chapter 9: the eight canonical domains.
- Chapter 10: the core loop.
- Chapter 11: the DAG.
- Chapter 12: the DFAOs.
- Chapter 13: the tokens.
- Chapter 14: the temporal mechanics.

The book's voice is conversational. The Codex's voice is precise. They aim at the same target.

### *H.4 The Defense*

The book's chapters 15 to 18 address the predictable objections:

- "Won't this be gamed?" The Goodhart resistance arguments. The architectural invariant. The retroactive-burn mechanism. The Signal paper's k-parameter taxonomy applied to the engine itself.
- "Won't bad actors take over?" The Sybil resistance arguments. The validator collusion analysis. The cartel-threshold result.
- "Won't this be co-opted by the state?" The Digital Autarky principle. The threshold reveal scheme. The user-sovereign architecture.
- "Won't this fail psychologically?" The intrinsic-motivation analysis. The reason people contribute when there is no extrinsic reward. The reason the engine works with that, not against it.

These chapters are the book's defense against the standard objections. The Codex's section 17 (Security, Attack Surfaces, and Failure Modes) is the technical equivalent.

### *H.5 The Call*

The book's final chapters (23 and 24) call for the protocol's deployment. The argument is that the existing systems will continue to fail in their predictable ways, and that the cost of building the alternative is low enough that it should be built. The book is explicit about the credentials it does not have, the funding it does not need, the permission it does not seek. It is the position of a solo developer with a working architecture and the stubbornness to assemble it without institutional support.

The Codex is the technical artifact of that assembly. The book is the case for why the assembly matters.

## I. Closing Reflection

### *I.1 The Protocol's Posture*

The Extropy Engine is not the only attempt to build a non-extractive coordination layer. It is one attempt. It has specific architectural commitments (Digital Autarky, the architectural invariant on reputation, the contribution graph, the eight canonical domains, the DAG substrate). Those commitments are testable and falsifiable. The protocol can be wrong.

The Codex is the protocol's offer to be read carefully, to be examined critically, and to be revised under pressure. It is not a marketing document. It is the technical specification that the system attempts to honor.

### *I.2 What Would Falsify the Project*

The Codex has stated the falsification conditions in section 19.9. To restate them briefly:

- If the architectural invariant (reputation does not enter XP) cannot be maintained in deployed practice.
- If the eight canonical domain instruments all turn out to be equally Goodhart-vulnerable.
- If cross-domain normalization cannot produce stable calibrations.
- If validator collusion can be sustained below the predicted detection threshold.
- If users prove able to farm metrics more efficiently than they can produce real entropy reduction, with the gaming cost less than the protocol's defenses anticipate.

Any of these would constitute project-level failure. The Codex documents them explicitly so that the protocol's success or failure can be assessed against pre-specified criteria, not retrospectively rationalized.

### *I.3 What Success Looks Like*

Project-level success would be a sustained deployment of the protocol in which:

- The HomeFlow pilot expands to additional households without protocol-level failure.
- A merchant pilot in a town demonstrates the two-sided market dynamics described in section 13.
- A small number of MESO and MACRO DFAOs adopt the protocol for their internal coordination.
- The peer review and academic community engage with the formal foundations (Randall's Feedback V4, the Signal paper) and produce additional theorems, falsifications, or refinements.
- The repository implementation matures toward the v3.2 and v4.0 targets.
- The architectural invariant is maintained under pressure.

None of this is guaranteed. The protocol is allowed to lose.

### *I.4 The Author's Note*

The Extropy Engine is the work of a solo developer, working on consumer hardware, without institutional backing or formal credentials. This is a feature, not a bug. The protocol does not require institutional credentials to be examined; it requires only the willingness to read the math and run the code. The Codex is the offer to be read.

The author's note in the one-pager states it plainly: "Read enough physics papers and you start to clock which physicists are bluffing. Build enough systems and you start to clock which architectures are vibes in a trench coat. That's the qualification on offer."

The Codex is the test. If it survives the reading, the architecture survives. If it does not, the architecture needs revision. Either way, the model is allowed to lose.

### *I.5 The Last Word*

The Extropy Engine pays people to be the kind of people they already wanted to be. It does so by measuring validated entropy reduction, not by measuring reputation or follower counts or compliance checklists. It does so under an architectural invariant that prevents the most common failure modes. It does so with a peer-reviewed mathematical foundation, an honestly enumerated gap catalog, and an explicit posture of falsifiability.

This Codex is the specification. The repository is the implementation. The book is the case. The protocol is the offer.

The system isn't broken. It's working exactly as designed. The Extropy Engine is the attempt to design a different one.

*End of Extropy Codex: Comprehensive Technical Specification (full)*

## J. Extended Treatment of Randall's Feedback V4

This appendix expands on the formal foundations of Randall's Feedback V4, which underpins the Extropy Engine's governance layer. The treatment here is more detailed than the main Codex section 7 but more compressed than the source paper. Readers seeking the full formal proofs should consult the source paper directly.

### *J.1 The Particular Partition*

Each RF agent is modeled as a quadruple (mu_i, s_i, a_i, eta_i) satisfying the particular partition conditions formalized by Friston and colleagues (2023). The components:

- **Internal states (mu_i):** the agent's goal representations and belief parameters. These encode the agent's generative model of the governance environment.
- **Sensory states (s_i):** incoming contribution signals and governance observations. These represent what the agent can observe about other agents' contributions and governance outcomes.
- **Active states (a_i):** outgoing contributions and governance actions. These represent the agent's decisions: what to contribute, how to vote, which contributions to validate.
- **External states (eta_i):** the governance environment, including other agents' states and the protocol rules.

The particular partition requires sparse coupling: internal states mu do not directly depend on external states eta, and sensory states s do not directly depend on internal states mu. The contribution validation protocol enforces this separation architecturally: an agent's contributions (active states) are computed from its internal goals and observed contributions (sensory states) but cannot directly access other agents' internal goal representations (external states).

This is not merely a modeling assumption. It is an architectural constraint enforced by the protocol's design. The Extropy Engine's contribution graph operationalizes this: contributors see what other contributors have publicly submitted, but they do not see other contributors' internal reasoning. The Digital Autarky principle and the personal-AI-at-the-edge architecture are the engineering manifestations of the particular partition.

### *J.2 The Governance Hamiltonian*

A central innovation of RF V4 is encoding the governance protocol as a Hamiltonian H_G. Formally, H_G(phi, o_tau) is defined as the integral functional over governance trajectories:

[ H_G(, o_) = _0^t dt' + C ]

Where phi = (u_tau, theta) with u_tau denoting the trajectory of governance decisions and theta the protocol parameters; f(x) is the solenoidal flow (conservative governance dynamics); g(x) is the action model; V(x, o) is the scalar potential coupling the governance system to its environment; and C is the integration constant encoding prior governance norms.

The key identification theorem: when the system's distribution p(phi) follows Fokker-Planck dynamics, the Hamiltonian H_G constitutes a generative model under which the system's Helmholtz energy is equivalent to variational free energy. This means the governance protocol's rules are literally the generative model that agents internalize through free energy minimization, and convergence to the governance attractor is equivalent to Hamiltonian matching: each agent's internal model converges to H_G.

In the Extropy Engine's operational form, the governance Hamiltonian is the set of provisional defaults plus the DFAO-specific overrides. An agent participating in the protocol internalizes these as their generative model: the rules of the game become the agent's expectation of how contributions translate to value.

### *J.3 Theorem V4-1: RF Governance Convergence (Expanded)*

The full statement: Agents satisfying the particular partition conditions, with internal dynamics following the variational free energy gradient, converge asymptotically to a governance attractor through Hamiltonian matching under the Free Energy Principle.

Proof sketch: The convergence proof rests on three components. First, the particular partition ensures the agent's internal states evolve according to a well-defined gradient flow. Second, the Hamiltonian formulation provides a generative model whose free energy is equivalent to the agent's variational free energy. Third, the Lipschitz constraints on the goal-update operator (Axiom 2) bound the rate at which the goal moves, ensuring the convergence is asymptotic rather than divergent.

Consequence for the Extropy Engine: in a deployed protocol with sufficient validator pool size and bounded parameter-update rates, agents' beliefs about what counts as contribution will converge over time. The convergence is not to a static fixed point but to an attractor: a region in belief-space within which agents' models are consistent with the protocol's rules.

### *J.4 Theorem V4-2: Collective Goal-Update Admissibility (Expanded)*

The full statement: Necessary and sufficient conditions on the goal-update operator L for the collective system to remain in a Banach fixed-point contraction on the product of feasible reward sets are: (1) L is Lipschitz continuous with constant kappa in (0, 1); (2) the pairwise overlap of feasible reward sets across agents is non-degenerate; (3) the rate bound T >= Omega(H^3 * S * A / epsilon^2) on goal updates is satisfied.

Proof sketch: The Banach contraction principle requires a contraction mapping on a complete metric space. The product of feasible reward sets is a convex polytope in reward-function space (Metelli and colleagues, 2023). The Lipschitz condition makes L a contraction. The pairwise overlap ensures the fixed point exists. The rate bound ensures sufficient observations are accumulated between updates to justify the update.

Consequence for the Extropy Engine: governance parameter updates cannot happen arbitrarily fast. The provisional defaults table includes implicit rate bounds (the deliberation period, the conviction half-life). These are not just process choices; they are the operational form of the rate bound that makes the theorem apply.

### *J.5 Theorem V4-3: Dynamic Goodhart Resistance (Expanded)*

The full statement: Under the four RF axioms, with bounded goal-update rates and Lipschitz metric continuity, the discrepancy process between the contribution metric m and the evolving goal G* has light-tailed marginals.

Proof sketch: The discrepancy process is the difference between the metric value and the goal value at each time step. Under the bounded goal-update rate, the goal moves slowly relative to the contributions. Under Lipschitz metric continuity, the metric does not jump unpredictably. The combination produces a discrepancy process whose tail probability decays exponentially in the discrepancy magnitude.

Consequence for the Extropy Engine: Goodhart pressure cannot accumulate unboundedly. The discrepancy between the contribution metric (XP) and the underlying goal (genuine entropy reduction) is bounded in distribution. This does not mean Goodhart pressure does not exist; it means Goodhart pressure cannot run away. The protocol's structural defenses (the retroactive-burn window, the falsifiability conditions, the governance review of failure rates) operate within the bounds the theorem establishes.

### *J.6 Theorem V4-4: Unhackability Under Constrained Policy Classes (Expanded)*

The full statement: RF's admissibility conditions on contribution strategies restrict the policy space to a finite (or effectively finite) set Pi_RF. Under this restriction, non-trivial unhackable metric-goal pairs exist (in the sense of Skalse and colleagues, 2022).

Proof sketch: Skalse and colleagues' linearity theorem shows that for the full stochastic policy class, the only unhackable metric is the trivial one (a constant metric). RF's admissibility conditions (Lipschitz contraction, rate

bound, feasible reward set constraint) jointly restrict the policy class. Within the restricted class, the linearity theorem does not apply, and non-trivial unhackable metrics become possible.

Consequence for the Extropy Engine: the protocol's restrictions on contribution strategies (the loop lifecycle, the falsifiability requirement, the validator-routing structure) are not just procedural conveniences. They are the operational form of the admissibility conditions that make non-trivial unhackability possible. Without these restrictions, the protocol's value formula would be hackable in the full-policy-class sense.

### *J.7 Theorem V4-5: Convergent Validation Without Ground Truth (Expanded)*

The full statement: Byzantine-robust peer prediction, extended to n validators with the CTF-PP (Convergent Truth-Finding Peer Prediction) protocol, achieves convergent validation in the presence of linearly many colluding validators.

Proof sketch: The Dasgupta-Ghosh peer prediction protocol provides truth-telling incentives when validators do not have ground truth. The CTF-PP extension generalizes from pairs to n-tuples and bounds the collusion tolerance. The convergence proof rests on the fact that, in expectation, honest validators outperform colluding validators on the truth-telling reward.

Consequence for the Extropy Engine: the validation model does not require a trusted oracle. Validators can reach consensus on claim validity through Bayesian aggregation of independent judgments. The collusion tolerance is bounded but non-trivial: validator pools of size 10 or more, with detection probabilities above 0.3, are safe against rational collusion strategies.

### *J.8 Conjecture V4-6: MeritRank Trilemma Resolution (Status)*

The full conjecture: RF achieves approximate Generalizability, approximate Trustlessness, and epsilon-Sybil-Resistance simultaneously.

Status: stated as a conjecture in RF V4. The formal proof is open. The conjecture is the strongest claim in the framework and is therefore the most likely to require revision under adversarial pressure.

Consequence for the Extropy Engine: the protocol claims to achieve generalizable, trustless, and Sybil-resistant operation, but the formal joint proof is not yet complete. The empirical evidence from the implementation supports the conjecture (the HomeFlow pilot, the integration test suite, the cartel-threshold analysis), but the formal proof is on the research roadmap.

### *J.9 The Two-Phase Bootstrap in Detail*

RF V4's two-phase bootstrap is the formal answer to a real practical question: how does a reputation-weighted system start when no one has reputation yet?

**Phase 1 (Contribution-Only).** All agents have equal standing. Contributions are validated purely on content quality, without reputation weighting. This phase establishes the initial contribution data needed to estimate the belief matrix B for CTF-PP and to compute the initial feasible reward sets R_i for each agent.

The Phase 1 duration is determined by the sample complexity bound: at least Omega(H^3 * S * A / epsilon^2) observations are needed before the first goal update can be triggered (Corollary 5.1 in the source paper, derived from Metelli and colleagues, 2023).

**Phase 2 (Reputation-Weighted).** After sufficient contribution data has been collected, the system transitions to reputation-weighted validation. Contributions are now evaluated with respect to the contributors'

reputation scores, which are themselves determined by past contribution quality. This creates the self-referential dynamics that the formal apparatus of V4 is designed to handle.

The transition from Phase 1 to Phase 2 is triggered when the estimated feasible reward set intersection G* has sufficiently small Hausdorff diameter (less than epsilon), indicating that the governance attractor is well-characterized.

In the Extropy Engine, the two-phase bootstrap is the formal justification for the cold-start period in any new DFAO. A newly-formed DFAO begins in Phase 1: all members have equal standing in validation routing. Once the DFAO has accumulated enough contribution data, governance can vote to transition to Phase 2, where validator routing begins to weight reputation.

### *J.10 The Non-Degeneracy Condition*

Carried forward from RF V3.2, the non-degeneracy theorem ensures that the governance system does not collapse to a trivial state. Formally, the system is non-degenerate if:

1. The governance attractor G* is not a singleton (multiple goals are consistent with observed behavior).
2. The policy set Pi_RF contains at least two distinct policies (agents have meaningful choices).
3. The contribution metric m discriminates between at least two contribution types (the metric is informative).

Under these conditions, the two-phase bootstrap produces a non-trivial reputation distribution (not all agents have the same reputation) and a non-trivial governance attractor (not all agents have the same goal).

The non-degeneracy condition is what prevents the protocol from collapsing to a homogeneous state where the metric is uninformative. The protocol's eight canonical domains, the multi-faceted XP formula, and the diverse contribution categories collectively ensure that the discriminative requirement is satisfied.

## K. Extended Treatment of Representational Fidelity Decay

This appendix expands on the Signal paper's theory beyond the main Codex section 6.

### *K.1 The Differential Equation in Detail*

The central equation:

[  = -k S(t) F(t) + r C(t) (1 - F(t)) ]

Both terms have intuitive interpretations.

The decay term $-k * S(t) * F(t)$ says: the rate of fidelity loss is proportional to the current fidelity (you cannot lose what you do not have), proportional to the current social load (more incentive weight produces faster gaming), and proportional to the domain's susceptibility coefficient k (some domains are more resistant to gaming than others).

The recovery term $r * C(t) * (1 - F(t))$ says: the rate of fidelity recovery is proportional to the gap from perfect fidelity (you can only recover what you have not already), proportional to the corrective feedback strength C(t), and proportional to the restoration rate r.

The system's equilibrium fidelity F* (when dF/dt = 0) is:

[ F^* =  ]

This shows that the equilibrium fidelity is determined by the ratio of corrective force to gaming force. When C is large relative to S (strong corrective feedback, low social load), F* is near 1 (high fidelity). When S is large relative to C (high social load, weak corrective feedback), F* is near 0 (full semantic capture).

### *K.2 The k-Parameter Domain Taxonomy in Detail*

The Signal paper presents three empirically-grounded k-parameter estimates:

**Physical domain (k ~= 0.02).** Reality pushes back quickly. A bridge that is rated for 100 tons but actually fails at 80 tons will visibly collapse, providing immediate corrective feedback. Engineering metrics, physical measurements, thermodynamic instruments operate in this regime. Atmospheric CO2 measurements, fluid-dynamics measurements, structural-load tests: these are domains where the territory pushes back fast.

**Institutional domain (k ~= 0.15).** Feedback is mediated by bureaucracies, peer review, regulatory processes. Corrective feedback exists but operates on timescales of months to years. University rankings, journal impact factors, credit ratings, drug-trial endpoints: these are domains where corrective feedback is slow but real.

**Social/moral domain (k ~= 0.45).** Corrective feedback is weak or absent. Status, identity claims, moral labels, content authenticity claims: these are domains where C(t) approaches zero and the decay term dominates. The "human-made content" label under the EU AI Act is the contemporary case study.

The k values are not arbitrary. They are estimated from documented Goodhart episodes: how quickly did the label decouple from its referent, what corrective feedback was available, and what was the recovery profile.

### *K.3 The Four Phases in Detail*

**Phase 1: Descriptive.** Low S(t), high F. The label is functional. People use it to track the underlying condition. Examples: "house" as a descriptive category, "kilometer" as a unit, "running" as an activity.

**Phase 2: Standardization.** Rising S(t). Institutions invest in verification. F may improve temporarily as measurement gets formalized. Examples: a credential becomes standardized through a recognized institution; a metric becomes mandatory in government reporting.

**Phase 3: Capture.** S(t) saturates. Gaming proliferates. F decays. Actors with the most to gain figure out how to optimize for the label without optimizing for the condition. Examples: SAT preparation industries optimizing test scores; SEO optimizing for search-engine signals; academic publishing optimizing for citation counts.

**Phase 4: Collapse or Correction.** Either semantic emptying (the label no longer tracks anything; "thought leader" is an example) or external C(t) shock triggers partial restoration (a regulator updates the instrument; a credible peer-review process corrects the field). Examples of collapse: the term "natural" in food labeling. Examples of correction: the financial-crisis-era updates to bond-rating methodologies.

The four-phase trajectory is not deterministic. Some labels stabilize in Phase 2 because the underlying domain supports continuous corrective feedback (atmospheric CO2 measurements). Most labels in the social/moral domain (high k) pass through all four phases on a multi-decade timescale.

### *K.4 The Bidirectional Extension*

The Signal paper's distinctive contribution is the bidirectional model: fidelity can recover, not just decay. The recovery requires structurally available corrective feedback (high C(t)) and a non-trivial restoration rate r.

The restoration rate r is a property of the system, not just the domain. A system with strong audit infrastructure, clear falsifiability conditions, and a path from observation to correction has high r. A system without these has r near zero, regardless of the underlying domain's k value.

This is the operational insight the Extropy Engine builds on: the protocol cannot change the domain's k value, but it can engineer the C(t) and r terms to be substantial. The 30-day retroactive-burn window is a structural increase in C(t). The validator-reputation penalty for falsified consensus is an increase in r. The transparency of the DAG is an enabler of both.

### *K.5 Self-Application*

The Signal paper applies its own model to itself. It claims k ~= 0.15 for itself (institutional domain). The corrective feedback C(t) is peer review, falsification attempts, and empirical challenge. The paper is allowed to lose.

This self-application is not a rhetorical move. It is a substantive claim: the paper's predictions about other labels apply to the paper's own labels. If the paper's framework drifts (the k-parameter taxonomy becomes a fixed dogma, the four-phase model becomes a procrustean bed), the framework will eventually be subject to the same Goodhart pressure it analyzes. The paper preempts this by submitting itself to falsification.

The Extropy Engine adopts the same posture at the protocol level. The protocol's specification is allowed to lose. The Codex is the explicit form of the protocol's self-application: documented falsifiability conditions, documented gaps, documented correction history.

## L. The HomeFlow Pilot Deployment Guide (Summary)

This appendix summarizes the HomeFlow family pilot deployment guide. The full FAMILY_PILOT.md document in the repository contains operational details; the summary here provides enough context for the Codex's technical reader to understand what the pilot does and how.

### *L.1 What Gets Deployed*

HomeFlow deploys the core Extropy Engine services (epistemology-engine, signalflow, loop-ledger, reputation, xp-mint, dag-substrate, dfao-registry, governance, temporal, token-economy, credentials, ecosystem) plus the HomeFlow application service (port 4015) and the homeflow-ui frontend (port 3002).

The default deployment uses file-backed storage (no Postgres required). The deployment can run on consumer hardware: a Mac mini or a small Linux server is sufficient.

### *L.2 Setup Steps*

1. Clone the repository.
2. Run `pnpm install` and `pnpm run build`.
3. Run `docker compose up --build -d` to bring up the core services, or use the Python orchestrator for non-docker local deployment.
4. Run the HomeFlow setup wizard (in homeflow-ui or via API).
5. Create the household DFAO at MICRO scale.
6. Onboard each family member through the identity flow.
7. Configure the household's weight vector w over the eight domains.
8. Configure the household's governance method (typically simple-majority for routine, conviction for substantial).

9. Begin submitting loops.

### *L.3 What Family Members See*

Each family member has a character sheet showing:

- Their accumulated XP balance (with provisional and confirmed breakdowns).
- Their rho values per domain.
- Their CT, CAT, IT, DT balances (CAT and DT typically zero for routine households).
- Their EP balance (the household's internal "spendable" currency for intra-household allocations).
- A timeline of recent loops (their own and others').
- A queue of validation requests.

The character sheet uses household-language translations of the protocol's terminology: "contribution score" for XP, "trust level" for rho, "household credit" for EP.

### *L.4 What Lessons Inform the Codex*

The HomeFlow pilot has informed several Codex decisions:

- The terminology translation layer is necessary; raw protocol terms confuse non-technical users. The character-sheet UI uses household-language; the underlying protocol uses canonical names.
- The 30-day retroactive-burn window is too long for routine household claims; HomeFlow uses a 7-day window for chore-class claims. The protocol's governance-tunable window default accommodates this.
- The validator pool size for household claims is 1 to 2 (one parent and one other member); the protocol's default of 5 to 12 is appropriate for non-household scales.
- The convergence vertex is useful for multi-person household contributions (a meal prepared by two people, a deep-clean done as a team); the convergence-split rules are intuitive for household members.

### *L.5 What HomeFlow Has Not Yet Tested*

The HomeFlow pilot has not yet exercised:

- The merchant layer (Layer 2). Intra-household EP conversion is not a true two-sided market test.
- The retroactive-burn falsification path at scale (most household claims are not falsified in the retroactive window because the validation is informal and trusted).
- Cross-DFAO interaction (the household is a self-contained DFAO; cross-DFAO portability of CAT or rho is not exercised).
- The validator-collusion threat model (in a trusted household, collusion is not adversarial).
- The Sybil resistance model (in a household with KYC-bound members, Sybil attempts are not a real threat).

These limitations are explicit. The HomeFlow pilot is the first deployment, not the comprehensive deployment. Subsequent deployments (a neighborhood DFAO, a workplace DFAO, a multi-DFAO ecosystem) will exercise the parts of the protocol that HomeFlow does not.

## M. The Repository's Three-Layer Separation Document

This appendix summarizes the THREE_LAYER_SEPARATION.md document from the repository, which articulates the protocol's intentional metric divergence as a Goodhart defense.

### *M.1 Why Separation*

The protocol's three layers (user-facing, merchant-facing, engine) are presented to three different audiences with three different metrics. This is not an accident of presentation. It is a structural choice grounded in the Goodhart resistance theory.

If the user sees the same metric the merchant sees, the user can optimize for the merchant's view, which collapses the merchant-side signal. If the merchant sees the same metric the engine uses, the merchant can game the engine's value calculation, which collapses the engine-level integrity. The separation is the defense.

### *M.2 What Each Layer Sees*

**Layer 1 (User-facing).** The user sees: their character sheet, their EP balance, their participating-merchant discounts. The user does not see: the XP formula coefficients, the rho values of other validators, the DFAO weight vector. The user experiences the system as a loyalty program with better economics.

**Layer 2 (Merchant-facing).** The merchant sees: their customer pipeline (anonymized), their operational signal dashboard (aggregated entropy patterns), their EP conversion rates, their merchant-services fee. The merchant does not see: individual user character sheets, validator routing decisions, the XP formula. The merchant experiences the system as a free POS with deeper analytics.

**Layer 3 (Engine).** The engine maintains: the XP formula, the validator routing, the DAG substrate, the token economy, the governance layer. The engine is what the Codex specifies.

### *M.3 The Goodhart Mechanism*

The separation works as a Goodhart defense because users and merchants cannot directly optimize for the engine's metrics. They can optimize for their layer's metrics, but their layer's metrics are downstream of the engine's metrics and are mediated by the engine's invariants.

A user trying to game their EP balance would need to game their XP balance, which requires gaming the five-term formula, which is structurally hard (R is loop-class, F penalizes repetition, Delta S requires validated instruments, w dot E requires multi-domain evidence, T_s is bounded). The user's direct optimization target (more EP) does not produce a direct gaming attack on the engine.

A merchant trying to game their operational signal would need to game the entropy patterns they observe, which they cannot directly write to (the patterns are derived from validated user contributions). The merchant can offer better products or better discounts to attract better customers; they cannot directly inject signal.

### *M.4 The Separation as Cultural Boundary*

The three layers also serve as cultural boundaries. Users encounter the system through cultural surfaces (the public site, the app's UX). Merchants encounter the system through commercial surfaces (the POS, the analytics dashboard). Engineers encounter the system through the repository, the Codex, and the academic papers.

These cultural boundaries route adoption appropriately. A user does not need to read the Codex to use the system; the user-facing layer is self-contained. A merchant does not need to read the academic papers to deploy the system; the merchant-facing layer is self-contained. An engineer reads the Codex and engages with the protocol's mathematics.

The separation is not a fence around expertise. It is a respect for the different mental models that different audiences bring to the system. The protocol does not demand that everyone understand the formula. It demands only that the formula be internally consistent and externally auditable.

## N. Implementation Status Snapshot

A final summary of the implementation status as of the v3.1.2 canonical snapshot.

### *N.1 What Is Complete*

- The full type system and inter-service contracts.
- The 22 typed inter-service events and the event-bus.
- The five-service happy-path integration test (12 of 12 passing).
- The canonical XP formula in a single source file with FORMULA_VERSION stamping.
- The reputation accrual service with per-domain decay.
- The DAG substrate's vertex and edge primitives (single-node PostgreSQL).
- The DFAO registry with the five canonical scales and lifecycle events.
- The governance service with proposals, conviction voting, and threshold execution.
- The temporal service with seasons, decay scheduling, and loop timeouts.
- The token economy with the six-token enum and balances.
- The credentials service with verifiable credentials and CAT issuance.
- The ecosystem aggregator.
- The HomeFlow family pilot with Google OAuth and file-backed storage.
- The identity layer's canonical-flow endpoints (in-memory storage, HMAC nullifier sandbox).
- The node-handshake VPS-to-laptop sandbox.

### *N.2 What Is Partial*

- The PSLL-sync service (skeleton, spec frozen).
- The quest marketplace (skeleton).
- The validation neighborhoods (skeleton).
- The libp2p gossip layer (planned, not implemented).
- The Pedersen-based nullifier production swap (planned, not implemented).
- The legacy v3.0 decomposition endpoints (present, marked deprecated, scheduled for v3.2 removal).
- The contract test suite (incomplete; the most likely source of future API drift).
- The cross-DFAO conflict resolution (specified, partial implementation).

### *N.3 What Is Open*

- The cross-domain Delta S normalization calibration.
- The c_l per-domain calibration.
- The Godel Boundary Watchdog.
- The cross-DFAO F arbitrage mitigation.
- The regulatory analysis for CT-to-fiat bridges.
- The cultural-friction mitigation for the optional base-10 temporal layer.
- The formal loop lifecycle transition table publication.
- The adversarial integration test suite.
- The automated Complexity Thermostat.

- The Phase 2 distributed DAG consensus.

### *N.4 What Is Architecturally Decided But Operationally Deferred*

- The production cryptographic primitives (BLAKE3, Ed25519, BBS+, Pedersen) are specified; the production swap is deferred.
- The multi-node deployment is specified; the production cluster orchestration is deferred.
- The full sample-complexity bootstrap (Phase 1 of the RF V4 two-phase) is specified; the operational triggering of Phase 2 is deferred.

### *N.5 The Honest Summary*

The Extropy Engine v3.1.2 is a working sandbox protocol with a coherent specification, a peer-reviewed mathematical foundation, an explicit gap catalog, and a first deployment vertical. It is not yet a production system. The Codex is the consolidation of what has been built, what has been specified, and what remains.

The reader who has reached this point has the comprehensive technical specification. The repository has the implementation. The book has the narrative. The academic papers have the formal foundations. The Codex is the bridge across all of them.

*End of Extropy Codex (extended appendices). Full document end.*

## O. Distinguishing the Extropy Engine from Adjacent Systems

This appendix addresses the predictable question: how does the Extropy Engine differ from other systems that share some of its surface features? The answer matters because the surface features (a token, a DAG, a reputation primitive, a governance layer) are individually unremarkable. The distinctiveness is in the specific combination and the architectural invariants.

### *O.1 vs. Conventional Cryptocurrencies*

A conventional cryptocurrency mints tokens by computational work (proof-of-work) or staked capital (proof-of-stake). The token's value is determined by market demand. The token's circulation is independent of any specific real-world action.

The Extropy Engine's XP is not a cryptocurrency. XP is non-transferable. XP mints only when validated entropy reduction is recorded. XP decays. XP's "value" is not a market price; it is a tally of contribution that conditions other things (EP conversion at merchants, IT accumulation for governance weight, rho accumulation for validator routing).

The closest analog among the canonical six tokens to a cryptocurrency is CT (the transferable, identity-bearing contribution token). But CT also differs from conventional tokens: it has a 14-day lockup, a 2% transfer friction, and an identity-bearing component (rho enters the formula). CT cannot be purely traded as a speculative instrument; it carries the contributor's identity weight.

The fundamental distinction is that the Extropy Engine's tokens are accounting receipts for validated work, not speculative assets. The protocol does not optimize for liquidity or for market cap. It optimizes for verified entropy reduction.

### *O.2 vs. Reputation Systems*

A reputation system attempts to capture trustworthiness in a numerical score. Examples: eBay's seller ratings, Yelp reviews, credit scores, academic h-indices, Uber driver ratings.

Reputation systems have predictable failure modes: reputation laundering (the highest reputations belong to actors who learned to game the signal, not to actors who are most trustworthy), reputation lock-in (new entrants cannot accumulate reputation against incumbents), reputation as social control (the entity controlling the score controls behavior).

The Extropy Engine treats reputation as a real but bounded quantity. Reputation density (rho) accrues from validated contributions. Reputation conditions validator routing (a high-reputation validator sees more claims and accrues more rho). Reputation does not multiply value: the architectural invariant prevents the reputation laundering vector.

The distinction matters because a system without separation of value and reputation will inherit the reputation system's failure modes. A system with separation can use reputation for routing without exposing the value calculation to reputation laundering.

### *O.3 vs. DAOs*

A Decentralized Autonomous Organization is a governance structure where decisions are made by token-weighted vote. Examples: MakerDAO, Uniswap governance, various Ethereum L2 governance bodies.

DAOs have predictable failure modes: governance concentration (tokens accumulate among whales who dominate votes), low participation (most token holders do not vote), governance theatre (votes are nominal; execution is unilateral), regulatory ambiguity (token-weighted governance attracts SEC attention).

The Extropy Engine's DFAOs differ from DAOs in several important ways:

- Voting weight is not token-weighted in the conventional sense. The Influence Token IT decays at 5% per month, preventing permanent influence accumulation. IT cannot be purchased; it accrues from contribution.
- The DFAO structure is fractal. NANO and MICRO DFAOs handle local decisions; MESO and MACRO DFAOs handle community-scale decisions; PLANETARY DFAOs handle protocol-scale decisions. Concentration of voting weight at one scale does not translate to concentration at another.
- Conviction voting (the default for substantial decisions) rewards patient positions over impulsive shifts.
- Governance proposals are visible and discussable; the deliberation period is structural, not optional.
- DFAOs are decentralized but not unconstrained: parent-DFAO safety bounds prevent child DFAOs from violating protocol-level invariants.

### *O.4 vs. Holochain*

Holochain is a distributed-application framework with personal source-chains and a DHT-based DAG. The Extropy Engine borrows several patterns from Holochain (the PSLL is conceptually a source-chain; the DAG substrate is influenced by Holochain's data model). The Extropy Engine reimplements these patterns natively rather than running on Holochain.

The distinction: Holochain provides a general framework for distributed applications. The Extropy Engine provides a specific protocol with a value formula, a domain taxonomy, a token economy, and a governance model. Holochain is infrastructure; the Extropy Engine is a system built with patterns inspired by Holochain's infrastructure.

### *O.5 vs. IOTA*

IOTA's Tangle uses a directed-acyclic-graph data structure where every transaction confirms two prior transactions. The Extropy Engine's DAG substrate is influenced by IOTA's Tangle (the tip selection algorithm, the dual-attachment, the continuous confirmation model).

The distinction: IOTA is a payment-focused DAG. The Extropy Engine is a contribution-focused DAG. The vertex types in the Extropy Engine (LOOP_OPEN, VALIDATION_RECORDED, XP_MINT, GOVERNANCE_VOTE, and others) are protocol-specific; IOTA's vertex types are transaction-focused. The two systems share architectural patterns but operate on different content.

### *O.6 vs. Universal Basic Income (UBI)*

Some proposals for non-extractive economics center on Universal Basic Income: a per-person stipend independent of contribution. The Extropy Engine is not a UBI mechanism.

The distinction: UBI distributes value without measuring contribution. The Extropy Engine distributes value only against validated contribution. The protocol does not provide a per-person stipend; it provides a way to mint accounting against verified entropy reduction.

The Extropy Engine and UBI are not contradictory. A society could operate both: UBI handles baseline economic security; the Extropy Engine handles contribution-based reward layered on top. The Codex does not take a position on UBI; it takes a position on what the Extropy Engine specifically does (and does not).

### *O.7 vs. Carbon Accounting*

Some proposals for non-extractive economics center on carbon accounting: a unit of value tied to carbon-equivalent measurement. The Extropy Engine's thermodynamic-domain claims include carbon-related entropy reduction, but the protocol is not a carbon accounting system.

The distinction: carbon accounting focuses on a single physical quantity (greenhouse gas equivalents). The Extropy Engine's thermodynamic domain is broader (waste heat, material flow, energy efficiency) and is one of eight canonical domains. Carbon accounting can be expressed as a sub-category of thermodynamic-domain claims in the Extropy Engine, but the protocol does not reduce to carbon accounting.

### *O.8 vs. Effective Altruism*

Effective altruism advocates for measurable, evidence-backed, comparative analysis of impact-per-dollar. The Extropy Engine's emphasis on falsifiable measurement and validated contribution overlaps with EA's methodological commitments.

The distinction: effective altruism is a framework for individual giving decisions. The Extropy Engine is a protocol for coordination at the system level. EA can be applied within the Extropy Engine (an EA-oriented DFAO can configure its weight vector to prioritize the domains EA judges most impactful). The Extropy Engine does not advocate for any specific philosophy of value; it provides the infrastructure for any value philosophy that admits measurement.

### *O.9 vs. Decentralized Identity (DID) Standards*

The W3C DID specification provides a framework for self-sovereign identity. The Extropy Engine's identity layer is W3C-DID-compatible (the DID format `did:extropy:<64-hex>` follows the DID method specification). Verifiable Credentials follow the W3C VC standard.

The distinction: W3C DIDs and VCs are infrastructure. The Extropy Engine is a protocol built on this infrastructure. The protocol adds: the per-context nullifier, the threshold reveal scheme, the PSLL with Merkle anchoring, the DFAO membership credentials, the CAT issuance flow.

### *O.10 The Synthesis*

The Extropy Engine combines patterns from cryptocurrencies (token accounting), reputation systems (rho accrual), DAOs (governance), Holochain (PSLL), IOTA (DAG substrate), W3C standards (DID, VC), and several other influences. The combination is novel; the components are not.

The architectural invariants (reputation does not enter XP; the user is sovereign; the DAG is event-sourced and append-only) are the protocol's distinctive contributions. These are not present in any of the adjacent systems. They are the operational consequences of the theoretical foundations (the representational fidelity decay theory, Randall's Feedback V4, the Shannon-Landauer-Bennett grounding).

A reader evaluating the protocol's distinctiveness should evaluate the invariants, not the components. The components are widely available. The specific combination, under the invariants, is what the Extropy Engine claims to contribute.

## P. The Universal Times Calendar

The repository contains a temporal-service package that exposes a "Universal Times" base-10 calendar system. This appendix summarizes the calendar's structure and addresses its status within the protocol.

### *P.1 The Calendar's Structure*

The Universal Times calendar uses a base-10 hierarchy of time units:

- **Eon:** the largest time unit (millions of years).
- **Age:** a sub-unit of an Eon.
- **Era:** a sub-unit of an Age.
- **Epoch:** a sub-unit of an Era.
- **Cycle:** a sub-unit of an Epoch (roughly a year-equivalent).
- **Season:** a sub-unit of a Cycle (roughly a quarter-equivalent).
- **Current:** a sub-unit of a Season (roughly a month-equivalent).
- **Spin:** a sub-unit of a Current (roughly a day-equivalent).
- **Tide:** a sub-unit of a Spin (roughly an hour-equivalent).
- **Wave:** a sub-unit of a Tide (roughly a minute-equivalent).
- **GQ:** the smallest unit (roughly a second-equivalent).

Plus solar-aligned units: Solar Loop, Arc, Tick.

### *P.2 Why Base-10*

The case for base-10 temporal units rests on coordination convenience. Decimal arithmetic on base-10 units is simpler than the modular arithmetic required for traditional units (60-second minutes, 24-hour days, 7-day weeks, 12-month years, 365.25-day years). A base-10 calendar admits scheduling math without conversion overhead.

The case against base-10 temporal units rests on cultural inertia. The seven-day week is a deeply embedded institutional rhythm. The 12-month year is the basis of fiscal accounting. The 24-hour day is the unit of human circadian biology. Replacing these with base-10 equivalents has high friction.

### *P.3 Status Within the Protocol*

The Universal Times calendar is **optional and DFAO-gated**. A DFAO can choose to operate on the standard Gregorian/24-hour-day timeline (the default) or on the Universal Times timeline.

The protocol's core temporal mechanics (loop timeouts, retroactive-burn windows, decay schedules) are expressed in canonical time units (seconds, days). The Universal Times calendar is a presentation layer on top of the canonical units; a DFAO using Universal Times sees its loop windows expressed in Spins or Currents, but the underlying timing is the canonical seconds.

### *P.4 The Port-Collision Note*

The temporal-service package's default port (4002) collides with the signalflow service's port. This is a known bug. The Codex documents the resolution in section 15.5: rename one of the packages and reassign the port.

The naming itself is also confusing: the temporal package (port 4011) handles seasons, decay scheduling, and loop timeouts (the core temporal mechanics). The temporal-service package handles the Universal Times calendar (the optional base-10 presentation). The Codex's recommended cleanup: rename temporal to temporal-heartbeat and temporal-service to temporal-universal-times, with non-colliding ports.

### *P.5 Cultural Friction as a P3 Risk*

If the Universal Times calendar is activated as a protocol-level requirement (rather than a DFAO-gated option), the resulting cultural friction with seven-day-week institutions is a P3 risk. The Codex acknowledges this in section 17.9 and in the gap catalog (section 19).

The current status preserves the cultural option: a DFAO that wants the base-10 calendar can use it; a DFAO that wants the Gregorian calendar can use that. The protocol does not force a calendar choice.

## Q. Implementation Patterns for Adopters

A short appendix for technical adopters integrating against the protocol or building applications on top of it.

### *Q.1 Use the Single-Source Formula*

Every minting service in the reference implementation imports the XP formula from packages/xp-formula/src/index.ts. There are no re-implementations. Adopters building on the protocol should follow the same pattern: import the canonical formula; do not reimplement.

If the canonical formula is updated (a future v3.2 or v4.0 might revise it), the FORMULA_VERSION stamp allows adopters to distinguish mints under different formula versions. Backward compatibility is preserved by the version stamp; forward compatibility requires importing the updated module.

### *Q.2 Use the Canonical Field Names*

The contracts package defines canonical field names for inter-service contracts. Use these names. Legacy field names (settlementTimeSeconds, feedbackClosure, context, reputation) are accepted during the v3.1.2 rollout for backward compatibility but will be removed in v3.2.

The canonical names are: rarityMultiplier (for R), frequencyOfDecay (for F), entropyDelta (for Delta S), domainWeights and evidenceVector (for w and E separately), settlementTimeFactor (for T_s), capability (for C in CT), reputationDensity (for rho).

### *Q.3 Anchor PSLL to DAG, Not to Backups*

The PSLL is the user's local source of truth. Adopters building applications should anchor PSLL Merkle roots to the DAG, not to off-PSLL backup systems. The DAG is the canonical witness for what the user has done; backup systems are convenience for the user's local resilience.

A PSLL that diverges from its DAG anchors triggers a dispute review, not silent loss. Adopters should design their applications to surface divergence to the user, not to hide it.

### *Q.4 Validate Inputs at the Boundary*

Adopters integrating with the protocol should validate inputs at the boundary: ensure T_s is in (T_floor, 1.0], ensure R is in [0.1, 10.0], ensure F is in (0, 1], ensure Delta S is strictly positive, ensure the evidence vector E is non-negative and sums to a reasonable value. The protocol services will reject malformed inputs, but client-side validation provides better user experience.

### *Q.5 Subscribe to the Right Events*

The event-bus has 22 typed events. Adopters should subscribe to the events relevant to their application: xp.minted.confirmed for character-sheet updates, loop.settled for retroactive-window closure, governance.proposal.passed for governance updates, and so on. The full event catalog lives in contracts/types.ts.

### *Q.6 Respect the Three-Layer Separation*

Adopters building user-facing applications should expose Layer 1 surfaces (the character sheet, EP balances, participating-merchant discounts). They should not expose Layer 3 internals (the XP formula coefficients, the validator routing decisions, the rarity table) to end users. The three-layer separation is structural; respecting it preserves the Goodhart defense.

### *Q.7 Plan for the v3.2 Migration*

The v3.2 migration will remove the legacy v3.0 decomposition endpoints, remove the GT and RT legacy field acceptance, swap to Pedersen-based nullifiers, and add the libp2p gossip layer. Adopters should:

- Migrate any integrations against legacy v3.0 endpoints to the v3.1 mesh-observability endpoints before v3.2.
- Migrate any code using GT or RT to the canonical six-token names.
- Plan for the production nullifier swap (application code does not change, but operational config will need updating).

- Plan for the production gossip layer (operational config will change; application code remains the same).

### *Q.8 Engage with the Open Gaps*

The protocol's gap catalog (section 19) is the explicit list of what remains unsolved. Adopters with relevant expertise are invited to engage with specific gaps: the cross-domain Delta S calibration is an open empirical question; the Godel Boundary Watchdog is open architectural work; the formal proof of Conjecture V4-6 is open research.

The protocol does not claim to be complete. It claims to be examinable, falsifiable, and open to contribution.

## R. The Philosophical Companion: Functional Divinity

This appendix summarizes the philosophical extension paper *God as Emergent Entropy Reduction*. The Codex treats this as a philosophical companion, not as load-bearing for the protocol's operational specification. Readers focused on the technical specification can skip this appendix without loss.

### *R.1 The Reframe*

The paper proposes that the question of God can be reframed from ontology (does a supreme entity exist?) to function (does reality contain measurable, recursive, feedback-driven entropy reduction?). Under the reframe, divinity is not a being but a functional name for a pattern: the emergence of coherent order through feedback across domains.

The reframe is not a claim that traditional religious experience is invalid. It is a proposal that traditional religious language may have been pointing imprecisely at real coherence-generating functions that lacked formal measurement.

### *R.2 The Functional Definition*

The paper's central definition:

> "God is the emergent function by which feedback-driven systems reduce entropy across domains."

The definition is functional, not ontological. It does not specify a supernatural being. It specifies a pattern that, when present in a system, manifests as the conversion of disorder to coherent structure through feedback.

### *R.3 The Falsification Conditions*

The paper specifies what would falsify the divine function claim:

1. **The alleged reduction is not measurable.** If no instrument, proxy, audit trail, or feedback structure can detect the reduction, the claim remains poetic rather than operational.
2. **The reduction is local but parasitic.** A corporation reducing its internal economic disorder by dumping thermodynamic disorder elsewhere is not exhibiting divine function. It is externalizing entropy.
3. **The reduction is temporary, suppressed, or compressed fraud.** A system that looks ordered because dissent has been suppressed, data has been hidden, or complexity has been pushed downstream is not exhibiting coherence. It is compression fraud.
4. **The measurement system itself drifts.** Once people gain status or reward from being seen as entropy reducers, they may optimize for the metric rather than the underlying reduction. Goodhart applies to divinity too.

These conditions are testable. The paper's claim is allowed to lose. This is the same posture the Extropy Engine adopts at the protocol level.

### *R.4 The Theological Translations*

The paper offers operational translations of classical divine attributes:

- **Omniscience** becomes the ideal of maximal signal integration: the conversion of dispersed information into coherent, actionable understanding.
- **Omnipresence** becomes the domain-general applicability of entropy dynamics.
- **Omnipotence** becomes the limit case of maximal transformation with minimal waste.
- **Goodness** becomes the tendency of actions to reduce disorder in ways that survive recursive feedback.

These are not literal proofs of classical attributes. They are operational reconstructions of what those attributes might have been pointing at when the framework of mathematical measurement was unavailable.

Other religious concepts also translate:

- **Prayer** becomes a request for feedback, orientation, and correction.
- **Worship** becomes alignment with coherence-generating patterns.
- **Sin** becomes disorder injection without compensating repair.
- **Salvation** becomes sustained participation in feedback loops that transform destructive drift into durable order.

The translations do not force themselves on religious tradition. They offer themselves as an operational bridge for those who find the bridge useful.

### *R.5 What the Paper Does Not Claim*

The paper is explicit about what it does not claim:

- It does not prove that a personal God exists.
- It does not prove that consciousness is fundamental.
- It does not prove that the universe has intentions.
- It does not prove that all religious experience is valid.
- It does not prove that all religious experience is invalid.

It offers a testable reconstruction. The reconstruction is useful for those who want to engage religious language operationally. It is silent on the metaphysical questions that traditional theology engages.

### *R.6 The Terminology Note*

The God paper carries one terminology slip relative to the canonical v3.1.2 notation: it glosses F in the XP formula as "Feedback closure strength." This is incorrect. F in the XP formula is frequency-of-decay. Feedback closure strength is a separate concept (and is captured at the validation level as V_c, or in the Signal paper as C(t)).

The correction is documented in the Codex's correction ledger as C6. Readers of the God paper should mentally substitute: "F is frequency-of-decay; the concept the paper calls feedback closure strength corresponds to V_c in the validation aggregation."

### *R.7 The Connection to the Protocol*

The God paper's connection to the Extropy Engine is explicit: the engine is described as "a prototype architecture for measuring contribution through validated entropy reduction. Its purpose is not to detect God as a supernatural object. Its purpose is to detect the functional pattern that this paper calls divine: measurable improvement produced through feedback."

The connection is a philosophical extension, not a foundational claim. Readers focused on the protocol can engage with the God paper or skip it. Readers focused on the philosophical question can read the God paper without committing to the protocol's specific operationalization.

### *R.8 Why This Appendix Exists*

The Codex includes this appendix because:

1. The God paper is part of the project's published material, and the Codex consolidates the project's material.
2. Some readers will encounter the God paper before or after the Codex, and the correction ledger entry C6 is easier to act on with this appendix's context.
3. The Codex's epistemic posture (falsifiability, validated measurement, the model is allowed to lose) extends consistently from the technical specification through the philosophical extension. The appendix shows the extension explicitly.

Readers who find the philosophical extension irrelevant or unwelcome are free to skip the appendix. The protocol specification stands on its own.

## S. Final Notes for the Reader

### *S.1 What You Have Read*

This Codex is the comprehensive technical specification of the Extropy Engine. It consolidates:

- The v3.1.2 canonical specification (corrected for the P0 errors).
- The architectural foundations (Digital Autarky, the contribution graph, the three-layer separation, the contribution-as-DAG model).
- The mathematical foundation (XP, CT, EP, the irreducible form, the cross-domain normalization, the dimensional analysis).
- The eight canonical domains and their instruments.
- The loop lifecycle and the validation model.
- The DAG substrate and the person-as-DAG architecture.
- The identity layer, the PSLL, and the Digital Autarky model.
- The token economy with the six canonical tokens.
- The DFAO governance model and the provisional defaults.
- The Goodhart resistance theory (representational fidelity decay).
- The formal foundations of governance (Randall's Feedback V4).
- The implementation status from the repository.
- The website and public narrative layer.
- The security model and threat analysis.
- The peer review response.

- The open engineering gaps and the roadmap.
- The glossary and formula reference.
- Extended treatments of the XP formula, the worked loop example, the DAG substrate, the identity flow, multi-party convergence, the HomeFlow pilot, the RF V4 theorems, the Signal paper's bidirectional model, the Universal Times calendar, the philosophical extension, and the distinguishing features from adjacent systems.

### *S.2 What This Codex Replaces*

This Codex replaces, as the canonical technical reference, the following prior documents:

- The prior synthesis-document final pass (the document previously labeled Final).
- The earlier v0.1 synthesis draft.
- The synthesis blueprint plan.
- The earlier raw-appendix output that prompted this consolidation.

These prior documents remain available as source material but are superseded by the Codex.

### *S.3 What This Codex Does Not Replace*

This Codex does not replace:

- The companion book (Unfuck the World for a Dollar). The book is the narrative case; the Codex is the technical specification. They are complementary.
- The repository (github.com/00ranman/extropy-engine). The repository is the canonical implementation. The Codex describes what the repository does; the repository is what the repository does.
- The academic papers (Randall's Feedback V4, the Signal paper, the God paper). The Codex condenses the load-bearing content of these papers; the papers themselves contain the complete formal proofs and the full case studies.
- The peer review document. The peer review is preserved as the historical record of external examination; the Codex documents the corrections applied in response.

### *S.4 How to Cite This Codex*

This Codex is internal documentation for the Extropy Engine project. It is not a published academic paper. References to the Codex's content should cite the underlying source material where appropriate:

- For the XP formula and v3.1.2 canonical specification: cite the repository's xp-formula/src/index.ts and docs/CHANGELOG.md v3.1.2 entry.
- For the formal governance foundations: cite the Randall's Feedback V4 paper.
- For the representational fidelity decay theory: cite the Signal paper.
- For the philosophical extension: cite the God paper.
- For the DAG architecture: cite the DAG Architecture paper.
- For the peer review response: cite the peer review document.

The Codex itself is the consolidation, not a standalone source.

### *S.5 The Closing Statement*

The Extropy Engine is a protocol for measuring validated entropy reduction. The protocol is documented in this Codex, implemented in the repository, narrated in the book, and grounded in the academic papers. The protocol is allowed to lose. Every claim in this Codex is subject to peer review, adversarial pressure, and operational falsification.

The Codex is the offer to be read. The protocol is the offer to be built upon. The repository is the offer to be examined. The book is the offer to be considered.

The system isn't broken. It's working exactly as designed. The Extropy Engine is the attempt to design a different one, and the Codex is the attempt to specify that design completely enough that another mind can audit it, build on it, falsify it, or improve it.

That is the entire point.

*End of Extropy Codex: Comprehensive Technical Specification. Full document complete.*

## T. Extended Treatment of the Eight Domains: Per-Domain Deep Dives

This appendix extends the section 8 treatment of the eight canonical domains, providing more depth on instruments, intersectionality, and current operationalization status per domain. The deep dives are organized to be readable independently; a reader concerned only with one domain can read that domain's section without reading the others.

### *T.1 Cognitive Entropy: Deep Dive*

Cognitive entropy is the disorder in understanding, explanation, learning, conceptual compression, and mental-model coherence. The domain is grounded in the information-theoretic observation that a good explanation compresses a complex phenomenon to a simpler form without losing fidelity, while a bad explanation either fails to compress or distorts the phenomenon.

**Operationalization paths.** The protocol supports several cognitive-domain instruments:

1. **Explanation length compression.** Under a content-preserving constraint (the explanation must preserve the original explanation's testable predictions), measure the explanation length reduction. A 1000-word explanation compressed to 200 words while preserving predictions represents a substantial cognitive entropy reduction.
2. **Error rate reduction in structured learning.** Pre-test and post-test a learner against a structured instrument. The error rate reduction divided by the time invested is a cognitive entropy reduction signal. Suitable for pedagogical contributions.
3. **Time-to-mastery improvement.** Standardized curricula have known mastery timelines. A pedagogical contribution that demonstrably reduces time-to-mastery (under controlled measurement) reduces cognitive entropy for future learners.
4. **Validator rubric assessments.** For contributions that cannot be measured by automated instruments (philosophical synthesis, complex reasoning), structured rubrics with multiple independent validators provide a measurement instrument. The inter-rater reliability is itself a quality signal.

**Failure modes specific to the domain.**

- **Explainer slop:** content that feels clarifying but adds no compression. Detected by post-hoc analysis showing readers do no better on test instruments after exposure than control groups.

- **Pseudo-pedagogy:** content optimized for engagement metrics rather than understanding. Detected by tracking learner outcomes against the engagement signal; if engagement is high but learning gain is low, the content is pseudo-pedagogical.
- **Authoritative-tone substitution:** cognitive entropy claims that rely on authorial reputation rather than measurement. The architectural invariant (R is rarity, not reputation) prevents this from inflating XP, but it can still inflate validator confidence if the validators are not careful.

**Intersectionality.** Cognitive entropy claims often have significant informational entropy components (an explanation that compresses content is also an informational improvement) and may have social components (a widely understood explanation reduces coordination friction in groups that depend on the understanding).

**Maturity status.** Medium. The cognitive instruments are well-understood in pedagogical and psychometric research, but the integration of these instruments into a protocol-grade measurement framework is still preliminary. The HomeFlow pilot exercises cognitive entropy claims at the household scale (a parent teaches a child a new skill; the skill acquisition is the cognitive entropy reduction); broader-scale cognitive contributions await more deployment.

### *T.2 Code Entropy: Deep Dive*

Code entropy is the disorder in software systems: architectural fragmentation, complexity accumulation, maintainability degradation, correctness failures, test-coverage gaps. The domain is the most mature among the protocol's eight because software engineering has decades of mature instrumentation.

**Operationalization paths.**

1. **Cyclomatic complexity delta.** A refactoring that reduces cyclomatic complexity (measured by static analysis) without reducing functionality is a code entropy reduction. The instrument is well-understood and automated.
2. **Bug density reduction.** Defects per thousand lines of code (KLOC) reduction, measured over a stable measurement window after the change. A change that demonstrably reduces post-deployment defect rates is a code entropy reduction.
3. **Test coverage improvement.** New tests that increase branch and statement coverage without becoming tautological represent informational entropy reduction in the test suite (the suite now has higher discriminatory power against future bugs).
4. **Mean-time-to-detect (MTTD) and mean-time-to-recover (MTTR) improvements.** Observability and incident-response improvements that reduce MTTD and MTTR are code-domain contributions with significant temporal-domain crossover.
5. **Formal verification coverage.** A theorem-prover-verified function or module is a substantial code entropy reduction relative to an untested function.

**Failure modes specific to the domain.**

- **Gaming via trivial refactors:** changes that improve a metric without improving substance. Detected by validator review of the refactor's downstream impact.
- **Test coverage padding:** tests that are tautological or test trivial paths. Detected by mutation testing: the test suite's quality is measured by how many introduced mutations the suite catches.
- **Complexity externalization:** a refactor that reduces local complexity by moving the complexity to a different module. Detected by system-level complexity tracking.

**Intersectionality.** Code entropy claims often have cognitive components (the refactored code is easier to understand), economic components (the refactored code is cheaper to maintain), and informational components (the refactored code has clearer interfaces, reducing informational uncertainty for callers).

**Maturity status.** High. The code-domain instruments are mature, automated, and broadly accepted in the software engineering community. The protocol can leverage existing tooling (static analyzers, mutation testers, code-quality dashboards) as the measurement infrastructure.

### *T.3 Social Entropy: Deep Dive*

Social entropy is the disorder in trust networks, cooperation patterns, conflict dynamics, and community coherence. The domain is the protocol's least mature domain because social measurement is genuinely hard.

**Operationalization paths.**

1. **Network cohesion deltas.** Clustering coefficients, reciprocity scores, and other graph-structural measures applied to the social-interaction graph. A contribution that demonstrably increases cohesion (within a validated context, not just any context) is a social entropy reduction.
2. **Conflict-incident reduction.** Documented conflict incidents in a community, measured over a windowed baseline. A mediation, an intervention, or a structural change that demonstrably reduces conflict-incident frequency is a social entropy reduction.
3. **Validated trust surveys with falsifiable questions.** Survey instruments where the survey itself can be checked against revealed-preference data (do participants actually behave as they say they trust each other?) provide a falsifiable social measurement.
4. **Mediation outcomes confirmed by all parties.** A mediation that all parties confirm as resolving the underlying issue is a social entropy reduction. The confirmation is itself a validation primitive.

**Failure modes specific to the domain.**

- **Social-desirability bias:** participants report higher trust or cooperation than they actually exhibit. Detected by triangulating survey data against revealed-preference signals.
- **Harmony suppression:** apparent reduction in conflict because dissent has been suppressed rather than resolved. Detected by exit behavior (do participants leave the community at higher rates than baseline) and by structured anonymous feedback.
- **Goodhart pressure:** the social domain has the highest k value in the Signal taxonomy (k approximately 0.45). Any visible social-domain metric will drift faster than any other domain's metric.

**Intersectionality.** Social entropy claims often have governance components (better social trust enables better governance decisions), informational components (trusted networks transmit better information), and economic components (cooperation reduces transaction costs).

**Maturity status.** Low. The social-domain instruments are the protocol's weakest. The protocol's response is to weight social-domain claims with appropriate humility: a DFAO operating heavily in the social domain should expect more retroactive burns, more validator disagreement, and more governance review of its instruments. The framework treats this honestly rather than pretending the social domain is well-instrumented.

### *T.4 Economic Entropy: Deep Dive*

Economic entropy is the disorder in resource allocation, exchange efficiency, waste, scarcity, and matching. The domain is medium-high maturity, with substantial instrumentation but persistent challenges around externality accounting.

**Operationalization paths.**

1. **Utilization improvement.** Asset utilization, capacity utilization, resource utilization deltas measured against operational baselines. A change that demonstrably increases utilization without degrading other variables is an economic entropy reduction.
2. **Throughput improvement.** Throughput per unit input (per dollar, per hour, per kilogram of material). Increased throughput represents more output produced from the same disordered-input flow.
3. **Allocation efficiency.** Under a structured market-clearing or matching test, demonstrated improvement in match quality (better allocations to need, less waste in the matching process) is an economic entropy reduction.
4. **Waste reduction.** Waste measured in domain-native units (kilograms of food, kilowatt-hours of energy, person-hours of labor) demonstrably reduced against a baseline.

**Failure modes specific to the domain.**

- **Rent extraction disguised as efficiency:** an "efficiency improvement" that extracts value from the system rather than creating it. Detected by tracking the distribution of the efficiency gain across stakeholders.
- **Externality dumping:** local economic gain produced by externalizing waste to other domains (thermodynamic, social, environmental). Detected by full cross-domain accounting at the claim's evidence vector.
- **Benchmark gaming:** narrow optimization that improves a specific metric while degrading the underlying economic state. Detected by validator review of the metric's relationship to the underlying state.

**Intersectionality.** Economic claims often have thermodynamic components (energy efficiency, material flow), temporal components (cycle-time, throughput per unit time), informational components (better information improves allocation), and social components (cooperation reduces transaction costs).

**Maturity status.** Medium-high. Economic instruments are well-developed but often fail to include externality accounting. The protocol's eight-domain evidence vector is the structural response: an economic claim that externalizes to thermodynamic or social must record the externality in the evidence vector or face validator pushback.

### *T.5 Thermodynamic Entropy: Deep Dive*

Thermodynamic entropy is the highest-precision reference domain. Physical disorder, energy efficiency, waste heat, material flow, environmental degradation. The domain provides the theoretical floor (Shannon-Landauer-Bennett) for the protocol's cross-domain framework.

**Operationalization paths.**

1. **Energy efficiency improvements.** Kilowatt-hours saved against a baseline, measured by direct energy metering. A retrofit, a process change, or a behavioral change that demonstrably reduces energy use is a thermodynamic entropy reduction.
2. **Joules per kelvin reduction.** Industrial processes that operate with less waste heat, measured against published baselines for the process type.
3. **Material recovery rates.** Recycled, reused, or repaired material against a baseline of disposed material. The recovery is a thermodynamic entropy reduction because the material's low-entropy structured form is preserved rather than dispersed.

4. **Emissions reductions.** Greenhouse gas equivalents, particulate emissions, water-pollutant emissions reduced against a baseline. The emissions are the system's externalized entropy; their reduction is a thermodynamic entropy reduction.
5. **Repair counts.** A repaired object retains its low-entropy form longer, reducing the long-run material flow into disorder.

**Failure modes specific to the domain.**

- **Measurement boundary games:** the "we don't count Scope 3" pattern. A company reports a thermodynamic improvement within its operational boundary while externalizing to upstream or downstream processes. Detected by extended-boundary accounting.
- **Baseline manipulation:** choosing a high-baseline year to make any reduction look favorable. Detected by published-baseline conventions and validator review.
- **Substitution with hidden costs:** replacing one waste stream with another that is less visible. Detected by full material-flow accounting.

**Intersectionality.** Thermodynamic claims often have informational components (better measurement instruments reduce informational entropy alongside thermodynamic), economic components (energy savings translate to cost savings), and social components (community-scale infrastructure reduces both thermodynamic and social entropy).

**Maturity status.** Highest. Thermodynamic measurement is the most mature domain and serves as the reference for cross-domain calibration. The protocol's c_l value for thermodynamic is set at the medium of the eight (1e-4), with cognitive and temporal at the extreme low end (1e-6) and economic at the high end (1e-2). The thermodynamic domain anchors the cross-domain calibration.

### *T.6 Informational Entropy: Deep Dive*

Informational entropy is the disorder in records, data quality, signal-to-noise, archival coherence, and retrieval latency. The domain is high-maturity, drawing on decades of information-retrieval and data-quality research.

**Operationalization paths.**

1. **Error rate reduction in records.** Deduplication, correction, validation of records measured against a structured-quality baseline. A records-improvement contribution that demonstrably reduces error rates is an informational entropy reduction.
2. **Completeness improvement.** Filled-in fields under a structured schema, where the filling-in is correct and the schema is appropriate.
3. **Retrieval latency improvement.** Faster access to relevant information, measured against an established baseline. A search improvement, an index improvement, or a content-organization improvement that reduces retrieval time is an informational entropy reduction.
4. **Signal-to-noise improvement.** Dataset improvements that increase the ratio of meaningful signal to noise. Measured in bits-of-information-per-unit using established information-theoretic instruments.
5. **Knowledge organization.** Restructuring information into more navigable forms (taxonomies, ontologies, semantic networks). The reorganization is an informational entropy reduction if it reduces the cognitive cost of finding relevant information.

**Failure modes specific to the domain.**

- **Noise reduction at the cost of legitimate variation:** smoothing that erases meaningful signal alongside the noise. Detected by post-hoc validation of the smoothed signal against known meaningful events.
- **Completeness padding:** filling in fields with fabricated or low-quality values to improve the completeness metric. Detected by validator review of field-value accuracy.
- **Cache-only optimization:** retrieval latency improvements that mask underlying corruption. Detected by content-validation alongside latency measurement.

**Intersectionality.** Informational claims often have cognitive components (better information enables better understanding), economic components (reduced informational uncertainty improves allocation), and code components (better data infrastructure is a code-domain improvement).

**Maturity status.** High. Informational instruments are mature in the information-retrieval and data-quality literature. The protocol can leverage established tooling.

### *T.7 Governance Entropy: Deep Dive*

Governance entropy is the disorder in decision systems: accountability, legitimacy, rule coherence, participation quality. The domain is low-medium maturity, with established but imperfect instruments.

**Operationalization paths.**

1. **Decision latency reduction.** Time from issue raised to decision rendered, measured against a baseline for the decision type. A process improvement that demonstrably reduces decision latency without sacrificing legitimacy is a governance entropy reduction.
2. **Reversal frequency reduction.** How often decisions get re-litigated, measured over a window. A governance process whose decisions reach more durable consensus is a governance entropy reduction.
3. **Participation quality improvement.** Under a structured measure (informed-participation rate, substantive-contribution rate), demonstrated improvement in participation quality is a governance entropy reduction.
4. **Conflict-resolution outcomes confirmed by all parties.** A governance process that produces durable, confirmed resolutions is a governance entropy reduction.
5. **Rule-coherence improvement.** Detecting and resolving contradictions in the rule set is a governance entropy reduction.

**Failure modes specific to the domain.**

- **Speed at the cost of legitimacy:** "we decided quickly, but nobody trusts the decision." Detected by post-decision participation drop-off and by reversal frequency.
- **Procedural compliance disguised as participation:** the participation metric is satisfied but the substantive engagement is absent. Detected by participation-quality measurement that distinguishes form from substance.
- **Rule simplification that externalizes complexity:** simpler rules that push complexity to enforcement. Detected by enforcement-cost tracking.

**Intersectionality.** Governance claims often have social components (governance shapes and is shaped by trust), informational components (transparent governance reduces informational uncertainty), and temporal components (faster decisions affect operational cadence).

**Maturity status.** Low-medium. The governance instruments are improving but remain less mature than physical or informational. The protocol's DFAO governance layer provides a rich substrate for instrumenting governance contributions: every governance proposal, vote, and outcome is a DAG vertex, enabling retrospective analysis.

### *T.8 Temporal Entropy: Deep Dive*

Temporal entropy is the disorder in time allocation, sequencing, synchronization, and operational cadence. The domain is medium maturity, drawing on operations research and queueing theory.

**Operationalization paths.**

1. **Cycle-time reduction.** Time to complete a defined cycle of work, measured against an established baseline. A process improvement that demonstrably reduces cycle time without degrading other variables is a temporal entropy reduction.
2. **Wait-time reduction.** Time spent waiting (in queues, in handoffs, in idle states). A change that demonstrably reduces wait time is a temporal entropy reduction.
3. **Throughput per unit time.** Output produced per unit time, where the output is measured for quality alongside quantity.
4. **Synchronization improvement.** Alignment of dependent processes, reducing cross-process friction. A scheduling improvement, a coordination protocol, or an automation that improves synchronization is a temporal entropy reduction.
5. **Schedule adherence.** Demonstrated improvement in meeting committed schedules against a baseline.

**Failure modes specific to the domain.**

- **Speedup at the cost of quality:** apparent temporal improvement that degrades the quality of the output. Detected by quality measurement alongside temporal measurement.
- **Batching that improves throughput but degrades responsiveness:** a process change that improves average throughput while making outliers worse. Detected by tail-latency tracking.
- **Schedule compliance through pre-emption of legitimate work:** meeting a deadline by deprioritizing other valuable work. Detected by cross-process tracking.

**Intersectionality.** Temporal claims often have economic components (faster cycle time often translates to lower cost), code components (automation improvements affect both code and temporal), and informational components (better information enables better scheduling).

**Maturity status.** Medium. The temporal instruments are well-developed in operations research but cross-domain temporal accounting is still preliminary. The protocol can leverage queueing theory and operations-research instruments for the in-domain measurement.

## U. Threat Catalog Worked Through

This appendix walks through specific attack scenarios and the protocol's responses, providing concrete illustrations of the security model in section 17.

### *U.1 Scenario: Sybil Attack on Validator Pool*

A determined attacker creates 50 identities through a combination of legitimate OAuth accounts, on-device KYC bypass (using purchased ID documents), and DID generation. The attacker's goal: bias the validator pool for high-value claims toward the attacker's own claims.

**Defense activated.** SignalFlow routing is multi-factor: domain match, reputation, load, accuracy. The attacker's 50 identities start at zero reputation. SignalFlow weights validators by reputation; the attacker's identities therefore have minimal weight in routing.

To accumulate reputation, the attacker must have their identities validate legitimate claims correctly over time. This requires the attacker to actually do correct validation work, which means the attacker is doing the protocol's legitimate work. The attack cost is the work itself.

If the attacker tries to use 50 identities to bias a specific claim, the routing must select multiple of them. The routing's load factor distributes assignments; selecting many of the attacker's identities for one claim is statistically unlikely without coordination that the cartel-threshold analysis detects.

If the attacker tries to use 50 identities to vote in lockstep on validation outcomes, the consensus aggregation's per-validator reputation weighting (combined with the retroactive-burn penalty for falsified consensus) makes the lockstep voting expensive in expectation.

**Residual risk.** The Sybil-by-fake-KYC vector remains. The protocol's defense depends on KYC quality. The KYC quality is governance-tunable per DFAO: a high-stakes DFAO can require trusted-issuer handoff (the strongest of the three KYC options), while a low-stakes DFAO might accept the weakest.

### *U.2 Scenario: Reputation Laundering*

An actor has a poor reputation in one DFAO. They attempt to launder reputation by joining a sympathetic DFAO and accumulating fresh rho there, then using that fresh rho to bias their value calculation in another DFAO.

**Defense activated.** The architectural invariant: reputation does not enter the XP formula. The actor's rho in DFAO B does not multiply their XP for claims in DFAO C. The XP formula does not see rho.

The actor's rho in DFAO B might let them route validators in DFAO B preferentially, but this affects only DFAO-B-internal claims. Their claims in DFAO C are routed by SignalFlow in DFAO C, which looks at the actor's rho in DFAO C (not DFAO B).

CT is portable across DFAOs, but the rho component of CT is per-domain, not per-DFAO. The actor's rho in cognitive-domain claims in DFAO B is recorded against the cognitive-domain reputation; transferring CT to DFAO C does not transfer cognitive-domain rho in a way that affects DFAO-C XP.

**Residual risk.** A coordinated reputation laundering across DFAOs (the actor systematically builds rho in many DFAOs, hoping the cross-DFAO rho aggregation will bias future routing) is theoretically possible but requires sustained legitimate contribution. The "laundering" becomes indistinguishable from legitimate contribution, which is the desired outcome.

### *U.3 Scenario: Validator Cartel*

A group of 8 validators in a 12-validator DFAO conspires to confirm fraudulent claims for a share of the payoff.

**Defense activated.** The retroactive-burn window opens after the loop closes. For 30 days, independent observers (including non-cartel validators and external auditors) can submit falsifying evidence. If the evidence stands, the colluding validators take rho penalties.

The cartel-threshold analysis: at validator population N >= 10 and detection probability p >= 0.3, no cartel strategy is profitable in expectation. The 8-of-12 cartel meets the population requirement (12 > 10). The detection probability depends on the claim's visibility: high-stakes claims have higher p; low-stakes claims have lower p.

For low-stakes claims, the 8-of-12 cartel might be profitable in expectation, but the per-claim payoff is also low. For high-stakes claims, the cartel is unprofitable because p is high.

**Residual risk.** A cartel that targets a specific niche of low-stakes claims that are not subject to outside scrutiny remains theoretically profitable. The protocol's response is to use the epistemology engine's pattern detection: clusters of validators whose judgments correlate suspiciously across many claims are flagged for governance review.

### *U.4 Scenario: Identity Compromise*

An attacker compromises a user's device and steals their private key. The attacker now has the user's DID and can sign actions as the user.

**Defense activated.** The PSLL hash chain makes retroactive mutation detectable: any attempt to modify the user's past PSLL entries invalidates the chain and triggers a divergence with the DAG anchor.

The user can rotate their DID through the documented key rotation flow. The rotation event is signed by both the old key and the new key; the rotation is recorded in the DAG. Once the rotation is anchored, actions signed by the old key are flagged as suspicious.

The user's CT, CAT, IT, DT, EP balances transfer to the new DID through the rotation event. The user retains their accumulated value despite the compromise.

**Residual risk.** During the window between compromise and detection, the attacker can sign actions as the user. The actions are recorded in the DAG and cannot be erased. The protocol's response is to allow corrective vertices: the user can submit a CORRECTION vertex stating that specific past actions were not legitimate; validators can then review and confirm the correction. The correction does not erase the original action but neutralizes its effects.

### *U.5 Scenario: Coordinated XP Inflation*

A coordinated group of contributors agrees to submit and validate each other's claims, generating XP without genuine entropy reduction.

**Defense activated.** SignalFlow routes validators based on multi-factor scoring, not on submitter preference. The colluding submitters cannot choose their own validators; the routing selects validators independently.

The F decay term penalizes repeated claim-type submissions from the same fingerprint. The colluding submitters cannot inflate XP by repeating the same claim type.

The retroactive-burn window allows independent observers to falsify the claims. Independent observers in the DFAO who notice the suspicious pattern (many low-substance claims confirmed) can submit falsifying evidence.

The epistemology engine's pattern detection surfaces clusters: submitters whose claims correlate suspiciously across many loops are flagged for governance review.

**Residual risk.** A small, tight cluster operating in an obscure niche might evade pattern detection for some time. The protocol's response is the gradual accumulation of evidence: if the cluster's claims never produce downstream value (the entropy reduction they claimed does not show up in actual outcomes), the cluster's reputation slowly degrades.

### *U.6 Scenario: Regulatory Compulsion*

A state actor compels the protocol's primary operators to reveal user identities and to restrict specific user actions.

**Defense activated.** The threshold reveal scheme: revealing a user's identity requires 7 of 12 ecosystem-validator shareholders to cooperate. The shareholders are distributed across multiple jurisdictions; compelling all 7 requires either multi-jurisdictional cooperation or international legal process.

User actions cannot be restricted by the protocol's operators: the protocol does not have the capability to deny a user from submitting actions. The actions are submitted from the user's device using the user's private key; the protocol cannot revoke the user's key.

The protocol can refuse to anchor specific PSLL roots or validate specific claims, but the user's PSLL remains on the user's device and can be re-anchored to alternative DAG instances (the gossip-layer roadmap supports multiple DAG instances).

**Residual risk.** The protocol does not provide protection against direct compulsion of the user (the user themselves can be ordered to reveal their actions). The protocol provides only the structural property that no third party can reveal the user without the user's involvement.

### *U.7 Scenario: Cryptographic Breakthrough*

A cryptographic breakthrough (a polynomial-time algorithm for Ed25519 discrete logarithms, for example) is published.

**Defense activated.** Governance approves a new signature scheme as the protocol default. New PSLL entries and DAG submissions use the new scheme. The transition is forward-only: existing signed events retain their original signatures, but the new ones use the updated scheme.

Users whose private keys are at risk under the broken scheme are encouraged to rotate to keys in the new scheme. The rotation events are themselves signed using a multi-signature flow (old key plus new key) until the old scheme can be fully retired.

**Residual risk.** During the transition window, signatures under the old scheme remain forgeable. The protocol's response is to compress the transition window: governance can mandate an accelerated rotation deadline for high-value identities.

For sudden catastrophic breaks (quantum attacks, for example), the protocol's defense is limited. The transition cannot be instantaneous; some level of forged-signature risk during the transition is unavoidable. The protocol does not claim to be quantum-secure under all circumstances; it claims to be cryptographically agile and able to migrate.

### *U.8 The Threat Catalog's Posture*

The threat catalog is not exhaustive. Real adversaries will find vectors the catalog does not anticipate. The protocol's posture is not "we have anticipated every attack" but rather "we have built the structural defenses that bound the impact of attacks, and we are committed to learning from attacks we did not anticipate."

The architectural invariants (reputation does not enter XP; the user is sovereign; the DAG is event-sourced and append-only) provide structural floor against attack: even attacks the catalog does not anticipate are constrained by these invariants. An attack that violates an architectural invariant is detectable; attacks that respect the invariants are limited in damage.