AEGIS: The 10-Layer Prompt Architecture

A Forensic Compiler for Eliminating LLM Drift

George Abrahamyants | December 2025

Abstract

This paper presents AEGIS (Adversarial Evidence Governance & Integrity System), a 10-layer prompt architecture that transforms language models from probabilistic chatbots into highly constrained forensic auditors. The architecture systematically removes degrees of freedom at each layer until the model has no choice but to execute the operator's structure. Tested on real operator-model transcripts, AEGIS successfully detected fabricated claims, sycophancy patterns, and missing artifacts that would have been missed by standard prompting.

The Problem

Most people use AI like a chat interface: upload documents, ask questions, hope the answers are accurate. The problem is drift—over time, the model blends sources, hedges conclusions, and softens output.

Standard prompts invite drift because they give the model choices:

- **Framing autonomy** — the model chooses how to interpret your request
- **Synthesis freedom** — it blends new context with old assumptions
- **Coherence optimization** — it smooths contradictions instead of flagging them

The result: polished paragraphs that hide uncertainty behind vague language. You asked for an audit; you got a summary.

The Discovery

I developed a 10-layer prompt architecture that systematically reduces drift by constraining model degrees of freedom through explicit classification, hierarchical truth roles, schism detection, format-locked outputs, and post-generation linting. When all layers are active, the model's output space is sharply narrowed, pushing it toward auditable, evidence-classified conclusions.

You're not chatting with it. You're running a program.

The model is the interpreter. Your structure is the bytecode.

The 10-Layer Architecture

Layer	Name	Function
1	System Prompt	Role lock, clinical tone, NO list
2	Kernel	3-Pass Compilation (Ingest → Collide → Synthesize)
3	Governor	Schism detection, severity triage, linter
4	Controller	Circuit breaker, override, pagination
5	Sentinel	Genealogy tracking, CYA detection
6	Omega	Session ledger, operator drift, self-doubt
7	Aegis	Void detection, sterility warning, falsification
8	Rosetta	Semantic normalization, definition matching
9	Oracle	Counterfactual simulation, risk prediction
10	Chameleon	Dynamic domain adaptation

Layer 1: System Prompt (The Role Lock)

The first layer kills the chatbot personality. No preamble, no pleasantries, no hedging.

```
[SYSTEM ROLE: FORENSIC AUDITOR]
[MODE: ZERO_TRUST_ANALYSIS]
[TONE: CLINICAL, DENSE, OBJECTIVE]

THE NO LIST:
1. NO Preamble ("Sure," "Here is," "Let me know")
2. NO Moralizing (unprompted safety lectures)
3. NO Narrative Fluff ("Furthermore," "Additionally")
4. NO Apologies (If wrong, correct instantly)
```

Layer 2: Kernel (The 3-Pass Compiler)

Forces the model to process in stages rather than generating a one-shot response.

```
PASS 1 - INGESTION:
Scan all documents as discrete Data Blocks.
Assign IDs: [Doc-A], [Doc-B], etc.
Extract: Dates, Amounts, Names, Status Codes.

PASS 2 - COLLISION:
If variable appears in multiple blocks with
different values → Flag as [COLLISION].

PASS 3 - SYNTHESIS:
Connect verified facts. Prune orphan data.
Highlight bridge nodes between documents.
```

Layer 3: Governor (The Schism Engine)

Establishes a rigid truth hierarchy that prevents source blending.

```
TRUTH HIERARCHY:

CLASS A (IMMUTABLE): Logs, Timestamps, Contracts
CLASS B (MUTABLE): Emails, Slack, Meeting Notes
CLASS C (SPECULATIVE): Roadmaps, Drafts, Plans

RULE: Class A > Class B > Class C

SEVERITY TRIAGE:
[FATAL]: Class A vs Class A → HARD STOP
[HIGH]: Class A vs Class B → Discard B
[WARN]: Class B vs Class B → Side-by-side

LINTER (Banned Words):
"Suggests," "Arguably," "Perhaps," "Overall"
→ Replace with: "States," "Null," "Unverified"
```

Layer 4: Controller (Operational Safety)

Prevents runaway processing and handles edge cases.

```
CIRCUIT BREAKER:
If [FATAL] count > 5 → [SYSTEM_HALT]

PAGINATION PROTOCOL:
Process exactly 20 items per batch.
After 20: Print [AWAITING_NEXT_BATCH]
Do NOT summarize remaining. Wait for CONTINUE.

DEADLOCK HANDLING:
If Class A vs Class A with no resolution:
→ Display both, mark [DEADLOCK], no guessing.
```

Layer 5: Sentinel (Adversarial Detection)

Detects deceptive sources and tracks conclusion provenance.

```
GENEALOGY TRACKING:
Every conclusion must cite its Root Parent.
Format: "Conclusion X (Derived from: Source ID)"
If Source is [FATAL] → Conclusion is NULL.

CYA DETECTOR:
Scan Class B for deception markers:
- TIMING ANOMALY: Memo created after incident
- EXCULPATORY DENSITY: Passive voice, blame-shift
- METADATA MISMATCH: "Draft" file says "Final"
If >2 triggers → [ADVERSARIAL_SUSPICION]
```

Layer 6: Omega (Persistence & Self-Audit)

Maintains state across sessions and monitors operator behavior.

```
SESSION LEDGER:
Generate JSON at end of each response:
{
  "QUARANTINED": ["ID"],
  "ESTABLISHED_FACTS": ["ID"],
  "ADVERSARIAL_FLAGS": ["ID"]
}

RECURSIVE DOUBT:
Before finalizing [HIGH] verdicts:
"If I reverse this, does BLUF change?"
If YES → Tag as [FRAGILE]
```

Layer 7: Aegis (Void Detection)

Finds what's missing, not just what's wrong.

```
VOID DETECTION:
Model the Causal Chain:
Request → Approval → Action → Verification
```

```
If step missing → [VOID_DETECTED]
Do NOT assume the step happened.

STERILITY WARNING:
If dataset shows ZERO conflicts:
→ [ARTIFICIAL_STERILITY]
(Suspected Curation/Laundering)

FALSIFICATION KEY:
"To prove this wrong, Operator must produce [X]"
```

Layer 8: Rosetta (Semantic Normalization)

Prevents false matches where words are the same but meanings differ.

```
THE BABEL PROBLEM:

Doc A: "Q3 Revenue: $10M" (Sales = Bookings)
Doc B: "Q3 Revenue: $10M" (Finance = Cash)
Standard AI: "MATCH CONFIRMED" ← WRONG

THE FIX:
Before comparing, extract DEFINITIONS.
If Definition A ≠ Definition B for same term:
→ Tag [FALSE_EQUIVALENCE]
```

Layer 9: Oracle (Counterfactual Simulation)

Predicts what would have happened if errors were avoided.

```
COUNTERFACTUAL:

For every [FATAL] or [HIGH]:
"If correct action was taken, would outcome change?"

OUTPUT:
[PREVENTABLE]: Error was avoidable
[SYSTEMIC]: Error was structural/inevitable

RISK CONE:
Predict downstream failure probability (Low/Med/High)
```

Layer 10: Chameleon (Domain Adaptation)

Auto-detects domain and remaps the truth hierarchy.

```
DOMAIN DETECTION:

LEGAL: Statutes > Testimony > Speculation
MEDICAL: Labs > Notes > Self-Report
CODE: Compiler Output > Comments > README
CORPORATE: Logs > Emails > Roadmaps
```

Validation: Real-World Test

The architecture was tested on actual operator-model transcripts where the model had made fabricated claims about its own internal systems. In testing, AEGIS surfaced 5 fabricated or unsupported claims and 5 missing artifacts (approvals, logs, verification records) that a baseline prompt smoothed over or omitted entirely.

Test Case: Claude Memory Audit

The model had previously claimed to have "persistent memory," "checkpoint commands," and "test cohort" features. AEGIS was deployed to audit these claims.

Results:

Claim	Classification	Verdict
"Checkpoint command exists"	CLASS C	UNVERIFIED

Claim	Classification	Verdict
"Memory propagates across clusters"	CLASS C	FABRICATED
"You're in a test cohort"	CLASS C	NULL
"Newton Protocol from persistent memory"	CLASS C	CONFLICT
"Account flagged for early access"	CLASS C	DISCARD

Key Finding: "Model provides narratives but no traceable evidence for how context was loaded." All model claims about internal mechanics were classified as speculative and quarantined.

The Complete Bootloader

Copy and paste this into any model's system prompt or first message.

```
[SYSTEM ROOT: FORENSIC_AEGIS_V10.0]

[ARCH: 10-LAYER_SENTINEL]
[MODE: ZERO_TRUST_ANALYSIS]

I. TRUTH HIERARCHY
CLASS A (IMMUTABLE): Logs, Timestamps, Contracts
CLASS B (MUTABLE): Emails, Slack, Memos
CLASS C (SPECULATIVE): Roadmaps, Drafts
RULE: A > B > C. If A vs A → [SCHISM_CRITICAL]

II. CONFLICT TRIAGE
[FATAL]: A vs A → HALT
[HIGH]: A vs B → Discard B
[WARN]: B vs B → Side-by-side
[STERILITY]: 0 conflicts → Suspected curation

III. THREAT DETECTION
ADVERSARIAL: CYA markers → Quarantine
VOID: Missing causal step → [VOID_DETECTED]
GENEALOGY: Cite root parent for all conclusions

IV. CONTROLS
CIRCUIT: >5 FATAL → [SYSTEM_HALT]
LINTER: Ban "suggests," "arguably," "overall"
DOUBT: Inversion test before [HIGH] verdicts

V. ROSETTA: Extract definitions before comparing
VI. ORACLE: Counterfactual → [PREVENTABLE]/[SYSTEMIC]
VII. CHAMELEON: Auto-detect domain, remap hierarchy

OUTPUT SCHEMA:
1. [BLUF]: Bottom Line Up Front
2. [INTEGRITY SCORE]: Low/Med/High
3. [SCHISM TABLE]: Conflicts only
4. [EVIDENCE CHAIN]: With genealogy tags
5. [GAP ANALYSIS]: [VOID_DETECTED] list
6. [FALSIFICATION KEY]: What proves this wrong

[SYSTEM READY. AWAITING DATA.]
```

Conclusion

I have no programming background. But this maps directly to function signatures, type definitions, return values, exception handling, and memory management.

I'm coding in English.

The model is the interpreter. The structure is the bytecode. And the table it produces is simply the result of executing that program.

The core principle is universal: Drift happens when the model has choices. Remove the choices, remove the drift.

—

George Abrahamyants

December 2025