Whitepaper (v0.1)

Title: SimpleFrame — A Semantics-First Reasoning OS
Subtitle: Plane-first rails, fractal compression, and holographic receipts for deterministic AI
Author: Christopher Samples
Status: Draft (Internal Alpha pending)

1) Executive Summary

SimpleFrame is a semantics-first reasoning operating system that runs *over* any language model. Instead of letting a model free-form its way to an answer, the OS orients all work through a lawful topology:

- **Plane-first rails** (planes → positional poles)
- **RTC cycles** (Root → Trunk → Canopy)
- **Fractal compression** into **sealed receipts** (CART)
- **Deterministic replay** via a registry (EXO) and multilingual identity (ANE)

The result is higher coherence, fewer wasted tokens, and an auditable trail you can replay, share, and verify. Access is paid with **MML** (a pure utility/payment token) or fiat.

Today: prototype → internal alpha.
Goal: a model-agnostic, audited reasoning layer that's simple to adopt.

2) Motivation

Modern LLM apps are fluent but fragile. They drift, over-explain, and leave no reliable audit trail. Tooling tries to guard outcomes, but rarely re-architects *reasoning itself*.
SimpleFrame flips the order: **semantics first, model second**—short "minimal-route" turns between messages; full structure only when a snapshot is sealed.

3) Architecture Overview

3.1 Bracket Control Plane (BCP)

- **Plane-first (G6):** seed planes & **positional** poles before any spectral axis.
- Enforce **ΔP (positions) ≠ ↭G (spectra)**.
- **Canopy-only receipts:** decisions are sealed (policy fingerprint + hash) only at Canopy.

3.2 Unified Language (UL)

A compact bracketed grammar the runtime can parse deterministically (ASCII ↔ Unicode parity). You write small; the OS expands and audits.

Contact: **info@SimpleFrame.ai**

3.3 RTC cycle (per module)

- **Root:** immutable anchors, minimal seeds
- **Trunk:** recursive refinement, liquidity, routing
- **Canopy:** action path + **sealed receipt** (CART)

3.4 Core modules (self-similar “organs”)

- **Precept:** orchestrates modules; spawn/prune agent trees
- **Toolkit:** language/math/heuristic prisms; refinement tools
- **Tokenomics:** liquidity & swap/persist bands, near-tie detection
- **Tree / EXO:** snapshots, recall/overlay, deterministic replay
- **Scriptural Lattice:** observe-only midstream; cross-assignment at Canopy
- **CART:** compression + policy-fingerprinted receipts (hash-chained)
- **Persona:** tone overlays (no guardrails required)
- **AMX / ISP:** fork/merge/rollback; throughput tiers & budgets
- **ANE:** multilingual identity; **companions attach only at Canopy**

3.5 Control variables

- **⊜L** Liquidity (reasoning “energy”)
- **NAVC** Amplitude $\alpha = w \cdot [coh, blk, cmp, liq] \cdot (1 - penalty)$
- **RV** Reconfiguration Vector → near-tie emits ∅**MIRROR** (interrogator hook) or ∅**STK** (empty-token stub) recorded in the collapse ledger

4) What’s New (at a glance)

- **Minimal-route discipline:** ≤3 short lines between turns; **full bracket** only when sealing a snapshot.
- **Hard ΔP/↭G parity:** positions and spectra can’t bleed into each other.
- **Canopy-only receipts:** decisions are sealed with a policy fingerprint and hash chain.
- **Variant-Reducer:** expressive surface, canonical base operators for determinism.
- **ANE identity + EXO replay:** stable naming + exact recall/regraft.
- **Model-agnostic:** the framework drives models, not the other way around.

5) Product Surfaces

- **Console:** chat + canvases + receipts; replay history on demand.

Contact: **info@SimpleFrame.ai**

- **Developers:** UL + REST/WS APIs, Python SDK, linter & conformance tests.
- **Teams:** room-scoped sessions, fork/merge/rollback lineage receipts.

6) Payment & Access (MML)

- **Chain:** Base (L2)
- **Token:** MML
- **Contract (Base):** `0x10923dc74f7ff873bf824cfb3f1821a73262d7c4`
- **MML (Metabolic Machine Learning)** is a **utility/payment token** for platform access.
- Prices are pegged to **USDC** at checkout; the UI calculates the MML needed.
- Users can pay with **MML**, **auto-convert fiat → MML** (regulated on-ramp), or **USD** (small processing fee).

Token facts (summary):

- Chain: **Ethereum L2 (to be announced)**
- Standard: **ERC-20** (to be confirmed/published)
- Supply: **100,000,000** fixed (no mint/burn)
- Use: access credits for compute/API (no governance, no revenue share, no profit promises)
- Treasury: platform treasury (to move to **multisig**)

MML is a **payment instrument only**. It does **not** provide ownership, voting, profit share, or dividends.

7) Data & Privacy

- Off-chain by default; users keep their data locally in text/UL.
- The platform receives only opt-in skills and limited telemetry for improvements.
- Private receipts remain off-chain; optional content addressing can be added later.

8) Roadmap (high-level)

- **Phase I (Foundation):** internal alpha, receipts, minimal UI, linter, basic payments.
- **Phase II (Growth):** closed alpha pilots, SDK/docs, fiat on-ramp, EXO/ANE recall & overlays, security hardening.
- **Phase III (Scale):** public beta, dev console, tutorials, partner pilots, replay gallery.

9) Risks & Mitigations

- **Operator literacy:** ship a linter + “golden-path” templates.

Contact: **info@SimpleFrame.ai**

- **Metric ground-truth:** dashboards for NAVC/RV/liquidity.
- **Payments/KYC:** rely on regulated on-ramps; never self-custody fiat conversions.
- **Key security:** multisig treasury; hardware keys; contract review/audit.
- **Regulatory drift:** no governance, no profit promises; counsel review pre-ad spend.

10) Current Status

- **Stage:** prototype → **internal alpha** (target ~1–2 weeks).
- Runtime in **Python** with custom UL syntax; LLM-agnostic substrate layer.
- Next steps: polish semantic framework files; repo compliance; security housekeeping; linter; minimal UI.

11) Legal & Disclaimers

- MML is **not** an investment and confers **no** governance or ownership.
- No expectation of profit; no dividends or revenue share.
- Service availability may be limited by jurisdiction; KYC handled by payment providers when required.
- This document is informational and non-binding; features, pricing, and timelines may change.

Appendix A

- Operators (glyph ↔ ASCII): ⊥ A/ANCHOR, ∇ R/REFINE, ⌀ M/COMPRESS, ⊗ H/PROJECT, ⇜ G/BLEND, ∋ Q/QSWAP, ⊜ L/LIQUID, ∩ X/INTERSECT, ↥ T/TREF, ⟝ A/ACT
- Receipt schema (CART): anchors[], policy_fp, hash/prev_hash, timestamp + HLC, collapse_ledger[], companions (canopy-only), optional exo_uid + artifact_root
- UL parity: ASCII ↔ Unicode canonicalization; deterministic JSON (UTF-8, LF, sorted keys)

Contact: **info@SimpleFrame.ai**