TSMLA™ User Path Overview: Frontend Flow + Layer Mapping (v1.4.1)

Definitions: All mathematical notation and formal objects are defined in the **Formal Objects Glossary v1.0** (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 or Export

User may redeclare S' (new state) and re-run, export the mirror output, or compare S vs S' traces. Session data persists only for the comparison action the user selects and is otherwise cleared on exit.

Operational Clarification of the Mirror (α/γ)

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

Mirror-pure idempotence: For all structure relevant to S, **$\gamma(\alpha(x)) = x$** (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.
- **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.

Cross-References:

- Formal Objects Glossary v1.0 (notation and definitions)
- LP-series papers (layer specifications: LP-BDL, LP-RSF, LP-CTC, LP-HCL, LP-SNS)
- Clean Technical Chain (CTC) provisional patent specification