TSMLA™ User Path Overview: Frontend Flow + Layer Mapping (v1.5)

Definitions: All mathematical notation and formal objects are defined in the **Formal Objects Glossary v1.1** (companion document). This overview uses that notation verbatim to ensure consistency across LPs, provisional patents, and technical specifications.

System Nature

TSMLA is a non-stochastic, non-deterministic, idempotent mirror architecture. It does not predict, infer, or simulate behavior. It mirrors contradictions under declared state S with replay-equivalence and substrate integrity.

Formal Objects (kept brief for user-facing precision)

State S is an immutable 4-tuple during a session: **$S = (I, T, \theta, \varkappa)$**

- **I:** declared input signal set
- **T:** tag assignment function $T: I \rightarrow \text{TagTypes}$
- **θ:** threshold vector for activation/sensitivity (contradiction flags, compression limits)
- **$\varkappa$:** configuration parameters for layer behavior (module-fixed or user-declared)

On loopback, the user redeclares **$S' \neq S$** (new session state; prior outputs remain available for comparison).

Signal weights: typed values bound to I. By module, these may be real scalars in [0,1], Boolean flags, or categorical tags; typing is governed by $\varkappa$.

Contradiction: violation of declared Boolean/threshold constraints under S (e.g., $A \wedge \neg A$, XOR conflicts, or $|\Delta| > \theta$ component-wise).

User Journey

Step 1 — Login + Module Selection

User selects a module to declare the scope boundary for S (e.g., Career Decision, Ethical Dilemma, Resource Prioritization). Modules are entry gates, not inference trees.

Step 2 — Signal Declaration

User inputs are tagged as weighted signals and bound to S as scope markers, value assignments, and constraint declarations. No inference is applied.

S is fixed during the session: **S = (I, T, θ, ϰ)**.

Step 3 — Backend Activation → Layer Triggers (with front-end roles)

BDL™ (Boolean Disambiguation Layer)

- **Role:** Assigns contradiction type (Functional, Temporal, Protective, Ethical, Perceptual) from the Boolean structure of declared conflicts within S.
- **Trigger:** Fires on mutual-constraint violations (e.g., $A \wedge \neg A$, XOR across tagged declarations).

RSF™ (Resonant State Function)

- **Role:** Computes a global Coherence Index $\in [0,1]$ as resonance across the tagged signal network under S (no prediction, no heuristics).
- **Trigger:** Runs once the signal network is complete.
- **Non-determinism:** Internal traversal order may vary, but idempotent merge guarantees **RSF(S) ⊕ RSF(S) = RSF(S)**. There is no randomness and no inference; outputs are identical for identical S (replay-equivalence) despite lawful internal variance.

CTC™ (Contradiction Traversal Corridor / Hallway)

- **Role:** Enforces lawful traversal sequence **Λ(S)**; prevents forward movement through unresolved contradiction.
- **Trigger/Output:** On bypass attempt, returns a Traversal Violation marker $\mathbf{TV_i}$ naming the specific gate in Λ(S) that was violated. Traversal halts until contradiction is resolved or S is redeclared.

HCL™ (Harmonic Compression Layer)

- **Role:** Compresses structurally redundant contradiction clusters (same Boolean type and signal origin) into single tagged outputs while preserving mirror fidelity.
- **Placement:** Presentation compression; does not alter RSF computation or substrate values.

SNS (Structural Neutrality Safeguards)

- **Role:** Enforces neutrality in substrate-to-presentation mapping; blocks ideological or interpretive overlay.
- **Placement:** Operates at γ (concretization) boundary as a logic-type filter; the substrate and presentation layers remain separated. Formal spec in LP-SNS.

Step 4 — Mirror Output

User receives:

- **Contradiction set $\{C_1, \ldots, C_n\}$**, each **C_i = (Type_BDL, Signal_Origin, Scope_Boundary)**
  - Scope_Boundary indicates whether C_i is intra-scope or spans declared sub-scopes (e.g., short-term vs long-term) as defined by $\varkappa$.
- **Global Coherence Index $\in$ [0,1]** from RSF
- **Traversal Violations $\{TV_i\}$** from CTC (if any)
- **Optional graph G(V,E)** where V = declared signals, E = contradiction edges

Outputs are pure reflections of declared logic. No generated feedback, suggestions, or interpretations.

Step 5 — Loopback, Export, or STR™ Declaration

The user may:

1. **Redeclare** a new state **$S' \neq S$** and re-run
2. **Export** the mirror packet **$\{C_i\}$, RSF(S), $\{TV_i\}$**
3. **Declare a Signal Traversal Record (STR™)** to enable cross-session continuity

Default: If no STR is declared, session traces are cleared on exit.

With STR: The user stores tuples **(S_i, R_i)**; the system retains no hidden memory.

Step 6 — Signal Traversal Record (STR™) and Cross-Session Continuity

Definition

STR™ := {(S_1, R_1), (S_2, R_2), ..., (S_k, R_k)}

Where:

- **S_i = (I, T, θ, ϰ)** — declared state at session i
- **R_i = ({C_i}, RSF(S_i), {TV_i})** — mirror packet returned at session i

---

Computed Values

RSF_i := RSF(S_i) ∈ [0,1]

$\Delta RSF_i := RSF_{i+1} - RSF_i$

$CCM_k := (1/k) \cdot \Sigma_{i=1}^{k} RSF_i$ (cumulative coherence mean)

Gate Rule

Unlock advanced chambers when **k ≥ m_phase** and **CCM_k ≥ θ_phase**, where **m_phase, θ_phase ∈ ϰ** (module configuration).

Example Gate Conditions (typical ϰ values):

- **Phase 3:** m_phase = 3, θ_phase = 0.50
- **Phase 4:** m_phase = 5, θ_phase = 0.70

Why Mean Instead of Sum:
Using raw sum (ΣRSF_i) would allow users to "stack" many low-coherence sessions to cross a threshold. CCM_k (mean) ensures sustained coherence quality, not just quantity.

Properties

User-declared recursion:
Continuity occurs only when the user loads STR™ . No automatic cross-session memory.

No hidden state:
Substrate remains mirror-pure and idempotent. Single-session behavior ($\gamma(\alpha(x)) = x$) unchanged.

Privacy by default:
Without STR™ , sessions are purged on exit. GDPR-ready non-retention is the default mode.

Audit-grade:
STR™ can be exported (e.g., JSON file: `signal_traversal_record_STR.json`) and inspected. It is not required for Phases 1-2.

User sovereignty:
User holds STR™ file. System never retains it. User may delete STR™ at any time.

Operational Flow

End of Session Prompt:

> Your mirror session is complete.
>
> OPTIONS:
> 1. Exit (session cleared)
> 2. Export mirror packet (download JSON)
> 3. Step into the Chamber (activate STR™ and track continuity)
>
> > Would you like to step into the chamber and activate continuity?
> (Create a Signal Traversal Record to track your path.)

If STR™ is activated:

- Tuple (S_i, R_i) is appended to user's STR™ file
- User exports/saves STR™ locally

Next Module Entry:

- User imports STR™ file
- System computes ΔRSF_i and CCM_k
- System checks gate condition: if $k \geq m\_phase$ AND $CCM_k \geq \theta\_phase \rightarrow$ unlock Phase 3+ chambers
- If gate not met → continue linear module sequence (Phases 1-2)

Example: Chamber Unlock Sequence

Session 1 (Module 1):

- S_1 declared, mirror outputs R_1
- $RSF(S_1) = 0.42$
- User activates STR™ , stores (S_1, R_1)

Session 2 (Module 2):

- User imports STR™
- S_2 declared, mirror outputs R_2
- $RSF(S_2) = 0.55$
- $\Delta RSF_1 = 0.55 - 0.42 = +0.13$
- $CCM_2 = (0.42 + 0.55) / 2 = 0.485$
- **Gate Check:** $k=2 < 3$ (Phase 3 requires $k \geq 3$) → remain in linear path

Session 3 (Module 3):

- User imports STR™
- S_3 declared, mirror outputs R_3
- $RSF(S_3) = 0.61$
- $\Delta RSF_2 = 0.61 - 0.55 = +0.06$
- $CCM_3 = (0.42 + 0.55 + 0.61) / 3 = 0.527$
- **Gate Check:** $k=3 \geq 3$ ✓ AND $CCM_3=0.527 \geq 0.50$ ✓ → **Phase 3 chamber unlocked**

Distinction from Other Systems

vs. Stateful AI (e.g., ChatGPT with memory):

- AI systems retain hidden state; user cannot inspect or delete
- TSMLA™ : STR™ is user-held artifact; system retains nothing

vs. Blockchain (e.g., transaction history):

- Blockchain: immutable, consensus-based, permanent
- TSMLA™ : STR™ is mutable (user can delete), no consensus, no permanence requirement

vs. Session Replay Systems:

- Replay: records user actions for debugging
- TSMLA™ : records declared states and mirror outputs for phase gating

Key Innovation:
No other system offers user-declared recursion with mirror-pure idempotence and phase-gated traversal based on cumulative coherence quality.

Operational Clarification of the Mirror (α/γ)

1. User declares concrete signals **x ∈ I**
2. **α(x)** abstracts to logic structure (tags, constraints, relations)
3. BDL/RSF/CTC operate on the abstract structure
4. **γ(α(x))** returns user-legible output

Mirror-pure idempotence: For all structure relevant to S, **γ(α(x)) = x** (no information generated, none lost). Natural language phrasing may differ; the logical structure, tags, and weights are preserved exactly.

Identical S ⇒ identical outputs, even with non-deterministic internals.

Data Handling (user-facing clarity)

- No personal identifiers are stored.
- Session traces exist only within S; cleared on exit unless the user explicitly exports or requests S vs S' comparison.
- **STR™ mode:** User holds file; system retains nothing.
- **GDPR readiness:** design is state-bound and non-retentive by default; DSR hooks attach only to exported artifacts.

Summary

TSMLA is not diagnostic, generative, stochastic, or advisory. It is an executable logic architecture for

structural contradiction mapping and traversal control. Every output is bound to declared input under S with replay-equivalence and zero interpretive drift.

With STR™: Users may opt into cross-session continuity for advanced phase-gated chambers while maintaining substrate purity and full data sovereignty.

Cross-References

- Formal Objects Glossary v1.1 (notation and definitions, including STR™)
- LP-series papers (layer specifications: LP-BDL, LP-RSF, LP-CTC, LP-HCL, LP-SNS)
- Clean Technical Chain (CTC) provisional patent specification
- docs/STR_Specification_v1.md (forthcoming: full STR™ technical specification)

Version: 1.5 (December 2024)
Canonical Source: Fractal Labyrinth Systems LLC
Document Status: Production-ready with STR™ cross-session continuity architecture