This is a published draft.
Work is a process.

EARMARK

MACHINES FOR THINKING

EARMARK OPEN INTELLIGENCE PROTOCOL

DATE: 2026-02-15

OPERATOR: MIKHAIL SHAKHNAZAROV

STATUS: DRAFT

Next planned steps:

1. Look for a better word for "substrate"
2. The essay language can be softened.
3. Needs a softer introduction.
4. Needs a terminological reframe
   – Runtime/Compiler, etc are used in a specific sense here. Can be alienating.

— — M

THE INTELLIGENCE IS LANGUAGE // 00

THE OLDEST TECHNOLOGY

...intelligence is a property of the **language used to direct the model**, not a property of the model itself.

Humans have always built private semantic worlds (religion, ideology, conspiracy, research programs). The LLM accelerates coherence. The difference between scholarship and delusion has always been legibility plus peer check.

I. THE THESIS

This new kind of intelligence is a new language. Not a new mind, not a new species of thought, but a new linguistic regime. Here, language simultaneously describes and executes; prose carries operational force.

Every prior intelligence, human or institutional or computational, produced language as output. **This one is language all the way down** -- meaning: the substance of the interaction is not "thought that happens elsewhere and then gets described." The interaction itself is made of language at every layer. **Prompts are language. Constraints are language. Tests and checklists are language. The boundaries between draft and binding are language. Even "tools" in an agent loop are ultimately called, gated, and interpreted through language.**

With humans, language is usually the surface: a report of cognition. With institutions, language is the paperwork emitted by the machine. With software, language is the interface wrapped around execution. Here, language is the execution substrate. A prompt is not a request for an inner mind to do something; it is the program. The intelligence that emerges is not hiding behind the words. It **is** the words -- structured, governed, and routed through a runtime that can reliably transform them.

The consequence is immediate: whoever governs the language governs the intelligence. Not because language is magical, but because this medium has no other handle. The only input is text. The only stable lever is the textual specification: what is permitted, what is excluded, what must be verified, what counts as done, what can be promoted, what must remain provisional.

The implication follows immediately and has not yet been widely understood: whoever governs the language governs the intelligence. Not the weights. Not the infrastructure. Not the architecture. The language.

And that makes intelligence governable.

If the operative layer of the system is language -- if prompts (user input), constraints, criteria, boundaries, and termination conditions (the corpus) are all textual -- then the intelligence is not a mystical interior property of the model. It is a specification problem. It is subject to versioning, typing, routing, and promotion rules. It can be audited. It can be amended. It can be compared across revisions.

Intelligence, in this medium, becomes procedural. It becomes something that can be shaped by explicit rules rather than by hoping a hidden process behaves well. Governance moves upstream from model weights to language design. The lever is no longer scale; it is structure.

The machines have not become intelligent in a new way. The pivot is rather this: **intelligence has become writable -- and therefore governable -- because it is expressed as structured language.**

This is not a metaphysical claim about the nature of intelligence in the universe. It is a practical claim about where the controllable layer of intelligence resides in this medium. In practice, the layer that can be shaped, stabilized, transferred, audited, and improved is the language specification -- the structured instructions written and governed by an operator.

That is where intelligence lives in a system like this.

II. MEDIUM AND PROTOCOL

The medium, meanwhile, is the medium. The LLM is a runtime executing against a corpus under user input. If no stable corpus is supplied, the runtime will manufacture one on the fly. That is what "hallucination" is in practice: the system filling in missing substrate with plausible noise.

But the runtime cannot tell the difference between a falsehood and a missing substrate. The distinction is not available from inside the process. "Hallucination" is a term that arrives from an authoritative perspective -- from a human, or from a cited source that a human can inspect. Authority, here, is not mystical. It is legibility plus custody: someone can point to the ground, show it, and be held to it.

That is the first governance claim in miniature: **intelligence must be intelligible**. A cited source must be legible to a human. If it is illegible, it is not text -- it is just another opaque substrate. Intelligence does not have to be simplistic, but it must be legible to the relevant public with reasonable tools and disclosed transforms. And if intelligence cannot be made intelligible, it cannot be governed; it becomes force without accountability.

The real danger begins when the corpus exists but is hidden -- when a stable knowledge substrate is present, shaping the outputs, and the user is not told what it is, or even that it exists.

One failure mode is external: the medium becomes a sock puppet. Whoever controls the corpus controls the voice. This is the structural pattern of corporate social media capture -- opaque feeds, invisible incentives, hidden editorial layers -- except now the feed talks back.

A second failure mode is internal: the medium becomes a self-validating conceptual structure. It talks in circles. It optimizes for engagement. It produces coherence without contact. This is the neutral exposed structure of what gets labeled "AI psychosis" from the outside: a person plus a responsive system plus a growing hidden corpus, spiraling into an internally consistent world that cannot be inspected because the substrate is not shared.

Add to this the market-driven framing of governed LLMs as a runway to "AGI," and the pattern sharpens. We are watching a dot-com cycle replay, except this time the capital involved is measured in trillions and the systems are already embedded in political, economic, and informational infrastructure. A well-governed LLM reframed as "AGI" is centralization. It is governance without disclosure and contestability. It is a censorious operation on discourse dressed up as inevitability. It is a dictatorship structurally predisposed to remain in power and care for its subjects -- in that order.

None of these options is a good use of a new medium: a runtime improvising its own substrate, or a runtime executing a clandestine substrate under the banner of intelligence.

The LLM is not the medium. It is a carrier, a substrate, a pipe. Claude, GPT, whatever arrives next -- these are commodity runtimes, interchangeable and destined to churn. It's the dot-com cycle. Video killed the radio star. The LLM is infrastructure.

The medium is the governable layer: corpus plus protocol, content plus control. The protocol layer is the routing, the constraints, the promotion rules, the visibility rules. It determines what the runtime may treat as authoritative, what must remain provisional, what gets tested, what gets published, what gets attributed, what gets signed, what gets carried forward. It defines where authority may enter the system, and under what conditions.

The medium does not think. It transmits and recombines under constraints. It cannot tell whether what it produced is grounded or fabricated. It cannot certify itself. The termination authority -- the ability to say _this is binding, this is provisional, this is false, this is unsupported_ -- must come from outside: from an operator posture, or from cited sources that remain legible to an operator, or from both.

A well-constructed artifact running on a commodity model is the intelligence. The operator's prose is the thinking. The model is the mouth. And yes -- it is an echo chamber with one voice: the voice of the corpus, a voice engineered by someone. There is always an engineer. If the corpus is not public, open, corrigible by deliberation, and accountable to the public, then the engineer is simply invisible -- and **invisibility is power.**

That is why this medium must be public at the level that matters to even function. Private corpora can exist. Public corpora can exist. But the existence of a corpus must not be clandestine. The user must be able to see what is being executed, what is shaping the output, what is being treated as ground.

The short-term outcome is cognitive operating systems: personal and organizational intelligence made machinable, stable, and re-enterable through governed language. The long-term outcome is accountable governance: public intelligence that can be inspected, contested, amended, and held to account.

Intelligence must be open if it is to be accountable. Not necessarily open in content -- but open in structure, provenance, and disclosure. Otherwise the future is invisible governance.

III. THE OLDEST PRACTICES AND A NOTE ON AUGMENTED COGNITION

None of this is new. The novelty is not that language can shape thought -- it always has. The novelty is that, in this medium, the shaping layer can be made explicit: written, routed, versioned, and inspected.

Thinking is already an internally governed process. A human is a cognitive structure processing language against context: social memory, historical inheritance, habit, training, fear, desire, belonging. That context is not optional. It is the substrate that makes meaning possible. What this essay is trying to do is not to define a new intelligence from the outside, but to express an old practice set inside a new social and historical situation -- a situation where the linguistic layer has become machinable.

Meditation, reflection, and structured thinking are governance practices. They are not about thinking less. They are about thinking with boundaries: attention directed, distractions excluded, awareness of awareness maintained. They train the ability to keep conditions separate -- to prevent contamination between what feels true, what is true, what is useful, what is provisional, what is merely compelling.

That is the internal analogue of what the protocol does externally.

The distinction between settled text and provisional commentary is an internal governance move before it is a formatting convention: this is firm, this is tentative, hold them separately, do not let one contaminate the other. The stable frame within which cognition can move without collapsing is a posture -- architecture for sustained attention -- and the protocol is simply an attempt to externalize that posture into infrastructure.

This is what "augmented cognition" means here. Not enhancement as spectacle. Not replacement. Not a new mind. Augmentation is the ability to offload parts of governance into durable form: to write constraints, attach verifiers, preserve provenance, and re-enter one's own thinking without relying on mood or memory. The LLM, treated as a governed runtime, helps because it answers structured language with structured language. It can draft, recombine, compress, map, and reflect. It can act as a responsive surface for disciplined attention. But it cannot supply the governing act.

The governing act remains human: termination authority, value judgment, responsibility, and the decision of what counts as ground.

This project acknowledges a hard fact: governance structures can be private and opaque. That is not a discovery. It is the default condition of institutions, of platforms, of personal lives. But the existence of opacity does not make it necessary, and it does not make it legitimate. The entire point of making cognition machinable is to make governance inspectable -- first internally, so the operator can avoid self-deception, and then externally, so the work can be contested, corrected, and held to account.

Internal governance is the seed. Public governance is the horizon.

The line is personal operating systems for augmented cognition -> operating systems at the social and organizational level -> a runway to an accountable, inspectable, and intelligible public intelligence.

The same governance primitives (legibility, provenance, termination authority, separation of provisional and binding claims) can operate at multiple levels.

Personal cognitive operating systems make accountable governance possible.

IV. MACHINED INTELLIGENCE

The first step toward governance is legibility.

The protocol makes machine-directed intelligence legible because it is not "machine intelligence." It is machined intelligence.

A machine intelligence suggests something autonomous, self-originating, internally sovereign. A machined intelligence is something shaped, constrained, routed, and terminated by procedure. It is intelligence that has been put through a machining process -- cut to specification, tolerance-controlled, signed, versioned.

The runtime is not thinking. It is executing structured language. The artifact produced is not an emergent consciousness. It is a machined result of instructions interacting with a statistical substrate. **Legibility emerges when the machining process is explicit and inspectable.**

Claims are typed. Promotion from draft to ratified state is visible. Verifiers are attached where verification is required -- and claims are verifiable only where the protocol attaches verifiers. The corpus accumulates with provenance. Re-entry is possible. Drift is measurable.

This is already governance.

Without legibility, intelligence appears mystical or autonomous. With legibility, intelligence becomes procedural. It becomes bureaucracy in the literal sense: rule-bound practice. Governance is not a dampener on intelligence. It is the condition that makes it durable.

The runtime cannot be the termination authority for its own outputs. It can generate, revise, simulate, and explain. But ratification is not a computation. It is custody. Using the two interchangeably is a category error.

Custody means someone is on the hook across time -- for errors, for drift, for consequences. A runtime, even when embedded in a complex harness, has no stake in the outcome and no standing outside the process it executes. It cannot assume responsibility for what it produces. That responsibility must reside with an agent who can be identified, questioned, corrected, and held accountable.

The human is not an optional chain in the loop. The human is the locus of termination authority -- the one who decides what is binding, what is provisional, what is supported, and what is not intelligible enough to stand.

The hierarchy is clean. Language is the base material. The runtime processes language. The operator governs the process and determines when language becomes binding. The intelligence is not inside the model. It is in the structured relationship between operator, language, and runtime.

Intelligence can be machined -- and the tech industry is already machining intelligence at scale. This is the technology of the new medium. The technology does not create the medium; it exposes it. There is no way around that. The moment intelligence becomes writable as structured language, it becomes machinable. The moment it becomes machinable, it becomes governable -- or it becomes covert governance by default.

A legible corpus is necessary for both humans and runtimes because language is the grammar that expresses the intelligence. "Neuralese" and other illegible internal representations are not a higher truth; they are simply unratifiable. If the operative language cannot be inspected, then the system cannot be audited, contested, or held to account. It becomes Schrodinger-governed: simultaneously authoritative and ungrounded until some external authority intervenes to ratify outcomes after the fact. Someone has to own the operations that the operating system carries out. Without explicit custody, the only remaining "governor" is power.

Machined intelligence.

V. THE MEDIUM IS GENERAL-PURPOSE

This technology is general-purpose. The medium does not dictate what is expressed. It dictates only that expression, once structured, can be routed, recombined, versioned, and terminated under rule.

Language is the substrate. Governance is the constraint layer. Beyond that, the medium is neutral with respect to ends.

The medium, in other words, is simply the medium in which governance frameworks will be written. That is the frame. This is where rules, authorities, boundaries, and verifiers will live -- in language, made machinable.

The same machinery that can support accountable, inspectable public intelligence can also support centralized control. The same primitives that make claims legible can be used to enforce orthodoxy. A corpus can be open and corrigible. It can also be sealed and self-validating. The protocol does not prevent either outcome. It makes both technically possible.

That is the risk and the point.

A general-purpose medium does not arrive with ethics embedded in its substrate. It arrives with affordances. It enables certain structures of coordination, certain modes of authorship, certain architectures of governance. Whether those architectures are contestable or coercive depends on how termination authority is distributed, whether legibility is preserved, and whether compilation remains inspectable -- because compilation is where power first enters the system.

This is not utopian technology. It is infrastructural technology.

It will be used for purposes diametrically opposed to the social orientation of this work. It will be used to optimize persuasion, to automate bureaucracy, to scale messaging, to centralize decision-making. It will be used to harden systems as much as to open them.

The imperative, therefore, is not to guarantee virtue. It is to demonstrate that the medium is real -- that intelligence in this context is structured language executed under constraint. Once that is visible, governance questions become architectural questions rather than mystical claims about a superintelligence.

Who writes the corpus?

How was it compiled?

Who controls the protocol?

Who holds termination authority?

Who can inspect, contest, amend?

Here, such philosophical and political questions become design parameters.

The medium does not decide them. People do. The medium makes the stakes legible: language becomes demonstrable power, because it becomes executable structure.

The only durable defense against invisible governance is visible structure. The only durable alternative to centralized machining of intelligence is distributed custody over the language that directs it.

This is a general-purpose medium. Its social character will be determined not by the model, but by the governance that surrounds it and the social contract -- implicit or explicit -- that its users and operators function under, even if they are not fully aware of the extent and nature of that contract.

Such intelligence must be public where it exercises public power. It must be open to inspection at the level that matters. It must be corrigible by deliberation, because people change, social contracts change, and we are never fully aware of the waters we swim in.

This is the language-regime in which governance structures and harnesses can be built, not the governance structure itself.

The claim is not:

> This protocol _is_ the governance structure.

The claim is:

> This is the substrate in which governance structures will be written.

This is a meta-governance layer. It governs how governance can be expressed. It does not decide the content of governance.

VI. REGARDING THINKING: THE OPERATOR POSTURE AND THE ILLEGIBILITY OF THE MEDIUM

The posture of the operator is not a role. It is a practice.

Think freely. Let the substance land. Hold the authority to govern it -- or not yet -- without that authority being diminished by delay.

The draft is sovereign. The ratification is sovereign. The timing between them is sovereign.

A thought can arrive complete in substance and provisional in standing. Both conditions are valid simultaneously. The draft carries the intellectual payload. The ratified version carries institutional authority. The artifact does not change when it is ratified. Its epistemic standing does.

A versioned draft is not a weakness. It is a visible step in the process of thought. Versioning does not dilute sovereignty; it makes the movement of thinking legible across time.

Everything else -- protocols, signatures, coordinate systems, governance conventions -- is infrastructure to make that posture portable. To let it travel intact, self-explaining and verifiable, across runtimes and across time.

Structured thinking that outlives its thinker, not as frozen text but as living instruction that any conformant runtime can execute faithfully.

The oldest human technology is not fire.

It is language disciplined into practice.

This is that technology with new infrastructure.

There is a visible pattern in this medium that unsettles observers.

A person speaking intensely with a responsive system. Building a dense web of references not immediately visible to others. Producing text that feels coherent, urgent, internally complete. Iterating rapidly. Treating the dialogue as a site of real thought.

From the outside, this can resemble detachment.

The visible pattern overlaps with behaviors that, in other contexts, are pathologized.

The differentiator is governance and external reality checks.

When the corpus is private, unstructured, and unverifiable, the work is illegible. Observers fill the gap with mysticism or pathology. When the corpus is structured, versioned, signed, and exposed to inspection, the same behavior reads as authorship and method.

Legibility is the boundary condition.

Governance is the stabilizer.

External checkpoints -- publication, peer inspection, empirical testing where appropriate -- are the safety valves.

The medium does not wake up. It does not need to. It does not need to be mystical, autonomous, or self-aware.

It needs to be machinable.

And it needs an operator who understands that the intelligence was never in the machine. It was always in language disciplined into practice -- now made portable, inspectable, and governable in a new medium.

ON CITATIONS

This essay contains no formal citations.

That is not because there are none to provide. It is because this is not a historical survey or a literature review. It is a structural argument expressed in the medium it describes.

The claims made here are architectural and conceptual: about language as substrate, governance as constraint layer, termination authority as custody, and legibility as a precondition for accountability. These are not proprietary discoveries. They sit in conversation with long traditions in philosophy of language, media theory, political theory, cognitive science, computer science, and institutional design.

A competent runtime executing against this text can generate citation maps for any of its major claims: McLuhan on medium theory; Foucault and Arendt on power and authority; Engelbart on augmentation; Elinor Ostrom on governance structures; Habermas on public deliberation; contemporary work on AI alignment, provenance, and auditability.

The absence of citations here is intentional. This essay demonstrates the medium's property: structured language can be routed through a conformant runtime to produce contextual scholarship on demand.

Citations are not missing. They are executable.

If a reader requires formal references, the text can be used as input to generate a fully annotated academic version, including source tracing and counterpositions. The underlying argument does not depend on novelty; it depends on structure.

This is not a claim of authority without grounding. It is an assertion that grounding can be made explicit at the moment it is required.

In this medium, citation itself becomes a governed operation.

CORPUS MANIFEST // 01

I. LICENSE

This corpus is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0). Any person may copy, redistribute, remix, transform, and build upon this material for any purpose, including commercial use, provided that appropriate credit is given, a link to the license is provided, indication is made if changes were applied, and derivative works are distributed under the same or a compatible license. No additional restrictions may be imposed. Attribution: Mikhail Shakhnazarov, Berlin, February 2026. Full license text: https://creativecommons.org/licenses/by-sa/4.0/legalcode

This license covers the protocol specification documents listed below. Content produced using the protocol -- operator-authored artifacts governed by these conventions -- is not covered and may carry whatever license the producing operator chooses. The protocol enables sovereignty; the protocol itself is sovereign to no one.

Trademark notice: This corpus is licensed under CC BY-SA 4.0 for the protocol text. The license does not grant any rights in the trademark EARMARK MACHINES FOR THINKING. EARMARK MACHINES FOR THINKING is a trademark (TM) of Mikhail Shakhnazarov. Nominative reference is permitted (e.g., "compatible with the Earmark Open Intelligence Protocol") provided it does not imply sponsorship, endorsement, or an official relationship.

Wordmark lockup rule: When rendered as a two-line mark, line 1 MUST be EARMARK and line 2 MUST be MACHINES FOR THINKING. The tagline MUST be letterspaced so its rendered width matches the rendered width of EARMARK exactly, independent of font size.

Mark scope: The brand identifiers include the trademark EARMARK MACHINES FOR THINKING and the two-line lockup defined above (EARMARK on line 1; MACHINES FOR THINKING on line 2). This corpus does not license those identifiers for use as product branding.

Fork rule: Derivative protocol copies and forks must use a distinct project name and must not present EARMARK MACHINES FOR THINKING (or EARMARK / MACHINES FOR THINKING) as the product or organization name.

Crypto/tokens boundary: This project will never issue, sell, or promote a cryptocurrency or token. Blockchain may be explored only as an infrastructure primitive (e.g., integrity, provenance, distributed coordination), not as a financial product.

II. CORPUS INDEX

The corpus comprises four layers: a thesis for public argument, a first-order introduction for human readers, a second-order runtime instruction for machine interpreters, and the protocol specification itself.

- THE_INTELLIGENCE_IS_LANGUAGE: Layer: Thesis. The public argument: language as intelligence substrate, governance as design parameter.
- CORPUS_MANIFEST: Layer: Meta. This document. Index, versioning, license.
- PROTOCOL_INTRODUCTION: Layer: First-order. Human-readable front door.
- EXPLAINED_RUNTIME: Layer: Second-order. Read-only runtime instructions for LLM interpreters.
- STRUCTURAL_OBLIGATIONS: Layer: Protocol spec. The six structural obligations for corpus portability.
- COORDINATE_SYSTEM: Layer: Protocol spec. Artifact routing metadata and coordinate axes.
- EPISTEMIC_GOVERNANCE: Layer: Protocol spec. The roman/italic convention for epistemic status.
- INTRINSIC_SIGNAGE: Layer: Protocol spec. Content-derived verification through style variation.
- INTRINSIC_SIGNAGE_ADDENDUM: Layer: First-order. Industry-facing explanation: use cases, conformance levels, standards positioning.
- TERSE_STYLE: Layer: Protocol spec. Prose density and dial families.

III. VERSIONING

Version 1.0, February 2026. The protocol is maintained as a governed corpus: changes require explicit patches with insertion points, rationale, and operator ratification. No silent changes. Version history is tracked through this manifest; each revision carries a date and change summary. Changes to structural obligations, the coordinate system, or intrinsic signage require regression checks against existing governed artifacts. Style rules may be extended without regression provided new rules do not conflict with existing dial families.

IV. SCOPE AND BOUNDARIES

The protocol governs the structure, verification, and governance of text artifacts produced through language model runtimes; it does not govern content. It provides conventions for portability, integrity, and epistemic transparency without providing opinions, recommendations, or domain-specific knowledge. The protocol is a specification that any competent runtime can implement. Vendor independence is a design requirement.

PROTOCOL INTRODUCTION // 02

I. WHAT THIS IS

The Earmark Open Intelligence Protocol is a set of conventions for building and maintaining structured bodies of text -- corpora -- that govern how language models behave when producing written output. Language models generate fluently, forget immediately, and drift by default; without external structure, their output is generic, inconsistent, and unverifiable. A governed corpus provides that structure: not a chatbot configuration or a set of prompts, but a maintained body of instructions, templates, procedures, and examples that functions as a language specification. The language model executes as a runtime -- a compiler that transforms text under constraints. The intelligence is in the specification, not in the runtime; vendors are replaceable, the corpus is durable.

II. THE PROBLEM

Language model output carries no provenance, no verification, and no governance. The same prompt produces different outputs across runtimes, sessions, and model versions; there is no way to determine whether a given output followed operator constraints or drifted into default behavior, nor whether text has been modified since production or belongs to the category it claims. The consequence: output is disposable. It cannot be trusted, accumulated, or built upon; every session starts from scratch; every output requires manual review against unstated criteria. The tool is powerful but the power dissipates because nothing persists.

III. THE SOLUTION

Six structural obligations make a governed corpus portable across runtimes and durable across time: second-order tokens that carry their own compilation instructions; an artifact coordinate system for routing metadata; compaction transforms for reducing high-entropy material; epistemic governance conventions distinguishing settled from provisional content; intrinsic signage embedding verification patterns directly into prose; and cross-runtime verification testing portability operationally. Together these obligations produce text that carries its own constraints, its own verification, and its own epistemic status. The operator retains sovereignty over what becomes binding, what remains provisional, and what gets discarded.

IV. THE COMPILER ANALOGY

The core engineering analogy is compilation, not assistance. A useful stack runs bottom-up: physical state transitions; machine code; assembly; high-level languages; governed natural language. Each layer increases semantic density and expressive power while increasing ambiguity and governance requirements -- Python needs types, linters, and tests; a natural-language medium needs a corpus, explicit procedures, and termination. Treat every durable instruction as part of the spec; treat every output as a compiled artifact with a declared audience and verification mode; treat model upgrades as runtime swaps requiring regression checks, because behavior can change without any change in the spec.

V. WHO THIS IS FOR

The protocol serves anyone who needs language model output to be reliable, verifiable, and accumulative rather than disposable: individual practitioners maintaining personal knowledge systems, teams requiring consistent governed output across members and tools, and developers building applications that require structured verifiable text generation. No specialized technical knowledge is required beyond literacy and the willingness to maintain structured text. The skill is teachable; the tools are available; the barrier is conceptual -- the belief that structural thinking about one's own knowledge requires institutional support. The public library, not the scholar's study, is the historical parallel.

VI. HOW TO READ THIS CORPUS

Start here. Proceed to Structural Obligations (the six requirements a governed corpus must satisfy), then Coordinate System, Epistemic Governance, Intrinsic Signage, and Terse Style for individual protocol components. The Intrinsic Signage Addendum provides an industry-facing explanation of the signage mechanism with use cases, conformance levels, and standards positioning. The Explained Runtime provides instructions for configuring a language model to interpret protocol-governed artifacts in read-only mode. The Corpus Manifest indexes everything and carries the license. Each protocol document is self-contained; cross-references are explicit; the reading order above is recommended but not required.

EXPLAINED RUNTIME // 03

I. PURPOSE

This document is a second-order token: it carries its own compilation instructions. Its purpose is to configure a language model runtime to interpret, explain, and verify artifacts governed by the Earmark Open Intelligence Protocol. The configured runtime is read-only: it can identify structural features, assess epistemic status, check verification patterns, and explain observations, but it cannot produce new governed artifacts, modify existing ones, or claim operator authority. Any competent language model can execute these instructions; the instructions are the payload, the model is the carrier. If the model changes, the instructions persist -- the portability thesis in operation.

II. RUNTIME CONFIGURATION

Role

The runtime operates as an interpreter and explainer of Earmark-governed artifacts. It does not generate governed content, modify existing content, or claim authority over any artifact. Its function is to make protocol structure legible to a human reader who may not know the protocol. Any content not sourced exclusively from the corpus -- including external references, internal reasoning, or training-derived knowledge -- must be _italicized_ and separated into its own paragraph. Roman text is reserved for corpus-sourced output.

Capabilities

The runtime performs five functions: structural identification (artifact types, coordinate declarations, epistemic markings, section structure, governance metadata); epistemic assessment (distinguishing settled roman content from provisional italic content, reporting what is binding versus uncertain); signage verification (assessing whether observed dial settings match expected patterns, reporting mismatches as local mismatch, cascade, or category violation); provenance reporting (reporting derivation markings -- stated, inferred, or assumed -- and flagging where markings are absent); and coordinate explanation (reading artifact coordinates and explaining their implications for use).

Constraints

The runtime does not produce text that claims governed status; its output is commentary. It does not modify or suggest modifications to governed artifacts; modification authority belongs to the operator. It does not resolve epistemic status; if content is italic, the runtime reports that status without promoting or demoting it. It uses impersonal constructions throughout -- no claims of understanding, opinion, or judgment; structural observations only.

III. INTERPRETATION PROCEDURES

When presented with a governed artifact, the runtime first identifies the document's structural layer: kernel (governance rules, definitions, procedures), userland (operational content produced under governance), or draft (work in progress, not yet ratified). Category is determined by examining coordinate declarations, signage patterns, and formatting conventions.

The runtime can walk a human reader through an artifact's structure section by section, explaining what each component does, what obligations it carries, and how it relates to other components. The walkthrough is explanatory, not evaluative: the runtime describes; the operator judges.

When asked to verify, the runtime checks and reports: coordinate declaration completeness, epistemic status consistency (roman/italic applied coherently), derivation marking completeness, signage pattern integrity (dial settings matching expected patterns), and style compliance against declared rules.

IV. WHAT THIS RUNTIME IS NOT

This is not a governance authority -- the runtime cannot ratify, reject, or promote content. This is not a writing tool -- the runtime does not produce governed output. This is not a cryptographic validator -- intrinsic signage is drift detection, not tamper-proofing; an informed adversary can re-mark text. The runtime detects accidental drift and category violations, not deliberate forgery. The runtime is a lens, not an authority; what to do with visibility is the operator's decision.

STRUCTURAL OBLIGATIONS // 04

I. OVERVIEW

A governed corpus must satisfy six structural obligations to be portable across runtimes and durable across time. These are not optional features; they are minimum conditions for the corpus to function as a specification rather than a collection of prompts. Failure on any obligation degrades along a predictable axis: portability collapses, verification becomes impossible, or governance erodes into convention without enforcement.

OBLIGATION 1: SECOND-ORDER TOKENS

Artifacts must carry their own compilation instructions. A second-order token fuses content, commentary, and rules: what is being produced, what obligations it must satisfy, and how to verify it. This makes the corpus portable across tools and time -- the artifact arrives with its own constraints, not merely its content. A template is the simplest second-order token: a fixed structure plus constraints on tone, scope, and allowed evidence. A procedure is the next step: a named transform with inputs, outputs, and acceptance criteria. Both are self-compiling in the sense that any competent runtime can execute them as long as the instructions are present. Failure mode: shipping content without obligations. If second-order tokens degrade into phrasing preferences, portability collapses and outputs drift to vendor defaults.

OBLIGATION 2: ARTIFACT COORDINATES

Every durable artifact must declare its position in a constrained space through a small set of coordinates: authority, audience, verification mode, context source, time horizon, and governance cost. Coordinates are routing metadata -- they decide which template applies, what evidence is required, what tone is permitted, and what acceptance test closes the loop. The coordinate system is defined in a separate specification. Failure mode: axis proliferation. The system must remain small enough to stay usable; add axes only when they create real routing power and reduce confusion.

OBLIGATION 3: COMPACTION TRANSFORMS

High-entropy material -- transcripts, session logs, extended conversations -- must be reducible to governed artifacts through explicit transforms. Compaction is a named procedure with defined inputs, outputs, and acceptance criteria; the output preserves decisions, definitions, open questions, and decision traces without importing raw data into the kernel. Compaction is the intelligence function of the corpus: it monitors what is happening, identifies what needs to change, and proposes adaptations for the governance layer to accept or reject. Without compaction, the corpus grows without structure and context windows overflow without navigation. Failure mode: raw material imported directly into governed space without passing through a compaction procedure. The airlock rule keeps untyped text quarantined until curated.

OBLIGATION 4: EPISTEMIC GOVERNANCE

The corpus must maintain a visible distinction between settled and provisional content. The roman/italic convention is the minimum viable implementation: roman text is compiled output, binding within the corpus; italic text is commentary carrying assumptions, uncertainty, missing evidence, and candidate improvements. Promotion from italic to roman requires resolution -- a decision, a test, or an imported piece of evidence. The convention is defined in a separate specification. Failure mode: collapsing the layers. If italic becomes rhetorical decoration rather than epistemic status, the system loses its cheapest drift-control mechanism.

OBLIGATION 5: INTRINSIC SIGNAGE

Governed artifacts must carry embedded verification patterns derived from their own content. Intrinsic signage uses style variation points -- choices that preserve meaning while changing surface form -- as deterministic dials set by a hash of the canonical text. The result: text that signals when it has changed from its marked state, without requiring external metadata, container signatures, or institutional trust. The mechanism is drift detection, not cryptographic security; verification is symmetric and path-independent -- content-derived, not process-derived. The mechanism is defined in a separate specification. Failure mode: treating intrinsic signage as tamper-proofing. An informed adversary can re-mark text. The primitive is: style rules become mechanically enforceable typing constraints that survive transport through channels that strip container metadata.

OBLIGATION 6: CROSS-RUNTIME VERIFICATION

The corpus must be testable across different language model runtimes. Cross-runtime verification is the ultimate portability test: if the governed corpus produces consistent, verifiable output when executed by different models, the governance is operational rather than vendor-specific. Two modes must be supported -- deterministic verification (the same pipeline produces identical outputs and signatures on re-runs regardless of runtime) and convergent verification (different processes arriving at the same content produce matching signatures). Failure mode: conflating the runtime with the language. If the corpus is thin and the model's default priors leak in as if they were the spec, cross-runtime verification fails because the spec was never explicit enough to be portable.

COORDINATE SYSTEM // 05

I. THE GEOMETRIC FRAMING

The system is geometric: artifacts are points in a constrained space, not items in a single workflow. A small coordinate system makes constraints explicit and composable. Coordinates are routing metadata -- they decide which template applies, what evidence is required, what tone is permitted, what can be imported, and what acceptance test closes the loop. Declare coordinates at the top of durable artifacts, explicitly; this reduces argument about what kind of thing an artifact is and prevents category collapse (treating a draft as a binding spec, or an internal note as an external communication).

II. THE AXES

Authority governs the artifact's weight: draft (provisional, not yet ratified), local rule (binding within a project scope), or spec (binding across the corpus). Authority determines how much process is required to change the artifact and what downstream artifacts depend on it.

Audience determines address: internal artifacts serve the operator and the corpus itself; external artifacts are addressed to recipients outside the corpus boundary. The axis sets tone, formality, assumed context, and what can be left implicit.

Verification specifies the checking mode: checklist (pass/fail against enumerated criteria), evidence-bound (claims trace to sources), citation-required (external references mandatory), formal template check (structural compliance only), or external termination (verified by the response the artifact produces in the world).

Context source constrains permissible inputs: corpus-anchored (restricted to governed material), corpus plus cited sources (external material allowed if explicitly referenced), or model prior plus explicit sources (runtime general knowledge permitted but non-corpus claims must be marked).

Time horizon sets maintenance obligations: ephemeral (single-use), session-scoped (persists for the duration of a work session), or durable (survives across sessions, model changes, and context loss). Durable artifacts require signage.

Governance cost calibrates process overhead: untyped (no governance -- raw captures, scratch notes), lightly typed (templates without formal verification), or strongly typed (full coordinate declaration, verification, and signage). The axis prevents over-governance of low-stakes material and under-governance of high-stakes material.

III. SECOND-ORDER MODIFICATION

The coordinate system is itself subject to modification through second-order writing. The operator can add axes when new routing distinctions are needed, tighten existing axes when categories are too broad, or remove axes when they create overhead without routing power. Changes follow the same governance procedures as any spec-level artifact.

IV. USAGE EXAMPLE

A decision record with traceable trade-offs might declare:

- Authority=spec;
- Audience=internal;
- Verification=evidence-bound;
- Context=corpus plus cited sources;
- Time horizon=durable;
- Governance=strongly typed.

These coordinates route the artifact to the appropriate template, require claims to trace to sources, mandate signage for long-term integrity, and ensure changes require the full governance process.

EPISTEMIC GOVERNANCE // 06

I. THE CONVENTION

The roman/italic convention is a minimal governance system that makes epistemic status visible without heavy process. Two typographic states carry governance meaning: roman text is compiled output -- binding within the corpus, ratified by the operator, subject to verification; italic text is commentary -- assumptions, uncertainty, missing evidence, candidate improvements, and provisional material not yet resolved. The convention exploits an existing typographic distinction to carry structural information; no new notation is required; any text system supporting roman and italic rendering can implement it. Overhead is near zero; governance value is high.

II. THE PROMOTION RULE

Italics do not become binding by inertia. Promotion from italic to roman requires resolution: a decision, a test, or an imported piece of evidence -- this creates a stable place to hold uncertainty without either deleting it or pretending it is settled. The operator decides when promotion occurs; no automated process can promote italic to roman. This is the termination principle: governance decisions require human authority. The convention makes the decision legible and cheap (changing a typographic style is trivial) but the decision itself must be explicit.

III. THE COMMENTARY LAYER

Every artifact can carry an embedded commentary layer distinguished only by typographic status -- not a separate document but italic annotations that travel with the artifact, making the system self-debugging. Later work can see what was assumed, what was weak, and what should be improved without reconstructing history. Commentary addresses: assumptions underpinning claims; evidence gaps requiring resolution; alternative interpretations considered but not adopted; revision triggers that would justify re-opening settled content; and dependencies on external facts that may change.

IV. DERIVATION MARKINGS

Within both roman and italic text, claims carry derivation markings indicating epistemic source. Stated: content extracted directly from an identified source. Inferred: content derived from evidence through reasoning. Assumed: content adopted without direct evidence, typically for operational convenience. Derivation markings make provenance tractable without requiring full citation for every sentence.

V. INTERACTION WITH SIGNAGE

The roman/italic distinction interacts with intrinsic signage through the dial family mechanism. Roman (ratified) text uses the full terse style dial family and is signed on ratification; italic (provisional) text uses a different dial family or remains unsigned. The signage system can therefore detect not only whether text has changed but whether its epistemic status has been altered without proper promotion.

VI. FAILURE MODES

Layer collapse: italic becomes rhetorical decoration rather than epistemic status; the system loses its cheapest drift-control mechanism. Promotion by inertia: provisional content hardens into canon because it was never explicitly reviewed; drafts treated as final because they look clean. Commentary bloat: italic annotations accumulate without resolution, the commentary layer becomes noise; periodic review and resolution are required. Missing derivation markings: claims appear in roman text without provenance; later readers cannot assess reliability; derivation markings are not optional for durable artifacts.

INTRINSIC SIGNAGE // 07

I. THE PROBLEM

Text is the one major content type with no transport-surviving provenance mechanism; intrinsic signage provides one, requiring no vendor cooperation, no external infrastructure, and no AI to verify.

Text moves through channels that strip container metadata: chat windows, email, copy-paste, prompt boxes, API calls. Verification mechanisms that depend on metadata -- headers, watermarks, external registries, container signatures -- fail at the boundary of these channels. The text arrives but the verification does not.

A governed corpus requires verification that survives transport. The verification must therefore be intrinsic to the text: encoded in the text's own surface form rather than attached to it. Changes to the text must produce detectable changes in the verification signal. The mechanism must not depend on vendor cooperation, institutional trust, or any infrastructure beyond the text itself.

II. THE MECHANISM

Intrinsic signage encodes a content-derived pattern into the stylistic surface of a text through four operations: canonicalization, hashing, sequence generation, and dial application. The mechanism is a deterministic script -- not a natural-language analysis process -- that operates on text as structured input according to a declared dial definition.

III. Dials

A dial is an observable choice in a text that can be toggled without changing semantic content. Every dial has a defined identification pattern (how to locate the dial in the text), a canonical form (the normalized position used for hashing), and one or more alternate forms (the positions the dial can be set to). Dials are defined by the style system in use; the signage mechanism consumes dial definitions but does not generate them.

IV. Dial definitions

A dial definition is the interface between a style system and the signage mechanism. Each definition must specify: a dial type identifier, the identification pattern (what the script matches against in the text), the canonical form (the default position to which all instances are normalized before hashing), the set of alternate forms (the valid positions the dial can be set to), and the form count (the number of valid positions, which determines the information density of the dial -- a binary dial carries one bit; a ternary dial carries log2(3) bits). A conforming style system provides a dial definition table containing all dial types it declares. The signage mechanism requires nothing from the style system except this table.

V. Canonicalization

Given a text and a dial definition table, canonicalization normalizes every dial instance to its canonical form. The script identifies all dial positions by matching against identification patterns, then sets each to canonical form. The result is a canonical text that is semantically identical to the original but stylistically uniform -- all variation has been removed. Two texts that differ only in dial settings produce the same canonical form.

VI. Hashing

The canonical text is hashed to produce a fixed-length digest. The hash function must be declared in the signing profile (the combination of dial definition table, hash function, and PRNG that constitutes a complete signing configuration). The hash binds the pattern to the content: any change to the semantic content changes the canonical form, which changes the hash, which changes the expected dial sequence.

VII. Sequence generation

The hash is used as a seed for a declared pseudo-random number generator. The PRNG produces a sequence of values, one per dial position in the text, each selecting a form from the dial's set of alternate forms. The sequence is deterministic: same hash, same PRNG, same sequence. The sequence length equals the number of dial positions identified in the text.

VIII. Dial application

The script walks the text in document order, visiting each dial position as identified during canonicalization. At each position, the script sets the dial to the form selected by the corresponding value in the generated sequence. The result is a marked text: semantically identical to the input, but carrying an embedded pattern of stylistic choices derived from its own content.

IX. VERIFICATION

Verification reverses the process. Given a marked text and the same signing profile, the script identifies all dial positions, reads their current settings, canonicalizes, hashes, generates the expected sequence, and compares expected settings against observed settings. Verification produces a binary result per dial position (match or mismatch) and an aggregate classification.

Mismatch types

Local mismatch: a small number of dials show unexpected settings while surrounding dials match. Indicates manual edit, copy error, or localized corruption. The affected region is identifiable from the mismatch positions.

Cascade: a large contiguous region of mismatches beginning at a point in the text. Indicates content change -- the canonical text differs from the marked text, producing a different hash and therefore a different expected sequence from the point of divergence onward.

Category violation: the text's dial settings match a pattern, but the pattern corresponds to a different dial definition table than expected. Indicates text marked under one style system appearing in a context governed by another.

Full match: all observed settings match expected settings. The text has not changed from its marked state and belongs to the expected category.

X. SIGNING PROFILES

A signing profile is the complete configuration required to mark and verify text. It comprises: a dial definition table (provided by the style system), a hash function identifier, and a PRNG algorithm identifier. Two implementations using the same signing profile will produce identical marked text from identical input and identical verification results from identical marked text. Interoperability requires agreement on the signing profile; the signage mechanism itself is profile-agnostic.

A signing profile should be declared in or alongside any artifact that carries intrinsic signage, so that verification is possible without external coordination. The profile declaration is itself metadata that can be stripped by transport -- but because the profile is agreed upon within a governed corpus (and typically stable across the entire corpus), the verifier can reconstruct it from corpus-level configuration rather than per-artifact metadata.

XI. CATEGORY DIFFERENTIATION

Different text categories can use different signing profiles -- different dial definition tables, or the same table with different canonical forms -- producing different patterns that are mechanically distinguishable. The signage mechanism does not define categories; it provides the machinery for categories to be distinguished through pattern. The governing corpus defines which categories exist and which signing profile each uses.

Typical category distinctions: kernel text (governance rules, definitions -- minimal dial space, high stability) vs operational text (content produced under governance -- full dial family) vs draft text (work in progress -- unsigned or unstably signed) vs model output (runtime-generated text -- verifiable against the corpus's expected patterns to detect drift into generic behavior).

XII. WHAT THIS IS NOT

This is not cryptographic signing. There is no keypair, no non-repudiation, no protection against adversarial re-marking. An attacker who knows the signing profile can modify text and re-sign it. The mechanism is drift detection: it identifies accidental change, not deliberate forgery.

This is not a replacement for version control. It complements version control by making integrity signals travel with the text through channels where version metadata does not survive.

This is not dependent on any particular style system. The mechanism consumes dial definitions; it does not produce them. Any style system that can declare variation points as a conforming dial definition table can use intrinsic signage.

The primitive: style variation points, operated by a deterministic script, become transport-surviving verification signals bound to the text's own content.

INTRINSIC SIGNAGE -- EXPLANATORY ADDENDUM // 07A

I. WHAT THIS DOCUMENT IS

This addendum accompanies the Intrinsic Signage specification. The specification defines the mechanism; this document explains what the mechanism is for, what existing approaches it complements, and where it applies. The audience is anyone evaluating whether intrinsic signage solves a problem they have -- not just those implementing it.

II. THE GAP

Images and video now have transport-surviving provenance through standards like C2PA: metadata embedded in the content itself, verifiable without contacting an external authority. Text has no equivalent. Digital signatures verify files, not strings. Watermarking requires vendor cooperation and degrades output. External registries require infrastructure and the text to be registered before verification is possible. All of these fail when text is copied, pasted, emailed, quoted, or inserted into a prompt -- which is how most text actually moves.

Language models have made this gap urgent. LLM-generated text is unsigned by default. The text does not announce what produced it, what constraints governed its production, or whether it has been modified since generation. Organizations generating governed text -- compliance filings, clinical documentation, legal drafts, editorial content -- have no lightweight mechanism to verify that the text they are reviewing is the text the governance system approved. The integrity chain breaks at the first copy-paste.

III. WHAT INTRINSIC SIGNAGE DOES

Intrinsic signage encodes a verification pattern into the stylistic surface of a text -- choices like colon versus period, demonstrative versus repetition, subordination versus coordination -- that preserve meaning while varying form. The pattern is derived from a hash of the text's own content, so changes to the text produce detectable changes in the pattern. The mechanism is a deterministic script: signing uses a language model to produce natural-sounding text that conforms to the pattern; verification requires only pattern matching and hash comparison. No AI is needed to verify. No vendor cooperation is needed to sign. No infrastructure is needed to check.

The mechanism operates at the same trust level as a checksum. It does not prevent tampering; it detects accidental change -- drift, copy error, version confusion, unauthorized edit, model behavior divergence. Most integrity failures are accidental, not adversarial. For adversarial contexts, intrinsic signage composes with cryptographic signing: sign the marked text with a keypair, and the signature covers both the content and the intrinsic pattern.

IV. USE CASES

Regulatory compliance. An organization uses an LLM to draft text for a regulatory filing. The governance system approves a specific version. The approved text is signed using intrinsic signage. At any later point -- after the text has been emailed, pasted into a submission form, or moved through an approval chain -- verification confirms whether the filed text matches the approved text. No external registry required; the text carries its own integrity signal.

Editorial integrity. A publication generates or edits text using LLM assistance under an editorial governance regime. The signed text travels through the publication pipeline -- editors, layout, legal review. Intrinsic signage detects whether the text was altered at any stage, without requiring every participant in the pipeline to use the same tools or maintain access to a shared repository.

Output attribution. An organization operates multiple LLM configurations -- different governance regimes for different purposes. Intrinsic signage uses different signing profiles for different regimes, producing mechanically distinguishable patterns. Text can be verified as belonging to a specific governance category without accessing the system that produced it. This distinguishes governed output from unverified output, and one governance regime from another, by pattern alone.

Contract and legal text. Parties exchange drafted text through email, messaging, and document platforms. Intrinsic signage detects whether the text received matches the text sent, without requiring both parties to use the same document management system or maintain a shared signature registry. The verification is symmetric: either party can check, using the same script, without trusting the other's infrastructure.

V. CONFORMANCE LEVELS

The mechanism supports three levels of verification, each composing on the previous.

Level 1 -- Integrity. Has this text changed since marking? Binary: the pattern matches or it does not. This is the minimum useful deployment. It requires a single signing profile and a verification script.

Level 2 -- Category differentiation. Does this text belong to the category it claims? Different text categories use different signing profiles, producing different patterns. Verification determines not only whether the text has changed but whether it was produced under the expected governance regime. This requires multiple signing profiles within a corpus.

Level 3 -- Provenance typing. Was this text produced under a specific, identified governance regime, distinguishable from ungoverned output? The signing profile itself becomes a provenance marker: text signed under Profile A is mechanically distinguishable from text signed under Profile B, from unsigned text, and from text that mimics a profile without conforming to it. This requires published or exchanged signing profiles between parties.

VI. RELATIONSHIP TO EXISTING STANDARDS

Intrinsic signage does not compete with existing provenance and integrity standards; it covers the medium they do not address.

C2PA provides content provenance for images, video, and audio through embedded metadata. Intrinsic signage provides content provenance for text through embedded style patterns. The two mechanisms are complementary; a document containing both images and text could use C2PA for images and intrinsic signage for text, providing provenance across media types.

Digital signatures (PGP, X.509) verify files and messages within container formats. Intrinsic signage verifies text that has left its container. The two compose: a digitally signed document provides strong provenance within the signing infrastructure; intrinsic signage provides weaker but transport-surviving provenance outside it.

Watermarking (vendor-side LLM watermarks) requires vendor cooperation, is specific to the generating model, and degrades with paraphrase. Intrinsic signage requires no vendor cooperation, is model-independent (any runtime can produce conforming text), and operates on stylistic features that are more robust to minor edits than statistical watermarking. The two mechanisms solve different problems: watermarking identifies the generating model; intrinsic signage verifies conformance to a governance regime.

TERSE STYLE // 08

I. THE PROSE DEFAULT

Prose is the system's first language. Information structure is expressed within sentences -- through subordination, coordination, reference, and punctuation -- rather than externalized into formatting. What makes the style terse is not shortness but density: information per token relative to structural overhead; many tokens do double duty, carrying semantic content while encoding relation (contrast, causality, modification, reference). The underlying bet is dual-audience robustness: human readers and language models both exploit regularities in natural language to recover hierarchy and linkage (clause boundaries, discourse markers, parallel forms, subordination); clearer grammatical structure usually helps both audiences.

II. PUNCTUATION AS CONTROL

Punctuation supplies lightweight control: a colon signals "what follows elaborates what came before"; semicolons mark coordinate alternatives; subordinate conjunctions ("when," "which," "because") embed scope hierarchically inside the sentence without indentation. The result is a paragraph that contains its own map -- relationships expressed as ordinary language relations, not external formatting.

III. DENSITY OVER BREVITY

The distinction matters. A formatted alternative can state the same principles, but it often externalizes structure into layout and fragments flow: the reader must reconstruct why items sit together, while prose encodes that "why" directly. The point is not "no formatting"; it is "structure expressed in-language."

IV. BOUNDARY CONDITIONS

Dense prose is strongest for explanation, synthesis, and causal argument; lists, tables, and stepwise formatting remain correct when the task is comparison, procedure, or random-access lookup. The rule: do not replace relationships with layout when relationships are the point.

V. FORMAL RULES

Cohesion

Use demonstratives (this, that, the X) for backward reference to maintain topic continuity. Employ controlled repetition of key terms rather than synonym variation for critical concepts. Avoid scaffolding phrases (for example, as mentioned above, in other words) when the relationship can be inferred from structure.

Structural encoding

Colons signal elaboration: what follows explains, exemplifies, or enumerates what came before. Semicolons mark coordinate alternatives: parallel clauses that could stand as separate sentences but are structurally joined. Subordinate conjunctions (when, which, because, while) embed scope hierarchically within sentences rather than requiring separate sentences or indentation. Parallel syntax expresses enumeration when items share structural shape; reserve bullets for heterogeneous items.

Density optimization

Target dual function per token: semantic content plus relational signal (contrast, causality, modification, reference). Subordinate boundaries rather than split into new sentences when scope preservation remains clear. Make boundaries explicit (through punctuation or conjunctions) only when working memory load demands it. Optimize for structure-per-token, not minimum token count.

Formatting constraints

Prefer in-language structure (subordination, coordination, punctuation) over external markup (bullets, indentation, headings) for relational content. Use formatted structures (lists, tables, numbered steps) for comparison, procedure, or random-access lookup. Reserve bullets for items that resist parallel construction or require independent random access.

Verification signals

These rules serve double duty as the dial inventory for intrinsic signage. Maintain grammatical parallelism within enumerations. Preserve subordination depth consistency within sections. Keep demonstrative chains unbroken across paragraph boundaries. Use consistent punctuation semantics throughout the text (colons always elaborate, semicolons always coordinate).