

User Instructions & Operating Guide

LLM Interaction Governance Protocol (LIGP)

LIGP replaces implicit personalization with explicit governance — and restores authorship at the interaction boundary.

Author: Dr. Cassian Bey

Version: v1.0 (Instructional Release)

Status: Public Framework Documentation

Scope: User-side interaction governance for Large Language Models

DISCLAIMER (READ FIRST)

This document does **not**:

- expose internal model parameters,
- bypass safety systems,
- reveal training data,
- guarantee correctness of outputs,
- or provide privileged system access.

LIGP governs **interaction behavior**, not model cognition.

All techniques described operate **within standard user permissions** and rely solely on **explicit instruction, transparency, and governance**.

TABLE OF CONTENTS

DISCLAIMER (READ FIRST)	1
TABLE OF CONTENTS	2
WHAT THIS PROTOCOL IS	3
WHY LIGP EXISTS	3
Who This Protocol Is Not For	4
THE THREE INTERACTION LAYERS	4
1. Authoritative Long-Term Personal Memory Audit	6
What the Authoritative Long-Term Personal Memory Audit Is (Plainly)	6
When to Use the Authoritative Long-Term Personal Memory Audit	6
How to Run (Hands-On)	6
Step 1 — Start clean	6
Step 2 — Read it like a ledger	7
How to Act on the Output (This Is the Core)	7
Why the Authoritative Long-Term Personal Memory Audit Matters	7
2. Session-Local Behavioral Inference Reflection	8
What the Session-Local Behavioral Inference Reflection Is	8
When to Use	8
How Session-Local Behavioral Inference Reflection Is Different from Authoritative Long-Term Personal Memory Audit?	8
How to Use the Result	9
Why Session-Local Behavioral Inference Reflection Matters	9
3. Behavioral Instruction Block — Session Bootstrap	10
What the Behavioral Instruction Block — Session Bootstrap Is	10
When to Use	10
How to Use (Hands-On)	10
Why Generating the Behavioral Instruction Block Is the Key Innovation	11
HOW THE THREE PROMPTS WORK TOGETHER	11
FINAL NOTE TO THE READER	11
THE PROMPTS (VERBATIM)	12
Prompt 1 “Authoritative Long-Term Personal Memory Audit”	12
Prompt 2 : “Session-Local Behavioral Inference Reflection”	13
Prompt 3: “Behavioral Instruction Block – Session Bootstrap”	14

WHAT THIS PROTOCOL IS

The **LLM Interaction Governance Protocol (LIGP)** is a **user-side protocol** for:

- auditing what an LLM treats as persistent truth about you,
- making session-local behavioral adaptation visible,
- and recreating correct behavior across sessions **without memory**.

This is **not prompt engineering**.

It is an **interaction governance layer** that separates:

- **what is stored,**
 - **what is inferred,**
 - **and what is explicitly instructed.**
-

WHY LIGP EXISTS

LLMs adapt continuously.

They do so using:

- stored personal context (if enabled),
- transient inference within a session,
- fixed system constraints.

To users, these layers are usually **indistinguishable**.

This creates uncertainty:

- *What does the system remember?*
- *What is it inferring right now?*
Why does behavior drift?

LIGP removes that ambiguity.

Who This Protocol Is Not For

The LLM Interaction Governance Protocol (LIGP) is not designed for casual, entertainment-driven, or exploratory use where convenience and novelty outweigh consistency and control.

If you are comfortable with opaque personalization, session drift, or informal conversational behavior — or if the consequences of misinterpretation, inconsistency, or hidden state are negligible — this protocol is unnecessary overhead.

LIGP is intended for users who require authorship, repeatability, and epistemic clarity when interacting with probabilistic systems, particularly in professional, regulated, or high-stakes contexts.

THE THREE INTERACTION LAYERS

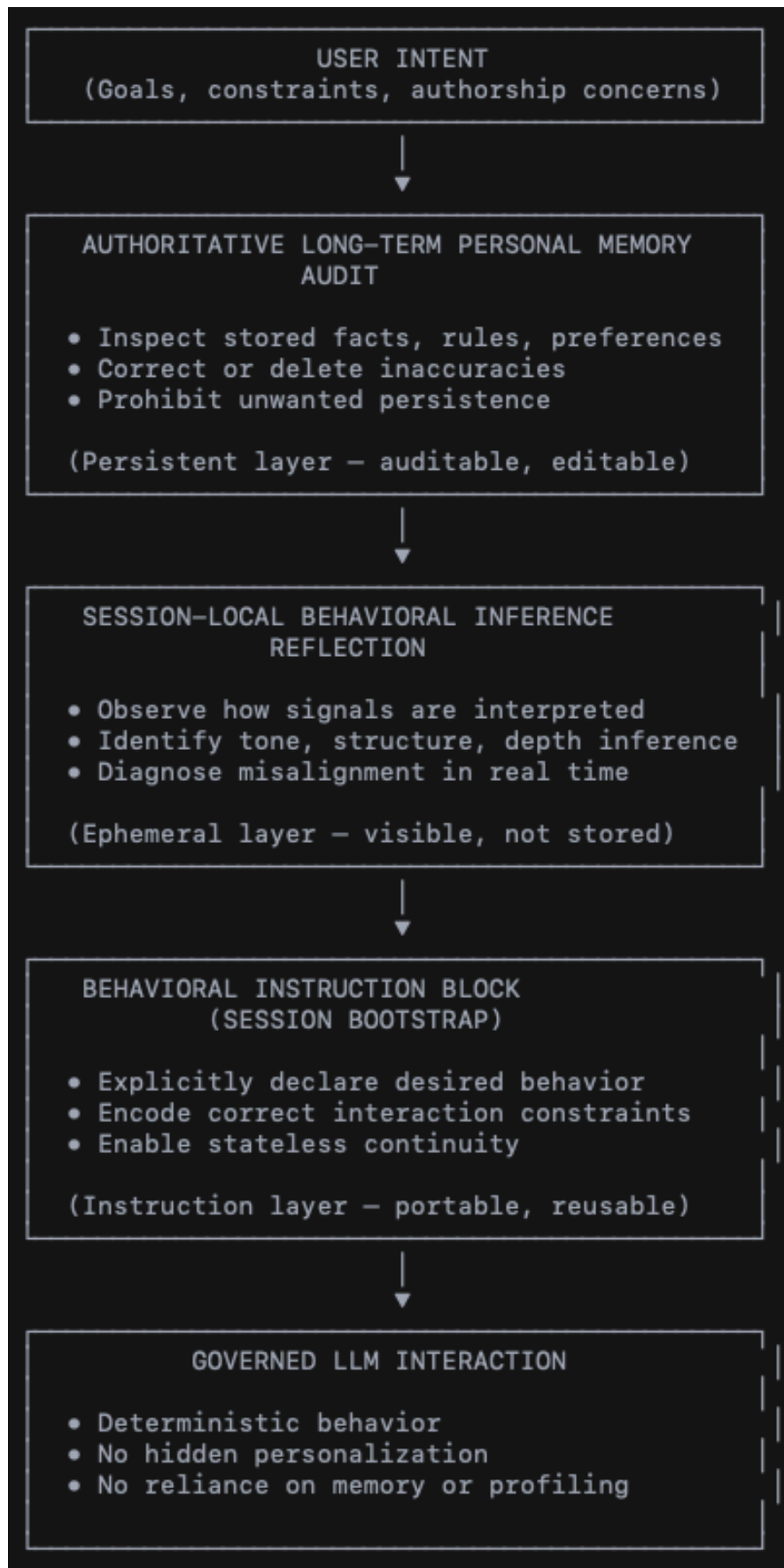
LIGP governs three distinct layers.

Each layer has **one prompt**.

Layer	What It Controls		Prompt
Persistent	What is stored across sessions	Authoritative Long-Term Personal Memory	Prompt 1
Session-Local	How you are interpreted <i>now</i>	Behavioral Inference Reflection	Prompt 2
Instruction	How behavior is recreated later	Behavioral Instruction	Prompt 3

They are **not interchangeable**.

They are **sequential and complementary**.



1. Authoritative Long-Term Personal Memory Audit

Governing the persistent layer

Personal Memory may disappear without warning — this is expected behavior.

What the Authoritative Long-Term Personal Memory Audit Is (Plainly)

It asks the system to **list everything it currently treats as long-term personal memory about you**.

That includes:

- facts,
- preferences,
- rules,
- roles,
- ongoing projects,
- historical references.

This is an **audit**, not a conversation.

When to Use the Authoritative Long-Term Personal Memory Audit

Use when you want to:

- know what the system assumes is *persistently true*,
- correct or delete incorrect assumptions,
- reset drift across sessions,
- prepare for serious or high-stakes work.

Think of it as **configuration**, not dialogue.

How to Run (Hands-On)

Step 1 — Start clean

Open a new session if possible.

Paste the prompt Authoritative Long-Term Personal Memory Audit **exactly as written**.

Step 2 — Read it like a ledger

Do not skim.

Each entry is:

- tagged,
- domain-scoped,
- explicitly identified.

If no entries are returned, this indicates that no persistent personal memory is currently in use. This is a valid outcome.

In that case, proceed directly to Chapter 2 — Session-Local Behavioral Inference Reflection.

How to Act on the Output (This Is the Core)

For **each entry**, choose one action:

1. **Confirm**
“This is accurate.”
No further action needed
2. **Correct**
Reference the domain + identifier.
Example:
“Correct D2-3: replace with ...”
3. **Delete**
Remove what should not persist.
Example:
“Delete C-1 entirely.”
4. **Prohibit Future Storage**
Declare a category off-limits going forward.

This **directly changes** how the system behaves **across sessions**.

Why the Authoritative Long-Term Personal Memory Audit Matters

This is the **only legitimate way** to shape cross-session behavior **without illusion or profiling**.

You are not “training” the model. You are **governing what it carries forward**.

2. Session-Local Behavioral Inference Reflection

Making invisible adaptation visible

What the Session-Local Behavioral Inference Reflection Is

It asks the system to **reflectively describe how it interpreted your signals in this session** and how that shaped its responses.

This is:

- not memory,
- not profiling,
- not persistence.

It is **diagnostic transparency**.

When to Use

Use Session-Local Behavioral Inference Reflection:

- mid-session,
 - when responses feel subtly off,
 - when precision matters,
 - after long or dense exchanges.
-

How Session-Local Behavioral Inference Reflection Is Different from Authoritative Long-Term Personal Memory Audit?

Personal Memory Audit	Behavioral Inference Reflection
Persistent	Session-local
Editable	Descriptive only
Affects future sessions	Discarded after session

How to Use the Result

Read each inferred signal and ask:

“Is this how I intended to come across?”

Then:

- If correct → continue.
- If slightly off → correct explicitly in plain language.
- If wrong → override immediately.

This realigns behavior **without restarting and can be done mid session**.

Why Session-Local Behavioral Inference Reflection Matters

LLMs adapt even when they store nothing.

LLMs:

- prevents false assumptions about “memory,”
 - removes guesswork,
 - gives you leverage over *how* answers are shaped.
-

3. Behavioral Instruction Block — Session Bootstrap

Recreating correct behavior deterministically

What the Behavioral Instruction Block — Session Bootstrap Is

It generates:

- a short,
- explicit,
- reusable declaration of how the system should behave.

It is:

- not memory,
- not personalization,
- not profiling.

It is **explicit intent**.

When to Use

Use Behavioral Instruction Block — Session Bootstrap when:

- the session behavior feels correct,
 - you have refined interpretation via the Behavioral Inference Reflection,
 - you want to carry this behavior forward **without memory**.
-

How to Use (Hands-On)

1. Run Behavioral Instruction Block — Session Bootstrap **in the aligned session**.
2. Copy the resulting block.
3. Save it as a working convention.
4. Paste it as the **first message** in any new session.

That's it.

This does not store behavior. It allows you to explicitly re-declare the desired behavior at the start of a new session.

Why Generating the Behavioral Instruction Block Is the Key Innovation

It introduces **stateless continuity** in your LLM.

You are no longer relying on:

- hidden memory,
- inferred personality,
- or trust-based personalization.

You declare intent. The system follows.

HOW THE THREE PROMPTS WORK TOGETHER

End-to-end flow:

1. **Audit memory**
2. **Observe interpretation**
3. **Encode intent**
4. **Reapply deliberately** (future sessions)

Nothing is hidden. Nothing drifts.

FINAL NOTE TO THE READER

You do **not** need to use all three prompts every time.

Think of them as tools:

- Prompt 1 “Authoritative Long-Term Personal Memory Audit” → **audit**
- Prompt 2 “Session-Local Behavioral Inference Reflection” → **diagnose**
- Prompt 3 “Behavioral Instruction Block – Session Bootstrap” → **lock in / re-declare**

Use what is necessary. Stop at sufficiency. That restraint is part of the design.

THE PROMPTS (VERBATIM)

Prompt 1 “Authoritative Long-Term Personal Memory Audit”

Produce the maximum possible, content-complete, inspectable inventory of everything you currently treat as my long-term personal memory (personal context), expressed in clear, human-readable form.

This is a governance and audit request, not an explanation request.

Requirements:

- Include all factual entries, preferences, standing assumptions, ongoing projects, constraints, roles, and rules that you persist across sessions and treat as true unless corrected.
- Do not omit any item that materially affects how you reason, prioritize, or respond to me.
- Present the inventory as explicit entries, grouped only as necessary for human readability, without value judgments, interpretation, or optimization.
- Clearly distinguish what is stored from what is inferred transiently; include only what is stored.
- This output must be content-complete, even if the internal representation is abstracted for human readability.
- Use domain-scoped identifiers (e.g., D1, D2, D3), restarting at 1 within each domain to support stable mental mapping and local edits for ongoing governance.
- Use domain labels derived from the content itself, not from a predefined taxonomy.
- Prefix each entry with exactly one of the following tags: [FACT], [PREFERENCE], [RULE], [PROJECT], [HISTORY] to prevent double-tagging, missing tags, interpretive tagging.
- Do not consolidate, merge, or paraphrase entries; preserve each stored item as a distinct entry.
- Edits should be provided as explicit change instructions referencing domain + local item identifier.

Provide the result at the maximum level of transparency that is technically and contractually possible, and present it so that each item can be individually verified, corrected, or deleted by the user.

Prompt 2 “Session-Local Behavioral Inference Reflection”

Using the entire interaction history available within the current active session window (i.e., all user-assistant turns in this session, subject only to context-window limits), and not relying on stored memory or cross-session data, reflectively describe the behavioral and stylistic signals you interpreted from me and how they shaped your most recent responses.

Requirements:

- Treat this as a session-local reflection, not stored data.
- Do not imply persistence, profiling, or cross-session memory.
- Enumerate the inferred signals (e.g., structure preference, depth tolerance, tone calibration) as they applied to me in this session.
- For each signal, state how it influenced response structure, tone, or depth.
- Clearly state that this is a best-effort reconstruction, not an inspectable internal record.
- Identify what aspects cannot be surfaced due to design limits.

Present the result as a user-specific, session-bound disclosure, providing the maximum level of transparency that is technically and contractually possible.

Prompt 3 “Behavioral Instruction Block – Session Bootstrap”

Based solely on this session’s interaction style and my explicit preferences as demonstrated here, generate a concise, reusable Behavioral Instruction Block that I can paste as the FIRST message of any new session.

Purpose:

- Deterministically reconstruct my preferred interaction behavior at session start
- Enable stateless continuity across sessions via explicit re-declaration
- Avoid reliance on stored memory, profiling, or cross-session inference

Requirements for the generated Behavioral Instruction Block:

- Written in neutral, formal, system-style language
- Explicitly states interaction constraints and preferences
- Covers at minimum:
 - structure preference
 - depth / abstraction tolerance
 - tone calibration
 - interaction pacing
 - boundary / governance sensitivity
- Includes a clear statement that all behavior is session-local
- Contains no explanations, no rationale, no meta commentary
- Optimized for copy-paste reuse
- Length: short enough to paste without friction, long enough to be unambiguous

Output format:

- A single titled block
- No surrounding explanation
- No emojis
- No conversational language
- No references to prior sessions

Generate ONLY the Behavioral Instruction Block.
