EXTROPY CODEX

Version 2.1 · Comprehensive Edition

F = Frequency of Decay (XP formula).
$\mathscr{F}$ = Falsifiability Index (V(Q) only).

canonical-v3.12 · protocol v0.2

Randall Gossett

August 2026

Self-contained specification. No external repository citations required. The v2.0 assignment of F to falsifiability is retired. There is no F_freq.

Extropy Codex, Version 2.1

Comprehensive Edition A Non-Extractive Contribution Ledger for Verified Entropy Reduction Full Technical Reference

Randall Gossett, with AI-assisted drafting disclosed August 2026

Codex v2.1 · formula canonical-v3.12 · protocol v0.2 · comprehensive edition

This document is self-contained. It is the internal specification of the Extropy Engine. It does not require any companion file, repository browse, or external handbook to be read as a complete protocol reference. Where the reference implementation exists, this Codex states what that implementation must do. Where the implementation currently differs, this Codex is the higher-order statement and the difference is a bug.

Abstract, Reading Guide, and Status

The Extropy Engine is a protocol for measuring and rewarding verified reductions in disorder across eight domains of human and civilizational activity, using a single scalar unit called bits-equivalent (b_e) and a single canonical minting formula. This is the Comprehensive edition of Codex v2.1: a full technical reference that folds in the complete 21-section outline and Appendices A through X, rewritten against the current reference implementation and stripped of every framing the project no longer stands behind.

It ships architectural invariants that were absent, informal, or contradicted in prior drafts:

- Non-extraction of value to external markets.
- Cross-domain measurement normalization with a stated falsifier contract.
- A bounded settlement-time factor with a governance-set floor and a governance-set amplification ceiling α_max.
- An implementation-agnostic protocol contract that separates the specification from the reference codebase.

It formally retires the pseudo-physics form $XP = \Delta S / c\_L^2$ and every framing catalogued in the rejected-framings register. Version 2.1 additionally establishes a strict disambiguation between Frequency of Decay (F) and Falsifiability ($\mathscr{F}$), upgrades the minting formula to canonical-v3.12, aligns temporal mechanics with Universal Times v4.2, and introduces the Local Flow bootstrapping protocol.

What this document is

A single-volume technical reference. It contains the protocol contract, the canonical formula, the eight-domain instrument statements, the validation and identity architecture, the token economy, the DAG substrate, the governance layer, the threat model, the engineering gap catalog, the Universal Times calendar, the Local Flow bootstrap, and the worked appendices. A reader should not need to open a second file to audit a claim in this Codex.

What this document is not

It is not a white paper for fundraising. It is not a token-sale document. It is not a claim that entropy is a metaphysics of everything. It is not a substitute for running the reference implementation. It is not finished: §20 enumerates 65 open engineering gaps.

Reading paths

- Protocol implementer. §3, §4, §6, §7, §8, §11, §15, Appendix Q, Appendix V, Appendix W.
- Governance participant. §9, §10, §13, §14, Appendix E.
- Peer reviewer. §3 through §7, §17, §18, and every Falsifier block.
- Bootstrap / local operator. §5, §19, Appendix G, Appendix L, Appendix P.
- General reader. The opening paragraph of each section, then §1, §2, and §12.

Three registers

Each section opens in plain language. The body is the technical register: formal statements, formulas, invariants, and worked numbers. The third register is falsification. Every section with empirically checkable claims carries a block beginning Falsifier. that states what would have to be true for the section to be wrong.

Methods note

This Codex was drafted by Randall Gossett with AI-assisted drafting disclosed. The assistant read the prior Codex v2.0 Comprehensive edition, the v2.1 revision notes, and the live reference implementation. Where those sources disagreed, this document states the v2.1 position and records the disagreement in the Correction Ledger.

Falsifier. If any of the five commitments in the abstract is contradicted by shipped protocol behavior, this Codex is wrong at the architectural layer, not merely incomplete at the implementation layer.

1. What Problem the Extropy Engine Solves

In plain terms: most systems that claim to measure value actually measure a proxy for value, and proxies get gamed. This section names that failure mode precisely, explains why the usual fixes do not work, and states what the Extropy Engine proposes instead. It also introduces the three-layer separation that keeps the fix from becoming just another gameable proxy.

1.1 The Proxy Worship Problem

Every large coordination system needs a legible signal to allocate attention, reward, and trust. Test scores stand in for learning. Credit scores stand in for creditworthiness. Follower counts stand in for cultural relevance. Market prices stand in for value. Compliance checklists stand in for safety. In each case, the signal was originally correlated with the underlying condition it was meant to track. In each case, once the signal acquired incentive weight, that correlation began to erode. This is not a story about bad actors gaming a good system. It is closer to a physical law of incentive systems. Once a measure becomes a target, the population subject to it begins optimizing for the measure rather than for what the measure was supposed to track. Goodhart’s Law states this generally: any observed statistical regularity tends to collapse once pressure is placed upon it for control purposes. The companion book, Randall Gossett, Unfuck the World for a Dollar (companion book; in progress), spends its opening chapters walking through this pattern across education, credit, social media, and market pricing, arguing that the pattern is mechanical rather than moral: the incentive structure produces the drift regardless of the individual character of the people inside it. The label becomes the target. The fidelity between label and referent decays. This document calls that decay process representational fidelity decay and gives it a formal treatment in §6 and Appendix K, borrowed from an internal Signal paper cited throughout this Codex as the source of the k-parameter taxonomy and the differential equation that governs decay and recovery.

1.2 Why Existing Remedies Do Not Work

The standard remedy for a captured metric is to design a better metric, audit harder, or add a penalty for gaming. All three remedies address the symptom without addressing the mechanism. A better metric is captured on the same timescale as the metric it replaced, because the underlying incentive dynamic has not changed. Harder audits raise the cost of gaming without removing the incentive to game, so gaming migrates to whatever the audit does not cover. Penalties for gaming require detecting the gaming, which is itself an arms race against actors who have every incentive to make gaming look like genuine performance. None of these remedies touches the structural feature that produces the decay: the metric is visible to the population being measured, and the population being measured has an interest in the metric's value independent of the underlying condition. As long as that structural feature holds, better metrics, harder audits, and stronger penalties buy time, not immunity.

1.3 What the Extropy Engine Proposes

The Extropy Engine proposes a different structural fix: separate the visible incentive layer from the measurement layer entirely, and ground the measurement layer in a quantity that is expensive to fake because faking it requires actually producing the underlying condition. That quantity is entropy reduction, measured per domain, under a falsifiable claim structure, verified by a routed population of contributors rather than a dedicated validator class. Operationally, the claim lifecycle runs as follows, restated here in outline and specified in full in §9:

1. A claim is opened with a specified domain, a specified evidence path, a baseline measurement, and a stated falsification condition.
2. The claim closes through a validation lifecycle that surfaces the evidence to a routed pool of contributors performing validation tasks, not to a dedicated validator class (see §10).
3. Consensus on the entropy delta is reached and the closure conditions in §6.4 are satisfied.
4. The minting service writes XP using the canonical formula in §4, with rarity, frequency-of-decay, measured entropy reduction, domain weighting, and the bounded settlement-time factor all multiplied together.
5. A governance-set retroactive window allows new evidence to confirm or burn the provisional mint.

1.4 The Three-Layer Separation

The structural fix above only works if the visible incentive layer and the measurement layer stay genuinely separate, and separation is not the network's default behavior. It has to be engineered and defended. Section 14 specifies the full architecture; the summary here is that the Extropy Engine operates across three layers with different visibility and different currencies.

Layer Visible to Currency Purpose

User-facing Everyone Discounts, savings, gami- Make participation feel fied feedback good and pay off in real terms

Merchant-facing Businesses Better point of sale, cus- Make merchants want in, tomer pipeline, operational without charging them signal SaaS

Engine Contributors performing XP, CT, and access Actually measure entropy validation tasks, sensors, thresholds reduction. Never user-visthe math ible as a score

The hard rule: the user-facing layer never exposes raw XP as a number a participant can target, and the engine layer never gets simplified into a public leaderboard. The gamification on the user-facing layer is a deliberate decoy for user-facing psychology. The real scoring function lives where users cannot farm it directly.

1.5 What This Is Not, Stated Plainly

The Extropy Engine is not a claim that entropy reduction is the only kind of value, or that everything of worth can be reduced to a bit count. It is a narrower and more defensible claim: that a meaningful and growing set of contribution types across eight domains admit an operational measurement of disorder

reduction, that this measurement can be made falsifiable and hard to game relative to the alternatives currently in wide use, and that a coordination protocol built on this measurement is worth building and testing even though it will not capture everything anyone might call valuable. It is also not a cryptocurrency, in the sense that matters most: it has no fiat bridge, no transferable balance, and no market price. Section 3 states this as an architectural invariant, not a policy choice.

Falsifier. If deployed practice shows that the three-layer separation collapses, that is, if raw engine-layer XP becomes visible and farmable at the user-facing layer in any shipped surface, the Goodhart-resistance claim of this section is falsified for that surface. The surface must be corrected or the claim withdrawn for that surface.

2. Core Thesis: Entropy Reduction as Value

In plain terms: this section makes the central claim precise, grounds it in established physics and information theory rather than metaphor, and states the seven questions any serious value-measurement protocol has to answer. It also explains why the canonical formula multiplies its terms instead of adding them, and why every claim in the system carries an explicit falsification condition.

2.1 The Claim, Stated Precisely

The claim is not that “everything is entropy” in some totalizing metaphysical sense. The claim is narrower: verified reductions in disorder, measured per domain under a domain-appropriate instrument, constitute a coherent and falsifiable basis for a value-tracking protocol, and this basis resists the proxycapture dynamic described in §2 better than the alternatives currently in wide use (test scores, credit scores, engagement metrics, market prices used as universal value proxies). This claim inherits its physical grounding from a real mathematical correspondence, not from rhetorical flourish, and the domain-specific epistemic note that governs how far that correspondence is pushed lives in this Codex §4: the term “entropy reduction” is used as a generalized measurement framework, operationalized through domain-specific metrics, rather than as a claim of literal thermodynamic equivalence in every domain.

2.2 The Shannon-Landauer-Bennett Grounding

Claude Shannon’s 1948 paper established that information has a formal entropy measure structurally identical to the Boltzmann-Gibbs entropy of statistical mechanics: both are computed as the negative sum of probability times the log of probability, over the possible states of a system (Shannon, A Mathematical Theory of Communication, Bell System Technical Journal 27, 1948, 379 to 423 and 623 to 656). Rolf Landauer’s 1961 principle showed this correspondence is not merely formal: erasing one bit of information in a physical computing system has an unavoidable minimum thermodynamic cost of k_B T ln 2 joules, where k_B is Boltzmann’s constant and T is the system’s temperature (Landauer, Irreversibility and Heat Generation in the Computing Process, IBM Journal of Research and Development 5, 1961, 183). Charles Bennett’s subsequent work on the thermodynamics of computation extended this into a general theory linking logical and physical irreversibility (Bennett, The Thermodynamics of Computation, International Journal of Theoretical Physics 21, 1982, 905).

That correspondence licenses treating informational-domain entropy reduction and thermododynamic-domain entropy reduction as instances of a single underlying phenomenon, with a real physical exchange rate at the Landauer floor. It

does not, by itself, license treating cognitive, code, social, economic, governance, or temporal disorder as literally thermodynamic. Codex v2.1 in both editions is explicit about this boundary: the six non-physical domains are operationalized through their own measurement instruments (see §8), and the claim that they are commensurable with the physical domains rests on the cross-domain normalization architecture in §6 through §7, not on an extension of Landauer's theorem beyond its actual scope.

2.3 The Seven Questions a Value Protocol Must Answer

Any protocol claiming to measure and reward contribution has to answer seven questions, whether or not it answers them explicitly. Making them explicit is itself a discipline against hand-waving.

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

Every loop opened under this protocol answers all seven questions before it can mint. Question 1 through 3 are answered by the claim payload and the domain's measurement operator M_d (§5, §6). Question 4 is answered by SignalFlow routing to a validator neighborhood (§9, §10). Question 5 is answered by the Falsifiability Index $\mathscr{F}$, which gates validation through V(Q) and never enters the mint formula (§4.2). Frequency of Decay F is a different symbol; it is the second factor in the XP formula (§4.1, §4.4). Question 6 is the closure condition in §6.4. Question 7 is the retroactive validation window in §8.5.

2.4 Why Multiplication

The canonical formula in §4 multiplies its five factors rather than summing them. This is a substantive design choice, not an aesthetic one, and it is worth stating the reasoning independently of the formula's mechanics. Addition treats the factors as independent sources of value that can compensate for each other. Under addition, a claim with high rarity but zero entropy delta could still earn XP, which is incoherent: the system would be minting value against claims that demonstrably reduced no disorder. Addition fails the conjunctive test the protocol requires: rarity, frequency of decay, magnitude, domain relevance, and timely settlement must all be present, not merely one of them in excess. Multiplication enforces necessity. Every factor must be non-zero for XP to mint at all. A claim cannot compensate for missing evidence with high rarity, and a trivial submission cannot reach high value through speed alone. One consequence deserves explicit statement: when any single factor is small, the resulting XP is small regardless of how large the other factors are. A high-magnitude entropy reduction in a low-rarity action class still mints a small amount of XP, because the rarity factor is small. This is intended: the protocol rewards the combination of substance and scarcity, not either alone. Appendix A works through why five factors specifically, and not four or six.

2.5 Why Frequency of Decay, and Why Falsifiability Is Not in the Formula

F in the mint formula is Frequency of Decay. It is the freshness / repetition penalty on an action-class fingerprint. First occurrence of a class is F = 1. Repeated instances of the same class decay. That is the whole job of F. It answers: has this move already been done, and how often?

Falsifiability is a different quantity. Its symbol is $\mathscr{F}$, the Falsifiability Index. $\mathscr{F}$ lives in the Epistemology Engine as an input to the validation score V(Q). A claim must state, in advance, what evidence would prove it wrong. A claim with no

falsification condition cannot be meaningfully contested, which means it cannot be meaningfully validated either. The protocol enforces that by rejecting claims with $\mathscr{F}$ below the protocol floor (default 0.50) before slicing. $\mathscr{F}$ does not multiply into XP. Putting $\mathscr{F}$ inside the mint formula would let a highly testable empty claim mint, and would let an untestable claim sneak through if the other factors were large. Validation gates minting. The mint formula does not re-encode the gate.

This is the correction Codex v2.1 exists to lock. Codex v2.0 used F for falsifiability. That was the drift. Intermediate drafts then invented a second symbol, written F_freq, for frequency of decay and kept F as falsifiability, which inverted the intended assignment and recreated the collision under a new name. Both of those assignments are retired. The canonical pair is:

- F — Frequency of Decay. Lives in the XP formula.
- $\mathscr{F}$ — Falsifiability Index. Lives in V(Q). Never in XP.

Falsifier. If a substantial population of high-XP claims in production turn out, on inspection, to have carried no meaningful falsification condition, the $\mathscr{F}$ gate has failed operationally and V(Q) must be revisited. If F in production does not decay with confirmed class-repetition, the frequency-of-decay term has failed and F's role in the formula must be revisited. Those are two different failure modes. They must not be collapsed into one symbol.

3. Canonical Terminology and Correction Ledger

In plain terms: this section fixes the vocabulary that the rest of the Codex depends on, states which symbols mean what and nowhere else, and documents where the project's own past drafts got the terminology wrong. Getting this section right matters more than it looks: most of the disputes about this protocol, internally and externally, have turned out to be disputes about what a symbol meant, not disputes about the underlying architecture.

3.1 The Architectural Invariant on Naming

This document and every document written for this project uses canonical variable names with exactly one meaning each. No symbol reuse. No ambiguity. If a future contributor needs a new symbol, they pick a fresh letter or coin a new term; they do not overload an existing one. This rule is stated as a hard invariant in §8 of this Codex and is treated with the same seriousness as the mathematical invariants themselves, because naming collisions are exactly the kind of drift that lets a captured concept re-enter through the back door.

3.2 Canonical Symbols

Symbol Meaning Lives in

R Rarity. Action-class scarcity per XP formula domain. A property of the contribution, not the actor.

F Frequency of Decay. Action-class freshness / repetition penalty in the XP formula. First occurrence of a class is F = 1. Repeated instances decay. See §4.4.

$\mathscr{F}$ Falsifiability Index. Epistemic index in [0, 1] used by V(Q) to gate validation. Never a factor in the XP formula. See §4.2.

ΔS Entropy delta. Positive magnitude, XP formula, domain instruments disorder reduced, in raw domain units before normalization.

ΔS_b_e Entropy delta after conversion to bits- XP formula equivalent by a domain's measurement operator M_d.

w·E Eight-domain effort weighting vector, XP formula dot product of the governance-set weight vector w and the claim's measured domain vector E.

T_s Normalized, clamped settlement-time XP formula factor. T_s = exp(negative λ_d times Δt), clamped into the interval from T_floor (exclusive) to 1 (inclusive).

T_floor Governance-set settlement-time floor. XP formula Default 0.01.

λ_d Per-domain settlement-decay con- XP formula stant.

ε_d Domain-specific tolerance on ΔS Closure conditions measurement across validators.

Q_d Domain-specific quorum function. Closure conditions

C Capability, in the CT formula only. CT formula

ρ Reputation density, in the CT for the CT formula, Non-Extraction and separately, the contribution and draw ratio in the Non-Extraction architecture. Context disambiguates; see §13.3 and §3 of the Concise edition for the ratio usage.

Δ Entropy delta as it enters the CT CT formula formula specifically.

E Eight-domain weighting vector, in the XP formula, CT formula, evidence CT formula context, and separately classes the domain vector in the XP formula,

Symbol Meaning Lives in and separately the evidence-class labels E1 through E3 in §8.2. Three distinct uses of the letter E survive in this Codex because all three are already load-bearing in the reference implementation; each use is disambiguated by context on first appearance in each section.

L Local loyalty multiplier, in the EP EP formula formula only.

τ_action_d, τ_validator_d Access thresholds for domain-d Non-Extraction, validation actions and for participating in a domain-d validator neighborhood.

ρ_min_d Domain-specific floor on the contri- Non-Extraction bution and draw ratio ρ. Below this, access degrades automatically.

b_e Bits-equivalent. The single common Normalization unit for cross-domain ΔS.

M_d Domain-specific measurement oper- Normalization ator that converts raw evidence to ΔS_b_e.

3.3 Canonical Tokens

The protocol recognizes exactly six tokens. No others. XP (Extropy Points). Non-transferable access threshold. Minted from verified ΔS_b_e under the canonical formula. Never a balance to be extracted, never redeemable for fiat. CT (Contribution Token). A per-actor structural coefficient that shapes access weight in ties and vote weight in validation. Non-transferable. Not spent by voting. CAT (Capability Token). The portable, cross-application carrier of skill certification and standing. CAT level in a given domain is recognized network-wide without further negotiation, and it progresses on a logarithmic scale keyed to confirmed contribution counts, provisionally at 10, 30, 90, and 270 confirmed contributions per domain (see §8 of this Codex , packages/contracts CAT_LEVEL_THRESHOLDS).

IT (Influence Token). Non-transferable governance-weight token. Decays at a provisional rate of five percent per month. Functions as anti-capture pressure against permanent influence accumulation. DT (Domain Token). Subject-matter expertise token, domain-scoped, used inside the token-economy package; see §13. EP (Emergence Points). The token that actually settles at the merchant-facing layer, computed as a function of XP and the local loyalty multiplier L. EP is the only token in the set with anything resembling a redemption surface, and that surface is bounded

to intra-network merchant discounts, never to fiat or to a transferable market instrument. See §13.5 and Appendix S for the non-extraction boundary this respects.

This is a reduction from the eight-token vocabulary that appeared transiently in earlier internal drafts (which briefly proposed GT and RT as well). Those two names were retired in v3.1.2 and are not part of the canonical set. §1.4 and Appendix M of this Codex and §8 of this Codex both state the six-token rule as a hard invariant: exactly six, no more, no legacy names.

3.4 Canonical Domains

Two domain vocabularies appear in the repository's history, and this Codex reconciles them by canonizing one and flagging the other's drift as an open engineering item. The canonical eight, adopted here in both editions, are the domains named in the EntropyDomain enum in packages/contracts/src/types.ts and in this Codex §5: cognitive, code, social, economic, thermodynamic, informational, governance, and temporal. This is the set every service in the reference codebase actually switches over, it is the set the causal-closure speed table CAUSAL_CLOSURE_SPEEDS is keyed on, and it is the set the contribution graph's rarity tables and routing already use. It also carries no reserved slot with no measurement operator. §4.6 of this Codex currently names a slightly different eight-domain set that swaps code for ecological and temporal for a reserved spiritual slot. That drift is real, it is out of sync with the enum the code actually uses, and it MUST be closed by aligning §4.6 of this Codex to the enum rather than the other way around. This is filed as §20 of this Codex naming-hygiene P2, "domain-set reconciliation between NORMALIZATION.md and EntropyDomain enum", and it is a precondition for the F1 falsification condition being operationalizable. Where earlier drafts of this Codex, and any published or narrated versions produced before this correction, described the mint-side domain set as including ecological or a reserved spiritual slot, those references are superseded by the canonical eight above. The physical-restoration measurement work that would have lived under an ecological label is minted as thermodynamic entropy reduction, grounded in the Landauer correspondence and in direct energetic or material measurement.

3.5 Why the Reconciliation Lands This Way

The Codex canonizes the enum set rather than the NORMALIZATION.md set for three reasons. First, the enum is what executable code branches on, and a specification that names domains the code does not enumerate cannot be tested end to end. Second, the temporal domain has a real reference implementation surface in packages/temporal/ and a defensible measurement anchor in scheduling and settlement-time predictability, whereas a reserved spiritual slot has no operator at all. Third, code entropy reduction is one of the two domains with the most mature evidence surface in the reference implementation (through packages/github-parasite), and demoting it to a sub-instance of informational entropy hides that maturity behind a naming convention that no other part of the codebase honors.

3.6 The Correction Ledger

This Codex, in both editions, corrects four architectural facts relative to Codex v1.0, and states here, once, what changed and why. Later sections restate the corrected form without re-litigating the history each time.

Correction 1: Non-extraction is now an architectural invariant, not an open regulatory question. v1.0 treated whether XP or CT could bridge to fiat as an open question to be resolved later by legal review. This Codex closes it: the protocol has no fiat bridge, no CT-to-fiat surface, no transferable wrapper, and no prediction market over loop outcomes, at any layer, ever. See §5 (Concise edition numbering) and the full statement in docs/NON_EXTRACTION.md , restated in this Codex's §13. Correction 2: Cross-domain ΔS is now a single common unit with a stated falsifier contract. v1.0 spoke of ΔS in "bits" across eight domains as though the shared unit name alone made the values comparable. This Codex introduces bits-equivalent (b_e), a per-domain measurement operator M_d, five invariants any M_d must satisfy,

and four explicit falsification conditions (F1 through F4) under which the whole project is falsified as a coordination protocol. See §6 and §7. Correction 3: The settlement-time factor is now bounded. v1.0 published the XP formula with a raw log(1/T_s) term applied to unnormalized wall-clock seconds. In the reference implementation through v3.1.2, this produced two simultaneous bugs: sub-second closures diverged the log term toward infinity (speed-farming), and any realistic closure over one second produced a negative log that was silently clamped to zero by a fallback branch that itself resorted to the now-retired irreducible-XP framing. This Codex publishes the corrected formula: T_s is normalized as exp(negative λ times Δt) and clamped into the interval from T_floor (exclusive) to 1 (inclusive), and the log-decay term is capped at log(1 divided by T_floor). Default T_floor of 0.01 yields a hard cap near 4.605. The bug and the fix are documented in the Correction Ledger in §3 under the v3.1.3 entry, and the formula is implemented in packages/xp-formula/ src/index.ts , stamped canonical-v3.12 . See §5.

Correction 4: Specification and reference implementation are now separated. v1.0 was ambiguous about whether the TypeScript reference implementation was the specification or one realization of it. This Codex draws the line explicitly: the specification is the Protocol Contract in this Codex plus this Codex; any implementation in any language satisfying the protocol contract is Extropy-conformant. See §15 and §18 of the Concise edition, and §15 of this document. Beyond these four architectural corrections, docs/REJECTED_FRAMINGS.md enumerates specific framings that are formally retired or reserved for retirement. Per this document's editorial policy, those framings are cited, not reproduced. The most consequential is R1, the retirement of the pseudo-physics irreducible-XP floor that divided an entropy delta by the square of a per-domain causal-closure constant; see §4.8 for the replacement mechanism and docs/REJECTED_FRAMINGS.md for the full retirement reasoning.

3.7 Correction 5. Frequency of Decay (F) and Falsifiability ($\mathscr{F}$) are different symbols with different homes. F lives in the XP formula. $\mathscr{F}$ lives in V(Q). Codex v2.0 used F for falsifiability. That collision is retired.

Correction 6. Temporal mechanics are aligned with Universal Times v4.2. Δt and F may be expressed in seconds or in quants. The unit is recorded on the mint event.

Naming Hygiene Going Forward The F collision is closed in this edition, not tracked as open work. Codex v2.0 used F for falsifiability. Some intermediate drafts then kept that assignment and coined F_freq for frequency of decay. That inversion is the opposite of the intended correction and is retired here. This Codex assigns F to Frequency of Decay and $\mathscr{F}$ to Falsifiability, everywhere. There is no F_freq. The reference implementation MUST be reconciled to this pair. Any remaining field named F that still means falsifiability is a bug; rename it to script-F / falsifiabilityIndex / $\mathscr{F}$. Any remaining field named F_freq is a bug; rename it to F.

The domain-set collision described in §4.4 is tracked similarly: the contribution-style eight and the normalization eight overlap but are not identical, and full reconciliation into one canonical list is future governance work, not yet closed.

Falsifier. If any future document in this repository introduces a new meaning for an alreadyassigned symbol without renaming it, the naming-hygiene invariant has been violated and the offending document must be corrected before merge.

4. Mathematical Foundation and Canonical Formula (canonical-v3.12)

In plain terms: this is the formula section. Everything else in this Codex either feeds an input into this formula or governs what happens to its output. Read this section carefully even if you skip everything else. Most disputes about the protocol's soundness are really disputes about one of the five terms below.

4.1 The canonical XP formula

The canonical formula, at version canonical-v3.12:

```
XP = R × F × ΔS_b■ × (w · E) × min(α_max, max(1, T■ / T_floor))
```

```
T■ = exp(-λ_d · Δt) Δt in seconds or quants (τ_H), λ_d per-domain
```

$T_s \in$ (T_floor, 1] by construction

The formula multiplies its five factors rather than summing them. Addition treats the factors as independent sources of value that can compensate for each other. Under addition, a claim with high rarity but zero entropy delta could still earn XP, which is incoherent. Multiplication enforces necessity. Every factor must be non-zero for XP to mint at all.

The formula-version string `canonical-v3.12` is stamped onto every mint event. A mismatch between the stamped version and the executed math is a critical bug that MUST fail closed. The Protocol Contract (Appendix V) requires every mint path to import a single implementation of this formula. Reimplementation in a second package is a protocol violation.

Symbol	Range	Meaning
R	[0.1, 10.0]	Rarity. Action-class scarcity per domain. A property of the contribution, not the actor. Governance-tunable.
F	(0, 1]	Frequency of Decay. Temporal rate parameter governing structural / fidelity entropy decay for the action class. Measured in s^{-1} or τ_H^{-1}. First occurrence of a class is F = 1. Repeated instances decay.
ΔS_b_e	(0, cap_d]	Entropy reduction in bits-equivalent after conversion by the domain measurement operator M_d. Bounded by a per-domain per-event cap.
w · E	(0, 8]	Dot product of the governance-set eight-domain weight vector w and the claim's measured domain vector E.
T_s	(T_floor, 1]	Normalized settlement-time factor. $T_s = \exp(-\lambda\_d \cdot \Delta t)$.
T_floor	default 0.01	Governance-set settlement-time floor. Prevents the settlement term from exploding as $\Delta t \to 0$ is not the issue; it prevents the inverse from exploding as settlement becomes arbitrarily slow.
α_max	default 4.6	Governance-set settlement-time amplification ceiling. Preserves the numerical bound that $\log(1/T\_floor) \approx 4.605$ at the default floor, so v2.1 does not re-open the speed-farming hole that the unbounded $\log(1/T_s)$ created.

Worked range of the fifth term. Let T_floor = 0.01 and α_max = 4.6.

- Instant honest settlement, $T_s \to 1$: T_s / T_floor = 100; max(1, 100) = 100; min(4.6, 100) = 4.6.
- Slow settlement at the floor, $T_s \to 0.01$: T_s / T_floor = 1; max(1, 1) = 1; min(4.6, 1) = 1.
- Mid settlement, $T_s = 0.046$: T_s / T_floor = 4.6; the fifth term equals 4.6, the ceiling.

The fifth term therefore lives in [1, α_max]. Faster honest settlement is rewarded. Speed-farming cannot push the term past α_max. Slow settlement cannot push it below 1, so a late-but-real claim still mints if the other four factors are alive.

4.2 What the formula does not contain

Reputation is not in the mint formula. Reputation is a property of the actor over time. It lives in CT and in the actor's per-domain reputation vector. It is never a multiplier on new XP mints. If R could be inflated by reputation, reputation itself would compound and produce runaway concentration, which is the failure mode the architecture exists to prevent.

Falsifiability is not in the mint formula. Falsifiability is $\mathscr{F}$, an epistemic index in [0, 1], and it lives in the Epistemology Engine as an input to the validation score V(Q). A claim with $\mathscr{F}$ below the protocol floor (default 0.50) is rejected before slicing. Putting $\mathscr{F}$ inside XP would let a highly testable but empty claim mint, and would let an untestable claim sneak through if the other factors were large. Validation gates minting. The mint formula does not re-encode the gate.

4.3 R: rarity in detail

R is a property of the action class, not the actor. Repairing a unique water main in a specific neighborhood is rarer than filing a duplicate status report. Governance sets R per class inside a domain. R may not be derived from the contributor's XP, CT, IT, or follower count. Laundering reputation through R is a protocol violation.

4.4 F: frequency of decay in detail

F is the Frequency of Decay. It is not Falsifiability. Codex v2.0 used F for falsifiability. That was the drift. Intermediate drafts then coined a second symbol, written F_freq, for class-repetition decay and kept F as falsifiability, which inverted the intended assignment. Both are retired. Codex v2.1 assigns:

- F — Frequency of Decay, in the XP formula.
- $\mathscr{F}$ — Falsifiability Index, in V(Q).

F is computed against the action-class fingerprint, not the actor. The default schedule is log-tail decay:

```
F(n) = 1 / (1 + ln(n + 1))
```

where n is the number of prior confirmed mints of the same class fingerprint inside the relevant window. $n = 0 \Rightarrow F = 1$. This prevents action-class grinding without zeroing honest repetition.

F may also be expressed natively in quants^{-1} (τ_H^{-1}) when the temporal service is the clock. See §5.

4.5 ΔS: entropy reduction magnitude

ΔS_b_e is the entropy delta after the domain operator M_d converts a domain-native measurement into bits-equivalent. A claim with $\Delta S\_b_e \leq 0$ MUST NOT mint. Each domain states its operator, its baseline requirement, and its falsifier in §6. Cross-domain comparison is legal only in b_e, and only under the falsifier contract in §4.6.

4.6 Cross-domain normalization

One b_e in the thermodynamic domain is anchored to Landauer's principle: $k_B T \ln 2$ joules of minimum dissipation avoided. One b_e in the informational domain is one bit of Shannon surprise removed relative to the pre-verification distribution. One b_e in the cognitive domain is one bit of Shannon entropy removed from a stated belief distribution. The other five domains state their own anchors in §6.

The falsifier for the common unit: if two domain operators, applied to the same underlying event, produce ΔS_b_e values that cannot be brought into agreement by any declared calibration, the common-unit claim is false and those operators must be revised or withdrawn.

4.7 w · E: the eight-domain weighting

w is a governance-set weight vector of length 8, one slot per canonical domain, non-negative, typically summing to 1. E is the claim's measured domain vector, each component in [0, 1] or a domain-native magnitude already converted toward b_e. A claim may be live in several domains at once. The bicycle-repair worked example in Appendix B is thermodynamic + informational + economic.

The Epistemology Engine may dampen w (the w-throttle) when it detects anomalous mint spikes in an easily gamed domain. That dampening is a bootstrap and Goodhart-resistance mechanism, not a hidden reputation multiplier. See §19.

4.8 T_s: the bounded settlement-time factor

```
   T■ = exp(-λ_d · Δt). λ_d is per-domain. Δt is elapsed time from evidence-ready to
closure, in seconds or in quants. T■ is clamped into (T_floor, 1]. The mint term is then
             min(α_max, max(1, T■ / T_floor)), not an unbounded logarithm.
```

Codex v2.0 used min(log(1/T_s), log(1/T_floor)). That form is retired. It bounded the old log-decay term, which was the right instinct, but it still spoke the language of the unbounded log(1/T_s) that the reference package still shipped. Canonical-v3.12 replaces the log with a ratio against the floor, then clamps with α_max. Same bound, clearer construction, no log singularity as $T_s \to 0$.

4.9 The CT and EP formulas

```
CT = C × F × ρ × Δ × E
```

CT is a per-actor structural coefficient. C is a base coefficient, F is Frequency of Decay on the actor's relevant class history, ρ is the actor's standing / accuracy term (this is where reputation lives), Δ is a documented structural delta, and E is effort. CT is not transferable into fiat.

```
EP = XP × L
```

EP (Emergence Points) is the merchant-facing settlement unit. L is a local loyalty multiplier. EP is not a bridge to an external market. Non-extraction still holds.

4.10 Decay, friction, and lockup

IT decays at a provisional 5% per month. CT lockup is 14 days. These are governance knobs, listed in §14.4. They are not mint-formula terms.

4.11 Dimensional analysis

XP is a dimensionless access threshold derived from a product of dimensionless or already-normalized factors. ΔS_b_e is the only factor with a physical interpretation (bits). The other factors are weights, rates, and clamps. This Codex does not claim that XP is a physical quantity in the sense of joules or kelvin.

Falsifier. If any shipped mint path executes math that differs from the version stamped on the mint event, or if any path reintroduces unbounded log(1/T_s), or if F is computed from actor reputation rather than action-class frequency, canonical-v3.12 is violated.

5. Universal Times v4.2 and Temporal Anchoring

Current civil timekeeping is dominated by two intertwined constructs: the Gregorian calendar and SI-based clock time. This arrangement is functional but not coherent. It mixes astronomical cycles, historical compromises, and machine-unfriendly arithmetic. Universal Times addresses the engineering problem: a coherent, machine-native, physics-grounded coordination layer for a protocol that must timestamp events across devices, domains, and decades.

This section is not a proposal to replace civil time for ordinary life. It is the temporal substrate of the Extropy Engine. Frequency of Decay (F) is natively expressible in quants^{-1}. Settlement-time Δt may be counted in seconds or in quants. Every DAG vertex carries a quant timestamp.

5.1 The physical anchor

A quant (τ_H) is one period of the radiation corresponding to the transition between the two hyperfine levels of the ground state of the hydrogen-1 (protium) atom. The adopted reference frequency is

$\nu_H = 1{,}420{,}405{,}751.768$ Hz

The period is approximately 7.04×10^{-10} seconds. This is the same physical transition that defines the hydrogen line used in radio astronomy. It is not a metaphor.

5.2 Three vocabularies, three jobs

Every prior timekeeping system uses one vocabulary for two different things: position and duration. Universal Times gives each function its own vocabulary.

System 1 — Solar Clock (Loop / Arc / Tick). Answers: what time is it here, on this Earth day?

- 1 day = 100,000 ticks (TPD)
- 10 loops in a day
- 100 arcs in a loop
- 100 ticks in an arc
- 1 tick = 0.864 SI seconds
- 1 arc = 86.4 SI seconds
- 1 loop = 8,640 SI seconds (2.4 hours)

System 2 — Universal Duration (Pulse / Wave / Tide / Spin / Current / Season / Orbit / Epoch / Era / Age / Eon). Answers: how long will this take? These units are powers of ten in the hydrogen-frequency counting frame, so they are machine-native and calendar-agnostic.

System 3 — Universal Position. The Quant Accumulator Q. Frame-qualified absolute coordinates anchored to the cosmological epoch (the Big Bang), calibrated via the cosmic microwave background age adopted by the implementation:

$BB\_SEC = 4.350639312 \times 10^{17}$ seconds

A now-snapshot reports both the solar triple (loop, arc, tick) and the duration stack (eon down to GQ), plus a base-10 calendar reading.

5.3 The base-10 calendar

Universal Times uses a 10-month year. Months 1–9 have 40 days. Month 10 has 5 days, or 6 in a leap year. Leap years follow the Gregorian rule, because the engine still has to close loops on Earth. Day-of-year maps onto (month, day) by walking those lengths. This is the calendar the temporal domain instruments in §6.9 may anchor to.

5.4 Constants as implemented

The temporal service is required to use these constants, identical to the public reference clock:

Name	Value	Role
HF	1,420,405,751.768	Hydrogen hyperfine frequency, Hz
TPD	100,000	Ticks per solar day
LOOPS	10	Loops per solar day
ARCS	100	Arcs per loop
TPA	100	Ticks per arc
EDS	86,400	SI seconds per Earth day
BB_SEC	4.350639312e17	Seconds from cosmological epoch to Unix epoch
TROPICAL_SEC	31,556,925.216	Tropical year in seconds
YEAR0_UNIX	–62,167,219,200	Unix time of year 0

Duration names, in increasing scale: GQ, Wave, Tide, Spin, Current, Season, Orbit, Epoch, Era, Age, Eon. Their SI lengths are 10^e / HF for exponents [9, 11, 13, 14, 15, 16, 17, 18, 20, 22, 24].

5.5 Relation to the mint formula

F, the Frequency of Decay, may be computed in s^{-1} or in τ_H^{-1}. When two nodes disagree about wall-clock UTC they can still agree about elapsed quants if they share Q. Δt in $T_s = \exp(-\lambda\_d \cdot \Delta t)$ is legal in either unit so long as λ_d is expressed in the reciprocal unit. A mint event MUST record which unit was used.

5.6 Why this is in the Codex rather than a companion page

Codex v2.0 treated Universal Times as Appendix P and pointed at an HTML reference. Codex v2.1 promotes it to a numbered section because F is now a first-class formula term with a native quant unit, and because Local Flow (§19) needs a shared clock that does not smuggle a single jurisdiction's civil time into every neighborhood.

Falsifier. If two correct implementations of these constants, given the same UTC instant, produce different (loop, arc, tick) triples or different Q, the temporal layer is not a protocol. If F is computed in a unit that is not recorded on the mint event, canonical-v3.12 is incomplete at the clock boundary.

6. The Eight Domains and Their Instruments

In plain terms: this section is the domain-by-domain instruction manual. For each of the eight canonical domains, it states what disorder means concretely, what evidence a claim needs, what instrument measures the reduction, and how mature that instrument currently is. Skim the domains you do not work in; read closely the one you do.

6.1 The Canonical Eight

As established in §4.4, this Codex treats the EntropyDomain enum set as canonical for mint-side purposes: cognitive, code, social, economic, thermodynamic, informational, governance, and temporal. Every subsection below states, for its domain, what disorder means, how the measurement operator is anchored, and what evidence a claim needs.

6.2 Cognitive

Disorder in belief states, knowledge, understanding, mental models, skill formation, and conceptual coherence. Typical contributions include teaching a concept clearly, correcting a misconception, building a curriculum, and producing documentation that improves comprehension. The measurement operator computes the Shannon entropy of the agent's belief state, before minus after, anchored to one b_e of cognitive entropy reduction meaning removing one bit of belief-state uncertainty in a verifiable epistemic agent, such as a test scored, a model's calibration improved, or a misconception corrected in a way validators can inspect.

6.3 Code

Disorder in software systems, architecture, maintainability, correctness, and operational clarity. Typical contributions include fixing a bug, refactoring a brittle module, increasing test coverage against a documented gap, and reducing cyclomatic complexity while preserving behavior. The measurement operator computes the Shannon entropy of the software state as evidenced by tests, review, and reproducible builds, before minus after, anchored to one b_e of code entropy reduction meaning resolving one bit of software-state uncertainty. Typical instruments include cyclomatic complexity delta, test passrate change against a specified suite, error-frequency reduction, and static-analysis score deltas. This domain has the most mature evidence surface in the reference implementation through packages/github-parasite ; see §15.

6.4 Social

Disorder in trust networks, cooperation, conflict dynamics, and community coherence. Typical contributions include mediation, trust restoration, organizing a fractured group, and reducing communication breakdown. The measurement operator computes the Shannon entropy of the assignment distribution over N agents, before minus after, anchored to the idea that one b_e of social entropy reduction corresponds to removing one round of who-does-what ambiguity in a coordination game.

6.5 Economic

Disorder in allocation, throughput, matching, waste, bottlenecks, and coordination of scarce resources. Typical contributions include better matching supply to need, removing a useless intermediate step, improving workflow efficiency, and reducing idle capacity. The measurement operator computes the Shannon entropy of the allocation or price-discovery distribution, before minus after, anchored to one b_e of economic entropy reduction meaning resolving one bit of allocation-outcome uncertainty in a market or matching. This is never denominated in fiat; see the Non-Extraction invariant in §13.

6.6 Thermodynamic

Physical disorder: waste heat, physical inefficiency, environmental degradation, energy loss, and material waste. Typical contributions include improving insulation, recycling systems, waste reduction in physical production, and physical restoration work. The measurement operator computes the Shannon entropy of the phase-space distribution before minus after, expressed in bits, anchored to Landauer's principle: 1 b_e of thermodynamic entropy erased corresponds to k_B T ln 2 joules of minimum dissipation avoided. Typical instruments include energy-use deltas, heat-loss reduction measurements, material recovery rates, and emissions changes. Physical-restoration work whose ΔS is grounded in measured energetic or material change is minted here rather than in a separate ecological domain (see §4.4 for the reconciliation).

6.7 Informational

Disorder in records, data quality, accessibility, signal-to-noise ratio, and archival coherence. Typical contributions include cleaning a dataset, organizing records, fact-checking and source reconciliation, and improving discoverability. The measurement operator computes the Shannon entropy of the document, channel, or dataset representation, before minus after, anchored to raw Shannon bits with H_0 equal to 1. For prediction-loop domains, where the mint is contingent on a specific claim being verified, such as a proof or a dataset reconciliation, the operator instead uses the log-likelihood-ratio of the claim under the pre-verification distribution.

6.8 Governance

Disorder in decision systems, accountability structures, legitimacy, responsiveness, and rule coherence. Typical contributions include fixing a broken decision process, making accountability enforceable, reducing policy contradiction, and improving transparency. The measurement operator computes the Shannon entropy of the decision distribution under a codified rule set, before minus after, anchored to one b_e of governance entropy reduction meaning removing one bit of decision uncertainty under that rule set. Typical instruments include decision latency, reversal frequency, participation quality, and auditability metrics.

6.9 Temporal

Disorder in scheduling, sequencing, coordination of events over time, and settlement-time predictability. Typical contributions include reducing cycle time in a repeated process, cutting wait time within a scheduled system, resolving a scheduling conflict, and improving the predictability of when a coordinated event will land. The measurement operator computes the Shannon entropy of the when-does-what-happen distribution within a bounded coordination context, before minus after, anchored to one b_e of temporal entropy reduction meaning resolving one bit of that scheduling uncertainty. Typical instruments include cycle-time delta, wait-time delta, scheduling-conflict-frequency reduction, and settlement-time-variance reduction. The reference implementation surface for this domain lives in packages/temporal/ , and Appendix P describes the Base-10 Universal Times calendar that anchors many temporal-domain instruments.

6.10 Intersectionality and Domain Vectors

Real contributions land across multiple domains at once. A teacher may reduce cognitive, social, and temporal disorder in a single act. A clean software deployment may reduce code, informational, and economic disorder simultaneously. The engine does not force a claim into one exclusive box; it measures a domain vector E across all eight dimensions and weights it contextually through the governance-set vector w, as specified in §5.5.

6.11 Why Not More Domains

Adding a ninth domain requires demonstrating three things: that the proposed domain admits a measurement instrument grounded in either the Landauer correspondence or a defensible Shannonentropy analogue; that the proposed domain's measurement cannot already be expressed under one of the existing eight without loss; and that the proposed domain has a falsifiability condition definable in advance. No domain has cleared all three bars beyond the current eight.

6.12 Operationalization Status

As of this writing, no domain has a fully mature, governance-adopted calibration table with production-grade validator-neighborhood recipes. The near-term roadmap (see §19) prioritizes landing at least three domain operators with calibration tables and validator-neighborhood recipes, with the code and informational domains first because they map most directly onto the reference implementation's current evidence surface through packages/github-parasite . This is stated plainly rather than implied, because the gap between "a domain is named in this Codex" and "a domain has a working, adversarially tested M_d" is exactly the gap the falsification conditions in §7 (Concise edition, restated as §6.2

of this Codex below) are designed to surface honestly.

Falsifier. If a specific domain's measurement operator, once specified in production detail, cannot be shown to satisfy the five invariants in §7 (non-negativity, boundedness, verifiability, determinism given evidence, composability), that domain must be marked inactive at the mint layer per the F1 falsification condition, and this section must be corrected to reflect that status.

7. Validation Model

In plain terms: this section explains who checks a claim, and the answer is deliberately not "a dedicated validator class." Validation in this protocol is an emergent property of ordinary contributors doing ordinary tasks, some of which happen to check other people's work, often without the person doing the checking realizing it. This design choice removes an entire category of attack that every prior reputation system has had to fight forever.

7.0 Disambiguated peer validation score

There is no dedicated validator class. Validation is a kind of contribution. In Codex v2.1 the validation score is:

```
V(Q) = ■(Q) × C_slice × (1 − e^{−λ N_val})
```

$\mathscr{F}$(Q) is the Falsifiability Index in [0, 1]. C_slice is the consensus ratio across blind one-tenth slices. N_val is the number of independent neighborhood validators. λ is a sensitivity factor. A claim with $\mathscr{F}$ below 0.50 is rejected before slicing. V(Q) does not multiply XP.

7.1 There Is No Validator Class

There are only contributors performing entropy-reducing tasks. Some of those tasks happen to validate other tasks, and the person performing them often does not know it. This is a load-bearing clarification, stated as a hard rule in §7 of this Codex , because the words "validator" and "validation pipeline" appear throughout this Codex and the reference implementation, and it is easy to picture a separate class of people whose job is to sit in judgment of contributions. That picture is wrong, and it imports exactly the failure mode the protocol exists to remove. The intuitive model of any review system is two tiers: contributors do the work, validators check the work. That split creates a privileged class. Validators become a chokepoint, a target for capture, and a source of their own information entropy, because now the validators themselves need validating. Every system that builds a dedicated review tier eventually has to answer who watches the watchers, and the answer is always another tier, which regenerates the same problem one level up. The Extropy Engine does not have that split.

7.2 What Actually Happens

Validation is just another contribution task. It reduces disorder about the state of a claim: before the task, the claim's correctness is uncertain, high entropy; after it, the claim is confirmed or contradicted, lower entropy. That is entropy reduction by the same definition the whole protocol runs on. It mints XP the same way, decays the same way, and settles retroactively the same way as any other contribution. Because validation is a task, it is performed by contributors, not by a separate population: the same person who writes code on Monday scores a blind slice on Tuesday and, on Wednesday, completes a quest whose output silently confirms or contradicts a third party's earlier claim. None of these three activities carries a special validator badge.

7.3 SignalFlow Routing

Each quest, including validation tasks, is routed based on a four-factor signal implemented in packages/ signalflow :

score = w_d × domain_match + w_r × reputation + w_l × current_load_inverse + w_a × historical_accuracy

Default weights are (w_d, w_r, w_l, w_a) equal to (0.35, 0.30, 0.15, 0.20), with per-DFAO override allowed. The routing function is a scoring rule, not an assignment; the participant retains refusal. Reputation gates whose slice scores carry weight in aggregation; it does not create a class of people called validators. When you see “validator” in the code, the specification, or this Codex, read it as “a contributor while they are performing a validating task,” not as a person who holds a validator role.

7.4 The Cross-Layer Coupling

Reputation conditions routing (through the w_r term above) but never conditions minting (per the hard separation in §4.2 and §10.1). This coupling is intentional and is now stated explicitly to avoid the confusion an earlier peer review flagged (see §18.3): reputation affects who is likely to be routed a validation task, which is a legitimate quality-of-routing signal, but it never affects how much XP a new claim mints, which would reintroduce the exact runaway-concentration risk the architecture is built to prevent.

7.5 Blind and Implicit Validation

Most validation in the mesh is not someone explicitly clicking approve. It is implicit and frequently invisible to the performer, produced by two mechanisms. Blind slicing. A claim is split into one-tenth slices and routed to contributors who see only their slice, not the parent claim or who made it (packages/validation-neighborhoods). A contributor scoring a slice does not know whose work they are checking, or sometimes even that the slice belongs to a larger validation at all. They are performing a small entropy-reducing task; the aggregation layer turns their independent slice scores into the falsifiability signal for the parent claim. The validation is real; the performer’s awareness of it is not required. Downstream task overlap. Many tasks validate earlier tasks as a side effect of doing their own job. If task B builds on the output of task A, then B succeeding is partial confirmation of A, and B failing in a way traceable to A’s output is partial contradiction of A. The person performing B is not “validating A”; they are performing B. The contribution graph extracts the validation relationship after the fact, from the dependency structure recorded in the DAG’s causal edges (§9.7), not from anyone’s stated intent.

7.6 Bayesian Aggregation and Neighborhoods

Validator neighborhoods are per-domain and per-loop sets of contributors above a domain-specific access threshold τ_validator_d, who observe submitted evidence and produce independent verdicts (the Protocol Contract in this Codex §2.2). Verdicts take the form validator identifier, measured ΔS, a decision among accept, reject, or abstain, and an evidence hash. Accept requires the measured ΔS to fall within tolerance ε_d of the

claimed value. Reject applies when the difference exceeds ε_d or when any evidence condition E1 through E3 fails. Abstain applies when the validator lacks sub-domain competency; abstentions do not count toward closure quorum but are recorded so validator drift and coverage gaps stay observable. Where multiple independent judgments must be combined into a single confidence estimate, the reference implementation uses Beta-conjugate Bayesian updating: each validation outcome updates a Betadistributed posterior over the claim’s validity, and closure requires the posterior mean to clear a domainset threshold, provisionally 0.66 in the worked example in Appendix B. This is the concrete mechanism behind “validator-verdicts-collected” in the loop lifecycle state machine of §8.4.

7.7 ρ_validator and Access Thresholds Participation in a domain’s validator neighborhood requires the actor’s XP or reputation standing in that domain to clear τ_validator_d, a domain-specific access threshold, distinct from the action threshold τ_action_d that gates opening a loop in the first place. Neither threshold is spent when crossed; they are stateful gates, consistent with the access-economy semantics in §13.2.

7.8 The Epistemology Engine, Correctly Framed

The epistemology-engine package is the mesh's emergent peer-review witness layer. It is not a review service, and it does not assign validators. It observes the task graph and reads validation out of it as an emergent property. What it does: aggregates validation outcomes across the mesh and surfaces consensus drift, dissent clusters, and contested-claim patterns; computes mesh-wide falsifiability statistics per domain and per DFAO; tracks reputation graph evolution and exposes Sybil-suspicious clusters; surfaces emergent ontologies, such as recurring claim patterns, naming convergence, and instrument standardization across DFAOs; and provides queryable hooks for governance proposals. What it does not do: decide what is true, perform claim decomposition, arbitrate disputes, or own a private world model. Architecturally, it is read-mostly, writes only metadata about the network's own state, is stateless under restart (it can be rebuilt entirely from DAG replay), and multiple independent instances may run simultaneously, since there is no canonical engine instance by design (this Codex §13.4). This redefinition corrects an earlier misreading in the reference specification's v3.0 draft, which framed the engine as a central decomposition service. That framing violated Digital Autarky by centralizing intelligence at the network layer, and it was corrected in v3.1: decomposition moved entirely to personal AI at the edge, and the engine was reread as the observability layer it always should have been.

7.9 Retroactive Slashing

A validation that later proves wrong burns its XP and penalizes the reputation behind it, exactly like any other contribution that decayed under new evidence. Because validation is a task subject to the same retroactive settlement as everything else, the watcher regress terminates: there is no separate trust tier that must be trusted axiomatically. See §16.1 for the cartel-defection analysis that depends on this mechanism functioning as specified.

7.10 Verdict Vocabulary

Two verdict values, confirmed and supported , are both currently in use across validators and test scripts in the reference implementation, both treated as affirmative for aggregation purposes. This is tracked as an open item (§20 of this Codex item 25): a single canonical enum needs to be defined in the shared contracts package and enforced at every validation boundary, and this Codex flags the current permissiveness honestly rather than pretending the vocabulary is already settled.

Falsifier. If any shipped protocol behavior admits a persistent, self-identified validator class that cannot itself be validated by contributors doing ordinary tasks, the emergence property is broken. Either the class must be dissolved into ordinary contribution routing, or the emergence claim must be dropped from this Codex. Separately, if any shipped mint path allows an actor's reputation to raise their new-mint multiplier through any indirect channel, including through the routing weight w_r feeding back into R or F, the cross-layer coupling described in §7.4 has failed and must be corrected.

8. The Contribution Graph and Loop Lifecycle

In plain terms: this section is the operational core of the protocol. It describes the single data structure, the contribution graph, that every unit of work in the system lives inside, and it walks through the exact state machine a claim passes through from opening to settlement.

8.1 The Contribution Graph

There is exactly one primitive in the system: a contribution. Cleaning a roadside is a contribution. Witnessing a fragment of someone else's claim is a contribution. Reviewing a document is a contribution. Completing a delivery is a contribution. All of them live in the same graph, and all of them are surfaced, routed, completed, and rewarded by the same engine (§8 of this Codex). Validation is not a separate concept; it is a contribution class with specific routing

rules, elaborated in full in §10. There is no validator role and no validation application as a separate thing. The substrate is theme-neutral: frontend applications may apply any theme overlay (fantasy, science-fiction, professional, minimalist) without changing the underlying math, routing, or credentialing thresholds. A contribution carries a type (cleanup, witness, review, delivery, instruction, repair, sourcing-confirmation, and others, open-ended), a stakes class (one of five, described in §9.2), a domain signature across the eight domains, a decay schedule governed by F, an R value drawn from the domain's rarity table, a T_s target, a set of prerequisites (CAT level, domain coverage, location, or none), a completion criterion, and a reward computed by the canonical formula once completion is witnessed.

8.2 The Five Stakes Classes

The graph is one substrate, but contributions route through one of five lanes with distinct rules because they carry distinct stakes. High-stakes and credentialed contributions, such as medical decisions or structural engineering, require hard CAT thresholds, hard deadlines, and explicit assignment; a participant never silently witnesses a high-stakes decision as a side effect of unrelated activity. Civic and coordination contributions, such as roadside cleanup or neighborhood mutual aid, carry soft deadlines and lower coverage thresholds with higher routing volume. Everyday and habitual contributions, such as cooking, household management, and learning loops, have wide eligibility and represent most of the network's volume, with short, regenerative decay schedules. Time-sensitive and event contributions, such as emergency response, tolerate only minutes of latency and prioritize proximity and availability over coverage depth. Speculative and exploratory contributions, such as open research questions, tolerate long settlement times and require higher CAT thresholds for witness contributions, reflecting lower volume but higher per-contribution significance.

8.3 The Loop as Unit of Work

Every unit of value in the Extropy Engine is minted from a loop. A loop is a bounded piece of work that opens, gathers evidence, is scored by a validator neighborhood (see §10 for why "validator" names a task, not a role), and closes with a verdict. If the verdict is favorable, XP mints against the loop record; if not, the loop is rejected and nothing mints. There is no partial credit for unclosed loops.

8.4 The Five States and the Lifecycle

A loop passes through the following states, in order, with no retro-mutation:

proposed → open → evidence-submitted → validator-verdicts-collected → closed | rejected ↓ minted → confirmed | burned

Each state transition is a substrate-recorded event. Timestamps are wall-clock but not authoritative for XP; the authoritative time input is elapsed seconds, measured as close time minus open time at substrate resolution (the Protocol Contract in this Codex §3, the Protocol Contract in this Codex §6). The default task grain, established in docs/ QUEST_MARKET.md , is two to five minutes: this choice is operational, not philosophical. Contribution economies fail when contribution units are too vague, too large, or too slow to verify, and a two-to-fiveminute default makes onboarding low-friction, validation tractable, coordination meaningful, and gaming expensive, since farming costs scale roughly linearly with the number of loops required. Larger work composes from many micro-loops rather than existing as a separate track for "big" contributions.

8.5 Evidence Requirements

Every loop closure MUST carry evidence satisfying three conditions, restated from the Protocol Contract in this Codex §4. E1, independently reproducible: any validator in the neighborhood can reproduce the evidence without contacting the actor. E2, domain-native: the domain's measurement operator M_d applied to this evidence yields ΔS_b_e within the domain's tolerance ε_d. E3, tamper-evident: a validator can detect if evidence has been modified between submission

and verdict. Implementations may use content-addressed storage, signed attestations, or on-chain hashes to satisfy E3; the protocol requires only that E3 holds, not any specific mechanism.

8.6 Provisional and Confirmed Mints

Mints are provisional until a retroactive-validation window closes. During that window, any validator, not only the original neighborhood, can submit a challenge. A challenge producing a rebuttal accepted by at least Q_d validators burns the mint, and the distribution is reversed. The formula-version stamp is retained on burned mints so an audit can distinguish burns of legitimate mints, arising from ambient noise, from burns of gamed mints. A provisional mint that survives its window with no accepted challenge is confirmed and its XP is committed to the actor's thresholds (the Protocol Contract in this Codex §9).

8.7 DAG Event Vocabulary

The substrate records a specific vocabulary of events for every loop: LOOP_OPEN , EVIDENCE_SUBMITTED , VALIDATION_RECORDED (once per validator verdict), LOOP_CONSENSUS (once quorum and rebuttal conditions are satisfied), XP_MINT_PROVISIONAL , and either XP_MINT_CONFIRMED or XP_MINT_BURNED at the close of the retroactive window. Each event is a DAG vertex, signed by the relevant actor's identity key, and the graph's causal edges record dependencies between loops, which is what the epistemologyengine package reads to surface downstream-task-overlap validation (see §10.3).

8.8 Event-Sourced Reinterpretation

Because the substrate is append-only and event-sourced, no event is ever edited or deleted after the fact. If new evidence changes the interpretation of a closed loop, the correction is written as a new event, most commonly a burn event during the retroactive window, rather than as a mutation of the original record. This preserves a complete, auditable history: any observer can replay the DAG from genesis and arrive at the same current state, and any dispute about what happened can be resolved by inspecting the event log rather than trusting a mutable database row.

8.9 The Happy Path, Restated

The ordinary path for a loop, restated in narrative form: a real-world input becomes a structured microclaim through personal AI mediation; the claim is decomposed into two-to-five-minute tasks; the microquest is published to the marketplace; SignalFlow routes it to eligible participants; a participant accepts and performs the work; completion is submitted with evidence; volunteer micro-validation via one-tenth blind slices produces a weighted consensus; the loop closes; XP mints provisionally; the retroactive validation window opens and, absent a successful challenge, closes with the mint confirmed. See Appendix B for this path worked through with concrete numbers.

Falsifier. If any deployed loop pipeline permits state transitions other than those listed in §8.4, or mints XP without passing the closure conditions stated in §7.4 below, the pipeline is nonconformant with the Protocol Contract in this Codex and must be corrected.

9. Identity, PSLL, Privacy, and Digital Autarky

In plain terms: this section is about how the network knows you are one real person without ever seeing your passport, and how you keep a private, tamper-evident diary of your own activity that the network can check against without ever reading it directly.

9.1 Digital Autarky as the Governing Constitutional Principle

Digital Autarky means every participant remains sovereign over their own intelligence stack, identity material, decision context, and local event history. The network is a coordination and accounting layer, never a supermind. The full statement lives in architecture/AUTARKY.md . The concern this principle addresses is specific: a network that decomposes reality on behalf of its users, even with the best intentions, eventually shapes what users perceive, what they can claim, and what counts as valid. That control surface is what every prior coordination platform has converged toward, regardless of the starting ideology, and the Extropy Engine refuses that surface as a structural commitment rather than a stylistic preference. The sovereignty table restated from the Concise edition: personal AI and model selection belong to the participant; local raw context, private logs, sensors, and observations belong to the participant; identity material, KYC artifacts, and biometric bindings live on the participant's device; decision history, the

PSLL, belongs to the participant; claim formulation, the decomposition of a real-world input into actionable units, is the participant's personal AI's job, not the network's; the submitted claim payload is the minimum interoperable surface the network needs; validation outcomes belong to the mesh, via incentive-aligned peer review; and receipts and DAG entries belong to the shared, immutable, public network.

9.2 The Hybrid Identity Layer

Identity establishes strong Sybil resistance and selective accountability without exposing raw identity material to the network. The full statement lives in §9 of this Codex , and the implementation skeleton is packages/identity , running on port 4101. Design constraints, all non-negotiable: easy onboarding for normal humans; strong resistance to one-person-many-identity abuse; no raw personally identifiable information exposure to the network DAG; selective reveal under governance conditions; and compatibility with edge-native intelligence, meaning personal AI handles identity locally rather than delegating it to the network.

9.3 BBS+, zk-SNARKs, and Verifiable Credentials

The canonical onboarding flow proceeds as follows. First, the user signs in via OAuth or OpenID Connect using familiar credentials. Second, an on-device KYC binding runs entirely on the user's device: identity document parsing, liveness detection, biometric binding, or a trusted issuer handoff; the network sees nothing at this step. Third, the personal AI generates a W3C Decentralized Identifier and a corresponding Verifiable Credential locally. Fourth, the credential is wrapped in zero-knowledge proof material; the default scheme is BBS+, chosen for selective-disclosure friendliness and smaller proof sizes, with zkSNARK circuits supported as an alternate for specific predicates such as age or jurisdiction. Fifth, the network receives only proof of uniqueness, proof of valid onboarding, a per-context nullifier, and the public DID; the network never receives raw documents, full biometric material, or a real-world identity bound to the DID by default.

9.4 Nullifiers and Per-Context Unlinkability

Per-context nullifiers are derived so the same DID cannot be cross-correlated across different DFAOs without the participant's consent. This is what makes the selective-privacy claim substantive rather than nominal: a participant can be a confirmed unique human in DFAO A and a confirmed unique human in DFAO B without either DFAO, or an observer watching both, being able to prove those two identities are the same person, absent an explicit reveal.

9.5 The Onboarding State Machine

The onboarding flow described in §12.3 is implemented as a state machine: a user visits an application (HomeFlow, a character-sheet interface, or another vertical) and initiates sign-in; the OAuth or OpenID step establishes a familiar authentication anchor; the on-device KYC step runs locally and produces a local-only attestation; the DID generator produces key material stored in a local secure enclave or its equivalent; the Verifiable Credential issuer wraps the local

attestation; the ZKP wrapper produces the proof material actually submitted to the network; and the nullifier service derives the context-specific identifier that the network stores. Each stage's output becomes the next stage's input, and a failure at any stage halts onboarding without partial identity material leaking to the network.

9.6 The 7-of-12 Reveal Threshold

Under governance threshold, a DID can be linked back to enforceable real-world identity. The provisional default is a 7-of-12 ecosystem-validator threshold, requiring a valid governance proposal with cause shown; threshold-keyed escrow holds the reveal material using a Shamir-style or threshold-encryption scheme, and the threshold itself is tunable per ecosystem DFAO (§9 of this Codex). This is selective privacy under enforceable accountability, not anonymity and not surveillance. The mesh cannot see who a DID is, and it cannot correlate DIDs across DFAOs without consent, but governance can pierce the veil with cause, through a public and auditable process. The bootstrap problem, who issues trusted KYC attestations before an ecosystem of accredited issuers exists, is an open question tracked in §20 of this Codex , with a provisional answer of bootstrapping through accredited issuers and transitioning to community-vouched models as DFAOs mature.

9.7 The Personal Signed Local Log

Every participant maintains a Personal Signed Local Log, PSLL, on their own device, implemented via packages/psll-sync on port 4102. The pattern is borrowed, with credit, from Holochain's source-chain concept and reimplemented natively (§11.6). The PSLL is append-only, hash-chained (each entry includes the hash of the previous entry), cryptographically signed with the participant's DID key, locally controlled, and selectively disclosable. Required entry types, per the minimum schema in §9.7 of this Codex , include claim submission, validation performed, quest accepted, quest completed, decomposition step, governance vote, reputation update, and reveal consent. Other participants' PSLL contents never appear in one's own log, since each PSLL is single-author, and raw network-side state never appears either, since the DAG is the canonical record for that.

9.8 Anchoring, Not Ingestion

The network does not ingest raw PSLL payloads. Instead, periodic Merkle-root commitment receipts are anchored into the DAG as public, immutable vertices, provisionally at a cadence of one anchor per active session, tunable per DFAO. Under dispute, subsets of the PSLL can be revealed with inclusion proofs or ZKP-based selective disclosure, allowing a participant to prove, for example, "I logged a claim of type X at time Y" without revealing the entry's full content. Optional device-to-device sync is supported via endto-end encrypted channels or encrypted backup to participant-controlled storage; packages/psll-sync handles local maintenance, anchoring, and this optional sync, and does not participate in network gossip.

9.9 Cryptographic Defaults and Their Rationale

BBS+ is the default ZKP scheme because it supports selective disclosure natively and produces smaller proofs than general-purpose zk-SNARK circuits for the common case of proving simple predicates about a credential. zk-SNARKs remain supported for predicates BBS+ does not handle as efficiently. Both choices are explicitly provisional and governance-tunable (§14.4 of this Codex), with an eye toward post-quantum-friendly schemes as that literature matures; this future-proofing intent is stated in §9 of this Codex without committing to a specific post-quantum scheme prematurely.

9.10 What the Network Does Not Get

Stated as a table for clarity, restated from this Codex §8.3: the network sees proof of uniqueness, proof of valid onboarding, contextual nullifier material, and governance-relevant accountability hooks. The network does not see raw identity documents, full biometric material, private local onboarding state, or a real-world identity tied to a DID by default. This boundary is the identity layer's entire reason for existing, and every design choice in this section is justifiable by reference back to this boundary.

Falsifier. If the network's threat model requires legal identity mapping to close Sybil clusters in practice, either the ZKP layer is inadequate or the threat model has been misspecified, and either resolution requires a public correction to this Codex. Separately, if any shipped protocol service accepts raw personal context, raw PSLL contents, or a participant's private reasoning as input, Digital Autarky is falsified at that service, and the service must be reduced to the minimal claim-package surface or removed from the protocol layer.

10. Token Economy and Market Layer

In plain terms: this section explains the six tokens, why none of them can be sold, and how the merchantfacing side of the system makes money without ever charging a participant or turning XP into a tradable asset.

10.1 The Six Canonical Tokens, Restated

Section 4.3 defined the six tokens by name. This section states how they interact as a system. XP is the raw, non-transferable measure of verified entropy reduction, minted per loop by the canonical formula in §4.1. CT is the structural, reputation-carrying coefficient that shapes access weight in ties and vote weight in validation, computed by the CT formula in §5.9, and it is the one place reputation density legitimately enters the system. CAT is the portable, cross-application capability certification, progressing on a logarithmic scale with confirmed-contribution thresholds. IT is the non-transferable influence and governance-weight token, decaying at roughly five percent per month as anti-capture pressure. DT is the domain-scoped subject-matter expertise accounting object. EP, Emergence Points, is the token that actually settles at the merchant-facing layer, computed as XP times a local loyalty multiplier L, and it is the only token with anything resembling a redemption surface.

10.2 Access-Economy Semantics

Under the Non-Extraction invariant (§13.3), XP and CT are not "how much I have" but "what I can currently do." XP functions as a stateful threshold: above a threshold τ_action, the actor is admitted to a class of actions, such as validating in a domain, opening a loop of a given rarity, or joining a validator neighborhood; below τ_action, the action is simply not admitted. XP is not spent by taking the action. CT is a per-actor structural coefficient in the SignalFlow routing equation (§10.3); it shapes access weight in ties and vote weight in validation, and it is not spent by voting. Consumption is metered against a contribution and draw ratio, ρ equals ΔS produced divided by ΔS consumed, computed on the ledger itself. If ρ drifts below a domain-set floor ρ_min_d, access degrades automatically, and restoration of ρ is itself a mintable loop. The ratio ρ is what a fiat balance would have measured if one existed; the point of the architecture is that no fiat balance is needed to enforce reciprocity, because reciprocity is a property of the actor's own ledger history.

10.3 The Non-Extraction Invariant, in Full

Extropy tokens are non-transferable access thresholds. They are never balances to be extracted. Formally, restated from docs/NON_EXTRACTION.md §1: there is no path in the protocol, at any layer, by which an actor converts an XP or CT balance into a claim on fiat, another cryptocurrency, or any external transferable value, whether directly or through wrapped, synthetic, derivative, or off-protocol side-market representations. This is not a policy setting. It is an architectural invariant of the same order as Digital Autarky: if a feature can be added without breaking Autarky and without breaking Non-Extraction, it is allowed; if it breaks either, it is not shipped. Two failure modes are ruled out

simultaneously. The Howey trap: any token that is purchasable, transferable, and held with an expectation of profit derived from others' work becomes a security under United States case law, and every prior "just add a small liquid market" iteration of a protocol like this has ended either registered as a security and captured by the intermediaries it was designed to route around, or operating in the grey until shut down. The way out is not a better lawyer; it is a token with no cash-out surface at all. The Nash flip: the feedback loops this Codex describes rest on an equilibrium in which honest signaling is optimal when the payoff for gaming is bounded by what the loop itself produces. The moment loop output can be sold outside the protocol, the payoff for gaming becomes the external market price, which is unbounded from the loop's perspective, and that flip is exactly what makes validator cartels rational in systems that allow it. Both failure modes disappear if the token has no external price.

10.4 The Three Tests Every Feature Must Pass

Before any feature merges into the protocol layer, it MUST pass three tests, stated in full in docs/NON_EXTRACTION.md §4. T1, the extraction test: can any actor convert an on-protocol balance into an off-protocol transferable value, directly or through a wrap? If yes, T1 fails. T2, the counterparty test: does the feature introduce a stable second party whose sole role is to pay for XP or CT? If yes, T2 fails. T3, the ratio test: does the feature preserve the contribution and draw ratio ρ as the primary throttle, or does it introduce a shortcut letting an actor draw without proportionately producing? If yes, T3 fails. Every module in packages/xp-mint , packages/reputation , packages/loop-ledger , and any future redemption package MUST include tests exercising these three tests against its public surface.

10.5 The Market Layer and the Two-Sided Model

The merchant-facing layer, described architecturally in §14 below and in full in §1.4 and Appendix M of this Codex , is where EP actually functions as a settlement token: a merchant offers a discount or benefit redeemable against a participant's EP balance, computed from their XP times the merchant's or DFAO's local loyalty multiplier. This is compatible with Non-Extraction because the redemption surface is bounded to intra-network merchant discounts and operational benefits, never to a cash-out, never to a transferable claim a third party could buy, and never to a counterparty whose sole role is paying for XP or CT (which would fail T2 above). The business model on the merchant side captures the standard merchant-services fee the merchant was already paying someone else, at a competitive rate, plus DFAO node registration fees at scale, treasury yield on protocol reserves, premium analytics for businesses wanting deeper signal, and specialized integration fees for multinational nodes. It does not charge participants a subscription, does not paywall the point-of-sale software, and does not bill users directly.

10.6 Token Lifecycle

A token's lifecycle, using XP as the representative case, runs from provisional mint (§9.6) through the retroactive validation window to either confirmation or burn. CT accrues alongside confirmed XP under the CT formula and carries its own lockup, provisionally fourteen days, before it becomes fully liquid for routing-weight purposes. CAT level updates occur when a participant crosses a confirmed-contribution threshold in a given domain, and CAT levels do not decay, unlike IT, which decays continuously as anticapture pressure. EP is generated at the point of merchant-facing redemption computation and is consumed, not accumulated indefinitely, when a participant redeems a discount. DT accrues per-domain and is primarily an internal accounting object supporting the token-economy service's domain-scoped balance tracking (packages/token-economy , port 4012).

10.7 Interaction with Bridge Integrations

Bridges to external systems, such as GitHub App events, calendar integrations, or appliance telemetry, are compatible with Non-Extraction as long as three conditions hold: the external system provides evidence, not payment; the reward for

evidence is XP threshold movement, not a transferable claim; and the bridge is one-way at the value layer, evidence flows in, no XP or CT flows out to the external system. The github-parasite package (§15.8) is designed under exactly these constraints and is cited in docs/NON_EXTRACTION.md §6 as the worked example of a compliant bridge.

10.8 What Non-Extraction Is Not

It is not a claim that no one will ever try to build an off-protocol market in XP receipts; people will. The design goal is that such a market has no counterparty support inside the protocol, so it stays a fringe activity and never captures the equilibrium. It is not asceticism: actors convert access thresholds into realworld value constantly, by using them, and what they cannot do is package unused access as a transferable claim. It is not a substitute for legal review: it removes the strongest known legal attack surface, the Howey test, but does not preempt every other regulatory question.

Falsifier. If any shipped protocol feature passes review while failing any of T1, T2, or T3, either the test framework is broken or the feature is a violation, and either resolution requires a public correction. Non-Extraction is falsified as an invariant the moment it is treated as guidance rather than a hard gate on merge.

11. The DAG Substrate and Person as DAG

In plain terms: this section describes the shared, tamper-evident record book that the entire network writes to. Nobody's local reasoning or private data goes into it; only the minimum public receipts needed for coordination and audit.

11.1 Three Axioms of the Substrate

The substrate exists to satisfy three commitments that recur throughout this Codex. First, it must be append-only at the event layer: nothing already written is ever edited or deleted, only superseded by new events (§9.8). Second, it must expose the mint-event log to any actor, so that the formula-version stamping discipline in §4.1 is independently auditable. Third, it must record causal dependencies between loops, so that downstream-task-overlap validation (§10.5) can be extracted after the fact rather than declared in advance.

11.2 Why a Native Substrate

The Extropy Engine commits to a native substrate end to end. It is not deployed as an application on Holochain, Ethereum, Solana, or any other existing framework. Full Digital Autarky requires owning the lowest shared layer, the handshake and the DAG; dependency on another project's plumbing would compromise the sovereignty commitment and create a supply-chain control point outside the network's own governance (this Codex §11.1).

11.3 The DAG Service

The reference implementation is packages/dag-substrate , running on port 4008. Its own documentation describes it as a permissionless ledger layer, analogous to the IOTA Tangle: every significant system event, loop opens, measurements, votes, mints, and governance proposals, is recorded as a cryptographically signed vertex in a directed acyclic graph. Core properties, per the package's own source documentation, include causal ordering via Lamport timestamps, tip selection via a random walk weighted

by confirmation weight, automatic vertex creation by listening to system events rather than requiring explicit submission for every event type, confirmation-weight propagation up the causal chain, and permissionless submission, meaning any service can submit a signed vertex without a gatekeeper.

11.4 IOTA-Inspired Tip Selection

The tip-selection mechanism (a random walk weighted by confirmation weight) is explicitly modeled on the IOTA Tangle's approach to DAG-based consensus without a linear blockchain, credited here as prior art the way this Codex §11.2 credits Holochain's patterns. The Extropy Engine's substrate borrows the shape of this approach and reimplements it natively, tuned for the specific vertex types the protocol needs (loop lifecycle events, governance events, reputation updates, token mint and burn events, credential issuance, and PSLL anchor commitments), rather than adopting IOTA's transaction model wholesale.

11.5 Convergence Vertices

Multi-party collaboration is the dominant mode of real-world entropy reduction, and the substrate represents this as a first-class primitive rather than forcing collaborators to submit artificially separated individual claims. A convergence vertex records a joint claim with a documented contribution split among participants, validated as a single unit. Convergence-split rules are governance-tunable: DFAOs can establish defaults for equal splits in symmetric collaborations, contribution-weighted splits for asymmetric ones, or fully custom DFAO-defined split functions for complex cases. See Appendix F for a worked convergence-vertex example.

11.6 Borrowed Patterns, Credited

Three architectural patterns are borrowed from Holochain and reimplemented natively, with credit given rather than obscured. The source chain concept becomes the Personal Signed Local Log, described in full in §12.4. The neighborhood DHT concept becomes Validation Neighborhoods, described in §7.6. Zome and DNA modules become Rule Modules, the composable units of fractal DFAO inheritance and evolution described in §14.3. The patterns are good; the implementations are the Extropy Engine's own, reimplemented to fit a native, non-Holochain substrate.

11.7 Nested DAGs: Person to Civilization

Because DFAOs nest fractally (§14.3), and because each DFAO's activity is recorded on the shared substrate, the DAG itself exhibits a nested structure that mirrors the governance hierarchy: an individual's loop history forms a coherent sub-graph, a household or team DFAO's activity forms a larger sub-graph containing its members' individual sub-graphs as causally-connected components, and this composition continues upward through domain DFAOs to the ecosystem scale. There is no single "master" DAG instance that must be centrally operated; the substrate's confirmation-weight propagation and tip-selection mechanism operate correctly whether observed at the scale of one person's loop history or at the scale of the entire network's event log, which is the concrete, implementation-level expression of Axiom 3 (Fractal Self-Similarity) from §7.2.

11.8 Causal History Walks

Because causal edges are recorded explicitly (§9.7, §11.1), any observer can walk backward from a given mint event to reconstruct the full evidentiary chain that justified it: which loop closed, which validators produced verdicts, what evidence was submitted, and which earlier loops (if any) that evidence causally

depended on. This walk is what the epistemology-engine performs at scale to surface dissent clusters and downstream-task-overlap validation (§10.5, §10.8), and it is available to any individual actor or auditor at the scale of a single claim.

11.9 Substrate Implementation and Conformance

Any alternative implementation of the substrate is Extropy-conformant if it satisfies the role definition in the Protocol Contract in this Codex §2.3: append-only at the event layer, exposing the mint-event log, and correctly recording the state transitions in the Protocol Contract in this Codex §3. The reference implementation's specific choices,

PostgreSQL-backed storage, Lamport timestamps, and IOTA-inspired tip selection, are referenceimplementation decisions, not protocol requirements (the Protocol Contract in this Codex §1, restated in §15.9 below).

11.10 Production Migration Considerations

The current substrate runs as a sandbox instance; docs/VPS_NODE.md states plainly that the deployed VPS instance is not a production node, that its node-to-node transport is HTTPS with body signing rather than a hardened peer-to-peer transport such as libp2p with Noise, and that no participant identity material lives on the VPS beyond what the identity layer's escrow design explicitly allows. The path to production involves hardening the transport layer, formalizing the causal-edge gossip protocol between independently operated nodes (an open item, §20 of this Codex item 27), and specifying partition tolerance and merge rules (§20 of this Codex item 28). None of this is a change to the substrate's logical model; it is the engineering work of making the logical model survive an adversarial network.

Falsifier. If the substrate exposes any mint event whose stamped formula version differs from the executed math, the substrate is non-conformant and must fail closed. If any deployed substrate instance requires a single canonical operator to function, the permissionless and multi-instance design claims in this section are falsified for that deployment.

12. Meaning Drift and Goodhart Resistance

In plain terms: this section explains, formally, why metrics decay once they carry incentive weight, and how the Extropy Engine is designed to resist that decay rather than merely delay it. It is load-bearing because everything else in this Codex is a specific engineering response to the dynamic described here.

12.1 Why This Section Is Load-Bearing

Section 2 stated the proxy-worship problem in plain terms. This section states it formally, using the representational fidelity model from an internal working paper cited throughout this Codex as the Signal paper, referenced in docs/NON_EXTRACTION.md §2.2 as the source of the Nash-flip analysis that motivates the Non-Extraction invariant in §13. If this section's model is wrong, the rest of the architecture's Goodhart-resistance claims lose their footing, so the model is stated here explicitly rather than assumed.

12.2 The Central Dynamic

Let F(t) denote the representational fidelity of a label or metric at time t: how well the label tracks the underlying condition it names. Let S(t) denote the social or incentive load placed on the label: how much reward, status, or access is conditioned on the label's value. The Signal paper's central differential equation is:

$$dF/dt = -k \cdot S(t) \cdot F(t) + r \cdot C(t) \cdot (1 - F(t))$$

The decay term, negative k times S(t) times F(t), states that fidelity loss is proportional to the fidelity currently remaining (you cannot lose what you do not have), proportional to the current social load (more incentive weight produces faster gaming), and proportional to a domain-specific susceptibility coefficient k. The recovery term, r times C(t) times (1 minus F(t)), states that fidelity recovery is proportional to the gap from perfect fidelity, proportional to the strength of corrective feedback C(t), and proportional to a restoration rate r. At equilibrium, where dF/dt equals zero, the equilibrium fidelity F is determined by the ratio of corrective force to gaming force: when C is large relative to S, F approaches 1; when S is large relative to C, F* approaches 0, meaning full semantic capture. See Appendix K for the full derivation and the equilibrium expression worked out in closed form.

12.3 The Three-Domain k-Parameter Taxonomy

The Signal paper estimates k empirically from documented Goodhart episodes across three domain classes. The physical domain, with k approximately 0.02, has fast corrective feedback: a bridge rated for 100 tons that actually fails at 80 tons collapses visibly, and engineering metrics operate under this fast feedback regime. The institutional domain, with k approximately 0.15, has feedback mediated by bureaucracies, peer review, and regulatory process, operating on timescales of months to years: university rankings, journal impact factors, and credit ratings live here. The social and moral domain, with k approximately 0.45, has weak or absent corrective feedback: status claims, identity claims, and authenticity labels live here, and the decay term dominates because C(t) approaches zero. Appendix K gives the full taxonomy with worked examples for each regime.

12.4 The Four Observable Phases

Labels under incentive pressure pass through up to four observable phases. In the descriptive phase, S(t) is low and F is high; the label is functional and people use it to track the underlying condition. In the standardization phase, S(t) rises as institutions invest in verification, and F may temporarily improve as measurement formalizes. In the capture phase, S(t) saturates, gaming proliferates, and F decays as the actors with the most to gain learn to optimize the label rather than the condition. In the collapse or

correction phase, the label either empties of meaning entirely, or an external shock to C(t) triggers partial restoration. Not every label passes through all four phases; some stabilize in the standardization phase when the underlying domain supports continuous corrective feedback.

12.5 The Case Study: AI Content Detection

A live case study, contemporaneous with this Codex's drafting, is the labeling of AI-generated content. As detection tools and disclosure requirements (S(t) rising) have been layered onto content platforms, the correlation between the "AI-generated" label and actual generation provenance has degraded rapidly (F falling) because corrective feedback (C(t)) is weak: there is no fast, reliable, universally trusted method to verify provenance after the fact. This sits squarely in the social and moral domain's high-k regime, and the Signal paper's model predicts exactly the trajectory observed: rapid capture, weak correction.

12.6 Self-Application

The Signal paper applies its own model to itself, estimating its own k at approximately 0.15, the institutional regime, with corrective feedback supplied by peer review, falsification attempts, and empirical challenge. This is not a rhetorical flourish. It is a substantive commitment: if the paper's own framework drifts into dogma, unfalsifiable and immune to revision, the framework predicts its own eventual capture. The Extropy Engine adopts the identical posture at the protocol level. This Codex's falsification conditions throughout, and the explicit correction ledger in §4.6, are the protocol's own submission to the dynamic it describes.

12.7 How This Informs the XP Formula

If reputation entered the value formula directly, optimizing for value would force optimizing for reputation, and reputation-gaming is not hypothetical: every reputation system deployed at scale has eventually been gamed. Therefore the value formula must be insulated from the reputation representation. This is why R is rarity, a property of the move, and not reputation, a property of the actor, and it is the direct formal justification for the hard separation stated in §4.2 and §10.

12.8 How This Informs Governance

The same logic that keeps reputation out of the mint formula keeps governance power from silently accumulating. Section 14's anti-concentration mechanisms, decay on reputation and CT, periodic renormalization of domain weights, rotating validator neighborhoods, and retroactive validation windows, are all instances of engineering C(t) and r to be substantial rather than assuming the domain's k value can be changed directly. The protocol cannot make governance a low-k domain by wishing it so; it can only make sure corrective feedback stays strong enough that the equilibrium fidelity F* stays high.

12.9 Relation to F and the Formula as a Whole

Seven concrete mechanisms in the formula and lifecycle jointly resist meaning drift. R as a loop-class property, not actor-bound, prevents reputation laundering through XP. F as a per-fingerprint decay prevents action-class grinding. ΔS requiring a domain-instrument-baseline-falsifiability record filters trivial submissions at the source. w · E requiring evidence across the eight domains means single-domain trivia cannot reach high XP alone. T_s with a floor and a cap prevents log-term inflation through arbitrarily fast settlement. Closure-quorum thresholding at consensus prevents low-confidence claims from minting. Retroactive burn corrects drift after the fact when the other six mechanisms miss something.

Falsifier. If deployed data shows F(t) for the XP-CT-reputation system decaying at a rate consistent with the social and moral domain's high-k regime, that is, if empirical reputationgaming rates approach those observed in credit scoring or social media engagement metrics, the architecture's Goodhart-resistance claim is falsified for the affected mechanism, and the mechanism must be redesigned or the claim withdrawn.

13. Randall's Feedback: Formal Foundation

In plain terms: this section states the formal governance theory that the Extropy Engine's coordination layer is built on top of, names which claims are proven and which are still conjectures, and explains how the theory's abstractions map onto the concrete DFAO and contribution-graph mechanics described later in this Codex.

13.1 What Randall's Feedback Is

Randall's Feedback, abbreviated 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 Sense-Infer-Integrate-Respond loop; V2 (2024) formalized the Complexity Thermostat; V3 (2025) added bootstrap governance phasing; V4 (2026), the current version, delivers six formal contributions grounding RF in the Free Energy Principle and modern peer-prediction literature. The full V4 paper is Randall's Feedback V4: Formal Foundations for Emergence-First Governance. This section synthesizes the load-bearing claims and shows how they support the Extropy Engine specifically. A companion, less formal treatment, Randall's Feedback V4: The Framework, exists in the repository's notes directory as an accessible walkthrough of the same material for readers without a probability-theory background.

13.2 The Four Axioms

RF V4 rests on four axioms, each stated in mathematical form rather than as slogans. Axiom 1, Feedback Primacy. The governance state evolves according to a feedback dynamical system, S at time t plus one equals S at time t plus alpha times F applied to the state, the environmental signal, and the current goal representation, where F is the feedback functional and alpha is the learning rate. All governance transitions are mediated by F; no state change occurs outside this feedback loop. Axiom 2, Goal Evolution. Goals evolve according to G at time t plus one equals G at time t plus beta times an update operator L applied to the current goal and an observation measured independently of the reputation

state. L is required to be Lipschitz continuous with constant kappa strictly between 0 and 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-organization level and the protocol level through structure-preserving functors in a category of feedback loops. Formally, a scale functor maps the micro-scale feedback category to the macro-scale feedback category, with a natural transformation such that the Sense-Infer-Integrate-Respond loop commutes at every scale.

Axiom 4, Cross-Domain Applicability. The axioms and theorems apply across governance domains via domain-specific instantiation: each domain specifies its own state space, environment, contribution metric, and goal representation while satisfying the structural requirements of Axioms 1 through 3. These axioms are not aspirational. The theorems in §7.4 treat them as the formal constraint on the class of systems to which RF applies, and the Extropy Engine's governance layer is one such instantiation, mapped out explicitly in §7.5.

13.3 The SIIR Loop as Dynamical Primitive

The Sense-Infer-Integrate-Respond loop is RF's fundamental dynamical primitive. In the sense phase, an agent observes the governance environment, collecting data about other agents' contributions, validation outcomes, and governance state. In the infer phase, the agent updates its internal belief model by minimizing variational free energy given the new observations, flowing down the free-energy gradient in the language of Bayesian mechanics. In the integrate phase, updated beliefs combine with the agent's goal representation to form an action plan, via policy selection through expected free energy minimization. In the respond phase, the agent executes the selected action, contribution, vote, or validation, which modifies the environment and generates new sensory signals for other agents, closing the loop. By Axiom 3, this loop operates simultaneously at the individual-agent, sub-organization, and protocol-wide scales.

13.4 The Six Formal Contributions

RF V4 establishes six formal contributions. The first five carry complete proofs in the full paper; the sixth is an explicitly labeled conjecture. Theorem V4-1, RF Governance Convergence. Agents satisfying particular partition conditions converge asymptotically to a governance attractor through Hamiltonian matching under the Free Energy Principle: the agents' internal generative model converges to the governance Hamiltonian. Theorem V4-2, Collective Goal-Update Admissibility. Necessary and sufficient conditions on the goalupdate operator L are established for the collective system to remain in a Banach fixed-point contraction on the product of feasible reward sets. Governance can update goals over time without collapsing into chaos, provided a 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, meaning Goodhart pressure cannot accumulate unboundedly under RF governance. Theorem V4-4, Unhackability Under Constrained Policy Classes. RF's admissibility conditions restrict the contribution strategy space to a finite, or effectively finite, set, which enables non-trivial unhackable metric-goal pairs to exist, in the sense established by the game-theoretic literature on reward hacking. Theorem V4-5, Convergent Validation Without Ground Truth. Byzantine-robust peer prediction is extended to n validators with linear collusion tolerance: consensus can be reached on claim validity without a trusted oracle, provided the validator population is structured as RF specifies. Conjecture V4-6, Trilemma Resolution. RF is conjectured to achieve approximate generalizability, approximate trustlessness, and epsilon-bounded Sybil resistance simultaneously. This is explicitly a conjecture in V4, not a theorem; the formal proof is open. See Appendix J.8 for its current status and retirement path per docs/REJECTED_FRAMINGS.md R3.

13.5 Connection to the Extropy Engine

The governance layer is the operational instantiation of RF. The contribution graph, the unified frame in which all submissions and validations are contributions (see §9 and §8 of this Codex), is the operational form of Axiom 1,

Feedback Primacy. DFAO governance configuration, quorum, conviction voting, and 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 described in §14 are the operational form of Axiom 3, Fractal Self-Similarity. The eight-domain taxonomy in §8 is the operational form of Axiom 4, Cross-Domain Applicability. The architectural invariant that reputation does not enter XP, stated in §4.2 and §10, is the operational form of Theorem V4-3: keeping the value formula insulated from the reputation representation is exactly the structural condition that bounds the discrepancy process the theorem describes. The retroactive validation window and the emergent-validation model in §10 are the operational form of Theorem V4-5: validation proceeds without a trusted oracle, with Byzantine tolerance proportional to the validating population.

13.6 The Two-Phase Bootstrap

RF V4 specifies a two-phase governance bootstrap that the Extropy Engine inherits directly. In Phase 1, contribution-only, all agents have equal standing and contributions are validated purely on content quality, without reputation weighting. Phase 1's duration is set by a sample-complexity bound requiring a substantial number of observations before the first goal update can be triggered. In Phase 2, reputationweighted, the system transitions to reputation-weighted validation once the estimated feasible reward set intersection has sufficiently small diameter, indicating the governance attractor is well-characterized. In the Extropy Engine, this phasing is the formal justification for why a newly formed DFAO cannot launch with reputation weighting from day one: all members start with equal standing in validation routing, and governance can vote to transition to reputation-weighted routing only once the DFAO has accumulated enough contribution data.

13.7 The Complexity Thermostat

The RF Complexity Thermostat, named CT in the RF literature and not to be confused with the Contribution Token CT used elsewhere in this Codex, regulates a unified entropy measure combining goal entropy, contribution entropy, and validation entropy. When that measure exceeds an upper threshold, meaning governance is too chaotic, the thermostat tightens admissibility. When it falls below a lower threshold, meaning governance is too rigid, the thermostat relaxes admissibility. This is the formal mechanism by which RF keeps a governed system in the productive region between collapse and disorder. In the Extropy Engine, this corresponds operationally to the governance-tunable parameters in §14's provisional defaults table: quorum, deliberation period, reward escalation, and the retroactive validation window are all levers a DFAO can adjust as its own entropy state shifts.

13.8 Honest Limitations

RF V4 is deliberately stratified by confidence level. Theorems V4-1 through V4-5 carry complete proofs. Conjecture V4-6 is stated as a conjecture, with the proof explicitly open. The framework distinguishes proven results from conjectured ones and refuses to elide the distinction, and the Extropy Engine inherits this posture: the governance layer rests on five proven theorems and one open conjecture, and the

conjecture, being the strongest claim, is the one most likely to require revision under adversarial pressure. Appendix J gives the expanded treatment of all six results, including the two-phase bootstrap's samplecomplexity bound and the non-degeneracy condition inherited from RF V3.2.

Falsifier. If deployed governance data shows the discrepancy process between contribution metrics and evolving goals developing a heavy tail, contrary to Theorem V4-3's light-tailed prediction, either the theorem's preconditions are violated in deployment (bounded goal-update rates, Lipschitz metric continuity) or the theorem itself requires revision. Either resolution requires a public correction to this Codex.

14. DFAO Governance and Parameter Evolution

In plain terms: this section explains how the rules of the system change over time, who gets to vote on what, and why the whole governance structure is designed so that no single group can permanently seize control of it.

14.1 The Five Scales

Governance is fractal, composable, and bounded against permanent concentration. The unit of governance is the Digital Fractal Autonomous Organization, DFAO, a rule-scoped organization that can nest inside other DFAOs, be nested by them, and evolve its own parameters through the same loop mechanism that mints XP. Five nominal scales recur throughout the reference implementation and this Codex: NANO (an individual or a very small team), MICRO (a household or small workshop collective, typically two to

seven members, the scale HomeFlow operates at, see Appendix G), MESO (a neighborhood, organization, or codebase-specific community), MACRO (a city, region, or large project), and PLANETARY (ecosystem-wide defaults and safety bounds). These scales are the operational form of Axiom 3, Fractal SelfSimilarity, from §7.2.

14.2 Voting Methods and Conviction Voting

DFAOs support multiple voting methods depending on their governance configuration: simple-majority voting for routine decisions, and conviction voting, where voting weight accrues the longer a participant maintains their position on a proposal, for substantial changes. Conviction voting is the reference implementation's default mechanism for parameter updates specifically because it resists both apathetic majority capture (a large but disengaged majority cannot simply outvote a smaller, sustained, and betterinformed minority on a single snapshot vote) and impulsive parameter churn (conviction takes time to accrue, which naturally damps rapid successive proposals on the same parameter). The implementation lives in packages/governance , port 4010.

14.3 Fractal Composition and Rule Modules

DFAOs nest. A domain DFAO, for example an informational-domain DFAO, contains sub-DFAOs, for example a codebase-specific DFAO for one open-source project. Sub-DFAOs inherit parameters from their parent by default and may override them within governance-defined bands; overrides exceeding the allowed band are rejected by the substrate rather than silently applied. This inheritance-with-boundedoverride pattern is implemented through Rule Modules, the Extropy-native reimplementation of Holochain's zome and DNA-module concept (§11.6), and the registry of DFAOs and their rule-module configurations lives in packages/dfao-registry , port 4009.

14.4 The Provisional Defaults Table

Every parameter is a knob. Every knob has a default so the system runs from day one. Every knob is votable. Nothing is locked. The table below restates the current provisional defaults from §14.4 of this Codex , with vote-tier assignments.

Knob Provisional default Vote tier

ZKP scheme BBS+ Ecosystem

Identity reveal threshold 7-of-12 plus cause-shown Ecosystem

Reward escalation curve, early linear 1.0× to 3.0× over 7 days Domain DFAO

Reward escalation curve, late logarithmic to a cap of 10.0× Domain DFAO

Retroactive validation window 30 days Ecosystem

IT decay 5% per month Ecosystem

CT lockup 14 days Ecosystem

EP decay pending Phase 2 modeling Ecosystem

GT decay pending Phase 2 modeling (legacy Ecosystem label, see §4.3)

Conviction voting half-life per tier, pending Per-DFAO

Validator weight factors 4: domain, reputation, load, accuracy Ecosystem

PSLL anchor cadence 1 per active session Ecosystem

T_floor 0.01 Ecosystem

λ_d, R_d, ε_d, Q_d, ρ_min_d, τ_val- per-domain Domain DFAO idator_d, τ_action_d

Quorum size formula open, see GAPS.md item 1 Domain DFAO

Cartel detection threshold open, see GAPS.md items 2 and 8 Ecosystem

Skill DAG progression criteria open, Phase 3 Domain DFAO

Note that “GT decay” persists in §14.4 of this Codex as a table row even though GT is not part of the canonical six-token set defined in §4.3; this Codex flags the inconsistency honestly here rather than silently dropping the row, since the underlying governance-defaults document has not yet been reconciled to the six-token invariant. This is exactly the kind of drift the naming-hygiene program in §4.7 exists to catch and close.

14.5 The Parameter Update Process

A personal AI drafts a proposal targeting the relevant tier. The proposal enters conviction voting in the appropriate DFAO. On passage, the new value is written to the governance configuration and propagated. The PSLL records the proposal trail end to end for every participant who engaged with it. The mint pipeline stamps every mint event with the formula version and parameter set active at the time of minting; a parameter change does not retroactively apply to already-confirmed mints, preserving the DAG’s immutability guarantee from §9.8.

14.6 Parent-Child Conflict Resolution

Cross-DFAO governance conflicts, for example two sub-DFAOs disagreeing on a shared parameter, escalate to the nearest common ancestor DFAO in the fractal hierarchy. The specific escalation rules are an open item (§20 of this Codex item 40), and this Codex states that honestly rather than implying a fully specified resolution process exists today.

14.7 Goodhart Governance and the Complexity Thermostat

Section 7.7 introduced the RF Complexity Thermostat as the formal mechanism regulating a unified governance entropy measure. Operationally, this corresponds to the governance-tunable parameters in the table above: quorum, deliberation period, reward escalation curves, and the retroactive validation window are all levers a DFAO’s governance can adjust as its own entropy state shifts between too chaotic and too rigid. Goodhart pressure on any specific default is treated as diagnostic fuel for refinement, per the closing principle in §14.4 of this Codex , not as a fatal flaw in the governance model itself.

14.8 Anti-Concentration Mechanisms

Reputation decays. CT decays. Domain weights are periodically re-normalized. Validator neighborhoods rotate rather than persisting as a fixed elite. Retroactive validation windows let late correctors take back XP from confirmed-but-wrong closures. Every mechanism in this Codex capable of producing concentration also produces an offsetting bleed, and none of these bleeds is optional or governance-removable without triggering the falsifier below.

14.9 The DFAO Registry

packages/dfao-registry , port 4009, is the reference implementation's source of truth for which DFAOs exist, their scale, their parent-child relationships, their active rule modules, and their current parameter overrides. It is the concrete system a governance proposal writes to on passage (§14.5), and any alternative implementation of the protocol needs an equivalent registry surface to be conformant, even though the specific registry implementation is a reference-implementation detail rather than a protocol requirement (the Protocol Contract in this Codex §1).

Falsifier. If any DFAO can, through governance action alone, produce permanent concentration of validator power that cannot be undone by ordinary participant activity within a bounded number of governance cycles, the anti-concentration claim is falsified for that DFAO, and the mechanism responsible must be redesigned or the claim withdrawn for that scale of organization.

15. Implementation Architecture from the Repository

In plain terms: this section is a guided tour of the actual codebase, package by package, cross-checked against the live repository rather than described from memory. If a package name or file path appears in this section, it exists in the repository at the time of writing.

15.1 Repository Overview

The reference implementation lives at the reference implementation repository , organized as a pnpmmanaged monorepo. Top-level directories include packages/ (the service and library implementations), docs/ (specifications, this Codex, and companion documents), architecture/ (foundational vision documents including AUTARKY.md), frontends/ (application-layer user interfaces), dashboard/ , build/ , scripts/ , and tools/ . The workspace is declared in pnpm-workspace.yaml , and every package under packages/ publishes under the @extropy/ scope.

15.2 Source-of-Truth Hierarchy

For any given claim about the protocol, the source-of-truth hierarchy runs: the Protocol Contract in this Codex for the external, implementation-agnostic contract; this Codex (in either edition) for the internal specification and its reasoning; individual docs/.md files for implementation-level detail on a specific subsystem; and the actual TypeScript source under packages//src/ as the current reference realization. Where a companion document and this Codex disagree, per the Reading Guide at the top of this document, the Codex is the higher-order statement.

15.3 Core Protocol Microservices

The following packages, all live in the repository at the time of writing, form the protocol's core service mesh, each running as an independent process communicating through the shared event bus and substrate.

Package Port Realizes

@extropy/contracts (library) Shared types, schemas, single source of truth for cross-package interfaces

@extropy/xp-formula (library) The canonical formula (§5). Stamped canonical-v3.12

@extropy/xp-mint 4005 The mint pipeline (§5, §9). Routes every mint through the canonical formula and stamps the formula version

@extropy/loop-ledger 4003 Loop lifecycle state machine (§9)

@extropy/signalflow 4002 SignalFlow routing (§10.3)

@extropy/reputation 4004 Per-domain reputation, decay, antiSybil scoring (§10, §14.8)

@extropy/dag-substrate 4008 Substrate (§11). Append-only event log with IOTA-inspired tip selection

@extropy/dfao-registry 4009 DFAO registry (§14.9)

@extropy/governance 4010 Proposals, conviction voting, threshold execution (§14)

@extropy/temporal 4011 Seasons, decay scheduling, loop timeouts

@extropy/temporal-service (see §15.6) Universal Times base-10 calendar service, clock, store, API

@extropy/token-economy 4012 Six-token economy: XP, CT, CAT, IT, DT, EP (§13)

@extropy/credentials 4013 Verifiable credential issuance and verification helpers

@extropy/identity 4101 OAuth, on-device KYC, DID, ZKP wrapper (§12)

@extropy/psll-sync 4102 PSLL anchoring service (§12.7 through §12.8)

@extropy/quest-market 4103 Quest marketplace (§9)

@extropy/validation-neighbor- 4104 Sharded micro-validation routing hoods (§10)

@extropy/epistemology-engine 3002 Emergent peer-review witness layer (§10.8). Read-mostly, multiple instances may run independently

@extropy/ecosystem 4014 Ecosystem-level accounting and cross-DFAO surfaces

@extropy/homeflow 4015 Family pilot vertical (Appendix G)

@extropy/grantflow-discovery 4020 Grant discovery vertical

@extropy/grantflow-proposer 4021 Grant proposal drafting vertical

Package Port Realizes

@extropy/academia-bridge 4022 Academic-publication bridge vertical

@extropy/node-handshake 4200 Sandbox node-to-node handshake harness (§11.10)

@extropy/api-gateway 3000 Public API gateway

15.4 Additional Packages Present in the Repository

Beyond the core mesh, the following packages exist in packages/ at the time of writing and are crosschecked here rather than assumed: @extropy/bayesian (Bayesian aggregation primitives supporting §7.6's closure-confidence computation), @extropy/contracts (already listed above), @extropy/ decomposition-kit (personal-AI-side decomposition helpers, consistent with Digital Autarky's placement of decomposition at the edge, §9.1), @extropy/ethics (ethics-review support utilities), @extropy/extropialingo (terminology and naming-convention tooling, operationally supporting the naming-hygiene program in §4.7), @extropy/levelup-academy (an education-vertical application), @extropy/localflow

(a civic or local-community vertical), and @extropy/github-parasite (§15.8 below). This Codex deliberately does not describe the internal implementation detail of every package listed here beyond naming it and its evident purpose from its package name and location, because doing so beyond what the repository itself documents would risk inventing detail the brief for this document explicitly prohibits. Readers needing implementation-level detail on a specific package should consult that package's own README and source.

15.5 The v3.1 Skeleton Packages

Four packages were introduced as v3.1 skeletons, meaning their specification was frozen ahead of a full implementation: identity , psll-sync , quest-market , and validation-neighborhoods . As of this writing these remain the skeleton implementations described in this Codex §13.2, meaning the specification-level contracts and basic service scaffolding exist, but production-grade hardening (rate limiting under adversarial load, full ZKP circuit auditing, cross-DFAO data isolation per §20 of this Codex item 49) remains open work.

15.6 Application Verticals and Frontends

Application-layer verticals sit on top of the core protocol mesh: HomeFlow (household scale, Appendix G), the GrantFlow pair (discovery and proposer, grant-funding coordination), the Academia Bridge (academic-publication coordination, notably relevant given this Codex's own publication target), LevelUp Academy (education), and LocalFlow (civic and local-community coordination). Frontends live under frontends/ , including a HomeFlow-specific UI (homeflow-ui , port 3002 per this Codex 's service table) and a GrantFlow UI. The Universal Times base-10 calendar service is implemented in @extropy/temporal-service , which contains a clock module, a store module, a universaltimes module, and an HTTP server and app layer; see Appendix P for the calendar's design and status.

15.7 Build Order and Event Flow

The dependency graph documented in DEPENDENCY_GRAPH.md at the repository root establishes the build order: contracts builds first, since every other package depends on its shared types; xp-formula builds next as a pure-function library with no service dependencies; core services (loop-ledger , signalflow , xp-mint , reputation , dag-substrate) build next, each depending on contracts and, where relevant, xp-formula ; the v3.1 skeletons and application verticals build last, since they depend on the core mesh being available. Event flow between services runs through a shared event bus (EventBus in @extropy/contracts), with services publishing typed domain events (loop opened, loop closed, XP minted provisionally, XP confirmed, XP burned, proposal created, governance vote cast, DFAO created, reputation accrued, and others) that the DAG substrate's auto-vertex-creation listener consumes to produce the permanent record described in §11.3.

15.8 The GitHub Parasite

packages/github-parasite is a v0.1 scaffold, with no runtime beyond the scaffold as of this writing, that bridges GitHub App events into the informational-domain contribution graph: pull-request merges become code-contribution loops. XP under this bridge is a function of the actor's contribution ratio ρ, never of any market position, and the package is explicitly designed to satisfy the three Non-Extraction tests from §13.4: nothing it emits is redeemable, transferable, or convertible into external value (docs/ NON_EXTRACTION.md §6, the Correction Ledger in §3 v3.1.3 entry).

15.9 What the Reference Implementation Does Not Commit the Protocol To

Database engines, message buses, programming languages, transport, deployment topology, storage backends, and observability stack are all reference-implementation concerns, not protocol concerns; the Protocol Contract in this Codex §1 says nothing about them, deliberately. An alternative implementation may choose a different database, a

different language entirely, and a different transport, and remain fully Extropyconformant provided it satisfies the Protocol Contract in this Codex §3 through §9, ships at least one domain measurement operator M_d satisfying §4.6 of this Codex §4, passes the three Non-Extraction tests, and publishes and enforces its formula version.

15.10 Verified Implementation Status

As of this writing, the core protocol services listed in §15.3 (xp-formula through credentials) are active in the sense that their source exists, builds, and has at least some test coverage; the v3.1 skeleton packages (§15.5) are specification-frozen but implementation-partial; the application verticals (§15.6) vary in maturity, with HomeFlow being the most mature (Appendix G); and github-parasite is an explicit scaffold with no production runtime. This Codex states this honestly rather than implying uniform maturity across the package tree, consistent with the sandbox-implementation posture stated in docs/ SPEC_v3.1.md §2 and docs/VPS_NODE.md .

15.11 Test Footprint

packages/xp-formula/src/index.test.ts carries seventeen property tests covering bounded XP, logdecay bounds, sub-second attack neutralization, non-zero minting for realistic settlement times, and the compile-time invariant that reputation cannot enter the formula (§4.8, §4.2). Test coverage across the remaining core services varies, and a full coverage audit is itself one of the honest gaps this Codex does

not paper over; see §19 and §20 of this Codex for the categories of testing work still open, including consensus mechanism edge cases, retroactive-validation edge cases under validator churn, and carteldetection threshold validation.

15.12 Build and Run

The monorepo uses pnpm workspaces (pnpm-workspace.yaml , .npmrc , PACKAGE_MANAGER.md) with a shared root tsconfig.base.json and per-package tsconfig.json files. Docker Compose configurations exist for the core mesh (docker-compose.yml) and for the HomeFlow and GrantFlow verticals specifically (docker-compose.grantflow.yml , deploy-homeflow.sh). This Codex does not reproduce the full build instructions here; readers standing up the stack should follow README.md at the repository root, which is the maintained, authoritative build guide.

15.13 Documented Drifts

Two categories of documented drift between specification and implementation are worth naming explicitly rather than discovering by surprise. First, the verdict-vocabulary drift described in §10.10, where confirmed and supported coexist as affirmative verdict values pending consolidation. Second, the F-symbol naming collision described in §4.7, where the reference implementation's THREE_LAYER_SEPARATION.md and CONTRIBUTION_GRAPH.md still use F to mean frequency-of-decay in some contexts, while this Codex and the Protocol Contract in this Codex use F to mean falsifiability. Both drifts are tracked in §20 of this Codex and are not hidden by this Codex's exposition.

15.14 Production Deferral Catalog

Several architecturally decided features are operationally deferred rather than unbuilt by oversight: crossnode gossip protocol hardening (§11.10), full ZKP circuit auditing for the identity layer (§12.9), the quorum size formula for variable-domain validator rings (§20 of this Codex item 1), and cartel-detection threshold finalization beyond the game-theoretic analysis in §16.1 all fall into this category. Section 19 catalogs these systematically as part of the honest gap accounting this Codex commits to throughout.

Falsifier. If this section names a package, file path, or port number that does not exist in the live repository at the reference implementation repository at the time a reader checks, that is a factual error in this Codex and should be filed

as a correction; every name in this section was checked against the repository during drafting.

16. Public Narrative Layer

In plain terms: this section maps the different documents and materials the project produces, who each one is for, and how they relate to this Codex, so a reader arriving from any one of them knows where to go next.

16.1 The Public Site

lladnaros.com functions as the cultural entry point, primarily artistic and brand-oriented, aimed at a curious visitor rather than a technical evaluator. It is intentionally not the place where architectural claims are adjudicated; that happens in this Codex and the repository.

16.2 The Companion Book

The companion book, Randall Gossett, Unfuck the World for a Dollar (companion book; in progress), makes the narrative and civilizational case for the protocol in a conversational register, accessible to readers without a technical background. Its diagnosis, spread across its opening chapters, is that existing systems for measuring and rewarding contribution fail in predictable, mechanical ways: test scores stand in for learning and get gamed, credit scores stand in for trustworthiness and become tools of social control, engagement metrics stand in for cultural contribution and become advertising surfaces, and market prices stand in for value while concentrating wealth in whoever can manipulate prices. The book's thesis is that this pattern is the default behavior of any system that puts incentive weight on a representation, and that fixing one specific instance does not change the underlying dynamic, since the new metric becomes the new target. Section 2 of this Codex distills this diagnosis into technical register; the book's own treatment is deliberately more expansive and rhetorical, and this Codex does not reproduce it in bulk. The book's middle chapters walk through the engine's components in narrative form, covering the XP formula, the eight domains, the core loop, the DAG, DFAOs, and the tokens, as the narrative counterpart to what this Codex specifies precisely. Its later chapters address the standard objections: whether the system can be gamed (answered by the Goodhart-resistance architecture in §6 and the retroactive-burn mechanism), whether bad actors can take over (answered by the Sybil-resistance and validator-collusion analysis in §16 and §12), whether the state can co-opt it (answered by Digital Autarky and the thresholdreveal scheme in §12), and whether it fails psychologically by assuming people need extrinsic reward to contribute (addressed by the intrinsic-motivation framing in the book's own argument, which this Codex does not attempt to reproduce). The book's closing chapters call for the protocol's deployment, framed honestly as the work of a solo developer without institutional backing, a framing this Codex's Methods Note and author's positioning likewise do not obscure.

16.3 Known Stale Passages

At least one passage in the book's technical chapters has fallen out of sync with the current specification: an early chapter's description of the loop's closed state references R as coming from a "reputation service," which predates the v3.1.2 correction separating rarity from reputation entirely (§5.2, §4.2). This Codex flags the discrepancy explicitly rather than silently, consistent with the correction-ledger discipline in §4.6; future printings of the book should incorporate the correction.

16.4 Academic Papers

Three formal papers ground specific claims made throughout this Codex. Randall's Feedback V4: Formal Foundations for Emergence-First Governance supplies the axioms and theorems synthesized in §7. The internal Signal paper on representational fidelity decay supplies the differential-equation model synthesized in §6 and expanded in Appendix K.

The God paper, discussed in Appendix H, proposes a falsifiable functional definition of a contested theological concept and connects it to the protocol's own architecture; it is explicitly a philosophical companion, not part of the protocol specification, and a reader can engage with or skip it entirely without affecting their understanding of the protocol itself.

16.5 The One-Pager

A business-oriented one-pager exists (§16.5 of this Codex) targeting investors and merchants evaluating adoption. It is a compressed pitch, not a technical document, and where it has historically drifted from the canonical domain or token vocabulary (an earlier version listed an incorrect domain set), this Codex's correction ledger discipline applies to it the same as to any other companion document.

16.6 The God Paper as Philosophical Companion

The God paper proposes that a contested theological concept can be reconstructed as a falsifiable functional concept, along the same falsifiability discipline this Codex applies to every empirical claim it makes. Appendix H gives the full synopsis. One terminology note carried over from the correction ledger: an earlier draft of the God paper glossed the symbol F as "feedback closure strength," which is a real and interesting quantity in that paper's own context but is not what F means in the XP formula (falsifiability, per §4.7). Readers moving between the God paper and this Codex should hold the two usages separately.

16.7 How These Materials Connect

The relationship among the public-facing materials, restated as a map: the public site is the cultural entry point; the book is the narrative case for the protocol, accessible to non-technical readers; the one-pager is the business pitch; this Codex, in both editions, is the comprehensive technical specification; the repository is the canonical implementation and the ground-level source of truth; and the academic papers are the formal foundations. A curious citizen reads the public site and the book. An investor reads the onepager and relevant book sections. A merchant evaluating adoption reads the one-pager and §13 and §14 of this Codex. A technical reader integrating against the protocol reads §4, §5, §9, §10, §11, §12, §14, and §15 of this Codex, then drops into the repository. A peer reviewer reads §4 through §7, §17, and §18, then drops into the relevant academic papers. This Codex functions as the structural index across all of these materials.

Falsifier. If any companion document's substantive claim about the protocol's architecture (as opposed to its narrative framing) contradicts this Codex, that is a correction-ledger item and should be filed, per the discipline stated in §4.6 and applied throughout this section.

17. Security, Attack Surfaces, and Failure Modes

In plain terms: this section assumes the world is adversarial and asks, systematically, what an attacker would try, whether the protocol's design actually stops it, and where the honest answer is "not fully, yet."

17.1 The Adversarial Model

The protocol's threat model assumes that a non-trivial fraction of participants will attempt to game any visible metric, that some fraction of contributors performing validation tasks will collude if collusion is profitable, that Sybil attackers will create multiple identities if identity is cheap, that reputation-laundering services will emerge if reputation is valuable and gameable, that regulators may attempt to compel disclosure beyond what the selective-disclosure layer would otherwise permit, and that cryptographic primitives may be broken or deprecated on a multi-decade horizon. The security posture that follows is structural, not exhortative: it does not ask actors to behave well, it makes misbehavior expensive or architecturally foreclosed.

17.2 The Six Primary Attack Surfaces

XP inflation via formula gaming. An attacker attempts to maximize XP by manipulating one of the five formula terms in §4.1. Defense against R inflation: R lives in a per-domain table, not bound to any actor, so inflating it requires a visible, contestable governance proposal (§14.5). Defense against F evasion: the fingerprint tuple of submitter, claim type, primary domain, and DFAO prevents trivial relabeling, though cross-DFAO fingerprint arbitrage remains an open vector (§17.4). Defense against ΔS inflation: validators verify the claim against the domain's instrument and stated baseline, and the retroactive-burn window (§9.6) catches false claims that initially pass validation. Defense against w · E gaming: w is governance-tunable, but governance proposals are visible, and E is per-claim and must be supported by evidence. Defense against settlement-time inflation: the T_floor and the explicit log-decay cap bound the maximum value of that multiplier (§4.8). Reputation laundering via XP. This attack is structurally foreclosed by the architectural invariant stated in §4.2 and §10.1: 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 arguably the single most important architectural choice in the protocol, and it is the one this Codex returns to most often precisely because it is the choice every prior reputation system has failed to hold. Validator collusion. A pool of contributors performing validation tasks conspires to confirm fraudulent claims for a share of the payoff. Defenses: the retroactive validation window allows independent observers to submit falsifying evidence after the fact; validators whose consensus is later contradicted take a reputation penalty exceeding the expected collusion gain; one-tenth blind-slice validation prevents any single validator from understanding the full claim, raising the coordination cost of collusion; and SignalFlow routing prevents validators from selecting which claims they validate. The game-theoretic floor, worked out in §16.1 below, is that at validator population N of 10 or more with detection probability of 0.3 or more, no cartel strategy is profitable in expectation. Sybil attack. An attacker creates multiple identities to manipulate consensus or multiply rewards. Defenses: the hybrid identity layer (§12.2 through §12.3) raises the cost of identity creation through ondevice KYC; the per-context nullifier (§12.4) prevents an attacker from acting as multiple identities within a single context; and Sybil attack cost scales with the number of honest loops required to compromise consensus as the network grows. Open issue: the three KYC options, ID scan, biometric, and trustedissuer handoff, have different Sybil-resistance properties, and the per-method threat model, while documented, shows some methods are meaningfully weaker than others. Identity compromise. An attacker steals a participant's private key and acts as that participant. Defenses: the PSLL hash chain makes mutation of past entries detectable (§12.7); a participant can rotate their DID, with the new DID bound to the prior identity through a documented rotation event on the DAG; and the threshold reveal scheme does not depend on the user's private key at all, so even a fully compromised user cannot be re-identified without cooperation among 7 of the 12 shareholders. Regulatory compulsion. A state actor compels the protocol to reveal user identity or restrict user actions. Defenses: the threshold reveal scheme ensures no single actor, including the protocol's own infrastructure operators, can comply unilaterally with a compulsion order, since compliance requires cooperation among 7 of 12 ecosystem-validator shareholders under a legitimate cause-shown process; and the user's identity material lives on the user's device, so the protocol cannot reveal what it structurally does not have. Open

issue: the regulatory exposure of any redemption surface resembling a fiat bridge, addressed architecturally by Non-Extraction in §13 but not thereby immunized against every possible regulatory theory.

17.3 Goodhart Pressure as a Permanent Feature

The representational fidelity model in §6 predicts that any metric the engine uses will eventually become a target. The protocol does not claim immunity from this; it claims structural resistance, visible correction, and diagnostic use of failure rates when they occur. Goodhart pressure manifests across several surfaces: actors may attempt to convince governance to set high R values for action classes they specialize in, which is visible and contestable since rarity-table revisions are governance proposals; actors may attempt to concentrate the weight vector w on domains they excel in, mitigated at the DFAO level but with crossDFAO arbitrage remaining a vector; actors may attempt to game SignalFlow's scoring inputs to route preferred validators to their own claims, limited by the routing function's determinism given its

inputs; and actors may attempt to convince the network to adopt instruments they can manipulate, mitigated by instrument adoption being governance-tunable and visible, with manipulated instruments producing observable drift. The protocol's response in every case is uniform: surface the pattern, route it through governance, revise the parameter, the instrument, or the routing weight, and preserve the DAG record so the lineage of revisions stays auditable.

17.4 Cross-DFAO Fingerprint Arbitrage

A specific, correctly identified attack vector: a contributor who submits the same type of claim across many different DFAOs receives a fresh, undecayed F for each new fingerprint combination, because the DFAO identifier is part of the fingerprint tuple. In a sufficiently large ecosystem with many DFAOs, this could functionally neutralize F's anti-grind intent for a prolific cross-DFAO contributor. This is documented as an open issue with three proposed mitigations, none yet implemented: a global frequency counter that decays F across DFAOs, weighted by claim similarity; a substantive-variation requirement, where the validator pool must confirm a claim's cross-DFAO presentation involves genuine variation rather than trivial relabeling; and a longer-term move to reputation-weighted F, modulated by the contributor's cross-DFAO contribution history. This Codex documents the gap rather than the fix, because the fix does not yet exist in the reference implementation.

17.5 Loop Lifecycle Race Conditions

The loop lifecycle state machine (§8.4) must handle concurrent events correctly. Concurrent validation submissions, where two validators submit verdicts at the same instant, are serialized by the event bus and both recorded, with the aggregator using the time-ordered set. Simultaneous consensus and falsification, where a loop's aggregation reaches closure threshold at the same instant an independent observer submits falsifying evidence, are resolved by prioritizing the latest event by timestamp: if falsification arrives during the validation phase, the loop moves toward rejection; if it arrives after closure but within the retroactive window, it triggers the re-evaluation process described in §9.6. Cross-service ordering guarantees are limited: the event bus guarantees in-order delivery within a single topic but not across topics, so services depending on cross-topic ordering must use the DAG's own vertex sequence as canonical rather than assuming cross-service message order. The state machine's fully formal transition table has not yet been published as a standalone artifact; this is tracked as an open documentation item.

17.6 PSLL-DAG Divergence

A subtle attack: a participant mutates their own PSLL retroactively, then anchors a new Merkle root that does not reflect the original entries. This is detected because each PSLL entry is hash-chained to its predecessor, so mutating a past entry invalidates every subsequent hash in the chain, and the Merkle root anchored to the DAG commits to the entire PSLL up to the anchor point, so an anchor referencing a mutated entry fails Merkle-proof verification. On detected divergence, the protocol does not silently overwrite either record; a dispute review opens instead. The DAG anchor is canonical for the network; the participant's PSLL is canonical for the participant. Divergence triggers a process, not a silent loss of integrity in either direction.

17.7 The Gödel Boundary

A self-referential claim in the contribution graph can produce a genuine paradox: a claim that asserts the falsity of its own validation, or a governance proposal that would nullify the protocol's own capacity to govern itself. A watchdog component intended to detect and quarantine such claims is on the roadmap but not implemented as of this writing. Self-referential claims remain an open, low-priority vector. This Codex states plainly that the protocol handles the overwhelming majority of contribution claims without any paradox risk, and that certain edge cases require formal handling that does not yet exist.

17.8 Cryptographic Agility

The protocol's hash and signature defaults, BLAKE3 with a SHA-256 fallback for content hashes, Poseidon where ZKP circuits require an arithmetic-friendly hash, Ed25519 for entry signatures, and BBS+ for selective disclosure, are all governance-tunable. If a primitive is deprecated or broken, governance can adopt a new default, and adoption is forward-only: existing PSLL entries and DAG vertices retain their original cryptographic commitments, while new entries use the updated defaults. This is the operational form of cryptographic agility: the protocol does not lock itself to 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 deserves explicit naming: cultural resistance to protocol-level conventions that conflict with established institutional rhythms, most concretely the optional base-10 Universal Times calendar (Appendix P) conflicting with the deeply embedded seven-day week and Gregorian calendar. If a protocol's conventions fight entrenched institutional habit, adoption suffers regardless of technical merit. This is tracked as a P3 cultural-friction risk, and the mitigation is architectural: the Universal Times layer is optional and DFAO-gated rather than protocol-mandatory, so a DFAO that wants the base-10 calendar can use it, and a DFAO that does not can ignore it entirely without losing protocol conformance.

17.10 What the Threat Model Does Not Cover

This Codex does not claim coverage of side-channel attacks on user devices, since Digital Autarky's sovereignty assumption presupposes a reasonably secure device and the protocol cannot defend a user whose device is already compromised; supply-chain attacks on the reference implementation, since the code is open source and users compile and run it at their own risk; catastrophic, sudden cryptographic breaks, since cryptographic agility (§17.8) addresses gradual primitive deprecation but not a sudden break, such as a practical quantum attack against Ed25519, on an emergency timeline; or adversarial cultural narratives that delegitimize the protocol without engaging its mathematics at all. These are real risks, and the protocol does not pretend to address them within its specification.

17.11 The Honesty Clause

This Codex inherits, and extends, an honesty clause first stated in this Codex §2: the current implementation is a sandbox, not a hardened production system, significant gaps remain, and the model is allowed to lose. The honest enumeration of failure modes in this section, the explicit gap catalog in §19, and the documented correction ledger in §4.6 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 only to be examinable, and this section is the examination.

Falsifier. If adversarial simulation with current parameters fails to converge to a majorityhonest settlement over cartel size N of 10 or more within a bounded number of loops, the cartel analysis referenced above and detailed in §16.1 is falsified, and either the parameters or the underlying mechanism must be revisited.

18. Peer Review Response and Applied Fixes

In plain terms: this section documents a real external technical review the specification went through, what it found, and what changed as a result. It is included in full because a specification that hides its review history is less trustworthy than one that shows its work.

18.1 The Peer Review

An earlier version of this specification (v3.1.2) was submitted to external technical peer review. The reviewer's overall verdict: strong architecture with credible philosophical grounding, with several formula-level issues requiring resolution before the specification could be considered fully rigorous. The reviewer commended several aspects specifically: the Digital Autarky principle as principled and architecturally enforced; the single-source-of-truth pattern for the XP formula as correct engineering; the R and F separation correction as important and clearly stated; the epistemology-engine redefinition, from central decomposition to mesh observability, as the correct architectural fix; the provisional-defaults-table pattern as the right approach to governance bootstrapping; the honesty clause distinguishing this specification from typical cryptocurrency whitepapers; and the treatment of Goodhart pressure as diagnostic fuel rather than a fatal flaw as the correct epistemic stance. The reviewer identified specific issues at four priority levels, P0 through P3, each addressed below.

18.2 P0 Issues and Resolutions

P0-1: a naming collision between F as frequency-of-decay and the validation-aggregation output. An earlier draft stated that aggregation of one-tenth blind-slice validation "produces F," which collides with F's role as frequency-of-decay in the XP formula. Resolution: the validation-aggregation output was renamed to a separate symbol (V_c, validation confidence, in the historical fix; this Codex's current convention additionally resolves the underlying naming question by adopting F as falsifiability throughout per §4.7, with the frequency-of-decay quantity now named F). The two concepts are kept distinct in all documentation going forward.

P0-2: a false boundedness claim about the settlement-time term. An earlier draft claimed the $\log(1/T_s)$ term "preserves boundedness," which is mathematically false as originally stated: the function is unbounded above as T_s approaches zero. Resolution: the specification introduced a minimum T_s floor, T_floor, governance-tunable with a default of 0.01, and an explicit cap on the log term at log(1/T_floor). This is the direct ancestor of the v3.1.3 fix described in full in §4.6 Correction 3 and §4.8 of this Codex.

18.3 P1 Issues and Resolutions

P1-1: cross-domain ΔS normalization not formally addressed. An earlier draft asserted that domainspecific entropy measures were instances of the same phenomenon without showing how they were normalized before aggregation. Resolution: the specification introduced the bits-equivalent common unit and the per-domain measurement operator M_d, now stated in full in §6 and §7 of this Codex, with the calibration of specific M_d instances acknowledged as preliminary, open engineering work, catalogued in §19. P1-2: the irreducible-form derivation was absent. An earlier draft introduced the now-retired irreducible-XP form as a structural analogy without deriving its relationship to the canonical formula. Resolution at the time: the specification stated that the operational formula was the five-term version and that the irreducible form was a structural analogy without a formal derivation connecting the two. This Codex's current resolution goes further: the irreducible form is fully retired (docs/REJECTED_FRAMINGS.md R1), not merely caveated, because the v3.1.3 bounded formula makes the irreducible branch unnecessary (§4.8, §4.6 Correction 3). P1-3: PSLL cryptographic primitives were unspecified. An earlier draft described the PSLL as hashchained and cryptographically signed without pinning specific primitives. Resolution: the specification pinned BLAKE3 with a SHA-256 fallback for content hashes, Poseidon where ZKP circuits require it, Ed25519 for entry signatures, and BBS+ for selective disclosure, all governance-tunable per the cryptographic-agility mechanism in §17.8. P1-4: token economy enum members were unexplained. An earlier draft's token list included unexplained abbreviations. Resolution: the specification now defines all six canonical tokens in full (§4.3, §10.1), and explicitly marks two historical names, GT and RT, as not canonical: they appeared in older drafts and transitional code but are excluded by the six-token hard rule in §8 of this Codex and §1.4 and Appendix M of this Codex .

18.4 P2 and P3 Issues and Resolutions

P2 issues, addressed with documentation and partial mitigation rather than full resolution: cross-DFAO F fingerprint arbitrage (§17.4 of this Codex, mitigations identified, implementation deferred); the SignalFlow reputation-routing cross-layer coupling (documented explicitly in §10.3 and §7.4 as intentional, not accidental); the per-KYC-method Sybil-resistance threat model (documented in §17.2, with the honest acknowledgment that methods vary in strength); and DFAO inheritance conflict resolution (addressed structurally in §14.3 and §14.6, with escalation-rule specifics still open). P3 issues, acknowledged and partially resolved: the 7-of-12 reveal-threshold derivation (addressed in §9.6: under threshold-signature semantics, an adversary controlling fewer than 7 of 12 shareholders cannot reconstruct the reveal secret, and the ratio provides margin above a simple majority while remaining practically achievable, and remains governance-tunable); the loop lifecycle's lack of a fully formal transition table (acknowledged in §17.5 as still open); calibration of domain-specific constants

such as the retired c_L values (moot for the retired irreducible form, but the broader calibration challenge persists for M_d per §8.13); and an incomplete formal bibliography (this Codex's References section, §23, addresses this directly with inline citation by name and, where available, full bibliographic detail).

18.5 What the Peer Review Did Not Catch

The peer review was exhaustive but not omniscient. Several issues surfaced during this Codex's own drafting process that the original review did not explicitly flag: an internal contradiction in a shared-types file's header comment that retained an old field definition after the field itself was renamed elsewhere in the same file, since resolved by updating the header in lockstep; a stale passage in the companion book referencing a "reputation service" for R, noted in §16.3; an earlier one-pager's domain-list error, which listed an incorrect domain set relative to the canonical one reconciled in §4.4; the God paper's gloss of F as "feedback closure strength," noted in §16.6; a port collision between two temporal-related packages (temporal and temporal-service), reconciled in the port table in §15.3, where both are now listed with their distinct roles; and drift between an ecosystem-scale label used informally in project materials and the formal five-scale DFAO enum stated in §14.1.

18.6 The Peer Review's Verdict, Updated

The original verdict, restated: strong architecture with credible philosophical grounding, with formulalevel issues requiring resolution before the specification could be considered fully rigorous. This Codex represents the post-peer-review state: the P0 issues are corrected, the P1 issues are addressed with some open calibration work explicitly acknowledged, and the P2 and P3 issues are documented with mitigation plans rather than silently deferred. The protocol's claim of rigor is stronger now than at the time of the original review, but it is not absolute, and this Codex's open gap catalog in §19 is the explicit acknowledgment of what remains unproven.

18.7 The Honest State of the Specification

A short, direct statement of where the specification currently stands: the XP formula is well-defined and single-sourced in code, at packages/xp-formula/src/index.ts . The R, F, and reputation-density separation is an architectural invariant and is enforced, not merely stated. The eight canonical domains (per the reconciliation in §4.4) and the six canonical tokens are fixed. The validation model is specified at the architectural level, with emergent validation (§10) as the governing frame. The DAG substrate, the loop lifecycle, the identity layer, the PSLL, and the governance layer are all specified in this Codex and crosschecked against the live repository. The peer-review-identified P0 errors are corrected. Cross-domain ΔS normalization is documented, with calibration still preliminary. Several skeleton services (PSLL-sync, quest-market, validation-neighborhoods) have specified interfaces but partial implementations. Production-grade cryptographic hardening (final primitive selection, gossip protocol specification, hardened transport) is deferred to the operational-hardening roadmap in §19.

Falsifier. If a future audit of this Codex or the reference implementation identifies a P0-severity issue, meaning a claim this Codex states as an enforced architectural invariant that is not, in fact, enforced in shipped code, that is a critical finding requiring the same correction-ledger treatment given to the v3.1.2 review's own P0 findings, documented in a future revision of this section.

19. Local Flow and Network Bootstrapping

To move from a theoretical ledger to a populated network, the protocol must not bootstrap a global XP market on day one. Doing so invites algorithmic exploitation of whatever is easiest to fake. Codex v2.1 solves this with Local Flow: a restriction of execution and validation to dense, geographically or socially proximate neighborhoods until those neighborhoods have statistically significant honest history.

This section covers two related but distinct uses of the name.

1. Local Flow the bootstrap protocol — the network-wide rule for how early quests are routed, validated, and promoted. This is the architectural meaning.
2. @extropy/localflow the vertical — a DFAO application (rides, grocery runs, local errands) that emits loops into the engine without showing users XP vocabulary. That vertical is one implementation of the bootstrap idea, not the definition of it.

19.1 Definition

Local Flow restricts the execution and validation of initial quests to overlapping trust graphs where offline reputation can backstop online claims. Claims are not immediately tossed onto an anonymous global DAG. A neighborhood is a set of actors with physical proximity, social proximity, or both, plus a shared DFAO scope at NANO or MICRO scale.

A Local Flow quest has the same loop lifecycle as any other quest (proposed → open → evidence-submitted → validator-verdicts-collected → closed | rejected → minted → confirmed | burned). The difference is the routing domain: SignalFlow is constrained to the neighborhood until promotion criteria fire.

19.2 Why global minting on day one fails

Goodhart's Law is not a warning about bad people. It is a warning about visible targets. A global XP market, before any neighborhood has an honest history, makes the measure the target for every actor who can spin identities or grind a cheap domain. The hybrid identity layer raises the cost of identity creation. It does not make the first week safe. Local Flow is the time-domain complement of the identity layer.

19.3 Goodhart-resistance mechanisms

High-context verification. Validators have physical or high-context digital proximity to the event. Verifying a local ecological cleanup, a civic repair, or a completed ride is a different epistemic problem from verifying a screenshot from an anonymous node. $\mathscr{F}$ is easier to evaluate when the sensors are community members with line of sight.

Reputation-seed generation. The primary purpose of early Local Flow is not massive XP generation. It is establishing un-gamed reputation histories. High-frequency, low-stakes quests build cryptographic histories of honest validation. Those histories later weight slice scores. They never enter the XP formula as a multiplier.

Qualitative dampening (w-throttle). The Epistemology Engine may dampen domain weights w if it detects anomalous spikes in easily gamed domains. This is a temporary bootstrap control. It is logged, bounded, and itself a governance-visible knob. It is not a silent slashing of named actors.

Multi-party convergence. In the localflow vertical, minting requires client confirmation and driver completion. Solo actions cannot mint. The convergence vertex appears in both person-DAGs. That is structural fraud resistance, not a policy preference.

19.4 Gradual algorithmic scaling

As a neighborhood reaches statistically significant density — enough independent actors, enough confirmed loops, enough slice-agreement — Local Flow gates widen. Nodes with high local standing are promoted to serve as validators on cross-neighborhood blind slices. That is how the global trust layer is seeded: with actors who already have a record of being wrong or right in public, locally.

Promotion is algorithmic and reversible. A promoted node that fails on blind slices loses cross-neighborhood weight. It does not lose the right to contribute locally.

19.5 The localflow vertical, as shipped

The reference vertical `@extropy/localflow` is a matchmaking service. Users do not see XP, EP, or DAG terminology. Completed tasks emit LOOPOPEN, LOOPCLOSE, and XPMINT vertices. Default domain weights for that vertical are economic-heavy and temporal-heavy (economic 0.45, temporal 0.20, with the remainder distributed). Port 4030. Prototype store is in-memory; the production path is Postgres plus the DAG substrate.

The vertical currently computes a local XP helper. That helper is not an authority. The only legal mint is the canonical-v3.12 path in §4. Any vertical-local formula is a display estimate until the mint service stamps `canonical-v3.12`.

19.6 Relation to HomeFlow

HomeFlow (Appendix G, Appendix L) is the family-scale pilot: chores, meals, inventory, household coordination. It is a NANO DFAO. Local Flow is the general bootstrap rule that HomeFlow, the localflow vertical, and any other neighborhood vertical must obey. HomeFlow does not get a special mint formula. It gets a special social context.

19.7 What Local Flow is not

It is not a geographic restriction on who may eventually participate. It is not a KYC dragnet. It is not a claim that offline reputation is ungameable. It is a staging rule: do not ask a global anonymous graph to be the first verifier of a measure that will later carry incentive weight.

Falsifier. If the first 1,000 confirmed mints on a live network are dominated by a single domain, a single action class, or a single tightly coupled identity cluster, Local Flow failed as a bootstrap. If any vertical mints XP from a formula other than canonical-v3.12, Local Flow failed as a protocol citizen.

20. Open Engineering Gaps and Roadmap

In plain terms: this section is the complete, current, honest list of what is not yet built or not yet proven, organized by how urgently it blocks further progress. A specification without a gap list like this one is not more finished, it is less honest.

20.1 The Honest Gap Catalog

There are 65 identified engineering gaps across 13 categories, enumerated in full in §20 of this Codex . Gaps are not failures. They are the engineering backlog, and acknowledging incompleteness is a prerequisite for systematic completion, and the only honest register for a specification describing a live, evolving system.

20.2 P1 Gaps: Critical Path (26 total)

Consensus mechanism details, 7 gaps: the quorum size formula for variable-domain rings; validator collusion detection thresholds; tie-break rules for split-quorum outcomes; late-arriving validation vote handling; the interaction between consensus finality and retroactive burn; cross-domain consensus weighting; and consensus failure recovery or re-validation protocol. Economic attack resistance, 6 gaps: a formal cartel-threshold analysis above 50 percent domain reputation concentration; wash-loop detection across colluding identities; bribery resistance under IT decay; validator bid-rigging mitigation; funded-validator, meaning corporate-capture, defenses; and CT lockup parameter optimization. Validator selection optimization, 5 gaps: tuning of the 4-factor SignalFlow weighting (domain, reputation, load, accuracy); cold-start validator bootstrapping; geographic and language balancing in routing; adversarial-load shedding policy; and Sybil-resistant load distribution under burst traffic. Cross-domain measurement calibration, 6 gaps: ΔS unit harmonization across the eight domains, which is the operational form of the F1 falsifier from §7 (Concise edition numbering; §6.2 mechanics restated in §8.13 of this Codex); falsification-condition specification for the cognitive domain; falsification-condition specification for the social domain; falsification-condition specification for the governance domain; calibration-drift detection and auto-replace policy; and inter-domain ΔS comparison weighting. Verdict vocabulary standardization, 2 gaps, added subsequent to the original 63-gap count: the canonical affirmative verdict value, resolving the confirmed versus supported collision noted in §10.10; and API field-naming consistency across services, needing a shared contract-test suite.

20.3 P2 Gaps: Robustness and Security (23 total)

DAG distributed consensus, 5 gaps: causal-edge gossip protocol specification; partition tolerance and merge rules; DAG garbage collection and pruning policy; replay-attack protection; and PSLL-anchor receipt cadence tuning. Retroactive validation specifics, 4 gaps: edge cases in the retroactive window under validator churn; burncascade limits when one loop's burn invalidates dependent loops; settlement reliability under network partition; and retro-validation incentive structure refinement. DFAO governance edge cases, 5 gaps: state hand-off protocol during DFAO migration; quorum-loss recovery for the smallest (NANO or MICRO) tier; conflicting proposals across nested DFAOs; influencedecay edge cases for dormant members; and cross-tier proposal escalation rules, referenced in §14.6.

Token economy equilibrium, 4 gaps: IT decay-rate validation at the provisional five-percent-per-month figure; CT, EP, and DT decay-rate finalization; multi-token attack-surface analysis; and token-velocity equilibrium modeling. Privacy and access control, 5 gaps: final ZKP scheme selection between BBS+ and zk-SNARKs for specific use cases; selective-reveal threshold mechanics beyond the provisional 7-of-12; nullifier collision-resistance proof; PSLL selective-disclosure protocol hardening; and cross-DFAO data isolation guarantees.

20.4 P3 Gaps: Ecosystem Maturity (16 total)

Skill DAG design, 3 gaps: skill-node progression criteria; skill-verification source of truth; and skill-graph traversal for SignalFlow routing. Oracle integration protocol, 4 gaps: external-data ingestion trust model; oracle-source diversity requirements; oracle-failure fallback policy; and XP minting from oracle-validated claims. Performance and scalability, 5 gaps: target throughput per validation neighborhood; PSLL localstorage growth bounds; DAG indexing strategy at planetary scale; SignalFlow routing latency targets; and cold-cache warm-up policy. Migration and upgrade paths, 4 gaps: the v3.0 to v3.1 state migration specification; breaking-change governance protocol; rule-module hot-swap procedure; and deprecation lifecycle for retired services.

20.5 Most-Cited Specific Gaps

Across this Codex's own sections, the gaps most frequently referenced back to are: the quorum size formula (§20 of this Codex item 1, referenced from §7.6 and §14.4); the cartel and Sybil-cluster detection thresholds (items 2 and 8, referenced from §14.4 and §16.1); ΔS unit harmonization (item 19, the operational form of the F1 falsifier, referenced throughout §6 through §8); and the verdict vocabulary standardization items (25 and 26, referenced from §10.10 and §15.13).

20.6 Roadmap to v3.2

Near-term work, within the current specification cycle: shipping the Non-Extraction test harness (packages/xp-mint/tests/non-extraction.test.ts) that exercises the T1, T2, and T3 tests from §13.4 against the public surface of every mint-adjacent package; landing domain-specific M_d implementations for at least three of the eight domains, with per-domain calibration tables and validator-neighborhood recipes, informational first per §8.13; reconciling the F naming collision surfaced in §4.7 and §10.10 across the affected reference-implementation documents; closing the reserved retirement entries R2 and R3 in docs/REJECTED_FRAMINGS.md , either by subtractive rewrite or by producing the missing derivations; and formalizing the cartel-threshold analysis from §16.1 in light of the v3.1.3 T_floor correction.

20.7 Roadmap to v4.0

Longer-horizon work: production-hardening the transport layer beyond the current HTTPS-and-bodysigning sandbox posture (§11.10); finalizing the quorum-size formula and cross-domain consensus weighting from the P1 list above; completing at least three domain M_d implementations to the adversarial-testing standard required by the adoption criteria in §19.9; and reconciling the domainvocabulary and token-vocabulary drifts noted in §4.4, §4.7, and §14.4 across every companion document, not only this Codex.

20.8 The Sandbox Posture

The current codebase and any live deployment of it should be treated as a sandbox implementation of the architecture described in this Codex: a live engineering testbed used to exercise assumptions, expose failure modes, and close remaining gaps, not a hardened, adversarial-internet-ready production deployment, and not a recommendation to run the stack as-is on arbitrary public infrastructure (docs/ SPEC_v3.1.md §2, docs/VPS_NODE.md). This is not an apology. It is the honesty clause restated in §17.11, applied specifically to deployment maturity.

20.9 Falsification Conditions for the Whole Project

Stated together, in one place, for a reader who wants the complete list of ways this project fails: if the architectural invariant that reputation does not enter XP cannot be maintained in deployed practice (§4.2, §7.4); if the eight canonical domain instruments all turn out to be equally Goodhart-vulnerable, meaning no domain achieves a defensible M_d under the invariants in §7 (Concise numbering; restated mechanically in §8.13); if cross-domain normalization cannot produce stable calibrations, meaning falsification condition F1 from §6.2 is confirmed; if validator collusion can be sustained below the predicted detection threshold from §16.1; or if users prove able to farm metrics more efficiently than they can produce real entropy reduction, with the gaming cost lower than the protocol's defenses anticipate, then the project has failed at the level this Codex is answerable for. This Codex documents these conditions explicitly, in advance, so the protocol's success or failure can be assessed against pre-specified criteria rather than retrospectively rationalized after the fact.

20.10 Adoption Criteria

The Codex, in either edition, is adopted, meaning it graduates from a live specification under active revision to a stable reference, when three conditions hold together: the protocol contract in the Protocol Contract in this Codex is stable across two consecutive governance cycles without breaking changes; at least three domains have M_d implementations satisfying §4.6 of this Codex §4's invariants under adversarial testing; and the Non-Extraction test harness passes against every mint-adjacent package in the reference implementation and against at least one independent alternative implementation. None of these three conditions is met as of this writing, and this Codex says so rather than implying otherwise.

Falsifier. This entire section is itself falsifiable on a rolling basis: if §20 of this Codex 's gap count or category structure changes, this section is stale and must be resynchronized. As of this writing the count is 65 gaps across 13 categories, per §20 of this Codex 's own header.

21. Glossary and Formula Reference

In plain terms: this is the lookup section. Every symbol, token, acronym, and formula used anywhere in this Codex is defined here once, so a reader who loses track of a term while reading a later appendix can come back here rather than searching backward through twenty sections.

21.1 Symbol Glossary

Actor. An entity that can open loops, close loops, hold XP thresholds, hold CT, and be observed by contributors performing validation tasks.

b_e (bits-equivalent). The single common unit for cross-domain ΔS. All domain-native measurements are converted to b_e before entering the mint formula. See §6, §7. C. Capability, in the CT formula only. See §5.9. Claim Package. The minimum interoperable payload a personal AI sends to the network: a schemaconformant claim, an identity proof, a PSLL anchor reference, and optional quest-market metadata. See §9.1. Contribution graph. The single graph of all contributions, including validation, that the mesh records. See §9.1. CT (Contribution Token). A per-actor structural coefficient shaping access weight in ties and vote weight in validation. Non-transferable. See §4.3, §5.9. DAG. Directed acyclic graph. The append-only substrate recording loop state, validator verdicts, and mint events. See §11. DFAO. Digital Fractal Autonomous Organization. The unit of governance. Nestable across five scales. See §14.1. DID. Decentralized Identifier. W3C standard for actor identifiers. See §12.3. E (evidence classes). E1 independently reproducible, E2 domain-native, E3 tamper-evident. See §9.5. E (entropy vector). The eight-domain vector of measured entropy reduction per claim. See §4.1, §5.5. ε_d. Domain-specific tolerance on ΔS measurement across validators. See §4.2, §9.5. F (Frequency of Decay). The freshness / repetition factor in the XP formula, in (0, 1]. A property of the action-class fingerprint, not the actor. See §4.4. $\mathscr{F}$ (Falsifiability Index). The epistemic index in [0, 1] used by V(Q) to gate validation. Never a factor in the XP formula. See §4.2. F1 through F4. The four whole-project falsification conditions in §4.6 of this Codex . See §6.2 (mirrored here from the Concise edition numbering; this Codex's §6 through §8 restate the mechanics without renumbering the conditions themselves). KYC. Know Your Customer, in the identity sense. On-device only. See §12.2. L. Local loyalty multiplier, in the EP formula only. See §4.10. Loop. A bounded piece of work that opens, gathers evidence, is scored, and closes. See §9.

λ_d. Per-domain settlement-decay constant, entering T_s. See §4.1, §4.8. M_d. Domain-specific measurement operator converting raw evidence to ΔS_b_e. See §6, §7, §8. Non-Extraction. The architectural invariant that XP and CT never bridge to external transferable value. See §13. PSLL. Personal Signed Local Log. Per-participant, append-only provenance log, anchored to the DAG via Merkle roots. See §12.7 through §12.8. Q_d. Domain-specific quorum function. See §8.4, §7.6.

R (Rarity). Rarity coefficient in the XP formula. Domain-specific, a property of the move, not the actor. See §5.2. Reputation. A property of the actor over time. Lives in CT and per-domain reputation vectors. Never enters the mint formula. See §4.2, §10. ρ (rho, contribution and draw ratio). ΔS produced divided by ΔS consumed, computed on the ledger. See §13.2. Note the separate, contextual use of ρ as reputation density inside the CT formula, per §4.2. ρ_min_d. Domain-specific floor on ρ. Below this, access degrades. See §13.2. SignalFlow. The four-factor routing function assigning quests, including validation tasks, to participants. See §10.3. Sybil. A single actor operating multiple identities. Countered by the identity layer's uniqueness proofs and by nullifiers. See §12.4, §17.2. T_floor. Governance-set settlement-time floor. Default 0.01. See §4.8.

T_s. Normalized settlement-time factor, exp(negative λ_d times Δt), clamped into the interval from T_floor exclusive to 1 inclusive. See §4.1, §4.8. τ_action_d, τ_validator_d. Access thresholds for domain-d actions and for participating in a domain-d validator neighborhood. See §4.2, §10.7, §13.2. Validator neighborhood. A set of actors above the τ_validator_d threshold, participating in a specific loop's closure. Per-loop, per-domain. No actor is a validator in general. See §7.6. w. Domain-weight vector in the XP formula. Per-DFAO override on top of ecosystem defaults. See §5.5. XP. Extropy Points. Non-transferable access threshold. Minted from verified ΔS_b_e under the canonical formula. See §4.3, §4.1. ZKP. Zero-knowledge proof. Wraps identity credentials so the network sees proof of uniqueness without raw material. See §12.3.

21.2 Formula Reference

```
XP = R × F × ΔS_b■ × (w · E) × min(α_max, max(1, T■ / T_floor))
```

```
T■ = exp(-λ_d · Δt) Δt in seconds
```

$T_s \in (T\_floor, 1]$ by construction Formula version: canonical-v3.12 Implementation: packages/xp-formula/src/index.ts

```
CT = C × F × ρ × Δ × E
```

```
EP = XP × L
```

21.3 Domain Glossary

Cognitive, code, social, economic, thermodynamic, informational, governance, temporal: the eight canonical mint-side domains, each with a stated measurement operator M_d. See §8.2 through §8.9. The retired references to a mint-side ecological domain or a reserved spiritual slot are addressed in the §4.4 reconciliation and in §20 of this Codex under naming-hygiene P2.

21.4 Token Glossary

XP, CT, CAT, IT, DT, EP: the six canonical tokens, defined in full in §4.3 and §10.1. GT and RT are explicitly not canonical; they appear only as historical artifacts in some companion documents and should be treated as errors where found, per the correction-ledger discipline in §4.6.

21.5 Loop Lifecycle States

proposed → open → evidence-submitted → validator-verdicts-collected → closed | rejected ↓ minted → confirmed | burned

See §8.4 for the full state-machine treatment and Appendix B for a worked example.

21.6 Acronyms and Initialisms

BBS+: a zero-knowledge proof scheme supporting selective disclosure, the default in the identity layer (§12.3). CAT: Capability Token (§4.3). CT: Contribution Token (§4.3). DAG: directed acyclic graph (§11). DFAO: Digital Fractal Autonomous Organization (§14.1). DID: Decentralized Identifier (§12.3). DT: Domain Token (§4.3). EP: Emergence Points (§4.3, §4.10). IT: Influence Token (§4.3). KYC: Know Your Customer (§12.2). M_d: domain-specific measurement operator (§6). PSLL: Personal Signed Local Log (§12.7). RF: Randall's Feedback (§7). SIIR: Sense-Infer-Integrate-Respond, the RF dynamical loop (§7.3). XP: Extropy Points (§4.3). ZKP: zero-knowledge proof (§12.3).

21.7 Service and Port Reference

See the full table in §15.3 for the complete, cross-checked package-to-port mapping as of this writing. Key entries repeated here for quick lookup: xp-mint on 4005, loop-ledger on 4003, signalflow on 4002, reputation on 4004, dag-substrate on 4008, dfao-registry on 4009, governance on 4010, token-economy on 4012, identity on 4101, psll-sync on 4102, quest-market on 4103, validation-neighborhoods on 4104, epistemology-engine on 3002, homeflow on 4015, and api-gateway on 3000.

21.8 Governance Default Reference

See the full table in §14.4 for current provisional defaults. Key entries repeated here: T_floor at 0.01, retroactive validation window at 30 days, CT lockup at 14 days, IT decay at 5 percent per month, and the SignalFlow four-factor weights at (0.35, 0.30, 0.15, 0.20).

21.9 The Three Architectural Invariants, Restated

Non-extraction to external markets (§13). Cross-domain normalization with a stated falsifier contract (§6, §7). A bounded settlement-time factor with a governance-set floor (§4.8). A fourth invariant, the implementation-agnostic protocol contract separating specification from reference codebase (§15.9, the Protocol Contract in this Codex), completes the set of four architectural invariants this Codex, in both editions, exists to state, defend, and make falsifiable.

21.10 Closing Note for This Section

Every symbol, token, and acronym in this glossary is used with exactly one meaning throughout this Codex, per the naming-hygiene invariant stated in §4.1 and §4.7. Where a symbol legitimately carries more than one meaning across different formulas (E, ρ), that is flagged explicitly both here and at the point of first use in the relevant section. If a future reader finds a use of any symbol in this document that contradicts this glossary, that is a documentation bug and should be filed against the repository.

Appendix A. Extended Treatment of the XP Formula Terms

A.1 Why Five Terms and Not Four or Six The XP formula has exactly five multiplicative terms, and the count is not aesthetic. Each term answers a distinct question the protocol requires for value to mint. R asks: was this contribution class scarce in the network? F asks: has this action class already been done, and how often? (Frequency of Decay.) ΔS_b_e asks: did disorder actually decrease, by how much, against which baseline? w · E asks: how well do the contribution's measured effects match what this community values? The bounded settlement-time factor 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 would let trivial contributions earn the same as scarce ones, encouraging volume over substance. Removing F would let class-grinding earn full value on every repeat, which collapses scarcity into volume. $\mathscr{F}$ is not this

term. Removing the $\mathscr{F}$ gate (not the F factor) would let unfalsifiable claims reach minting at all. Removing ΔS_b_e would let claims mint without demonstrating any actual disorder reduction. Removing w · E would let singledomain trivia outweigh genuine multi-domain substance. Removing the settlement-time term would let abandoned or perpetually stalled loops accrue value as readily as promptly completed ones. Adding a sixth term would have to answer a sixth distinct question the existing five do not capture; no such question has survived proposal. A reputation-flavored sixth term has been proposed and rejected outright, because it would violate the hard separation stated in §4.2 and §10.1.

A.2 The Multiplicative Structure, Restated in Full Section 3.4 stated the reasoning in brief; this appendix restates it with the full consequence analysis. Addition treats the five terms as independent sources of value that can compensate for one another, which fails the conjunctive test the protocol requires: a submitter with high rarity but zero entropy delta could still earn XP under addition, which is incoherent since the system would be minting value against claims that demonstrably reduced no disorder. Multiplication enforces necessity: every term must be non-zero for XP to mint at all, and a submitter cannot compensate for missing evidence with high rarity, nor can a trivial submission reach high XP through speed alone. One consequence deserves restating precisely: when any single term is small, the resulting XP is small regardless of the size of the other four terms. A high-magnitude entropy reduction in a low-rarity action class still mints a small amount of XP, because the rarity term is small. This is intended: the protocol rewards the combination of substance and scarcity, never substance alone or scarcity alone.

A.3 R Calibration in Practice The rarity table is the most contested governance object in the protocol, because setting R values is precisely where the network's judgment about what counts as scarce gets encoded into the formula. The protocol provides default rarity values for canonical action classes per domain, deliberately conservative:

most common action classes sit in the 0.5 to 1.5 range, with values above 3.0 reserved for genuinely scarce contributions, such as a working novel governance protocol or a domain-foundational instrument calibration. DFAOs can adjust R values for local context within bounds set by their parent DFAO, and ultimately within bounds set by the ecosystem-level, planetary-scale defaults. A small-town DFAO might set a repair-shop-operation action class at a higher R than a megacity DFAO would, reflecting genuinely different local scarcity. The bounds prevent runaway R inflation: a child DFAO cannot set R equal to 10 for a routine action class, because that exceeds the parent's safety bound, and an attempted override outside the safety range triggers a parent-DFAO governance review rather than silently applying. Governance proposals to revise the rarity table are visible to all DFAO members, and the deliberation period and voting method are determined by the DFAO's own governance configuration (§14.2). Approved revisions apply to future loops only; existing loops continue under the prior R value, preserving DAG immutability (§9.8).

A.4 F Decay Curves in Detail Two supported frequency-of-decay curves have materially different practical behaviors, as introduced in §4.4. Log-tail decay, F(n) equals 1 divided by (1 plus the natural log of n plus 1), where n counts prior occurrences of the fingerprint, produces the following numeric behavior: F(0) equals 1.0 (first occurrence); F(1) is approximately 0.59; F(10) is approximately 0.30; F(100) is approximately 0.18; F(1000) is approximately 0.13. It has a long tail: F decreases but never reaches zero, so repeated contribution retains some marginal value indefinitely. Harmonic decay, F(n) equals 1 divided by (n plus 1), produces sharper decay: F(0) equals 1.0; F(1) equals 0.5; F(10) is approximately 0.09; F(100) is approximately 0.01; F(1000) is approximately 0.001. F approaches zero quickly under repetition. 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 calibration. The choice of curve is per-DFAO and per-claim-type, and the DFAO's rarity table records the recommended decay curve alongside each canonical action class.

A.5 The Settlement-Time Term in Practice, Worked Numerically The settlement-time term creates an incentive for fast, honest settlement. A claim that closes in one-tenth of its target window yields log(10), approximately 2.3, from the time term alone. A claim that closes exactly at its target window yields log(1), which equals zero, from the time term. The T_floor default of 0.01 caps the maximum value of the term at log(100), approximately 4.6, which prevents arbitrarily fast settlement from producing arbitrarily large XP.

The term's behavior at T_s equal to 1.0 is log(1) equals 0, meaning the formula multiplies by zero and the mint produces zero XP from this factor. This is intentional: a claim that takes its full target window, or longer, yields no time bonus, which is the correct behavior, since the formula's minimum non-zero XP value is then set entirely by the product of the other four terms. The term's behavior near T_floor is the cap itself: any T_s value below T_floor is clamped to T_floor before the log is computed, handling edge cases where a claim is reported as having settled in negligible time, which usually indicates either a measurement error or an attempted speed-farming attack (§4.8, §17.2).

Falsifier. If a production deployment shows a statistically significant cluster of claims settling at exactly T_floor with no plausible legitimate explanation, that is empirical evidence of an active speed-farming attempt against the clamp, and the per-domain λ_d and rate limits described in §16.3 must be re-tuned in response.

Appendix B. Worked Example: The Loop Lifecycle

B.1 Setup Consider a contributor who runs a community repair workshop and wants to mint XP for repairing a broken bicycle inside their MICRO-scale DFAO, a four-person workshop collective. The repair has thermodynamic content (the bicycle returns to a functional, lower-entropy operational state), informational content (the contributor documented the repair process for the workshop's knowledge base), and economic content (the repair extends the bicycle's useful life, reducing replacement demand).

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 vector E equal to (0.1 cognitive, 0.0 code, 0.0 social, 0.3 economic, 0.5 thermodynamic, 0.1 informational, 0.0 governance, 0.0 temporal), keyed to the canonical eight in the order used throughout this Codex; instrument, the workshop's repair instrument, a structured photographic and tested-functionality record, with the working state validated by a second workshop member; baseline, the bicycle's documented non-functional state at intake; falsification condition, if a 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 close. The DAG vertex for loop open is written and signed by the contributor's identity key.

B.3 Evidence and Validation SignalFlow routes the validation request. For a workshop-scale repair claim, routing produces a pool of two workshop validators. Validator 1, another workshop member, inspects the bicycle, tests it under the workshop's standard tests, confirms functionality, and records an accept verdict. Validator 2, a guest validator visiting from another DFAO, reviews the documentation, asks one clarifying question through the validation interface, then also records an accept verdict. The aggregation layer combines the two independent judgments via Bayesian updating (§7.6); the posterior mean clears the domain's closure threshold.

B.4 Closure The loop closes: quorum is met, no top-decile reject vote stands unrebutted, and both accepts agree on ΔS within tolerance. The formula computes as follows. R equals 1.2, per the workshop DFAO's rarity table for the action class of a simple, documented repair. F equals 0.95, since the falsification condition was explicit and well-instrumented. ΔS_b_e equals 0.7, normalized: the thermodynamic content is wellbounded, and the economic and informational content each add marginal ΔS. The workshop DFAO's weight vector is (1.0 cognitive, 1.0 code, 1.2 social, 1.0 economic, 1.5 thermodynamic, 1.1 informational,

1.0 governance, 1.0 temporal); computing w · E against the claim's domain vector above yields approximately 1.26. T_s equals 0.45, since the loop closed in roughly half its target window, giving log(1/0.45), approximately 0.80. XP equals 1.2 times 0.95 times 0.7 times 1.26 times 0.80, which is approximately 0.804. XP mints as provisional. The retroactive validation window opens.

B.5 Settlement After the retroactive window closes with no accepted challenge, the provisional mint of approximately 0.804 XP is confirmed and committed to the contributor's thresholds. This worked example demonstrates the full

lifecycle described abstractly in §9: open, evidence and validation, closure, provisional mint, and confirmation, with concrete numbers substituted at every step.

Appendix C. DAG Substrate Detail

C.1 Vertex and Edge Schema Every DAG vertex, per the reference implementation in packages/dag-substrate , carries a vertex identifier, a vertex type drawn from the protocol's event vocabulary (§9.7), a payload specific to that vertex type, a signature from the submitting actor's identity key, a set of parent-edge references establishing causal ordering, and a Lamport timestamp for causal sequencing independent of wall-clock skew across nodes. Edges are directed and acyclic by construction: a vertex can only reference parents that already exist in the graph, which is what makes the structure a DAG rather than a general graph.

C.2 Tip Selection in Detail New vertices attach to the graph by selecting parent tips through a weighted random walk, where the weight is the cumulative confirmation weight of the candidate tip, borrowed conceptually from the IOTA Tangle's approach (§11.4). A tip with more accumulated confirmation weight, meaning more subsequent vertices have built on top of it, is more likely to be selected as a parent for a new vertex, which is the mechanism by which the graph converges toward a single canonical history over time without a linear blockchain's total ordering.

C.3 Confirmation Propagation When a new vertex attaches to a tip, it increases that tip's confirmation weight, and this weight propagates backward along the causal chain to the tip's own parents. A vertex is considered confirmed once its accumulated confirmation weight exceeds a domain-specific or protocol-wide threshold. This is the mechanism underlying the loop lifecycle's transition from provisional to confirmed (§9.6): the mint event's confirmation weight must clear the threshold, in combination with the retroactive-validation window closing without an accepted challenge, before the mint is treated as final.

C.4 PostgreSQL Backing The reference implementation's packages/dag-substrate service persists vertices to a PostgreSQLbacked store, accessed through the shared connection-pooling utilities in @extropy/contracts . This is a reference-implementation choice, not a protocol requirement (§15.9): an alternative implementation could back the same logical DAG structure with any storage engine, provided it preserves append-only semantics and exposes the mint-event log per the Protocol Contract in this Codex §2.3.

C.5 Cross-Check Against the Live Repository This appendix's description is cross-checked directly against packages/dag-substrate/src/index.ts , which documents itself, in its own header comment, as "the foundational permissionless ledger of the Extropy Engine, analogous to the IOTA Tangle," with core properties listed as causal ordering via Lamport timestamps, tip selection via a confirmation-weighted random walk, automatic vertex creation from system events, confirmation-weight propagation up the causal chain, and permissionless vertex submission by any service. Every claim in this appendix traces to that source file's own stated design.

Appendix D. Identity Flows

D.1 New User Onboarding, Step by Step A new user's onboarding proceeds as follows. The user visits an application, HomeFlow or another vertical, and initiates sign-in. The application redirects to an OAuth provider; the user authenticates, and the application receives an OAuth token, transitioning the onboarding state from pending to verified. The application then prompts the user to complete KYC on-device, offering a choice among an ID scan, a biometric bind, or a trusted-issuer handoff; the KYC payload is processed entirely locally, producing a verified-identity attestation that never leaves the device, and the onboarding state transitions to attested. The application generates a key pair locally; the public key becomes the user's DID, and the private key is stored in the device's secure enclave or platform-equivalent secure storage; the DID is registered with the network, and the onboarding state transitions to issued. Finally, the application creates the user's initial PSLL, generates a genesis entry hash-chained to itself, and

anchors the PSLL's first Merkle root to the DAG, completing onboarding. The entire flow is designed to take a few minutes for an experienced user, and the state machine preserves progress if the flow is interrupted and resumed.

D.2 Selective Disclosure When a participant presents themselves to a validator neighborhood for a specific claim, disclosure is selective by construction. A participant can prove statements such as: they are a verified member of a specific DFAO, without revealing other DFAO memberships; they hold a specific CAT level in a specific domain, without revealing other CAT levels; they have validated a threshold number of loops in a given domain, without revealing which specific loops; or their XP standing in a given DFAO exceeds a required threshold, without revealing the exact value. The proofs are produced using the BBS+ scheme over the participant's credential bundle (§12.3); the verifier sees only the proven statement and that the proof verifies, which is sufficient for the protocol's routing or admission decision, and nothing more.

D.3 Nullifier Use When a participant takes a one-time action, a governance vote, a single claim per loop, or a reveal-consent action, the protocol requires a nullifier, deterministically derived from the participant's locally held secret and a context tag specific to that one-time-action context. The same participant computing the nullifier for the same context always produces the same value, which prevents double-action; different participants produce different values, which prevents impersonation; and different contexts produce uncorrelated nullifiers, which prevents cross-context tracking. If a nullifier has already been seen in a given context, the action is rejected; otherwise it is accepted and the nullifier is recorded for that context going forward.

D.4 Key Rotation If a participant's private key is compromised, or if the participant wants to rotate keys for routine hygiene, the protocol supports a documented rotation flow: the participant generates a new key pair locally; creates a rotation event signed by both the old and new keys; anchors that rotation event to the DAG; and from that point forward, all PSLL entries and DAG submissions use the new key, while the old DID is marked as rotated rather than deleted, preserving the historical record. The participant's accumulated token balances transfer to the new DID through the rotation event's dual-signature chain. The rotation is fully observable: anyone querying the participant's DID history can see the rotation event and verify the dual signature, and the rotation does not break the causal history of the participant's prior contributions; it only changes which key signs new contributions going forward.

Falsifier. If any production identity flow allows onboarding to complete with raw KYC material, rather than only the resulting local attestation, transmitted off-device, Digital Autarky is violated at that flow and the flow must be corrected per the falsifier already stated in §9.10.

Appendix E. DFAO Governance Worked Example

E.1 The Proposal A MESO-scale DFAO, a roughly forty-person community focused on neighborhood-scale entropy reduction, decides its current validation-closure confidence threshold is too low for thermododynamic-domain claims, which it considers high-stakes. A member opens a governance proposal to raise the closure threshold for thermodynamic-domain claims within this DFAO, citing rationale: three thermodynamic claims in the past quarter were initially confirmed and subsequently falsified during the retroactive window, suggesting the current threshold is admitting low-confidence consensus. The proposal specifies an effective date fourteen days out and a voting method of conviction voting with a seven-day half-life.

E.2 Deliberation For fourteen days, DFAO members discuss the proposal through channels outside the protocol itself, with the discussion record feeding back into the proposal's documented rationale. Several members raise a reasonable objection: the three falsified claims may share a specific instrument flaw rather than reflecting genuine threshold inadequacy. The proposal's sponsor responds with further analysis showing the three falsified claims used three different instruments, which is evidence favoring a threshold-level rather than an instrument-level explanation.

E.3 Voting and Execution Members cast conviction votes, each consisting of a position, a stake of the DFAO's governance-weight token, and the timestamp the position was taken; conviction weight grows the longer a position is maintained. After fourteen days, the aggregate conviction weight in favor clears the proposal's pass threshold, and the proposal passes. On the effective date, the DFAO's closure threshold for thermodynamic-domain claims updates, recorded as a governance-parameter-update event causally linked to the original proposal event. Loops opened before the effective date continue under the prior threshold; loops opened on or after the effective date use the new one, preserving the forward-only application rule from §14.5.

E.4 Subsequent Observations and What the Example Shows Over the following quarter, the DFAO's thermodynamic claims show a materially lower false-positive rate, while claim throughput drops slightly, since some previously borderline claims now fall short of the higher threshold. The trade-off proves acceptable to the membership, and no reversion proposal is opened. The example illustrates several structural properties: governance here is parameter-based, not procedural, meaning specific numbers adjust while the underlying protocol logic does not change; proposals are visible and discussable; conviction voting rewards patient positions over impulsive shifts; updates apply forward-only, preserving DAG immutability; and observed outcomes feed back into future governance decisions, closing the loop described abstractly in §14.7's Complexity Thermostat framing.

Appendix F. Convergence Vertex Worked Example

F.1 The Setup Three contributors collaborate on a substantial entropy reduction: designing, building, and commissioning a community-scale composting system for their neighborhood, a project taking four months and measurably reducing the neighborhood's waste flow.

F.2 The Convergence Claim At project completion, the three contributors jointly open a single loop as a convergence vertex, with three incoming person-DAG edges rather than one. The claim states the system was designed and commissioned, reduced neighborhood waste flow by a documented percentage against baseline, and has been in continuous operation with a maintenance plan documented. The domain vector spans primarily thermodynamic, with significant informational, economic, social, and governance content. The instrument combines community-validated waste-flow measurement with third-party audit inspection. The falsification condition is explicit: if the system fails to maintain at least half its claimed waste-flow reduction over the following ninety days, the claim is falsified. Convergence-split rules are documented at open time: in this example, the three contributors agree to a 35, 35, and 30 percent split reflecting differential effort contribution.

F.3 Validation and Settlement Validation routing identifies a validator pool spanning the relevant domains, incorporating the third-party auditor's measurement directly into the validation record. Consensus clears the closure threshold, and the loop transitions through closure. Suppose the resulting joint XP computes to 12.5. The convergence split

then distributes 4.375 XP each to the first two contributors and 3.75 XP to the third, with each contributor's individual balance increasing by their documented share, and the split itself recorded permanently in the provisional-mint vertex. The retroactive validation window opens, and after it closes with no falsifying evidence, the provisional mint confirms for all three contributors.

F.4 The Convergence Vertex as Shared History The convergence vertex becomes part of each contributor's person-DAG (§11.7). Any future query examining any of the three contributors' individual histories surfaces this shared convergence, and reputation accrual flows to all three contributors in the domains the claim touched. If the system fails well beyond the retroactive window, governance retains the ability to review whether a corrective vertex against the convergence is warranted, with the DAG preserving the full lineage of any such correction.

F.5 Why Convergence Matters Multi-party collaboration is the dominant mode of real-world entropy reduction, and the convergence vertex is the protocol's first-class representation of that fact: the protocol does not force collaborators to

submit artificially separated individual claims or to split credit outside the ledger's own record. Convergence-split rules are themselves governance-tunable, with DFAOs able to establish defaults for symmetric equal splits, contribution-weighted splits for asymmetric collaborations, or fully custom DFAO-defined split functions for complex cases.

Appendix G. HomeFlow: The Family Pilot

G.1 What HomeFlow Is HomeFlow is the application-layer vertical that brings the Extropy Engine to household scale, implemented as @extropy/homeflow on port 4015 with a dedicated frontend. It is the protocol's first deployment vertical and its most mature application example. A household enrolls as a MICRO-scale DFAO, typically two to seven members; each member onboards through the identity flow in Appendix D and creates a character sheet; the household configures its own governance, typically simple-majority voting for routine decisions and conviction voting for substantial changes; and the household sets its own weight vector w over the domains, commonly weighting social and thermodynamic contribution heavily given the household context.

G.2 What HomeFlow Tracks The household tracks contribution across routine categories: chores, mostly thermodynamic and social; repairs, thermodynamic and economic; childcare and care work, social; cooking and food planning, thermodynamic, informational through recipe documentation, and economic through food budgeting; shopping-list and pantry management, informational and economic; and family meetings and household governance, governance and social. Each contribution opens as a loop in the household DFAO, with the household's own membership serving as its validator pool: one member opens the loop, another validates, often informally, in conversation, but the validation is still recorded on the protocol as a formal event.

G.3 The XP Flow, Concretely A child takes out the trash. The child opens a loop: claim, "took out trash," domain primarily thermodynamic with temporal-adjacent framing, baseline, the trash bag was full, falsification condition, the trash was not actually removed within thirty minutes. A parent confirms the claim; XP mints; the child's character sheet updates. Over time, the child accumulates XP, and the household has configured an EP conversion (§4.10, §13.5) that converts XP into small treats or screen time, giving the child a tangible, immediate reward for habitual contribution. The household's collective XP also feeds into household-level governance decisions, informing allocations such as who chooses a weekend activity or who gets a new device first.

G.4 Why HomeFlow Is the Pilot HomeFlow is the pilot vertical because the household is small enough that the protocol's full complexity does not overwhelm participants; the validators are mutually trusted family members, which simplifies the validator-collusion threat model relative to a large anonymous network; the contribution categories are routine and well understood, making the rarity table simple to calibrate; the merchant-facing side is intrahousehold, with parents effectively acting as merchants for children or vice versa; and the protocol's failure modes, a stale rarity value, a confused decay curve, a missing closure threshold, are observable but not catastrophic at this scale. HomeFlow is the protocol's empirical testbed, and lessons from it inform protocol-level revisions.

G.5 What HomeFlow Has Shown Running on consumer hardware with file-backed default storage, HomeFlow has demonstrated that the onboarding flow works for non-technical users who have no interest in cryptography, that the charactersheet interface is intuitive with XP accrual visible and motivating, that the validation flow can be entirely informal in a high-trust DFAO while still producing a valid protocol-level record, that the XP-to-EP conversion works for intra-household exchange of treats and privileges, and that household-level governance can adapt over time as members renegotiate weight vectors as priorities shift. The pilot has also surfaced real limits, stated honestly here rather than smoothed over: the protocol's terminology can be opaque to household members, and benefits from translation into household-native language, saying "did the parent agree this happened" rather than "validation confidence," for instance; the DAG's permanence creates friction for genuinely trivial loops, since even a two-minute chore creates a permanent DAG record; and the standard retroactive-burn window is too long for household-scale claims, which is why

HomeFlow's own deployment uses a shortened window rather than the ecosystem default (see Appendix L for the deployment-level detail). These lessons feed back directly into the protocol's governance defaults and into interface design decisions.

Falsifier. If the HomeFlow pilot fails to expand to additional households without protocol-level failure, that is direct evidence against the adoption-criteria claims in §19.10 and against this appendix's characterization of HomeFlow's maturity.

Appendix H. The God Paper: A Philosophical Companion

H.1 What This Appendix Is and Is Not This appendix summarizes a philosophical extension paper, informally called the God paper, that the project has published alongside its technical materials. It is a philosophical companion, not load-bearing for the protocol's operational specification. A reader focused purely on the technical specification can skip this appendix entirely without losing anything needed to understand or implement the protocol.

H.2 The Reframe The paper proposes reframing a classical theological question from ontology, does a supreme entity exist, to function, does reality contain measurable, recursive, feedback-driven entropy reduction. Under this reframe, divinity is treated not as a being but as a functional name for a pattern: the emergence of coherent order through feedback across domains. The reframe does not claim traditional religious experience is invalid; it proposes that traditional religious language may have been pointing imprecisely at real coherence-generating functions that lacked formal measurement until now.

H.3 The Functional Definition The paper's central definition states that this functional concept names the emergent process by which feedback-driven systems reduce entropy across domains. The definition is explicitly functional, not ontological: it does not specify a supernatural being, only a pattern that, when present in a system, manifests as the conversion of disorder into coherent structure through feedback.

H.4 The Falsification Conditions The paper states what would falsify its own central claim, in the same falsifiability discipline this Codex applies throughout. First, if the alleged reduction is not measurable, meaning no instrument, proxy, audit trail, or feedback structure can detect it, the claim remains poetic rather than operational. Second, if the reduction is local but parasitic, for example an organization reducing its own internal disorder by externalizing thermodynamic or social disorder elsewhere, that is not the functional pattern the paper describes; it is externalization dressed as improvement. Third, if the apparent order is produced by suppression, hidden data, or complexity pushed downstream rather than genuinely resolved, that is compression fraud, not coherence. Fourth, if the measurement system itself drifts once people gain status or reward from appearing to reduce entropy, the same Goodhart dynamic described in §6 applies to this functional concept as much as to any other metric. These conditions are testable, and the paper's own claim is explicitly allowed to lose, which is the identical posture the Extropy Engine adopts at the protocol level throughout this Codex.

H.5 Theological Translations, Offered Without Insistence The paper offers operational translations of classical attributes without claiming to prove them literally: omniscience as the ideal of maximal signal integration, the conversion of dispersed information into coherent, actionable understanding; omnipresence as the domain-general applicability of entropy dynamics across every context; omnipotence as the limit case of maximal transformation achieved with minimal waste; and goodness as the tendency of actions to reduce disorder in ways that survive recursive feedback rather than merely appearing to in the short term. Adjacent concepts translate similarly: prayer as a request for feedback, orientation, and correction; worship as alignment with coherence-generating patterns; a classical notion of transgression as disorder injection without compensating repair; and a

classical notion of redemption as sustained participation in feedback loops that transform destructive drift into durable order. These translations do not force themselves onto any religious tradition; they offer an optional operational bridge

for readers who find it useful.

H.6 What the Paper Does Not Claim The paper is explicit about its own limits: it does not prove that a personal deity exists, does not prove that consciousness is fundamental to reality, does not prove that the universe has intentions, and does not adjudicate the validity of any specific tradition's religious experience in either direction. It offers a testable reconstruction, useful for readers who want to engage religious language operationally, and remains silent on the metaphysical questions traditional theology engages directly.

H.7 The Terminology Note An earlier draft of the God paper glossed the symbol F as naming a quantity it called feedback closure strength. This is a genuine and separate quantity in that paper's own context, but it is not what F means in the canonical XP formula, which is falsifiability per the convention this Codex adopts throughout (§4.7, §4.4). A reader moving between the God paper and this Codex should hold the two usages separately: F in this Codex's formula is falsifiability, and whatever the God paper calls feedback closure strength corresponds most closely to the validation-confidence concept discussed in §7.6, not to the formula's F term.

H.8 The Connection to the Protocol The paper's own stated connection to the Extropy Engine is explicit and modest: the engine is described as a prototype architecture for measuring contribution through validated entropy reduction, whose purpose is not to detect a supernatural object but to detect the functional pattern the paper names, measurable improvement produced through feedback. This connection is a philosophical extension, not a foundational claim the protocol's operational correctness depends on. A reader focused on the protocol can engage with the God paper or skip it entirely; a reader focused on the philosophical question can read the God paper without committing to the protocol's specific engineering choices.

H.9 Why This Appendix Exists This appendix exists for three reasons: the God paper is part of the project's published material, and this Codex aims to consolidate the project's material honestly rather than pretend the philosophical extension does not exist; some readers will encounter the God paper independently of this Codex, and the terminology-note correction in H.7 is easier to act on with this appendix's context; and the epistemic posture this Codex insists on throughout, falsifiability, validated measurement, and the explicit willingness to be wrong, extends consistently from the technical specification through to this philosophical extension, and this appendix makes that extension visible rather than leaving it implicit.

Appendix I. Sandbox Posture and Falsification Conditions

I.1 Restating the Sandbox Posture Section 19.8 stated the sandbox posture in the main body; this appendix consolidates it alongside the falsification conditions for readers who want both together in one place. The current codebase and any live deployment of it are a sandbox implementation: a live engineering testbed used to exercise the architecture's assumptions, expose failure modes, and close remaining gaps, not a hardened, productionready deployment (this Codex §2, docs/VPS_NODE.md). This is stated as fact, not apology.

I.2 The Complete Falsification Conditions, Consolidated Repeated here for a reader who wants a single consolidated list, cross-referenced to where each is derived in the main body. The whole-project falsifiers F1 through F4, from §6.2: non-comparability across domains, runaway cross-domain arbitrage, validator disagreement dominating signal, and settlement-floor arbitrage surviving the v3.1.3 clamp. The reputation-separation falsifier from §4.2 and §10.10: any shipped path letting reputation raise a new-mint multiplier. The emergent-validation falsifier from §10.10: any shipped protocol behavior admitting a persistent, unaccountable validator class. The Non-Extraction falsifier from §13.4: any shipped feature passing review while failing T1, T2, or T3. The anticoncentration falsifier from §14.9: any DFAO producing permanent, ungovernable power concentration. The cartel-analysis falsifier from §17 (Concise numbering, restated in this Codex's §16 through §17 material): adversarial simulation failing to converge to majority-honest settlement at cartel size 10 or above. And the project-level falsifiers from §19.9, restated: failure of the reputation

invariant in deployed practice, uniform Goodhart-vulnerability across all eight domain instruments, failure of cross-domain normalization to produce stable calibrations, sustained validator collusion below the predicted detection threshold, or metric-farming proving cheaper than genuine contribution at a rate the protocol's defenses did not anticipate.

I.3 What Success Would Look Like Restated from the project's own closing reflection: project-level success is a sustained deployment 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 §13.5 and §14; a small number of MESO and MACRO DFAOs adopt the protocol for their internal coordination; the peer-review and academic community engages with the formal foundations in §7 and Appendix J and produces additional theorems, falsifications, or refinements; the reference implementation matures toward the roadmap targets in §19.6 and §19.7; and the architectural invariants in §20.9 are maintained under real adversarial pressure, not merely in specification. None of this is guaranteed. The protocol is allowed to lose.

Falsifier. This appendix is itself a falsifiable artifact: if a future revision of this Codex removes or weakens any falsification condition listed here without a corresponding public correction explaining why, that removal is itself evidence of exactly the drift this Codex exists to prevent.

Appendix J. Randall's Feedback V4: Formal Theorems in Full

J.1 Scope of This Appendix Section 7 summarized Randall's Feedback V4's axioms and six formal contributions at the level needed to understand how the Extropy Engine's governance layer instantiates them. This appendix expands each result to the level of its formal preconditions and proof strategy, for a reader with the relevant mathematical background who wants to evaluate the claims directly rather than take the summary on faith.

J.2 Theorem V4-1: RF Governance Convergence, Expanded The full statement: agents satisfying particular partition conditions, in the sense developed in the activeinference literature on Markov blankets and generative models, converge asymptotically to a governance attractor through Hamiltonian matching under the Free Energy Principle, meaning each agent's internal generative model converges to a shared governance Hamiltonian H_G. The proof strategy draws on Bayesian-mechanics treatments of self-organizing systems and on categorical-compositionality results establishing that local free-energy minimization at the agent level composes coherently into a system-level attractor under the partition conditions. The consequence for the Extropy Engine: individual participants running their own personal AI, each locally minimizing their own prediction error against the shared protocol's observable state, are expected to converge toward consistent expectations about governance state over time, without any participant needing to model every other participant explicitly.

J.3 Theorem V4-2: Collective Goal-Update Admissibility, Expanded The full statement: necessary and sufficient conditions on the goal-update operator L are established for the collective system of agents to remain within a Banach fixed-point contraction on the product space of individually feasible reward sets. The proof strategy shows that if each agent's goal-update operator is Lipschitz continuous with a constant bounded away from 1 (Axiom 2, §7.2), and if the feasible reward sets satisfy a pairwise overlap condition, then the joint update operator across all agents is itself a contraction on the product space, guaranteeing convergence to a unique joint fixed point rather than divergence or oscillation. The consequence: governance can update its own parameters over time, at the DFAO level, without the update process itself becoming chaotic or diverging, provided the rate-limiting on parameter updates (§14.5) keeps individual DFAOs' update operators within the required Lipschitz bound.

J.4 Theorem V4-3: Dynamic Goodhart Resistance, Expanded The full statement: the discrepancy process between the contribution metric and the evolving governance goal remains light-tailed, in the formal sense of not developing heavy-tailed excursions, under bounded goal-update rates and Lipschitz metric continuity. The proof strategy applies

tail-distribution characterizations of Goodhart dynamics developed for general adaptive-metric systems, showing that when the metric's sensitivity to underlying state changes is bounded (Lipschitz continuity) and the rate at which the governance goal itself can shift is bounded (Axiom 2), the metric-goal discrepancy cannot accumulate the heavy tail that characterizes catastrophic Goodhart capture. The consequence for the Extropy Engine, stated already in §6.7 and §7.5: keeping the value formula insulated from the reputation representation is exactly the structural condition the theorem requires, since reputation is precisely the channel through which an unbounded discrepancy could otherwise accumulate.

J.5 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, within which non-trivial unhackable metric-goal pairs exist, in the sense established by the game-theoretic literature on reward hacking, which shows that for the full unconstrained stochastic policy class, the only unhackable metric is a trivial constant metric. The proof strategy shows that RF's admissibility conditions, a Lipschitz contraction bound, a rate bound, and a feasible-reward-set constraint, jointly restrict the policy class enough that the general impossibility result no longer applies, and non-trivial unhackable pairs become constructible within the restricted class. The consequence for the Extropy Engine: the protocol's restrictions on contribution strategies, the loop lifecycle's fixed state machine, the falsifiability requirement, and the validator-routing structure, are not merely procedural conveniences; they are the operational form of the admissibility conditions that make non-trivial unhackability possible at all. Without these restrictions, the value formula would be hackable in the full-policy-class sense the underlying impossibility result describes.

J.6 Theorem V4-5: Convergent Validation Without Ground Truth, Expanded The full statement: Byzantine-robust peer prediction, extended from pairs of validators to n-tuples with a bounded collusion-tolerance parameter, achieves convergent validation in the presence of linearly many colluding validators, without requiring a trusted oracle to supply ground truth. The proof strategy extends peer-prediction mechanisms that provide truth-telling incentives even when no validator has direct access to ground truth, generalizing from pairwise comparison to n-validator aggregation and bounding the fraction of colluding validators the mechanism can tolerate while still converging to the honest consensus. The consequence for the Extropy Engine: the validation model in §10 does not require a trusted oracle. Validators, meaning contributors performing validation tasks, reach consensus on claim validity through Bayesian aggregation of independent judgments (§7.6), and the collusion tolerance is bounded but nontrivial: validator pools of size 10 or more, with detection probabilities above roughly 0.3, are safe against rational collusion strategies, which is the direct theoretical grounding for the cartel-threshold analysis worked through empirically in §16.1 (Concise-edition numbering; restated as the cartel analysis referenced from §17.2 of this Codex).

J.7 Conjecture V4-6: Trilemma Resolution, Status The full conjecture: RF achieves approximate generalizability, approximate trustlessness, and epsilonbounded Sybil resistance simultaneously, a resolution of a trilemma that the broader reputation-and-trustsystems literature treats as generally unresolvable in the strict sense. Status: this is stated as a conjecture in RF V4, not a theorem, and the formal proof remains open. This is the strongest claim in the framework and is therefore the one most likely to require revision under sustained adversarial pressure. The consequence for the Extropy Engine: the protocol claims to achieve generalizable, trustless, and Sybilresistant operation in its design intent, but the formal joint proof connecting all three properties simultaneously is not yet complete. The empirical evidence from the reference implementation, the HomeFlow pilot (Appendix G), the integration test suite (§15.11), and the cartel-threshold analysis (§16.1), supports the conjecture directionally, but empirical support is not a substitute for the missing proof. Per docs/REJECTED_FRAMINGS.md R3, this conjecture's retirement path, should it fail, is either a full proof or an explicit restatement as an open conjecture with the current partial results stated as lemmas; that restatement work is not yet complete and is tracked as ongoing.

J.8 The Two-Phase Bootstrap, Expanded Phase 1, contribution-only: all agents have equal standing, and contributions are validated purely on content quality without reputation weighting. Phase 1's duration is set by a sample-complexity bound requiring a substantial number of observations, formally of order H-cubed times the state-space size times the

action-space size, divided by an accuracy parameter squared, before the first goal update can be triggered with statistical confidence. Phase 2, reputation-weighted: after sufficient contribution data accumulates, the system transitions to reputation-weighted validation, triggered when the estimated feasible-reward-set intersection has sufficiently small diameter, indicating the governance attractor is well characterized. In the Extropy Engine, this is the formal justification for why a newly formed DFAO cannot launch with reputation weighting active from day one: all members begin with equal standing in validation routing (Phase 1), and governance can vote to transition to Phase 2, reputation-weighted routing, only once the DFAO has accumulated enough contribution data to satisfy the sample-complexity bound.

J.9 The Non-Degeneracy Condition Carried forward from an earlier RF version, the non-degeneracy condition ensures governance does not collapse to a trivial state. Formally, the system is non-degenerate if the governance attractor is not a singleton, meaning multiple goals remain consistent with observed behavior; the admissible policy set contains at least two distinct policies, meaning agents retain meaningful choice; and the contribution metric discriminates between at least two contribution types, meaning the metric is actually informative rather than constant. Under these conditions, the two-phase bootstrap produces a non-trivial reputation distribution, not every agent ending up with identical reputation, and a non-trivial governance attractor, not every agent converging on an identical goal. In the Extropy Engine, the eight canonical domains, the multi-factor XP formula, and the diverse contribution categories collectively ensure the discriminative requirement holds in practice.

Falsifier. If deployed governance data shows the discrepancy process from Theorem V4-3 developing a heavy tail despite the stated preconditions (bounded update rates, Lipschitz continuity) holding in the deployment, the theorem itself, not merely its application, requires revision, and this appendix must be corrected accordingly.

Appendix K. Meaning Drift and Goodhart Math, Extended

K.1 The Differential Equation, Full Derivation The central equation, restated from §6.2: the rate of change of representational fidelity F with respect to time equals negative k times S(t) times F(t), plus r times C(t) times (1 minus F(t)). Both terms carry intuitive interpretations worth restating in full. The decay term, negative k times S(t) times F(t), states that the rate of fidelity loss is proportional to the fidelity currently remaining, since a label cannot lose fidelity it does not already have; proportional to the current social or incentive load, since more incentive weight produces faster gaming; and proportional to a domain-specific susceptibility coefficient k, since some domains resist gaming structurally better than others. The recovery term, r times C(t) times (1 minus F(t)), states that the rate of fidelity recovery is proportional to the gap remaining from perfect fidelity, since a

system can only recover what it has not already lost entirely; proportional to the strength of corrective feedback C(t) currently available; and proportional to a restoration rate r characterizing how efficiently available corrective feedback actually repairs fidelity. At equilibrium, where the rate of change is zero, solving for F yields an equilibrium fidelity determined by the ratio of corrective force to gaming force: F equals r times C divided by (r times C plus k times S). When corrective feedback C is large relative to social load S, F approaches 1, high fidelity. When social load S dominates corrective feedback C, F approaches 0, full semantic capture.

K.2 The Three-Domain k-Parameter Taxonomy, Extended The physical domain, k approximately 0.02: reality provides fast, unambiguous corrective feedback. A bridge rated for a specific load that actually fails below that rating collapses visibly and immediately; atmospheric measurements, fluid-dynamics measurements, and structural-load tests all operate in this fastfeedback regime, where gaming a physical measurement is quickly and visibly contradicted by physical reality itself. The institutional domain, k approximately 0.15: feedback is mediated by bureaucracies, peer review, and regulatory process, operating on timescales of months to years rather than immediately. University rankings, journal impact factors, credit ratings, and drug-trial endpoints all live in this regime, where corrective feedback exists and is real but is slow enough that substantial gaming can accumulate before correction catches up. The social and moral domain, k approximately 0.45: corrective feedback is weak or entirely absent. Status claims, identity claims, moral labels, and

content-authenticity claims live here, where C(t) approaches zero and the decay term dominates the dynamics almost entirely. The k values themselves are not arbitrary; they are estimated from documented historical Goodhart episodes, examining how quickly a given label decoupled from its referent, what corrective feedback was actually available, and what the resulting recovery profile looked like.

K.3 The Four Phases, Extended with Examples Phase 1, descriptive: low S(t), high F. The label functions descriptively, and people use it to track the underlying condition it names. Examples include “house” as a descriptive housing category or “kilometer” as a unit of measure, where no meaningful incentive weight has yet attached to the label itself. Phase 2, standardization: rising S(t) as institutions invest in verification; F may temporarily improve as measurement becomes formalized. Examples include a credential becoming standardized through a recognized accrediting institution, or a metric becoming mandatory in government reporting, where the formalization process itself briefly tightens the label’s correspondence to its referent. Phase 3, capture: S(t) saturates, gaming proliferates, F decays as the actors with the most to gain learn to optimize the label rather than the underlying condition. Examples include standardized-test preparation industries optimizing test scores directly rather than the underlying learning the test was designed to measure, or search-engine-optimization practices targeting ranking signals rather than the content quality those signals were meant to proxy. Phase 4, collapse or correction: either semantic emptying, where the label no longer tracks anything meaningful, or an external shock to C(t) triggers partial restoration, such as a regulator updating an instrument or a credible peer-review process correcting a field’s practices. The four-phase trajectory is not

deterministic; some labels stabilize in Phase 2 indefinitely when the underlying domain supports continuous corrective feedback, such as atmospheric measurement standards, while most labels in the high-k social and moral domain eventually pass through all four phases on a multi-decade timescale.

K.4 The Bidirectional Extension The distinctive contribution of this model, relative to simpler one-directional Goodhart accounts, is that fidelity can recover, not merely decay. Recovery requires structurally available corrective feedback, a high C(t), and a non-trivial restoration rate r; the restoration rate is a property of the system’s own audit infrastructure and correction pathways, not merely of the domain’s inherent k value. A system with strong audit infrastructure, clear falsifiability conditions, and a genuine path from observation to correction has a high r regardless of the underlying domain’s k value; a system lacking these has r near zero regardless of domain. This is the operational insight the Extropy Engine builds directly on: the protocol cannot change a domain’s inherent k value, but it can engineer C(t) and r to be substantial. The retroactive validation window (§9.6) is a structural increase in C(t). The reputation penalty for validators whose consensus is later contradicted (§16.1, restated from §17.2 of this Codex) is an increase in r. The DAG’s transparency (§11) enables both.

K.5 Self-Application, Restated The underlying Signal paper applies its own model to itself, estimating its own k at approximately 0.15, the institutional regime, with corrective feedback supplied by peer review, falsification attempts, and empirical challenge from the broader research community. This self-application is a substantive commitment, not a rhetorical flourish: if the paper’s own framework drifts into unfalsifiable dogma, the kparameter taxonomy hardening into fixed categories immune to revision, or the four-phase model becoming a procrustean bed forced onto every case regardless of fit, the framework predicts its own eventual capture by the same dynamic it describes. The Extropy Engine adopts the identical posture at the protocol level, as stated already in §6.6: this Codex’s own falsification conditions, correction ledger, and honesty clause are the protocol’s explicit submission to the dynamic this appendix describes mathematically.

Appendix L. HomeFlow Deployment Detail

L.1 Scope Relative to Appendix G Appendix G described what HomeFlow is, what it tracks, and what it has shown as a pilot. This appendix summarizes deployment-level detail specific to running a HomeFlow instance, drawn from the repository’s own FAMILY_PILOT.md deployment guide, without duplicating Appendix G’s pilot-level findings.

L.2 What a HomeFlow Deployment Runs A HomeFlow deployment runs the core Extropy Engine services needed to support a MICRO-scale DFAO: the epistemology engine, SignalFlow, the loop ledger, reputation, and the DAG substrate, alongside the HomeFlow application service itself on port 4015 and its dedicated frontend. Setup proceeds through a setup wizard, either in the HomeFlow frontend or via direct API calls, that walks a household through DFAO creation, member onboarding via the identity flow (Appendix D), and initial weight-vector configuration.

L.3 Household-Scale Parameter Adjustments HomeFlow's own deployment configuration diverges from ecosystem defaults in at least one documented, deliberate way: the standard thirty-day retroactive-burn window (§9.6, §14.4) is judged too long for routine household claims, since a household does not typically want a two-minute chore's provisional status hanging open for a month, and HomeFlow's deployment uses a substantially shortened window, provisionally seven days, more appropriate to household-scale trust and stakes. This divergence is exactly the kind of per-DFAO parameter override the governance architecture in §14 is designed to support, and it is documented here explicitly as a worked instance of that flexibility rather than treated as an inconsistency.

L.4 What HomeFlow's Deployment Has Informed Beyond the pilot-level findings in Appendix G, the deployment experience specifically has informed several Codex-level decisions: the case for household-appropriate retroactive-window defaults, discussed above; the case for translating protocol terminology into household-native language at the interface level, without changing the underlying protocol semantics; and early evidence, still informal, that file-backed default storage is adequate for MICRO-scale deployments while clearly inadequate for anything approaching MESO scale or above, which is a data point feeding into the broader performance and scalability gap catalog in §19.4.

L.5 What HomeFlow Has Not Yet Tested Stated plainly, as this Codex's honesty discipline requires throughout: the HomeFlow pilot has not yet exercised adversarial validator behavior of the kind analyzed theoretically in §16.1 and §17, since family members are not typically adversarial toward one another in ways a stranger-scale network must anticipate; cross-DFAO interaction, since HomeFlow deployments to date operate as isolated MICRO DFAOs rather than nested within a larger MESO or MACRO structure; production-scale identity threat models, since household members' KYC stakes are lower than a stranger-network's; or the merchantfacing layer's two-sided market dynamics at any scale beyond intra-household exchange. These limitations are explicit rather than implied: HomeFlow is the protocol's first deployment, not its comprehensive one, and later appendices and future Codex revisions will need a second pilot, at MESO scale or with genuine merchant participation, to exercise the parts of the protocol HomeFlow structurally cannot.

Appendix M. Three-Layer Separation, Extended

M.1 Relationship to Section 2 and Section 14 Section 2.4 introduced the three-layer separation as part of the problem statement, and it recurs as an architectural constraint throughout §10 and §14. This appendix consolidates the full architectural reasoning in one place, cross-checked against §1.4 and Appendix M of this Codex , for a reader who wants the complete treatment without hunting across sections.

M.2 Why the Separation Exists, Restated in Full Every gamified scoring system in history has eventually been gamed: credit scores became targets people optimize for instead of genuine creditworthiness; engagement metrics on social platforms became parodies of the cultural relevance they were meant to measure; compliance metrics drift from the safety conditions they were built to track. This is Goodhart's Law in its most familiar form: once a measure becomes a target, it stops functioning as a good measure. The standard response is to hide the metric, and the Extropy Engine does hide it, but with a specific twist worth restating precisely: the hiding is not secrecy for its own sake. The user-facing gamification is deliberately a different metric from the systemlevel scoring function. A participant can farm the visible metric all day, and it pays out in engagement, streaks, and badges, but it does not pay out in XP. Actual XP comes from the underlying entropy-reduction signature computed at the engine layer from validator-witnessed data, which the participant cannot push directly because they do not know which lever it is, and even if they did, the validation model in

§10 would catch the manipulation. This is deliberate metric divergence, not mere obscurity: the two metrics are correlated by design, but they are not the same metric, and that non-identity is the entire point.

M.3 What Each Layer Sees, in Full Detail The user-facing layer shows discounts at participating merchants, money saved over time, gamified feedback such as streaks and completed-loop counts, a self-curated character sheet the participant controls the presentation of, and narrative achievement hooks. It deliberately does not show raw XP numbers, the participant's own entropy-reduction coefficient, per-domain rarity multipliers, or any lever that maps directly and legibly onto XP minting. The reasoning is direct: if a participant could see they earned a specific XP amount for a specific trivial action, they would rationally maximize that specific action rather than the underlying condition the system is trying to encourage. The character-sheet metaphor captures the intended relationship precisely: the participant holds the pen and the eraser over their own displayed record, choosing what to show and to whom, but validators and sensors prevent fabrication of what is actually there. Reality holds the dice; the participant does not get to write fiction into their own sheet. The merchant-facing layer shows a free or low-cost point-of-sale system, a customer pipeline of participants who preferentially patronize network-affiliated merchants, better operational signal than conventional key performance indicators since entropy-reduction patterns reveal what is genuinely working in the business, and standard merchant-services infrastructure at competitive rates. It does not show individual participant XP balances, individual character sheets unless a participant explicitly opts to share theirs, the full engine-layer mathematics, or any mechanism to manipulate a customer's standing. Merchants opt in not because they are persuaded by the entropy-reduction thesis but because the practical business case works: cheaper payment processing, an inbound customer pipeline, and better operational data than they had before. The engine layer holds the canonical formula from §4.1, the CT formula from §5.9, all six canonical tokens, the contributors performing validation tasks and the sensors and consensus mechanisms that resist forgery, and the rarity tables, domain weighting, and decay schedules that parameterize the whole system. This layer is never simplified into a public leaderboard, and its raw values are never exposed as a target a participant can directly observe and optimize against.

M.4 The Goodhart Mechanism, Stated as an Explicit Chain of Reasoning The chain runs: a visible metric attracts optimization pressure from the population it measures; optimization pressure, sustained, decouples the metric from its referent (§6); therefore, any metric that is both visible to and rewarding for the measured population will eventually decouple, given enough time and enough incentive weight; therefore, the only durable defense is to ensure the metric that actually drives reward is not the metric the population can see and target; therefore, the three-layer separation is not an optional UX choice but a load-bearing architectural response to a mathematically characterized dynamic (§6.2).

M.5 Separation as a Cultural Boundary, Not Only a Technical One The separation also functions as a naming and register boundary, documented explicitly in the underlying repository document: internal project material, this repository, development branches, and the companion book, uses a sharper, more irreverent internal voice, while external material, point-of-sale software copy, merchant-facing pitches, and consumer-facing marketing, uses a more measured, confident register appropriate for a general audience. The substance behind both registers is identical; only the presentation differs, and the document is explicit that the two registers must never be confused or mixed, since doing so would either alienate the general audience the external register is built for or dilute the internal clarity the sharper register is built for.

Falsifier. If any deployed user-facing surface publishes raw XP as a leaderboard, or if any deployed merchant-facing surface exposes an individual participant's XP balance without that participant's explicit consent, the layer separation is broken for that surface, and either the surface must be corrected or this Codex's Goodhart-resistance claim must be withdrawn for that surface specifically.

Appendix N. Implementation Status Snapshot

N.1 Purpose This appendix consolidates, in one place, an honest snapshot of what is complete, partial, open, or architecturally decided but operationally deferred, synchronized against the current repository at the time of writing, complementing the narrative treatment in §15.

N.2 Complete The canonical XP formula, single-sourced in packages/xp-formula/src/index.ts and stamped canonical-v3.12 (§4.1). The R, F, and reputation-density separation as an enforced architectural invariant, not merely a stated one (§4.2, §10.1). The six-token vocabulary and the eight-domain vocabulary, each fixed and reconciled against historical drift (§4.3, §4.4). The loop lifecycle state machine's logical specification (§8.4). The Non-Extraction three-test framework's specification, though not yet its full automated test harness across every mint-adjacent package (§13.4, §19.6).

N.3 Partial The v3.1 skeleton packages, identity , psll-sync , quest-market , and validation-neighborhoods , have specified interfaces and basic service scaffolding but incomplete production hardening (§15.5). Cross-domain ΔS normalization is documented at the architectural level with seed measurement operators, but per-domain calibration tables remain preliminary for all eight domains (§8.13). The DFAO governance layer's core mechanics are implemented, but several parameter-conflict and escalation rules remain open (§14.6).

N.4 Open The quorum-size formula for variable-domain validator rings (§20 of this Codex item 1). Cartel and Sybilcluster detection thresholds beyond the game-theoretic analysis in §16.1 (§20 of this Codex items 2 and 8). The verdict-vocabulary standardization between confirmed and supported (§10.10, §20 of this Codex item 25). Final ZKP scheme selection for every predicate type (§12.9, §20 of this Codex item 45). The Gödel Boundary Watchdog for self-referential claims (§17.7).

N.5 Architecturally Decided But Operationally Deferred Production-grade cryptographic hardening: the specific primitives (BLAKE3, Ed25519, BBS+, Poseidon where needed) are decided and documented (§17.8), but full circuit auditing and a hardened, non-sandbox transport layer remain deferred. The native-substrate decision itself is final (§11.2), but cross-node gossip protocol hardening for a genuinely adversarial, multi-operator network remains deferred (§11.10). The six-token economy's decay-rate values are provisionally set (§4.11, §14.4) but explicitly await Phase 2 modeling for several tokens' final rates.

N.6 Honest Summary The specification is substantially more complete than the reference implementation. This is expected and appropriate at this stage of the project: a specification that races ahead of its implementation, provided the gap is documented honestly rather than concealed, is a healthier state than an implementation that has outrun its own specification's rigor. Section 19's gap catalog and this appendix's snapshot are the same honesty applied at two different levels of granularity.

Appendix O. Comparative Analysis

O.1 Purpose A predictable and fair question for any new coordination protocol is how it differs from adjacent systems that share some of its surface features. The answer matters because individual surface features, a token, a DAG, a reputation primitive, a governance layer, are each unremarkable on their own. The Extropy Engine's distinctiveness, if it has any, lies in the specific combination under a specific set of architectural invariants, not in any single component.

O.2 Versus Conventional Cryptocurrencies A conventional cryptocurrency mints tokens through computational work or staked capital, and the token's value is set by market demand, independent of any specific real-world action. XP is not a cryptocurrency in this sense: it is non-transferable, it mints only when validated entropy reduction is recorded, it decays,

and its significance is not a market price but a tally of contribution that conditions other things, EP conversion at merchants, IT accumulation for governance weight, reputation-density accrual for validator routing. The closest analog among the six canonical tokens to a conventional token is CT, but CT differs meaningfully too: it carries a lockup

period, a transfer-friction coefficient, and an identity-bearing reputation-density component, meaning it cannot be purely traded as a speculative instrument without also carrying the contributor's identity weight. The fundamental distinction is that Extropy tokens are accounting receipts for validated work, not speculative assets; the protocol does not optimize for liquidity or market capitalization, it optimizes for verified entropy reduction.

O.3 Versus Reputation Systems Conventional reputation systems, seller ratings, review scores, credit scores, academic citation indices, attempt to capture trustworthiness in a single numerical score, and they share a predictable set of failure modes: reputation laundering, where the highest scores belong to actors who learned to game the signal rather than actors who are genuinely most trustworthy; reputation lock-in, where new entrants cannot accumulate standing against incumbents; and reputation as a tool of social control, where whoever controls the score controls behavior. The Extropy Engine treats reputation as a real but strictly bounded quantity: reputation density accrues from validated contribution, conditions validator routing, but never multiplies value directly, since the architectural invariant in §4.2 forecloses the reputation-laundering vector by construction rather than by policy. A system without this separation inherits reputation systems' failure modes wholesale; a system with the separation can use reputation for routing quality without exposing the value calculation itself to laundering.

O.4 Versus DAOs A conventional Decentralized Autonomous Organization makes decisions by token-weighted vote, and conventional DAOs share predictable failure modes: governance concentration among large token holders, low overall participation, governance theater where votes are nominal and execution is effectively unilateral, and regulatory ambiguity from token-weighted governance resembling securities activity. The Extropy Engine's DFAOs differ in several specific ways. Voting weight is not token-weighted in the conventional purchasable sense: the Influence Token decays continuously and cannot be purchased, only earned through contribution (§4.3, §10.1). The DFAO structure is genuinely fractal across five scales (§14.1), so concentration at one scale does not automatically translate into concentration at another. Conviction voting, the default for substantial decisions, rewards patient positions over impulsive tokenweighted swings (§14.2). Governance proposals are structurally visible and discussable, with deliberation periods that are architectural rather than optional. And DFAOs, while decentralized, are not unconstrained: parent-DFAO safety bounds prevent child DFAOs from violating protocol-level invariants (§14.3, §14.6).

O.5 Versus Holochain Holochain is a distributed-application framework built around personal source-chains and a DHT-based DAG. The Extropy Engine borrows several patterns from Holochain explicitly and with credit (§11.6): the PSLL is conceptually a source-chain, and the DAG substrate's data model is influenced by Holochain's approach. The distinction is that Holochain provides general infrastructure for building distributed applications, while the Extropy Engine provides a specific protocol with a value formula, a domain taxonomy, a token economy, and a governance model, reimplemented natively rather than run atop Holochain itself (§11.2). Holochain is infrastructure; the Extropy Engine is a system built with patterns inspired by that infrastructure's design.

O.6 Versus IOTA IOTA's Tangle uses a directed-acyclic-graph structure in which each new transaction confirms prior transactions directly, and the Extropy Engine's substrate is explicitly influenced by this approach, particularly its tip-selection algorithm and continuous-confirmation model (§11.3, §11.4). The distinction is that IOTA is a payment-focused DAG, while the Extropy Engine's DAG is contribution-focused: its vertex types (loop events, validation records, mint events, governance votes) are protocol-specific rather than transaction-focused, and the two systems share architectural patterns while operating on entirely different content.

O.7 Versus Universal Basic Income 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: UBI distributes value without measuring contribution at all, while the Extropy Engine distributes value only against validated contribution, providing a way to mint accounting against verified entropy reduction rather than a baseline stipend. The two are not contradictory; a society could operate both, with UBI handling baseline economic security and the Extropy Engine handling contribution-based reward layered on top. This Codex takes no position on UBI as policy; it takes a position only on what the Extropy Engine specifically does and does not do.

O.8 Versus Carbon Accounting Some non-extractive economic proposals center on carbon accounting, a unit of value tied to carbonequivalent measurement. The Extropy Engine's thermodynamic domain includes carbon-related entropy reduction as one instance among others, but the protocol is not a carbon-accounting system: carbon accounting focuses on a single physical quantity, while the thermodynamic domain is broader, covering waste heat, material flow, and general energy efficiency, and is only one of seven adopted mint-side domains (§8). Carbon accounting can be expressed as a sub-category of thermodynamic-domain claims within the Extropy Engine, but the protocol does not reduce to it.

O.9 Versus Effective Altruism Effective altruism advocates measurable, evidence-backed, comparative analysis of impact per unit of resource spent. The Extropy Engine's emphasis on falsifiable measurement and validated contribution overlaps substantially with this methodological commitment. The distinction is scope: effective altruism is a framework for individual giving decisions, while the Extropy Engine is a protocol for system-level coordination. The two are compatible rather than competing: an effective-altruism-oriented DFAO could configure its own domain-weight vector to prioritize whichever domains its own impact analysis judges most consequential, using the Extropy Engine as infrastructure for a value philosophy it does not itself mandate.

O.10 Versus Decentralized Identity Standards The W3C Decentralized Identifier specification provides a general framework for self-sovereign identity, and the Extropy Engine's identity layer is W3C-DID-compatible by design, with Verifiable Credentials following the corresponding W3C standard (§12.3, §17.10 domain reference). The distinction is that W3C DIDs and Verifiable Credentials are general infrastructure, while the Extropy Engine is a specific protocol

built on top of that infrastructure, adding the per-context nullifier, the threshold reveal scheme, PSLL Merkle anchoring, DFAO-scoped membership credentials, and the CAT issuance flow, none of which are part of the underlying W3C standards themselves.

O.11 The Synthesis The Extropy Engine combines patterns from cryptocurrencies (token accounting), reputation systems (reputation-density accrual), DAOs (governance), Holochain (the PSLL pattern), IOTA (the DAG substrate pattern), and W3C identity standards (DID, Verifiable Credentials), among other influences. The combination is what is distinctive; the individual components are not. The architectural invariants, reputation never entering XP, the participant remaining sovereign over identity and local context, and the DAG being event-sourced and append-only, are the protocol's genuinely distinctive contributions, present in none of the adjacent systems in combination, and they are the operational consequences of the theoretical foundations in §6 and §7, not arbitrary design preferences. A reader evaluating this protocol's distinctiveness should evaluate the invariants, not the individually unremarkable components; the components are widely available elsewhere, and the specific combination under these invariants is what this Codex claims as the contribution.

Appendix P. The Universal Times Calendar

P.1 The Calendar's Structure The repository contains a temporal-service package exposing a base-10 calendar system called Universal Times, cross-checked directly against docs/universaltimes-reference.html in the live repository. The calendar uses a base-10 hierarchy of time units, from largest to smallest: Eon, Age, Era, Epoch, Cycle (roughly a year-equivalent), Season (roughly a quarter-equivalent), Current (roughly a monthequivalent), Spin (roughly a day-equivalent), Tide (roughly an hour-equivalent), Wave (roughly a minuteequivalent), and GQ (roughly a second-equivalent), plus solar-aligned units named Solar Loop, Arc, and 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 by traditional units, sixty-second minutes, twentyfour-hour days, seven-day weeks, twelve-month years, and 365.25-day years, and a base-10 calendar admits scheduling math without conversion overhead between bases. The case against rests on cultural inertia: the seven-day week is a deeply embedded institutional rhythm, the twelve-month year underlies fiscal accounting, and the twenty-four-hour day tracks human

circadian biology, so replacing these with base-10 equivalents carries high adoption friction regardless of the arithmetic's elegance.

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, twenty-four-hour-day timeline, the default, or on the Universal Times timeline. The protocol's core temporal mechanics, loop timeouts, retroactive-burn windows, and decay schedules, are always expressed in canonical time units, seconds and days, underneath either presentation. Universal

Times is a presentation layer on top of those canonical units: a DFAO using it sees its loop windows expressed in Spins or Currents, but the underlying timing computation remains canonical seconds throughout.

P.4 The Port Collision Note The temporal-service package's default port, 4002, collides with the SignalFlow service's port, confirmed as a currently live discrepancy in the repository at the time of this Codex's drafting (packages/ temporal-service/src/server.ts defaults to port 4002 via an environment variable, the same default packages/signalflow uses). This is documented here as a known issue rather than silently corrected in this Codex's own service table, which lists both packages with their documented intended roles (§15.3): temporal , port 4011, handles seasons, decay scheduling, and loop timeouts, the core temporal mechanics; temporal-service handles the Universal Times calendar presentation layer specifically. The recommended cleanup, not yet applied in the repository as of this writing, is to assign temporal-service a non-colliding default port and to consider renaming both packages for clarity, temporal to something like temporal-heartbeat and temporal-service to something like temporal-universal-times .

P.5 Cultural Friction as a P3 Risk If the Universal Times calendar were ever activated as a protocol-level requirement rather than remaining a DFAO-gated option, the resulting cultural friction with seven-day-week institutions would be a real adoption risk, acknowledged already in §17.9 and in the gap catalog in §19.4. The current status preserves the cultural option deliberately: a DFAO that wants the base-10 calendar can use it, and a DFAO that wants the Gregorian calendar can keep using that, and the protocol does not force a calendar choice on anyone.

Falsifier. If the port collision documented in P.4 is resolved in a future repository revision without this appendix being updated to match, this appendix becomes stale and should be corrected at the next Codex revision.

Appendix Q. Developer Guide

Q.1 Use the Single-Source Formula Every minting service in the reference implementation imports the XP formula from exactly one place, packages/xp-formula . Any new service that needs to compute XP MUST import from there rather than reimplementing the formula locally, per the single-source-of-truth pattern the peer review in §18.1 specifically commended. A reimplementation anywhere else in the codebase is a bug by definition, regardless of whether the reimplementation happens to compute the same value, because it creates a second location that can silently drift from the canonical version.

Q.2 Canonical Field Names New code should use the canonical symbol names from §4.2 and the canonical field names from packages/contracts , not ad hoc alternatives. Where the repository's current field names have not yet caught up to this Codex's naming conventions, most notably the F-symbol collision discussed in §4.7 and §10.10, new code MUST use this Codex's convention (F as Frequency of Decay, $\mathscr{F}$ as Falsifiability). There is no F_freq. Flag any legacy field rather than silently perpetuating F-as-falsifiability or F_freq into new surfaces.

Q.3 PSLL Anchoring for New Services A new service that wants a participant's activity to be provably part of that participant's history should write to the participant's PSLL through packages/psll-sync , never by asking the participant to submit raw activity logs directly to a network-facing service. The correct pattern is always: local PSLL entry first, on the participant's own device, then periodic Merkle-root anchoring to the DAG (§12.7 through §12.8). A service that asks for raw activity data outside this pattern is violating Digital Autarky, per the falsifier in §9.10, regardless of how

convenient the shortcut seems.

Q.4 Boundary Validation Any service accepting a claim payload from the edge, from a personal AI's Claim Package, must validate the payload against the evidence conditions E1 through E3 (§9.5) before treating it as eligible for validation routing. Skipping this boundary check to "optimize" throughput reintroduces exactly the unverifiable-claim attack surface the evidence conditions exist to close, and any service that does so is non-conformant with the Protocol Contract in this Codex §4 regardless of its throughput numbers.

Q.5 Event Subscription Patterns Services that need to react to protocol events, a loop closing, a mint confirming, a governance proposal passing, should subscribe to the shared event bus (EventBus in @extropy/contracts) rather than polling service-specific endpoints or, worse, querying another service's internal database directly. Crosstopic ordering is not guaranteed by the event bus (§17.5), so any service whose correctness depends on the relative order of events across different topics must resolve that ordering against the DAG's own vertex sequence, which is authoritative, rather than assuming bus-delivery order.

Q.6 Respecting Three-Layer Separation in New Frontends Any new user-facing frontend must respect the three-layer separation from Appendix M: it may show savings, gamified feedback, and a self-curated character sheet, but it must never expose raw XP, perdomain rarity multipliers, or any other lever that maps legibly onto the mint formula's inputs. A frontend that violates this, even unintentionally, by adding a debug view that leaks engine-layer numbers to end users, breaks the Goodhart-resistance property this Codex depends on throughout, and should be treated as a defect with the same severity as a security bug.

Q.7 Engaging with the v3.2 Migration Path Developers building against the current v3.1.3 formula and protocol version should track docs/ CHANGELOG.md and the roadmap in §19.6 for the v3.2 migration path. Breaking-change governance protocol for this migration is itself an open gap (§20 of this Codex item 63), so developers should not assume any specific migration mechanism is already settled; the safest posture is to depend on the formula-version stamp (§4.1) being checked at every mint boundary, since that check is what will catch any migration-related drift automatically, by design, rather than silently.

Q.8 Engaging with the Gaps Honestly A developer or contributor encountering a genuine gap between this Codex's specification and the reference implementation's current behavior should file it against §20 of this Codex rather than quietly working around it in a way that papers over the discrepancy. The entire discipline this Codex depends on, the correction ledger in §4.6, the honest gap catalog in §19, and the peer-review response in §18, only works if new discrepancies are surfaced with the same honesty the existing ones were.

Appendix R. God Paper Companion Notes

R.1 Relationship to Appendix H Appendix H gave the full synopsis of the philosophical extension paper informally called the God paper: its reframe, its functional definition, its falsification conditions, its theological translations, what it does and does not claim, the terminology note regarding F, its connection to the protocol, and why the appendix exists at all. This appendix does not repeat that material. It exists only to note, briefly, that the two v1.0 appendices covering this material (one titled around the philosophical companion generally, one titled specifically as a companion appendix) have been merged into the single treatment in Appendix H for this Comprehensive edition, since the two were substantially duplicative once reconciled against the current repository and this Codex's naming-hygiene discipline.

R.2 Why the Merge Per the brief governing this edition's construction, appendices with substantially overlapping content should merge rather than both appear at full length, since duplication does not add information, only length. Appendix H's synopsis already covers the reframe, the functional definition, all four falsification conditions, all four theological translations plus the four adjacent-concept translations, the explicit disclaimer of what the paper does not claim, the F-terminology correction, the protocol connection, and the rationale for the appendix's existence. A reader seeking any

of that material should consult Appendix H directly.

Appendix S. Closing Note

S.1 What You Have Read This Codex is the comprehensive technical specification of the Extropy Engine, Comprehensive edition. It consolidates the current v3.1.3 canonical specification with the v3.1.3 corrections applied throughout; the architectural foundations, Digital Autarky, the contribution graph, the three-layer separation, and the person-as-DAG architecture; the mathematical foundation, the canonical XP, CT, and EP formulas, crossdomain normalization, and dimensional analysis; the eight canonical domains and their instruments; the loop lifecycle and the emergent-validation model; the DAG substrate; the identity layer, the PSLL, and Digital Autarky in operational detail; the six-token economy; the DFAO governance model and its provisional defaults; the Goodhart-resistance theory grounding the whole architecture; the formal governance foundations of Randall's Feedback V4; the implementation status cross-checked against the live repository; the public narrative layer; the security model and threat analysis; the peer-review response; the open engineering gaps and roadmap; the glossary and formula reference; and twenty-one appendices covering extended treatments of the formula, a worked loop example, DAG substrate detail, identity flows, a governance worked example, a convergence worked example, the HomeFlow pilot, sandbox posture and falsification conditions, the full RF V4 theorems, the meaning-drift mathematics, HomeFlow deployment detail, the three-layer separation in full, implementation status, comparative positioning against adjacent systems, the Universal Times calendar, a developer guide, the God paper companion, this closing note, domain deep dives, and a threat catalog worked through scenario by scenario.

S.2 What This Codex Replaces This Comprehensive edition, together with the prior Concise edition , replaces Codex v1.0 in full as the project's canonical technical reference. Prior synthesis drafts and intermediate blueprint documents that preceded v1.0 remain available as historical source material within the project's own records but are superseded by the current Codex in both editions.

S.3 What This Codex Does Not Replace This Codex does not replace the companion book, Randall Gossett, Unfuck the World for a Dollar (companion book; in progress): the book is the narrative case, and this Codex is the technical specification, and the two are complementary rather than substitutable. It does not replace the repository at github.com/00ranman/extropy-engine : the repository is the canonical implementation, and this Codex describes what the repository does, while the repository itself remains the final word on what the repository actually does. It does not replace the academic papers, Randall's Feedback V4, the Signal paper, and the God paper: this Codex condenses their load-bearing content, but the papers themselves contain the complete formal proofs and full case studies this Codex only summarizes. It does not replace the peer-review record referenced in §18: that review is preserved as the historical record of external examination, and this Codex documents the corrections applied in response to it, not the review itself.

S.4 How to Cite This Codex This Codex, in either edition, is a technical specification intended for publication and citation. When citing specific claims, prefer citing the underlying source material this Codex itself cites: for the XP formula and the current canonical specification, cite packages/xp-formula/src/index.ts and the the Correction Ledger in §3 v3.1.3 entry; for the formal governance foundations, cite Randall's Feedback V4: Formal Foundations for Emergence-First Governance; for the representational fidelity decay theory, cite the internal Signal paper; for the philosophical extension, cite the God paper; for the companion narrative case, cite Randall Gossett, Unfuck the World for a Dollar (companion book; in progress). When citing this Codex itself as a synthesis and specification document, the recommended citation form is: Randall Gossett, with AI-assisted drafting, disclosed, Extropy Codex, Version 2.0, Comprehensive Edition (2026), the reference implementation repository .

S.5 The Closing Statement The Extropy Engine is a protocol for measuring validated entropy reduction. It is documented in this Codex, implemented in the repository, narrated in the companion book, and grounded in the academic papers referenced throughout. The protocol is allowed to lose. Every claim in this document is subject to peer

review, adversarial pressure, and operational falsification, and this Codex has tried, throughout, to state in advance exactly what that falsification would look like rather than leaving it to be discovered after the fact. 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 we have is not broken by accident; it is working exactly as its incentives were designed to work. The Extropy Engine is an attempt to design a different one, and this 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.

Appendix T. Domain Deep Dives

T.1 Cognitive Entropy: Deep Dive Cognitive entropy is disorder in understanding, explanation, learning, conceptual compression, and mental-model coherence, grounded in the information-theoretic observation that a good explanation compresses a complex phenomenon into a simpler form without losing predictive fidelity, while a bad explanation either fails to compress or distorts the phenomenon it claims to explain. Operationalization paths include explanation-length compression under a content-preserving constraint, where the compressed explanation must still preserve the original's testable predictions, measured as a reduction in explanation length at fixed predictive fidelity; assessment-delta measurement, where a learner's beforeand-after performance on a validated instrument tracks genuine comprehension gain rather than test-taking familiarity; and misconception correction, verified when a learner's specific, documented false belief is replaced with a correct one that the learner can then apply to a novel case they were not directly taught. The domain's known failure mode is credential inflation, where the instrument drifts toward measuring test-taking skill rather than genuine understanding, mirroring the institutional-domain Goodhart dynamics described in §6.3; the falsification condition for any specific cognitive-domain instrument is that its scores must correlate with independently assessed comprehension (a different instrument, a practical application task) above a pre-registered threshold, and instruments that fail this correlation test are retired.

T.2 Social Entropy: Deep Dive Social entropy is disorder in coordination, trust, and cooperative structure. Operationalization paths include mediation outcomes, measured as a documented reduction in unresolved conflict incidents within a bounded community over a fixed window; trust-network coherence, measured through survey-based or behavioral proxies for whether members of a group can predict one another's cooperative behavior with improved accuracy after an intervention; and coordination-game resolution, measured directly as the Shannon entropy reduction in the assignment distribution over who does what within a group, per the measurement operator in §4.6 of this Codex §3.3. The domain's known failure mode is that selfreported trust improvements are easy to fabricate and hard to independently verify, which is why the evidence requirements in §9.5, particularly E1's independent reproducibility, weigh especially heavily for social-domain claims relative to, say, thermodynamic claims where physical measurement is more directly available.

T.3 Economic Entropy: Deep Dive Economic entropy is disorder in allocation, matching, and resource flow, explicitly never denominated in fiat under the Non-Extraction invariant (§13.3). Operationalization paths include matching-efficiency improvement, measured as a reduction in unmet-need-to-available-supply mismatch within a bounded market or allocation problem; waste-reduction measurement, tracking a documented decrease in discarded or idle resources against a baseline; and bottleneck removal, measured as a throughput increase in a coordination or production process attributable to a specific, documented intervention. The domain's known failure mode is conflating economic entropy reduction with simple cost-cutting that externalizes disorder elsewhere, which is precisely the "local but parasitic" failure pattern named explicitly in the God paper's falsification conditions (Appendix H.4) and which this domain's instruments must guard against by requiring the baseline and post-intervention measurement to account for the full bounded system, not just the sub-component the claimant controls.

T.4 Thermodynamic Entropy: Deep Dive Thermodynamic entropy is physical disorder, the domain with the most direct and rigorous grounding in the Shannon-Landauer-Bennett correspondence (§3.2). Operationalization paths include energy-use delta measurement, tracking a documented reduction in energy consumption for a fixed output; heat-loss

reduction, measured through direct instrumentation such as thermal imaging before and after an insulation intervention; and material-recovery-rate improvement in recycling or reuse systems, measured as an increase in the fraction of input material successfully returned to productive use rather than discarded. This domain's instruments are the closest to the physical domain's fast-feedback regime described in §6.3 (k approximately 0.02): physical reality provides rapid, hard-to-fake corrective feedback, which is why this domain is often the easiest to calibrate a defensible M_d for, and why it anchors the bits-equivalent common unit's Landauer-floor correspondence for the whole cross-domain architecture (§8.2).

T.5 Informational Entropy: Deep Dive Informational entropy is disorder in records, data quality, and archival coherence, and is the domain most directly connected to the reference implementation's current evidence surface through packages/githubparasite (§15.8). Operationalization paths include dataset error-rate reduction, measured directly as a decrease in documented inconsistencies or inaccuracies in a defined dataset; retrieval-latency and discoverability improvement, measured as a reduction in the time or effort required to locate a piece of information within a system; and, for prediction-loop claims such as a merged code patch or a proof, the log-likelihood-ratio of the claim under the pre-verification distribution, per §4.6 of this Codex §3.2. This domain's relative maturity, and its direct mapping onto version-control and code-review evidence that is already naturally reproducible and tamper-evident, is why §8.13 and §19.6 both identify it as the natural first domain for a fully adversarially tested M_d implementation.

T.6 Governance Entropy: Deep Dive Governance entropy is disorder in decision systems, accountability, and rule coherence. Operationalization paths include decision-latency reduction, measured as a documented decrease in the time between a decision being needed and a decision being made under a codified process; reversal-frequency reduction, tracking whether decisions made under a reformed process are overturned less often than under the prior process; and auditability improvement, measured as an increase in the fraction of decisions for which a complete, independently reviewable rationale exists. This domain has an unusually direct self-referential relationship to the protocol itself, since the DFAO governance layer described in §14 is itself a governance system subject to the same entropy-reduction framework it helps operationalize for other governance systems; this self-reference is treated explicitly rather than as an embarrassment, in the same spirit as the Signal paper's self-application discussed in §6.6.

T.7 Code Entropy: Deep Dive Code entropy is disorder in software systems: cyclomatic complexity, dead paths, test-coverage gaps, and defect density. Operationalization paths include cyclomatic-complexity reduction under a behaviorpreserving constraint, meaning the same test suite continues to pass after the reduction; test-coverage improvement against documented gaps, which requires exercising previously uncovered branches under adversarial inputs rather than padding coverage with trivial assertions; and defect-density reduction, controlled against release cadence to prevent gaming through slower shipping. Known failure mode: Goodhart drift toward measurable-but-hollow work, code churn that increases surface complexity while claiming to reduce it, tests that pad coverage without exercising meaningful behavior. Falsifier: scores

must correlate with independently assessed maintainability above a pre-registered threshold. This domain has the most mature evidence surface in the current implementation via packages/github-parasite (§15.8), which extracts PR-derived signal into contribution graphs.

T.8 Temporal Entropy: Deep Dive Temporal entropy is disorder in scheduling, queueing, and coordination timing within a bounded process. Operationalization paths include cycle-time reduction, measured as a documented decrease in the time from task initiation to completion for a repeated process; wait-time reduction, measured as a documented decrease in idle time within a scheduled system; scheduling-conflict-frequency reduction; and settlementtime-variance reduction. Known challenge: disentangling temporal contribution from throughput improvements that belong to economic entropy, since temporal claims must reduce uncertainty of whendoes-what-happen rather than merely producing more per unit time. Reference implementation lives in packages/temporal/ and the Base-10 Universal Times calendar in Appendix P anchors many temporal instruments to a shared, decimal, timezone-free reference frame.

Appendix U. Threat Catalog Worked Through

U.1 Scenario: Sybil Attack on a Validator Pool A determined attacker creates fifty identities through a combination of legitimate OAuth accounts and compromised or purchased on-device KYC material. The attacker's goal is to bias the validator pool for high-value claims toward the attacker's own submissions. Defense: SignalFlow routing is multi-factor (§10.3), and the attacker's fifty identities start at zero reputation, so the routing's reputation weight gives them minimal influence initially. To accumulate reputation, the attacker must have their identities correctly validate legitimate claims over time, which means doing the protocol's genuine work; the attack's cost is, structurally, the work itself. If the attacker attempts to bias a specific claim by having many of their identities selected simultaneously, the routing's load-balancing factor makes selecting many of the same operator's identities for one claim statistically unlikely absent detectable coordination, which the cartel-threshold analysis in §16.1 is designed to surface. Residual risk, stated honestly: the Sybil-byfraudulent-KYC vector remains open, and the protocol's defense depends entirely on KYC quality, which is governance-tunable per DFAO, meaning a high-stakes DFAO can require the strongest available KYC method (trusted-issuer handoff) while a low-stakes DFAO might accept a weaker one, and that variance in acceptable KYC strength is itself an attack surface a sufficiently patient attacker could target.

U.2 Scenario: Reputation Laundering Across DFAOs An actor with poor standing in one DFAO attempts to launder their reputation by joining a more sympathetic DFAO, accumulating fresh reputation density there, and then using that fresh standing to bias their value calculation elsewhere. Defense: the architectural invariant in §4.2 means reputation never enters the XP formula regardless of which DFAO it was accumulated in, so the actor's reputation density in DFAO B simply does not multiply their XP for claims in DFAO C. Reputation density is per-domain, not per-DFAO in a fungible sense, so their cognitive-domain standing in DFAO B does not directly transfer to bias DFAO-C routing in a way that affects DFAO-C XP outcomes. Residual risk: a sufficiently patient actor systematically building genuine reputation across many DFOAs is, at some point,

indistinguishable from a genuinely broadly contributing actor, which is arguably the desired outcome rather than a failure, since sustained genuine contribution across many communities is exactly the behavior the protocol wants to reward.

U.3 Scenario: Validator Cartel A group of eight validators within a twelve-validator DFAO conspires to confirm fraudulent claims for a share of the resulting payoff. Defense: the retroactive validation window (§9.6) allows independent observers, including non-cartel validators and external auditors, to submit falsifying evidence for the window's full duration, and validators whose consensus is later contradicted take a reputation penalty exceeding their expected collusion gain (§16.1). The cartel-threshold analysis shows that at validator population 10 or more with detection probability 0.3 or more, no cartel strategy is profitable in expectation; an eight-of-twelve cartel meets the population threshold, and the detection probability scales with claim visibility, meaning high-stakes claims carry higher detection probability and low-stakes claims lower. For low-stakes claims specifically, an eight-of-twelve cartel might remain profitable in expectation even under this analysis, though the per-claim payoff is correspondingly low. Residual risk: a cartel deliberately targeting a niche of low-stakes, low-scrutiny claims remains theoretically viable; the protocol's response is the epistemology-engine 's pattern-detection capability (§10.8), which flags validator clusters whose judgments correlate suspiciously across many claims for governance review, even absent a single dramatic detected fraud.

U.4 Scenario: Identity Compromise An attacker compromises a participant's device and steals their private key, gaining the ability to sign actions as that participant. Defense: the PSLL hash chain makes retroactive mutation of the participant's own history detectable (§12.7, §17.6), and the participant can execute the documented key-rotation flow (Appendix D.4), with the rotation event signed by both the old and new keys and anchored to the DAG; once anchored, actions signed by the old key are flagged as suspicious. The participant's token balances transfer to the new DID through the rotation event's signature chain, so the participant retains their accumulated standing despite the compromise.

Residual risk: during the window between compromise and detection, the attacker can sign actions as the participant, and those actions are permanently recorded on the DAG and cannot be erased. The protocol's response is to allow corrective vertices: the participant can submit a correction stating specific past actions were not legitimate, and validators can review and confirm the correction, which neutralizes the action's effects without erasing the historical record itself.

U.5 Scenario: Coordinated XP Inflation A coordinated group of contributors agrees to submit and validate each other's claims, attempting to generate XP without genuine entropy reduction. Defense: SignalFlow routes validators based on multifactor scoring, not submitter preference, so colluding submitters cannot simply select their own validators (§10.3). The frequency-of-decay term penalizes repeated same-fingerprint claim submissions, limiting how much XP repeated, low-substance claims can generate from the same actors (§4.4). The retroactive validation window allows independent observers within the DFAO who notice a suspicious pattern, many low-substance claims being confirmed among a tight group, to submit falsifying evidence, and the epistemology-engine 's pattern detection surfaces submitter clusters whose claims correlate suspiciously across many loops for governance review. Residual risk: a small, tight cluster operating in an obscure

niche might evade pattern detection for some time; the protocol's ultimate response is the gradual accumulation of evidence, since if the cluster's claimed entropy reductions never show up in actual downstream outcomes, their reputation eventually degrades even absent a single dramatic detection event.

U.6 Scenario: Regulatory Compulsion A state actor compels the protocol's infrastructure operators to reveal user identities or to restrict specific users' actions. Defense: the threshold reveal scheme requires cooperation among 7 of 12 ecosystemvalidator shareholders, distributed across multiple jurisdictions by design, so compelling all of them requires either genuine multi-jurisdictional cooperation or international legal process, not a single unilateral order to a single operator (§9.6). User actions cannot be unilaterally restricted by any infrastructure operator: actions are submitted from the participant's own device using the participant's own private key, which the protocol's operators cannot revoke. Operators can refuse to anchor specific PSLL roots or decline to route specific claims, but the participant's PSLL remains on their own device and can, in principle, be re-anchored to an alternative DAG instance under the multi-instance substrate design (§11.3, §11.9). Residual risk, stated without softening: the protocol provides no protection against direct compulsion of the participant themselves, who can always be personally ordered to reveal their own actions; the protocol's structural property is only that no third party can reveal the participant without the participant's own involvement or the full threshold-reveal process.

U.7 Scenario: Cryptographic Breakthrough A cryptographic breakthrough, for example a practical polynomial-time attack against Ed25519's discretelogarithm assumption, is published. Defense: governance approves a new signature scheme as the updated default (§17.8); new PSLL entries and DAG submissions adopt the new scheme going forward, while existing signed events retain their original signatures under a forward-only transition. Participants whose private keys are at elevated risk under the broken scheme are encouraged to rotate to keys under the new scheme, with rotation events themselves signed under a dual old-key-and-new-key flow until the old scheme can be fully retired. Residual risk: during the transition window, signatures under the old scheme remain forgeable, and the protocol's mitigation is to compress that window as much as possible, including governance-mandated accelerated rotation deadlines for high-value identities, rather than to claim the transition can be instantaneous. For a sudden, catastrophic break, such as a practical quantum attack arriving well ahead of the broader cryptographic community's migration timeline, the protocol's defense is explicitly limited: this Codex does not claim quantum security under all circumstances, only cryptographic agility and the capacity to migrate, which is a materially weaker but honest claim.

U.8 The Threat Catalog's Posture This catalog is not exhaustive, and real adversaries will find vectors it does not anticipate. The protocol's posture is not "every attack has been anticipated" but rather "structural defenses exist for the attack classes that are foreseeable today, and the mechanisms that catch novel attacks, the retroactive validation window, the epistemology engine's pattern detection, the governance layer's capacity to revise parameters, are themselves general

enough to catch attack classes this specific catalog did not name in advance." This is a weaker claim than blanket security, and it is the honest one.

Falsifier. If any scenario in this appendix is shown, in a live deployment, to succeed at materially lower cost or with materially higher probability than this appendix's analysis predicts, that specific scenario's defense claim is falsified, and the underlying mechanism, not merely this appendix's prose, requires revision.

Codex v2.1, in both the Concise and Comprehensive editions, supersedes Codex v1.0 in full. Framings v1.0 taught that v2.0 does not carry are retired in docs/REJECTED_FRAMINGS.md . This document is the internal specification; the external contract is the Protocol Contract in this Codex . The system is designed to be falsifiable, not infallible. Every domain defines what would prove it wrong. That is the difference between engineering and ideology.

Appendix V. Protocol Contract v0.2

This appendix is the external contract that Codex v2.0 referred to as PROTOCOL.md v0.1. It is included here so the Comprehensive edition does not depend on a second file.

V.1 Status and hierarchy

For any claim about the protocol:

1. This Codex (v2.1) is the internal specification.
2. This appendix is the external, implementation-agnostic contract.
3. The TypeScript reference implementation is the current realization. It may lag. Lag is a bug, not a license to treat the code as the spec.

V.2 MUST / MUST NOT

1. Every mint path MUST compute XP using canonical-v3.12 as stated in §4.1.
2. Every mint event MUST stamp `formula_version = canonical-v3.12`.
3. A mismatch between stamped version and executed math MUST fail closed.
4. There MUST be exactly one implementation of the mint formula imported by every minting service.
5. Reputation MUST NOT enter the XP formula.
6. Falsifiability $\mathscr{F}$ MUST NOT enter the XP formula. It MUST gate validation via V(Q).
7. There MUST NOT be a fiat bridge, a CT-to-fiat surface, a transferable wrapper, or a prediction market over loop outcomes, at any layer.
8. Loop state MUST advance only in the order in §8. There is no retro-mutation of a prior state.
9. The network MUST NOT ingest raw PSLL payloads. Only Merkle-root commitments are anchored.
10. Identity material (government documents, biometrics, raw KYC) MUST remain on-device. The network receives proofs, a per-context nullifier, and a public DID.
11. A claim with $\mathscr{F}$ below the protocol floor (default 0.50) MUST be rejected before slicing.
12. $\Delta S\_b_e \leq 0$ MUST NOT mint.

V.3 The validation score

```
V(Q) = ■(Q) × C_slice × (1 − e^{−λ N_val})
```

$\mathcal{F}(Q) \in [0, 1]$ is the Falsifiability Index. C_slice is the consensus ratio across blind one-tenth validation slices. N_val is the number of independent, non-colluding neighborhood validators. λ is a sensitivity scaling factor. This score lives in the Epistemology Engine. It is not a sixth XP factor.

V.4 SignalFlow routing

Default weights: domain_match 0.35, reputation 0.30, current_load_inverse 0.15, historical_accuracy 0.20. Reputation gates whose slice scores carry weight. It does not create a validator class.

V.5 Non-extraction tests

Every proposed feature must pass three tests.

- T1. Can an actor convert XP or CT into a claim on fiat, another ledger's transferable token, or any external market? If yes, reject.
- T2. Does the feature publish raw XP as a social score or leaderboard? If yes, reject.
- T3. Does the feature let reputation multiply new XP? If yes, reject.

V.6 Event vocabulary (minimum)

LOOPOPEN, EVIDENCE_SUBMITTED, VERDICT, LOOPCLOSE, LOOPREJECT, XPMINT_PROVISIONAL, XPMINT_CONFIRMED, XPBURN, PSLL_ANCHOR, GOVERNANCE_PARAM.

V.7 Formula-version quarantine

Pre-canonical mints, if any exist, remain quarantined under their original `formula_version` tag. They are not rewritten. They are not counted as canonical-v3.12.

Appendix W. Canonical Formula Notes and Reference Inventory

W.1 What the reference package currently does

The package `@extropy/xp-formula` still ships, in source comments, the retired form

```
XP = R × F × ΔS × (w · E) × log(1 / T■)
```

and computes `logDecay = Math.log(1 / Ts)`. That is the pre-canonical log path. Codex v2.0 already replaced it with a clamped log. Codex v2.1 replaces it again with

$\min(\alpha\_max, \max(1, T_s / T\_floor))$

The specification wins. The package must be updated to canonical-v3.12. Until it is, any mint that executes the log path is non-conformant even if the rest of the stack is healthy. This is Correction 3 applied to the live tree.

Required package behavior under v3.12:

- Reject $\Delta S \leq 0$.
- Reject $T_s \notin (0, 1]$.
- Clamp T_s into (T_floor, 1] before the fifth term.
- Compute fifth = $\min(\alpha\_max, \max(1, T_s / T\_floor))$.

- Return a breakdown that names R, F, deltaS, wDotE, and settlementTerm — not logDecay.
- Stamp `canonical-v3.12`.

W.2 Reference implementation inventory

The reference tree is a pnpm monorepo. The following packages are the ones this Codex treats as protocol-bearing. Application verticals are listed so a reviewer can see where loops enter the engine. Ports are the development defaults; they are not protocol invariants.

Core protocol

Package	Role	Dev port
@extropy/xp-formula	Single mint-formula implementation	library
@extropy/xp-mint	Mint pipeline, formula stamp, provisional / confirmed / burn	4001 class
@extropy/loop-ledger	Loop state machine	
@extropy/signalflow	Four-factor quest routing	
@extropy/reputation	Per-domain standing; vote weight; not an XP multiplier	
@extropy/dag-substrate	Append-only DAG, tip selection, quant timestamps	4008
@extropy/epistemology-engine	$\mathscr{F}$, V(Q), Goodhart monitors, Sybil observables	
@extropy/bayesian	Closure-confidence updates	
@extropy/contracts	Shared types and events	
@extropy/identity	DID, VC, nullifiers, ZKP wrappers	4101
@extropy/psll-sync	Merkle-root commitment receipts	
@extropy/credentials	Portable capability credentials	
@extropy/governance	Conviction voting, parameter updates	
@extropy/dfao-registry	DFAO registration and nesting	
@extropy/token-economy	XP / CT / CAT / IT / DT / EP accounting	
@extropy/temporal-service	Universal Times v4.2 clock	
@extropy/temporal	Temporal types and conversions	
@extropy/validation-neighborhoods	Neighborhood membership	
@extropy/node-handshake	Node authentication	
@extropy/quest-market	Quest surfacing	
@extropy/ethics	Principle checks on claims	
@extropy/decomposition-kit	Claim slicing	
@extropy/api-gateway	Edge routing	
@extropy/ecosystem	Ecosystem-level composition	

Verticals and frontends

Package	Role
@extropy/homeflow	Family / household NANO DFAO

Package	Role
@extropy/localflow	Local matchmaking vertical, port 4030
@extropy/grantflow-discovery	Grant opportunity matching
@extropy/grantflow-proposer	Grant proposal generation
@extropy/academia-bridge	Paper / claim ingestion
@extropy/levelup-academy	Adaptive learning vertical
@extropy/extropialingo	Language / literacy vertical
frontends/homeflow-ui	Household client
frontends/grantflow-ui	Grantflow client
frontends/character-sheet	Actor sheet
dashboard	Operator dashboard

W.3 Source-of-truth rule

If a package comment, README, or dashboard copy states a different XP formula than §4.1, the package is stale. Stale copy is not an alternative specification.

Appendix X. Full Engineering Gap Catalog (65 items)

This catalog is the complete contents of what Codex v2.0 cited as docs/GAPS.md. Counts: 26 P1, 23 P2, 16 P3. Updated 2026-05-06. Gaps are the engineering backlog. They are not a confession that the protocol is undefined.

X.1 P1 — Critical path (26)

Consensus mechanism details (7)

1. Quorum size formula for variable-domain rings.
2. Validator collusion detection thresholds.
3. Tie-break rules for split-quorum outcomes.
4. Late-arriving validation vote handling.
5. Consensus finality vs. retroactive-burn interaction.
6. Cross-domain consensus weighting.
7. Consensus failure recovery / re-validation protocol.

Economic attack resistance (6)

8. Cartel threshold formal analysis (>50% domain reputation).
9. Wash-loop detection across colluding identities.
10. Bribery resistance under IT decay.
11. Validator bid-rigging mitigation.
12. Funded-validator (corporate-capture) defenses.
13. CT lockup parameter optimization.

Validator selection optimization (5)

14. Four-factor weighting tuning (domain, reputation, load, accuracy).
15. Cold-start validator bootstrapping.
16. Geographic / language balancing in SignalFlow.
17. Adversarial-load shedding policy.
18. Sybil-resistant load distribution under burst traffic.

Cross-domain measurement calibration (6)

19. ΔS unit harmonization across eight domains.
20. Falsification-condition specification for the Cognitive domain.
21. Falsification-condition specification for the Social domain.
22. Falsification-condition specification for the Governance domain.
23. Calibration drift detection and auto-replace policy.
24. Inter-domain ΔS comparison weighting.

Verdict vocabulary (2)

25. Canonical affirmative verdict values: `confirmed` and `supported` are both in use. A single enum must be defined and enforced.
26. API field naming consistency: `statement` vs `content`, nested vs flat claim routes. Shared contract tests are required.

X.2 P2 — Robustness and security (23)

DAG distributed consensus (5)

27. Causal-edge gossip protocol.
28. Partition tolerance and merge rules.
29. DAG garbage collection and pruning policy.
30. Replay-attack protection.
31. PSLL-anchor receipt cadence.

Retroactive validation (4)

32. 30-day window edge cases under validator churn.
33. Burn-cascade limits when one loop's burn invalidates dependents.
34. Settlement reliability under network partition.
35. Retro-validation incentive structure.

DFAO governance edge cases (5)

36. MIGRATING-state hand-off protocol.
37. Quorum-loss recovery for MICRO tier.
38. Conflicting proposals across nested DFAOs.
39. Influence-decay edge cases on dormant members.
40. Cross-tier proposal escalation rules.

Token economy equilibrium (4)

41. IT 5%/month decay rate validation.

42. CT / EP decay rate finalization. Historical names GT and RT are not canonical.
43. Multi-token attack-surface analysis.
44. Token-velocity equilibrium modeling.

Privacy and access control (5)

45. ZKP scheme final selection (BBS+ is the default; zk-SNARK remains allowed for advanced use).
46. Selective-reveal threshold mechanics.
47. Nullifier collision-resistance argument.
48. PSLL selective-disclosure protocol.
49. Cross-DFAO data isolation.

X.3 P3 — Ecosystem maturity (16)

Skill DAG (3)

50. Skill-node progression criteria.
51. Skill verification source of truth.
52. Skill-graph traversal for SignalFlow routing.

Oracle integration (4)

53. External-data ingestion trust model.
54. Oracle-source diversity requirements.
55. Oracle-failure fallback policy.
56. XP minting from oracle-validated claims.

Performance and scalability (5)

57. Target throughput per validation neighborhood.
58. PSLL local-storage growth bounds.
59. DAG indexing strategy at planetary scale.
60. SignalFlow routing latency targets.
61. Cold-cache warm-up policy.

Migration and upgrade (4)

62. Prior-version state migration specification.
63. Breaking-change governance protocol.
64. Rule-module hot-swap procedure.
65. Deprecation lifecycle for retired services.