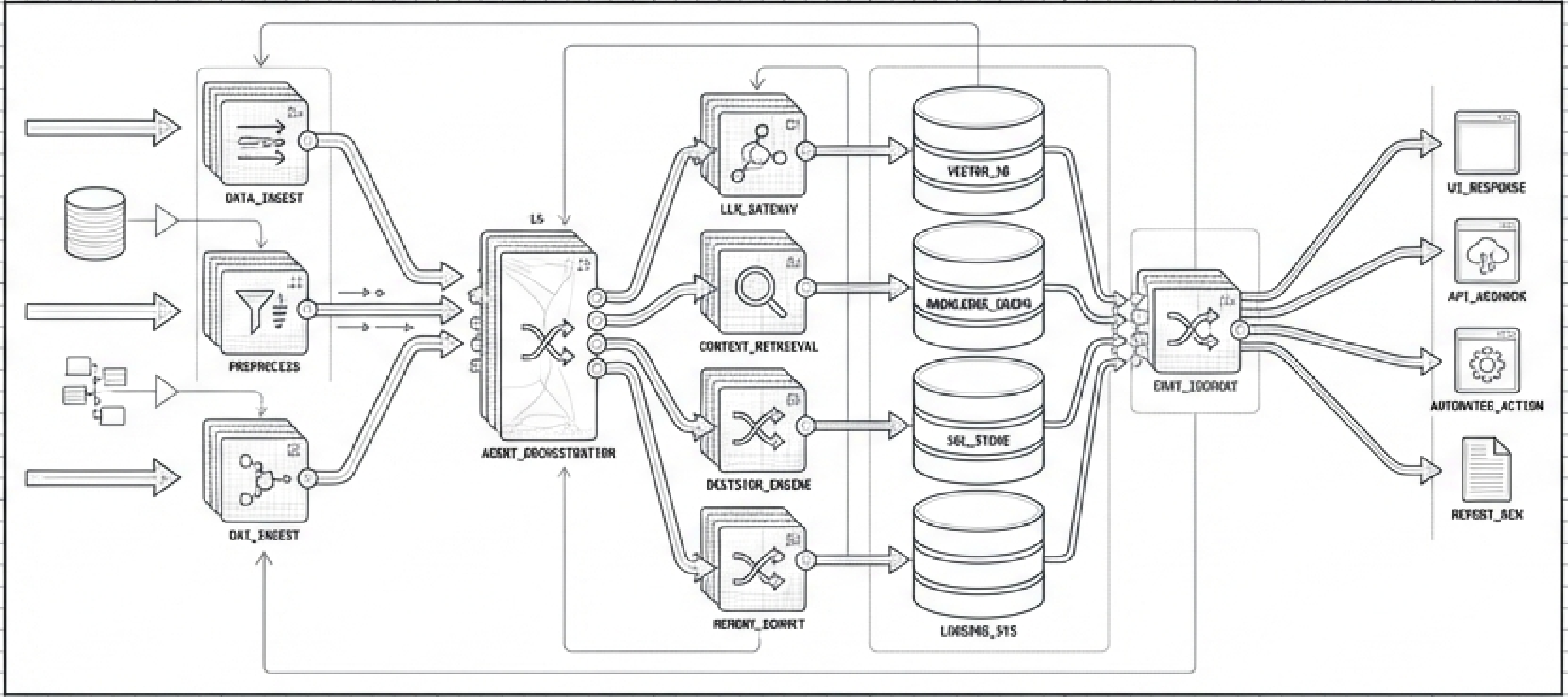
THE TEN AXIOMS OF AGENTIC DEVELOPMENT

A Methodology for AI-Native Software Engineering



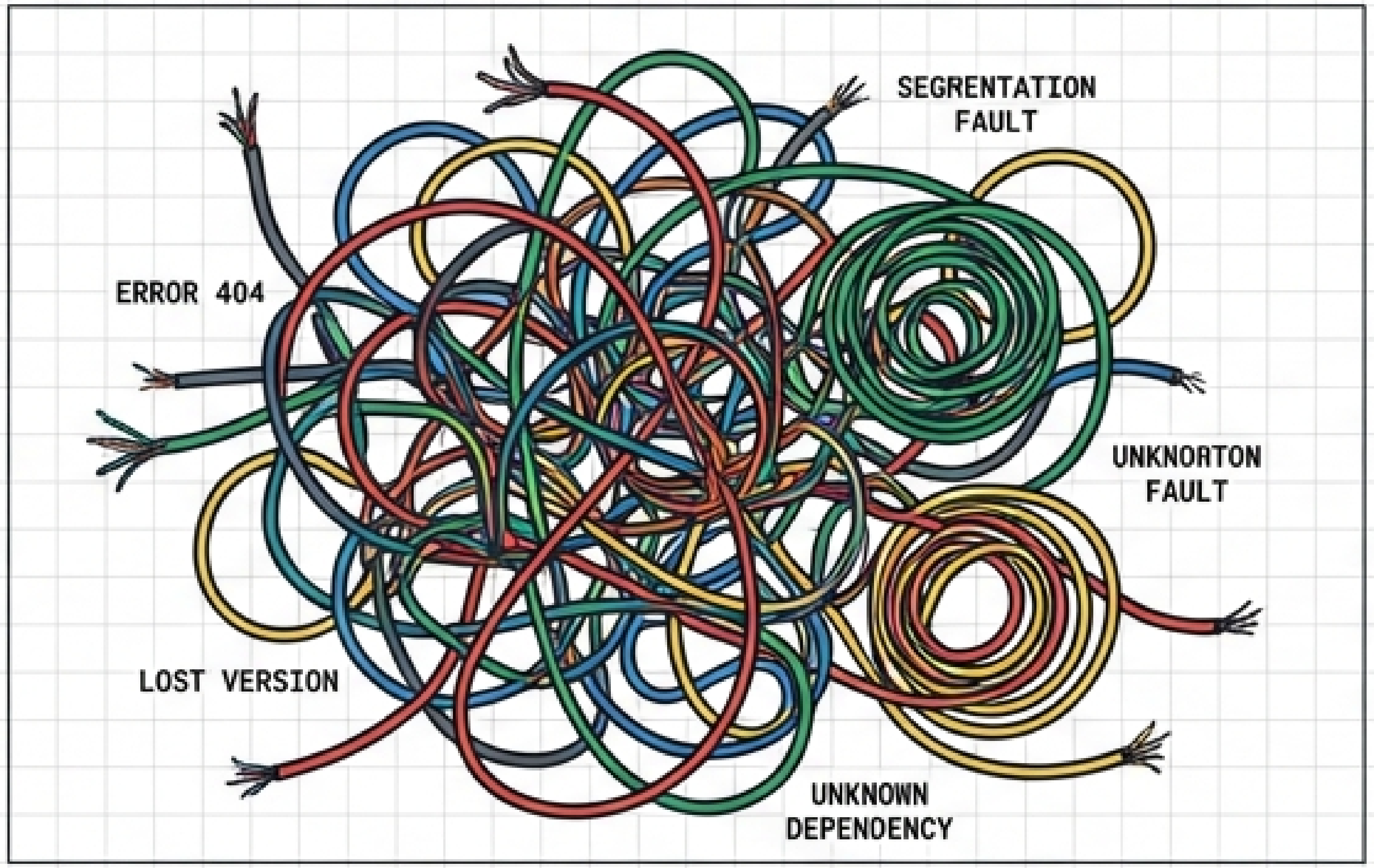
SOURCE MATERIAL:	Chapter 14
DOCUMENT TYPE:	Technical Manifesto
STATUS:	Final Specification

Bridging the gap between experiential learning and technical rigor.

NotebookLM

FROM EXPERIMENTAL SCRIPTS TO ENGINEERED SYSTEMS

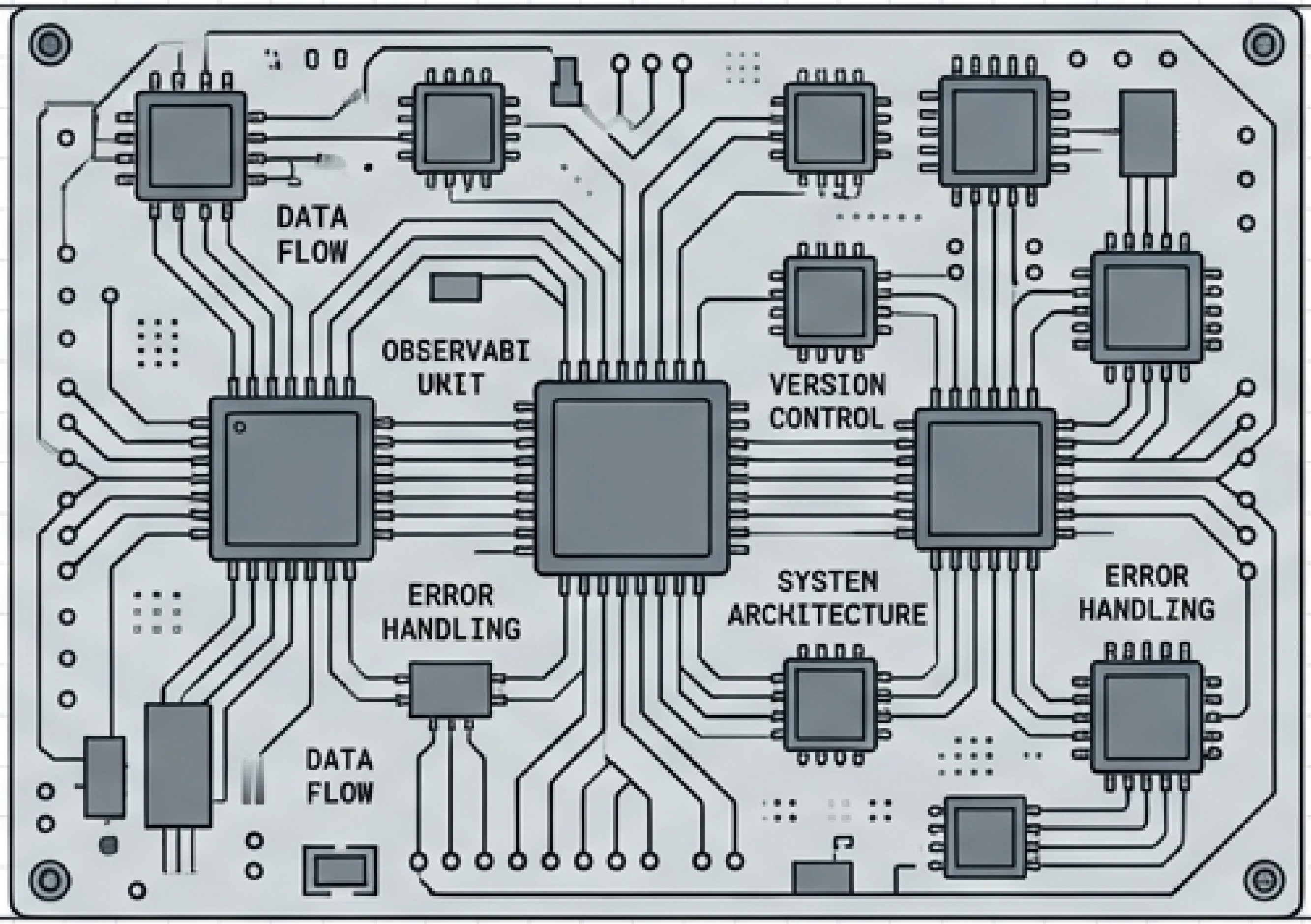
The Experimental Phase



Ad-hoc Scripting

Characterized by "Tomas". Fragile prototypes, 2 AM crashes, lost history, and vague error logs. The "It works on my machine" trap.

The Engineering Phase



Agentic Engineering

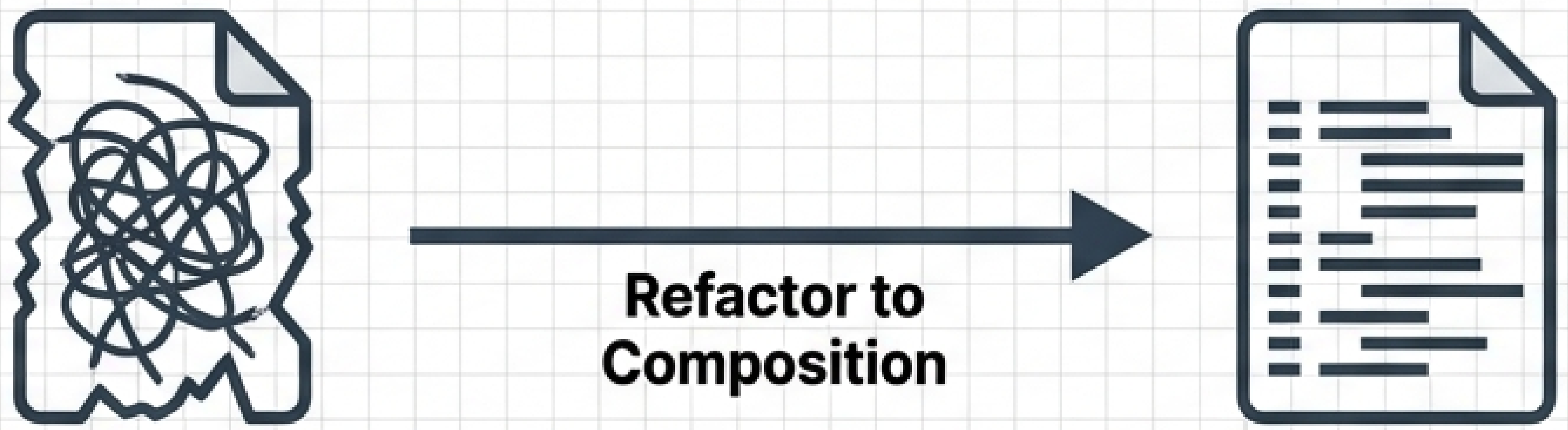
Characterized by "Lena". Rigorous discipline, historical principles applied to AI, predictable systems, and complete observability.

Core Thesis: AI agents require MORE discipline, not less.

NotebookLM

I. ORCHESTRATION

AXIOM I: SHELL AS ORCHESTRATOR



`deploy.sh`
400 lines of complex logic

`Makefile`
12 lines of declarative dependency

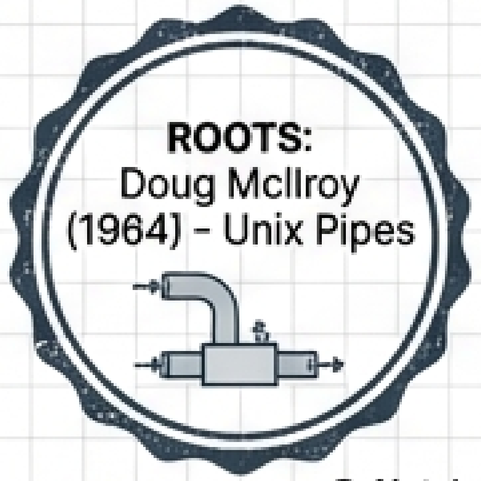
The **Shell** coordinates; Programs compute. Use the composition primitives:

- **Pipes** (Input/Output flow)
- **Exit Codes** (Error protocol)
- **Redirection** (Decoupling data)

THE TOMAS TRAP

Maria's deploy script crashed at 2:14 AM because complex application logic was hidden in fragile shell loops.

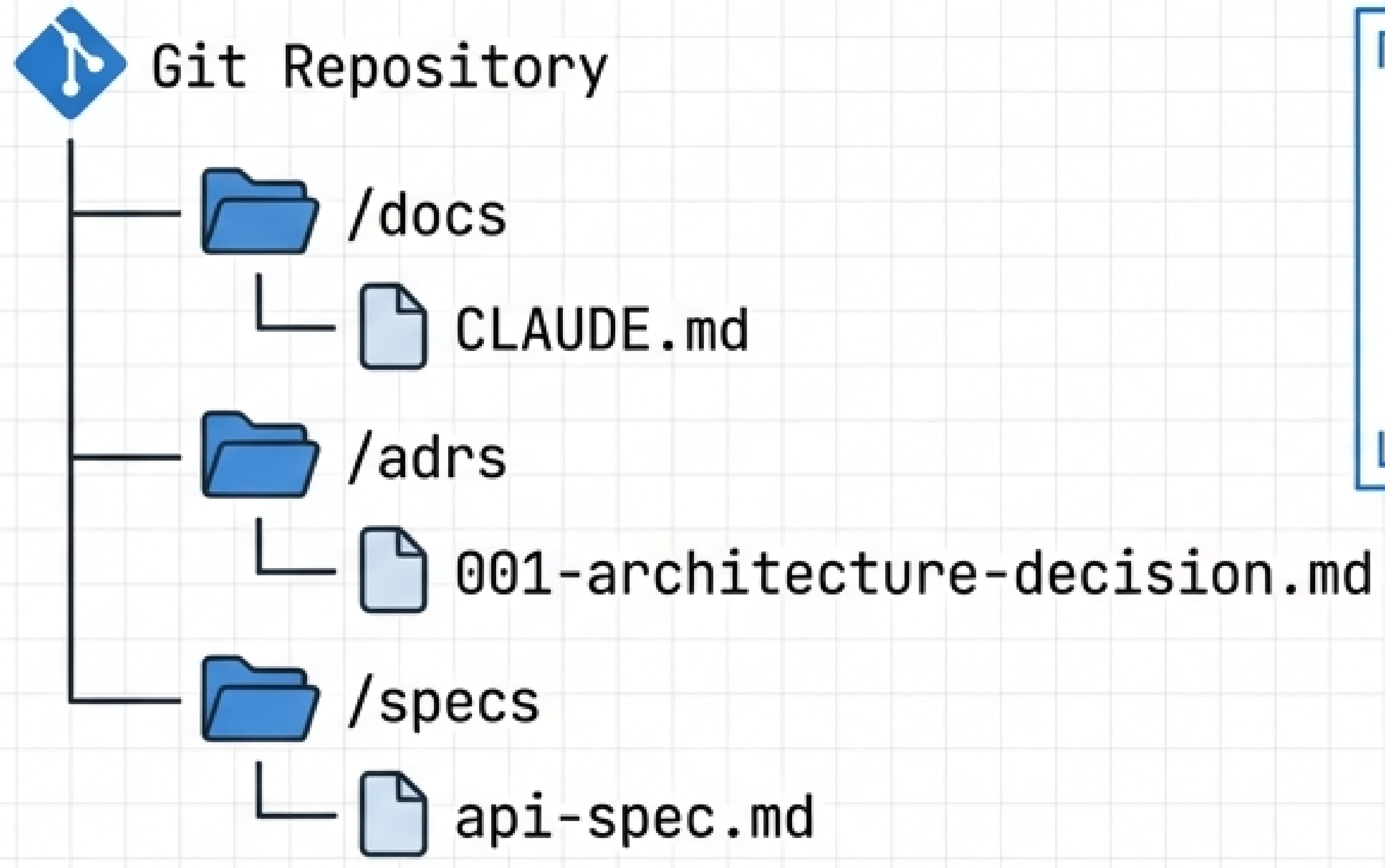
Solution:
Extract logic to programs. Use Makefiles for orchestration.



NotebookLM

II. SPECIFICATION

AXIOM II: KNOWLEDGE IS MARKDOWN



Why Markdown?

- Human-Readable
- Version-Controllable (Git)
- AI-Parseable (LLM Context)
- Tool-Agnostic

If it is not in the repository, it does not exist.
Persistent knowledge lives in Markdown.

THE TOMAS TRAP

Tomas spent two weeks building a REST integration the team had rejected. The decision was lost in a Slack thread.

Solution:

Record all decisions in ADRs (Architecture Decision Records).

ROOTS: John Gruber & Aaron Swartz (2004)

NotebookLM

AXIOM III: PROGRAMS OVER SCRIPTS

THE PYTHON DISCIPLINE STACK

4. Testing (`pytest`)
3. Linting & Formatting (`ruff`)
2. Type Checking (`pyright`)
1. Dependency Management (`uv`)

Helvetica Now Display

The Threshold - When to Switch

- ☐ Someone else runs the code
- ☐ It processes important data
- ☐ It exceeds 50 lines
- ☐ An AI generated it

THE TOMAS TRAP

A 15-line image renamer script crashed on file 847/2000 due to a Unicode character, leaving the directory in a corrupted half-state.

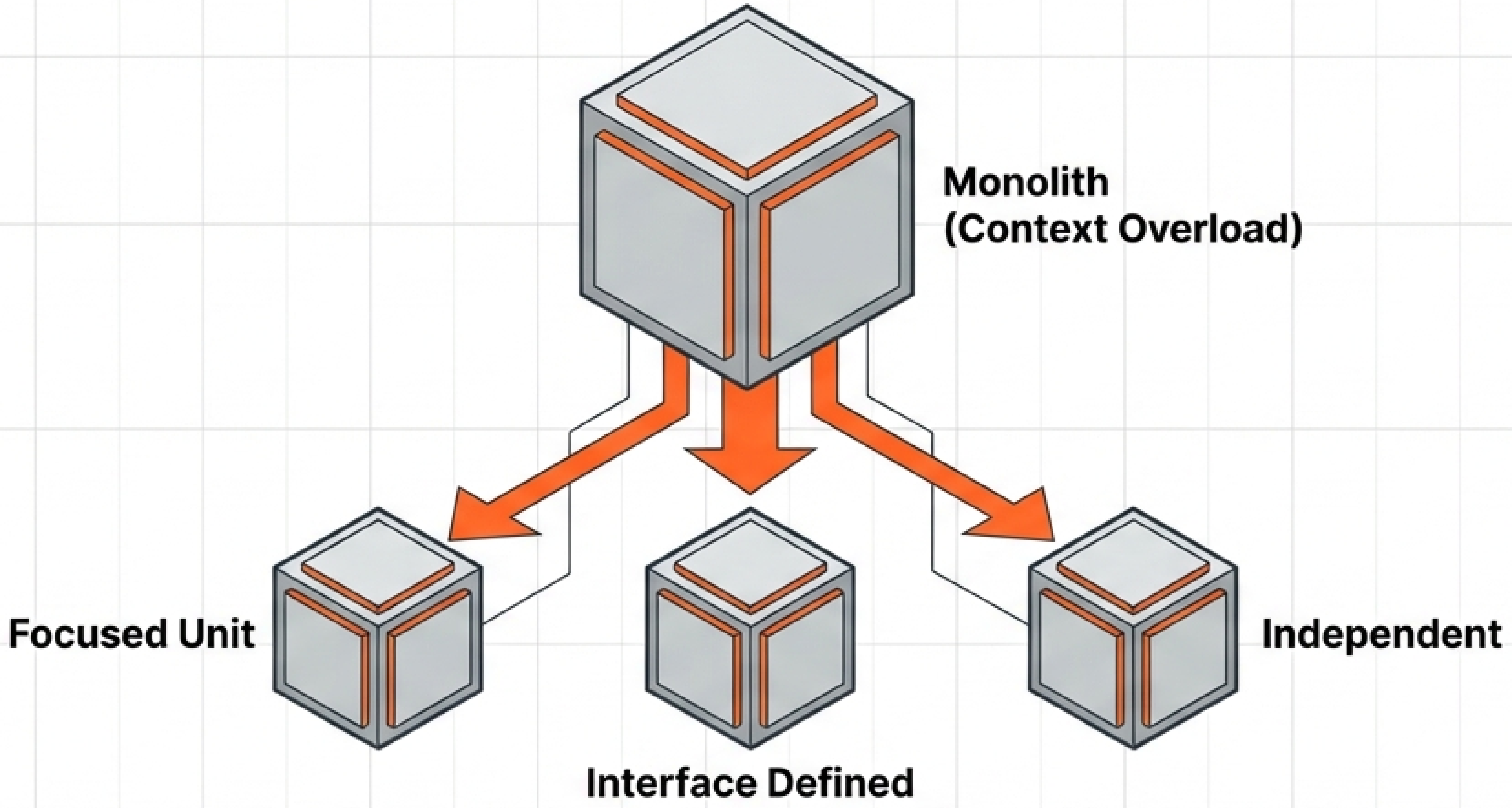
Solution:

Stop scripting when you add a dependency.

ROOTS:
Robin Milner
(1973) - ML Language

NotebookLM

AXIOM IV: COMPOSITION OVER MONOLITHS



AI cannot handle massive context windows.
Decomposition is essential.

`Use Dependency Injection to compose behavior.`

THE TOMAS TRAP

The 1,400-line "process_order()" function. Adding a discount feature accidentally broke tax calculations 300 lines away.

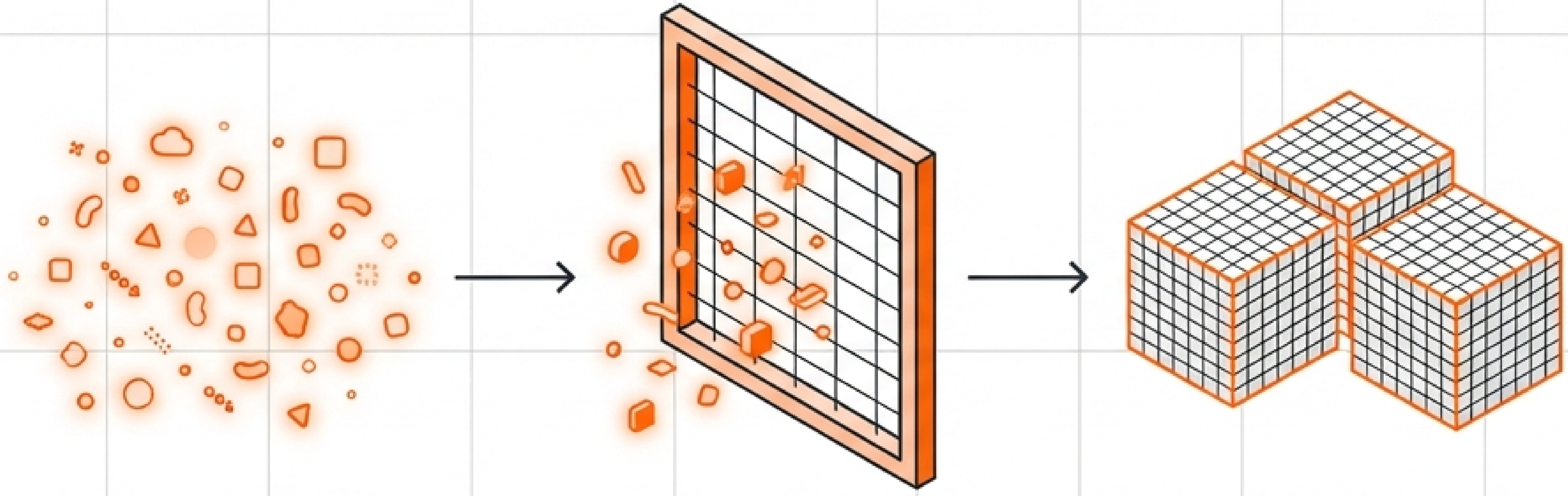
Solution:
Small, testable units with explicit inputs/outputs.

ROOTS: David Parnas (1972) - Information Hiding

NotebookLM

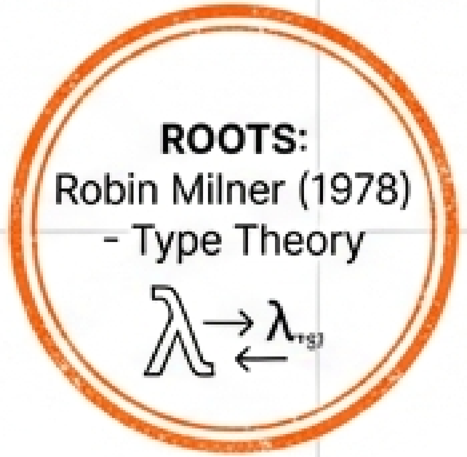
III. ARCHITECTURE

AXIOM V: TYPES ARE GUARDRAILS



The Three-Layer Discipline:

1. `Hints` (Contracts)
2. `Pyright` (Static Enforcement)
3. `Pydantic` (Runtime Validation)



THE TOMAS TRAP

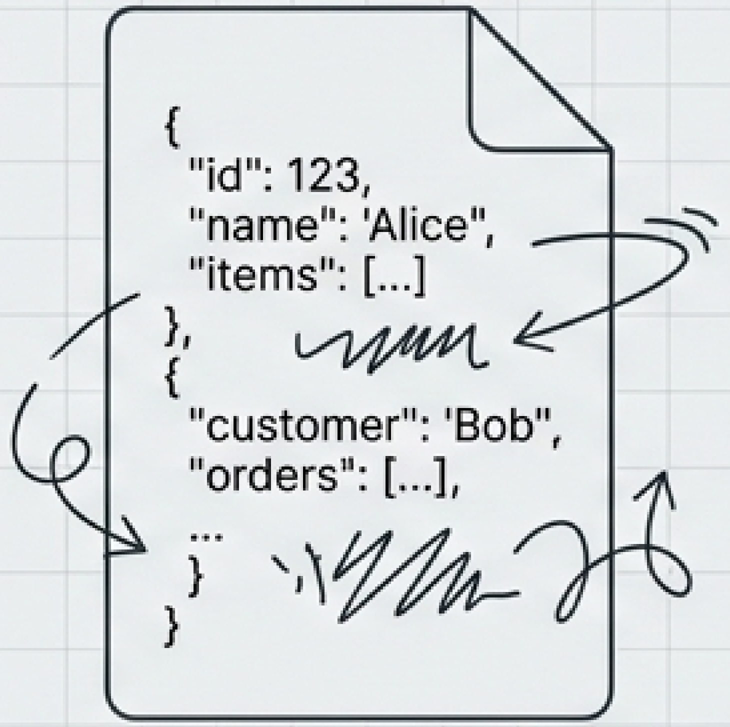
Production crash: **AttributeError**. The AI generated code that treated a Dictionary as an Object.

Solution: `Pydantic` at the edges (external data), `Dataclasses` at the core.

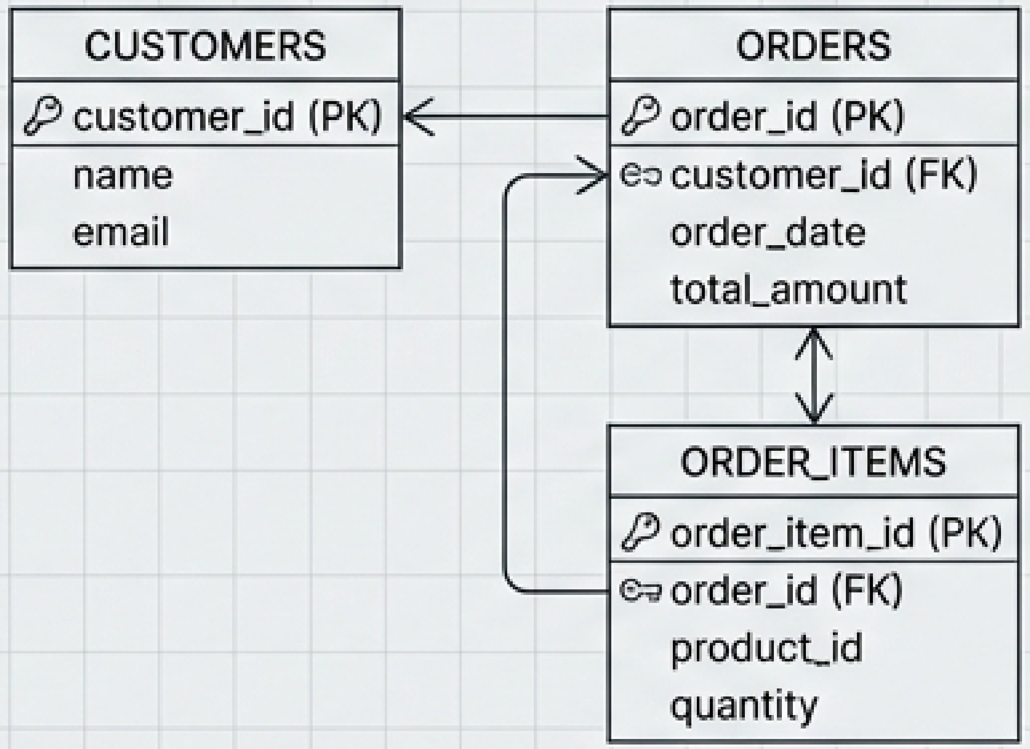
NotebookLM

III. ARCHITECTURE

AXIOM VI: DATA IS RELATIONAL



JSON (Loose Structure)



SQL (Relational Specification)

Schema is the specification for data. SQL is declarative, constrained, and transactional.

`SQLite for single-user. PostgreSQL for multi-user.`

THE TOMAS TRAP

orders.json grew to 2,000 records. Queries took 11 seconds. Customer names were inconsistent.

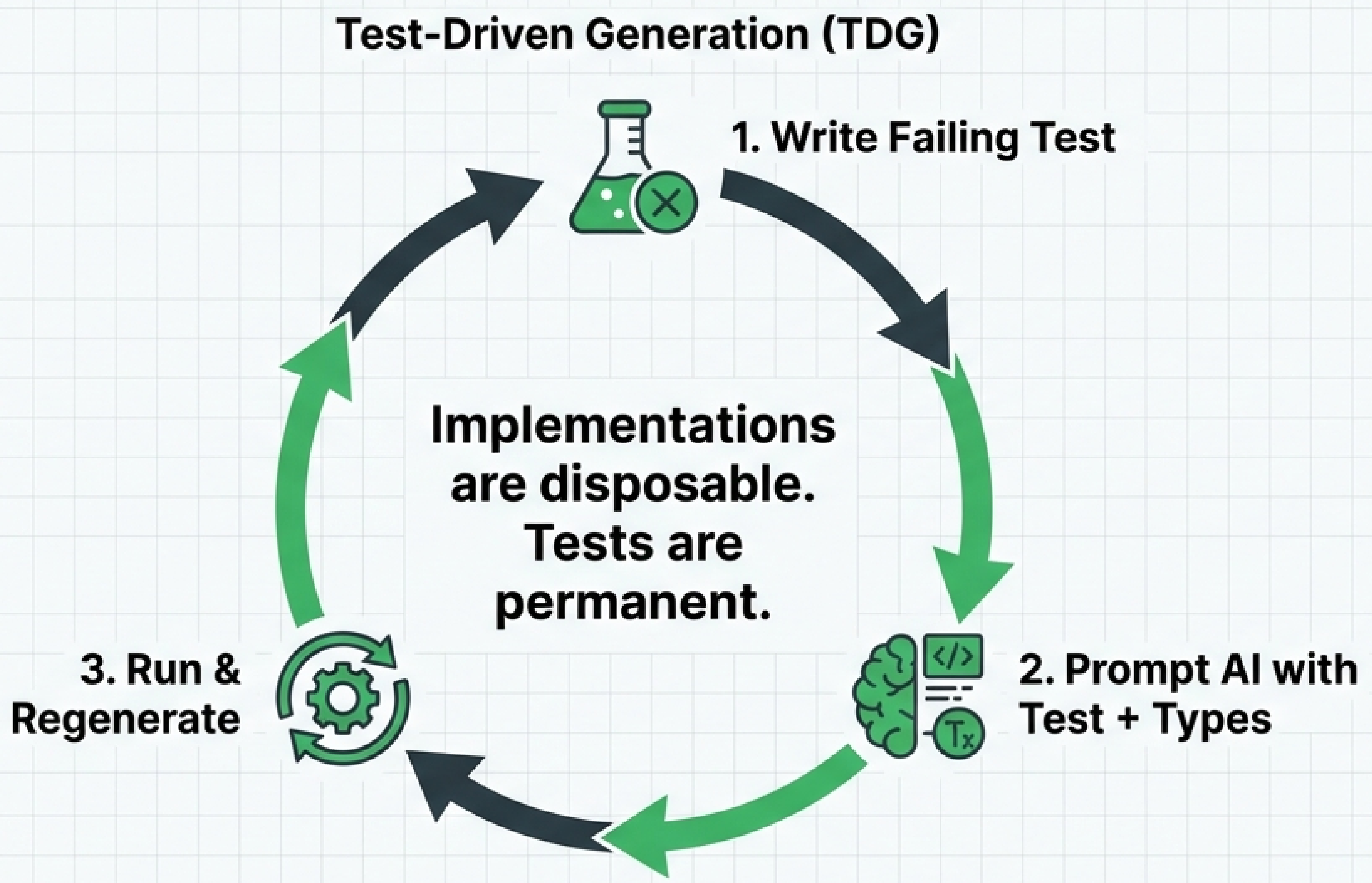
Move to SQL. The schema enforces the data quality.

ROOTS:
Edgar F. Codd (1970)
- Relational Model

NotebookLM

IV. VERIFICATION

AXIOM VII: TESTS ARE THE SPECIFICATION


Test-Driven Generation (TDG)
1. Write Failing Test
2. Prompt AI with Test + Types
3. Run & Regenerate
Implementations are disposable. Tests are permanent.

THE TOMAS TRAP

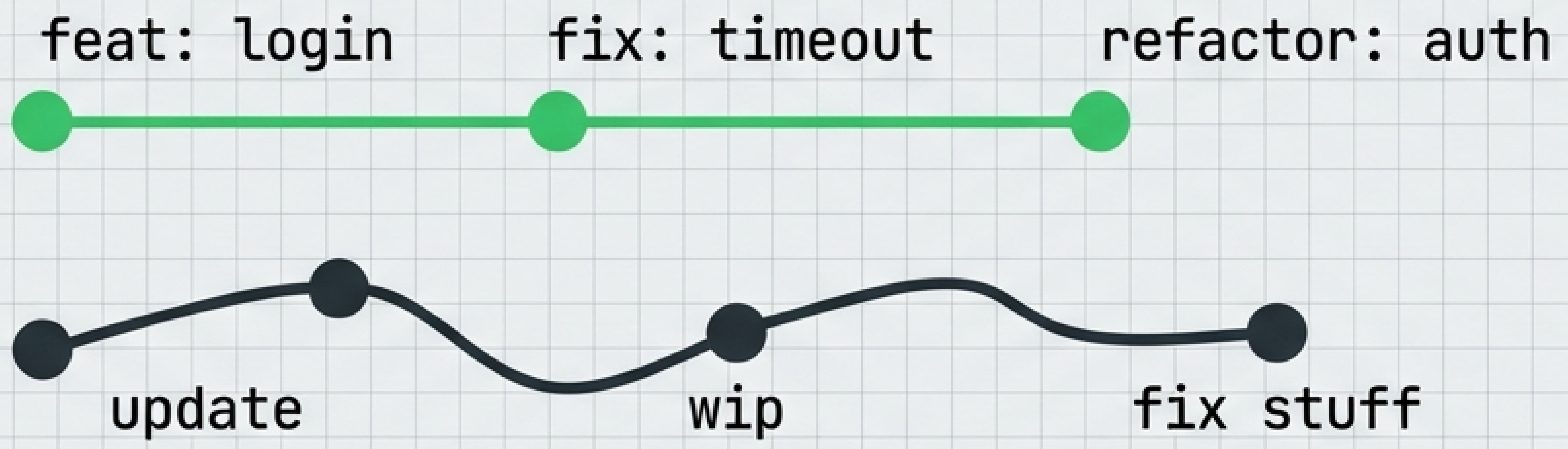
The $12,000 Bug. The code looked perfect and had types, but the logic returned 0.15 instead of 0.85.

Solution: Write the test *before* the code.

ROOTS: Kent Beck (2002) - TDD

NotebookLM

AXIOM VIII: VERSION CONTROL IS MEMORY



Git records decisions, not just code. Follow the 'WHY' rule in commit messages.

- Atomic Commits (One logical change)
- Conventional Commits (Standard prefixes)

THE TOMAS TRAP ⚠

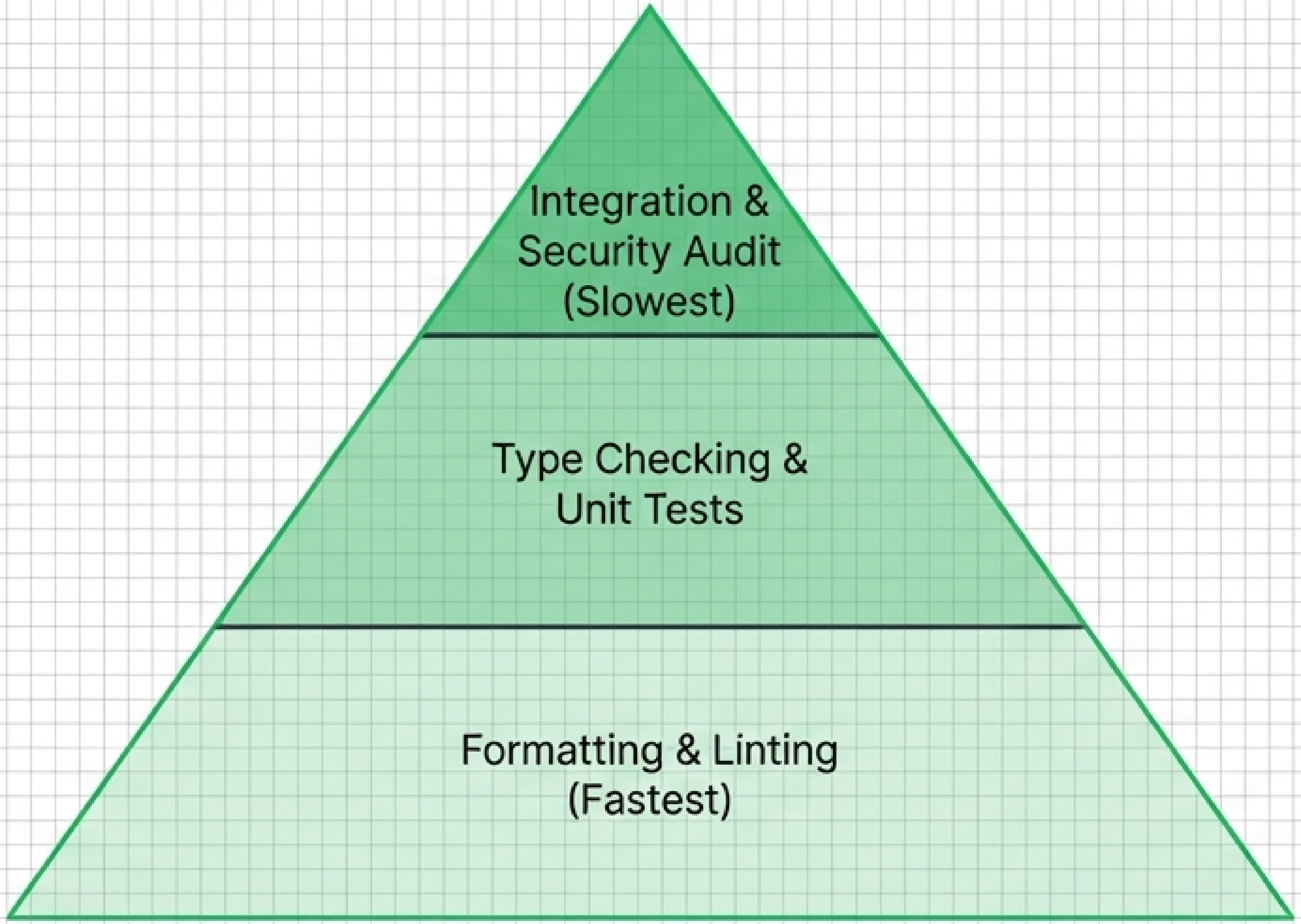
The original bug history was lost because the commit messages were just "wip" and "fixes".

Solution:
Treat version control as a historical record of intent.

ROOTS: Linus Torvalds (2005) - Git

NotebookLM

AXIOM IX: VERIFICATION IS A PIPELINE


Integration & Security Audit (Slowest)
Type Checking & Unit Tests
Formatting & Linting (Fastest)

If it is not in CI, it is not enforced.

THE TOMAS TRAP

The "Shallow Pipeline". Confidence was based on local runs, but the build failed 4 minutes into the remote CI process.

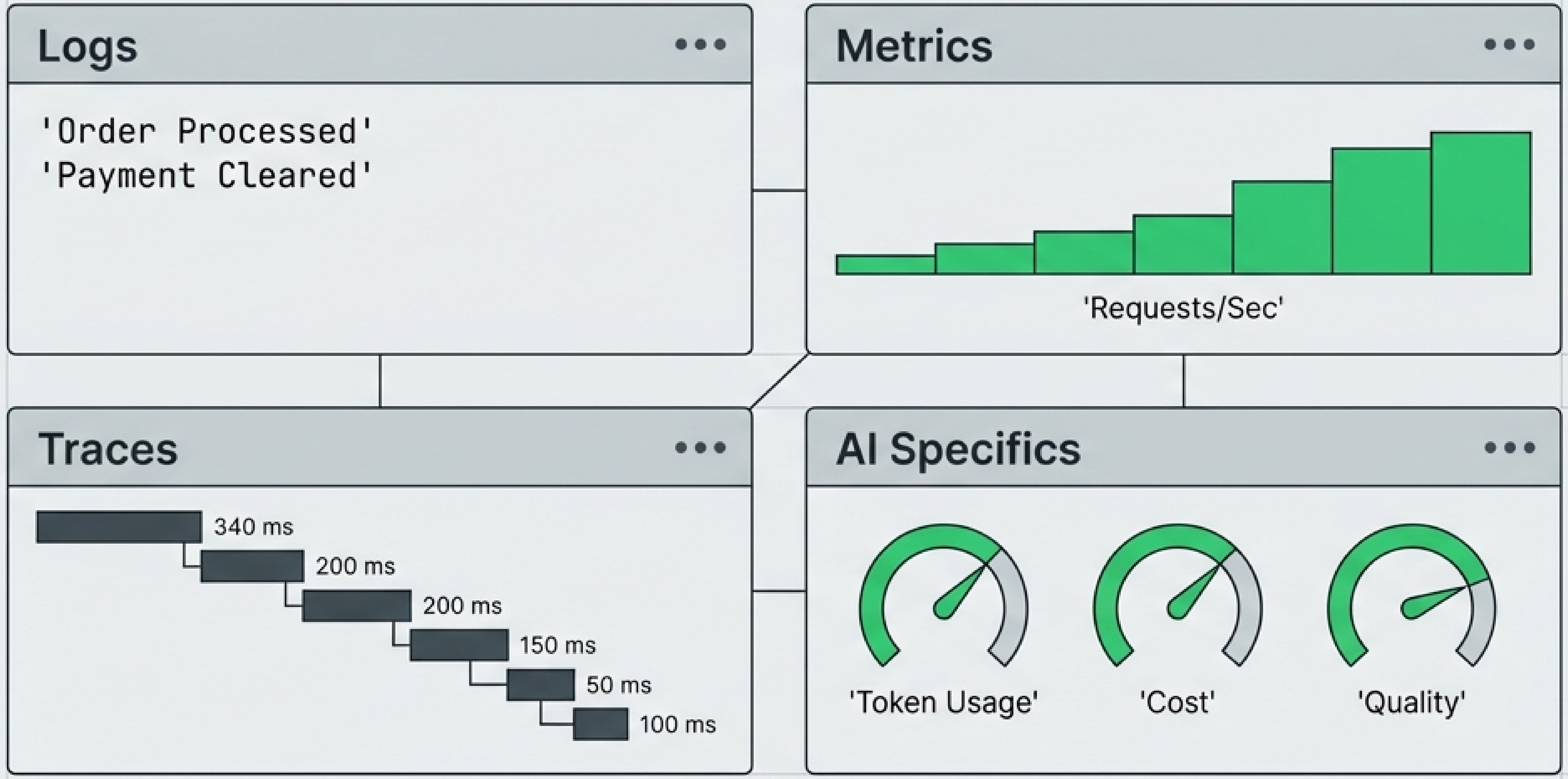
Solution: Mirror remote pipelines locally (`make ci`).

ROOTS: Martin Fowler (2000) - Continuous Integration

NotebookLM

IV. VERIFICATION

AXIOM X: OBSERVABILITY EXTENDS VERIFICATION



Testing doesn't stop at deployment.

⚠ THE TOMAS TRAP

Shipping rates were wrong during peak traffic. Logs only said 'ERROR something went wrong'.

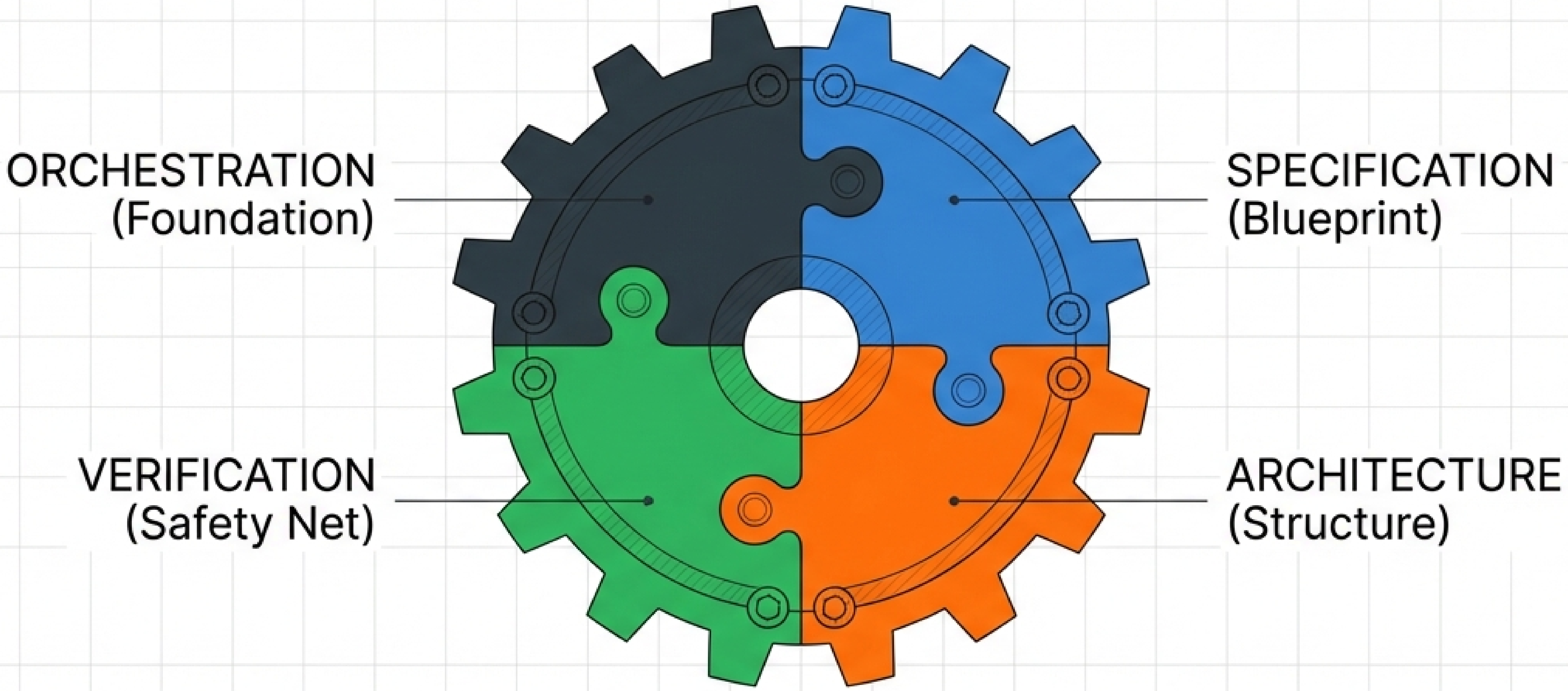
Solution: Structured JSON logs and detailed tracing.

ROOTS: Google SRE (2016)

NotebookLM

THE COMPLETE SYSTEM

The Interlocking Phases of Agentic Engineering



Software development is a system. Skip one axiom, and a vulnerability opens. These layers provide the complete coverage required for reliable AI agents.

NotebookLM

SUMMARY & CHECKLIST

AXIOM	THE TRAP (Failure Mode)	THE TOOL (Engineering Fix)
Shell	400-line script crash	Makefile & Composition
Markdown	Lost decisions/Context	ADRs in Repo
Programs	Prototype fragility	uv / ruff / pyright
Composition	Context overload	Dependency Injection
Types	Hallucinated Objects	Pydantic / Dataclasses
Data	Slow, messy JSON	SQL (SQLite/Postgres)
Tests	Logic errors ($12k bug)	Test-Driven Generation
Git	“Fix stuff” history	Atomic Conventional Commits
Pipeline	“Works on my machine”	GitHub Actions
Observability	Vague errors	Structured Logs + Metrics

“AGENTIC DEVELOPMENT REQUIRES MORE DISCIPLINE, NOT LESS.”

NotebookLM