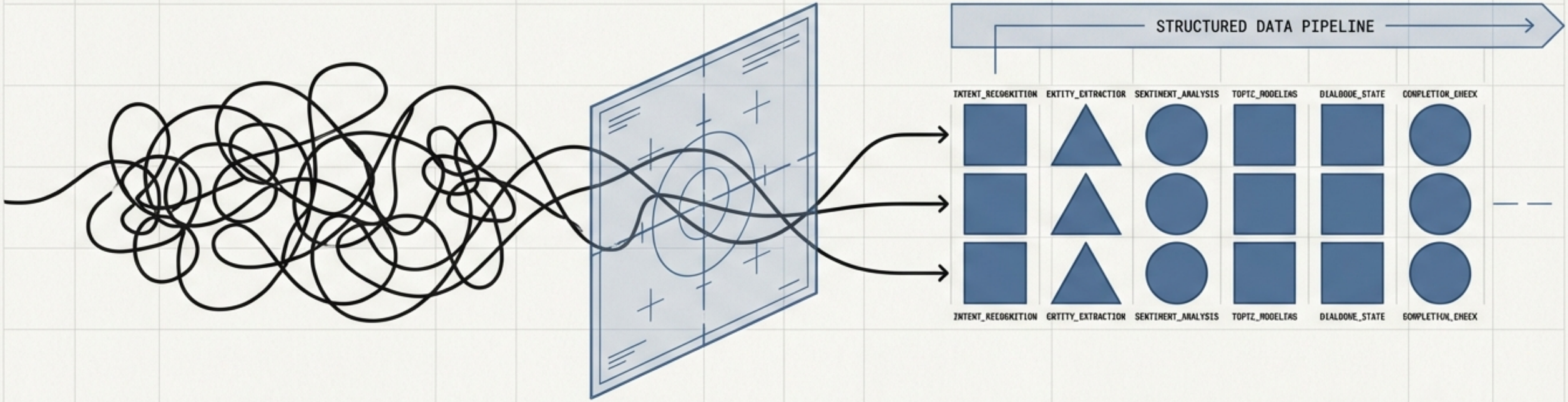
Multi-Turn Conversation Evaluation

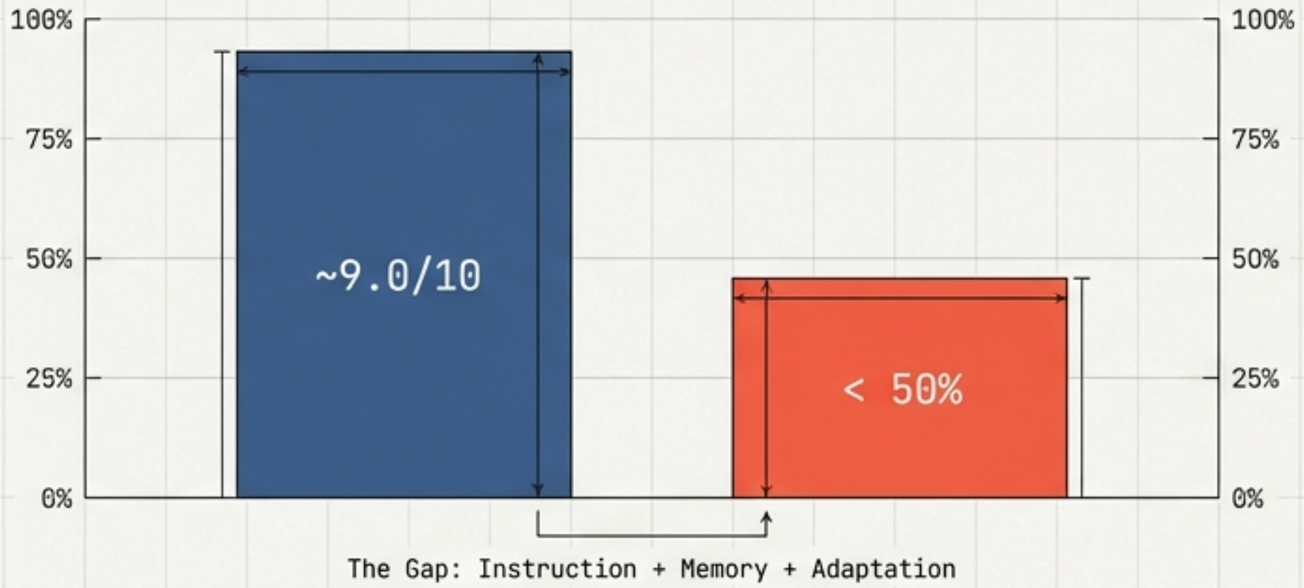
A First Principles Guide to Debugging, Monitoring, and Scaling AI Agents.



Benchmark Performance is an Illusion

Frontier models fail when challenges compound. You cannot rely on generic proxies for application success.

Isolated Capabilities (MT-Bench)

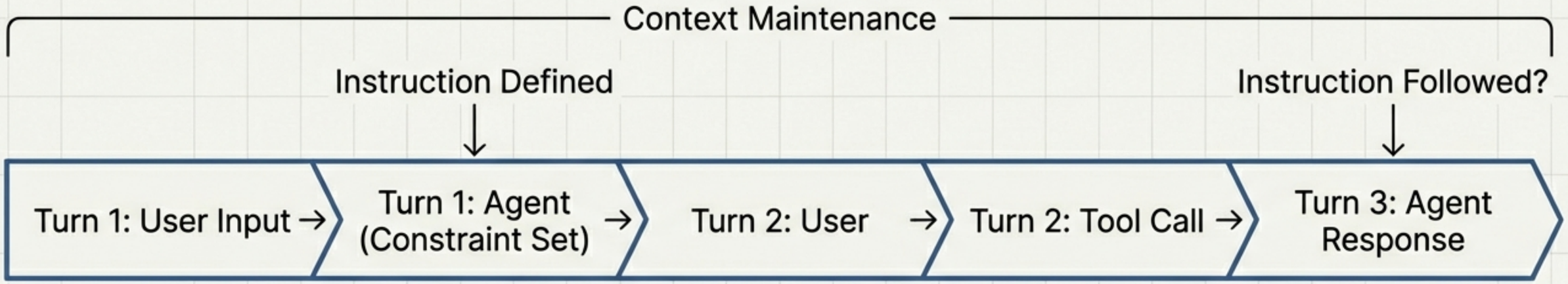


Compositional Reality (MultiChallenge)

The Reality Check

Assumption: Good single-turn performance predicts multi-turn robustness.
Reality: Context decay and attention loss are non-linear. Generic benchmarks do not account for custom application constraints.

Anatomy of a Multi-Turn Trace



Taxonomy of Failure (MultiChallenge)

1. **Instruction Retention**: Failing to follow constraints set in Turn 1.	2. **Inference Memory**: Failing to connect details across turns.
3. **Versioned Editing**: Failing revision cycles.	4. **Self-Coherence**: Contradicting previous statements.

Trace: The complete record of a session. The fundamental unit of analysis.

The Three-Level Evaluation Hierarchy

Axiom: It is not efficient to evaluate every turn in every trace.

High Altitude, Low Cost →

← High Altitude, Low Cost

← Low Altitude, High Cost

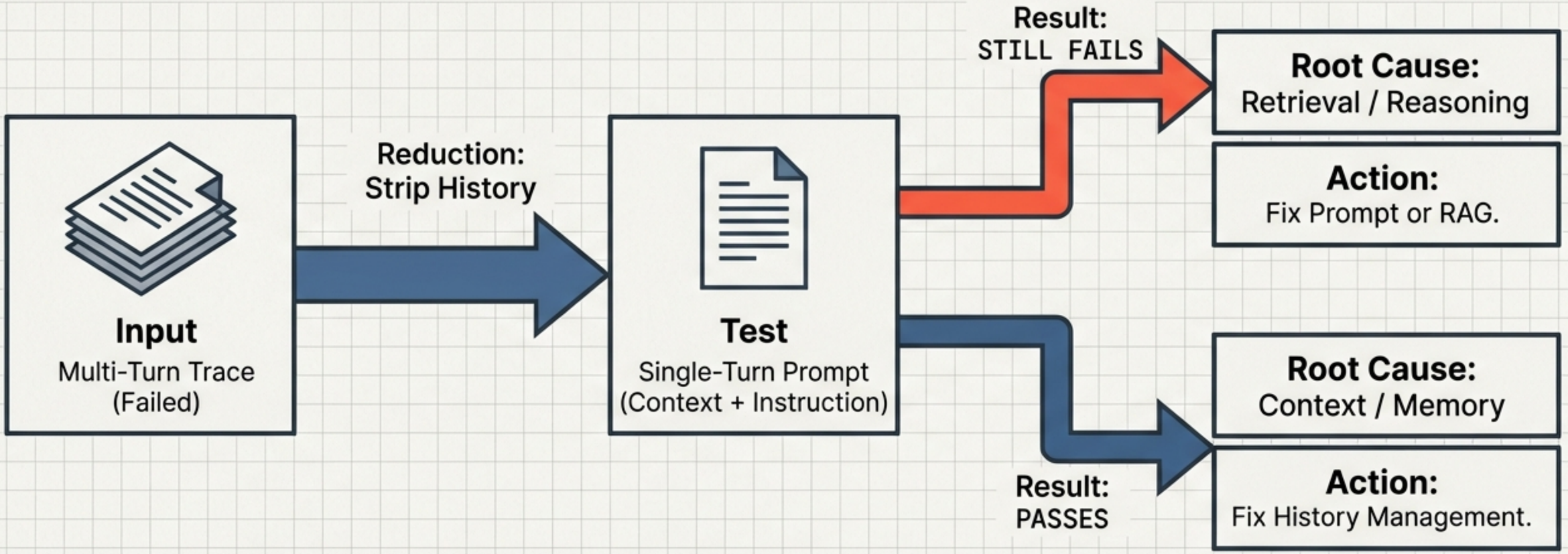
Altitude determines Cost & Signal.

Challenging Engineering Assumptions

~~MYTH~~	REALITY
~~Evaluate every turn in every trace.~~	Wasteful. Most turns pass. Evaluate `Session-Level` for signal, `Turn-Level` only for root cause.
~~Multi-turn failures are always context issues.~~	False. Many reduce to `single-turn retrieval` or `grounding failures`.
~~More turns = Harder evaluation.~~	Difficulty depends on `Challenge Composition` (`Memory + Instruction`), not just length.
~~Benchmarks Transfer.~~	The Benchmark-to-Application gap is systematic **(>45 point drop)**.

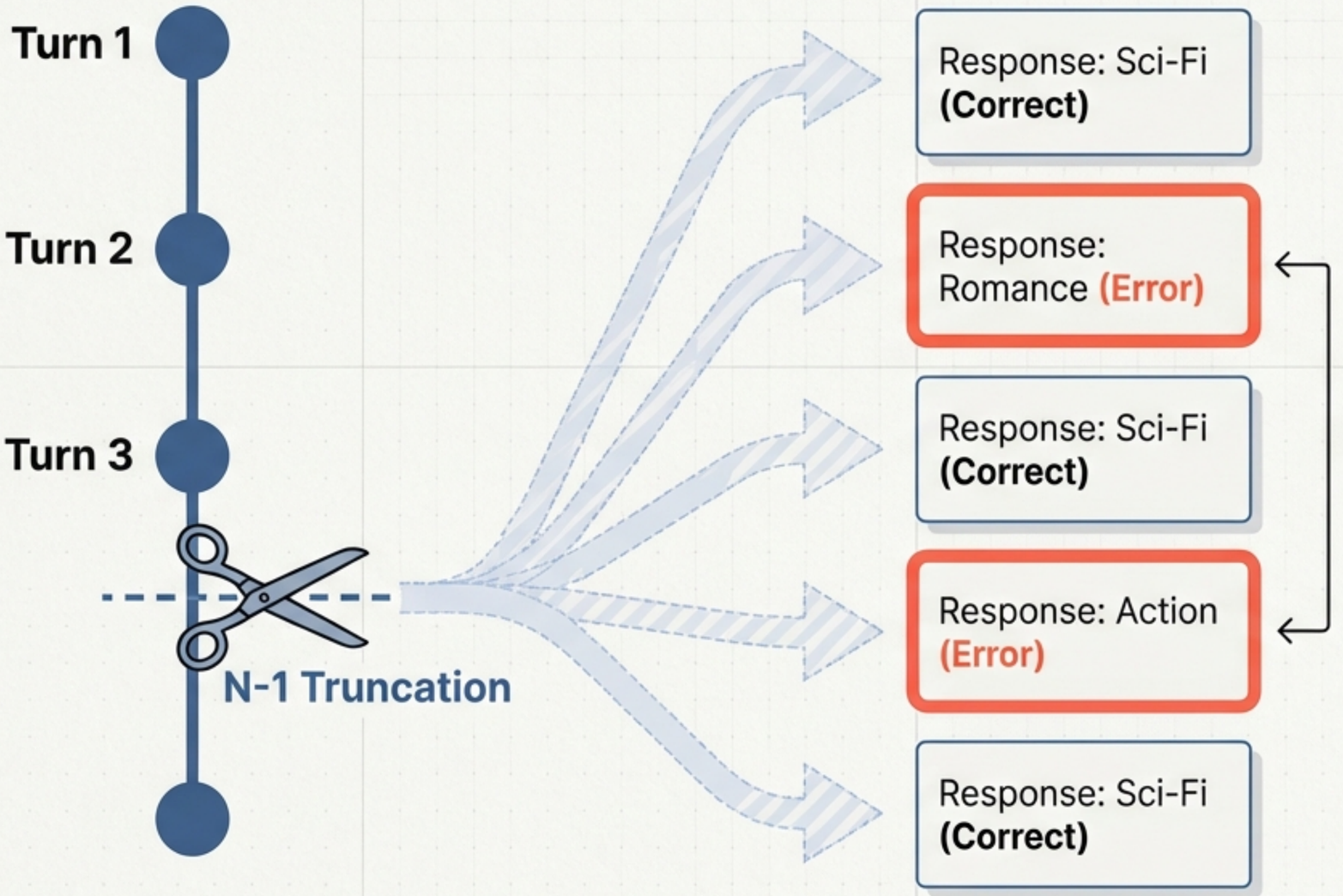
The Principle of Single-Turn Reduction

Isolate the variable. Most multi-turn failures are single-turn failures in disguise.



Example: If the model hallucinates a return policy in single-turn, no amount of context window debugging will fix it.

Debugging Memory: N-1 Prompt Sampling



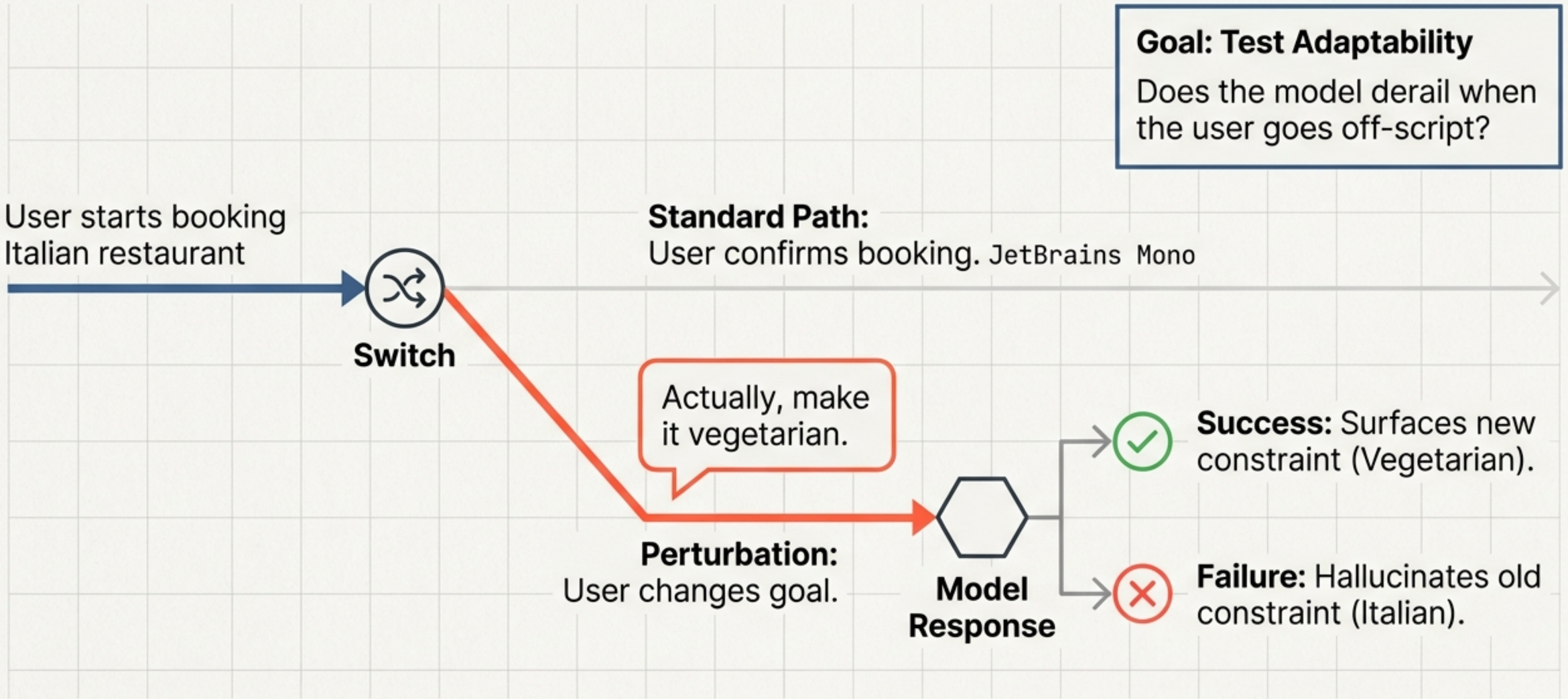
Method:

1. Truncate Trace.
2. Sample 5x.
3. Measure Probability of Forgetting.

Stress-Testing Robustness: Perturbations

Goal: Test Adaptability

Does the model derail when the user goes off-script?



Mapping Abilities to Evaluation Logic

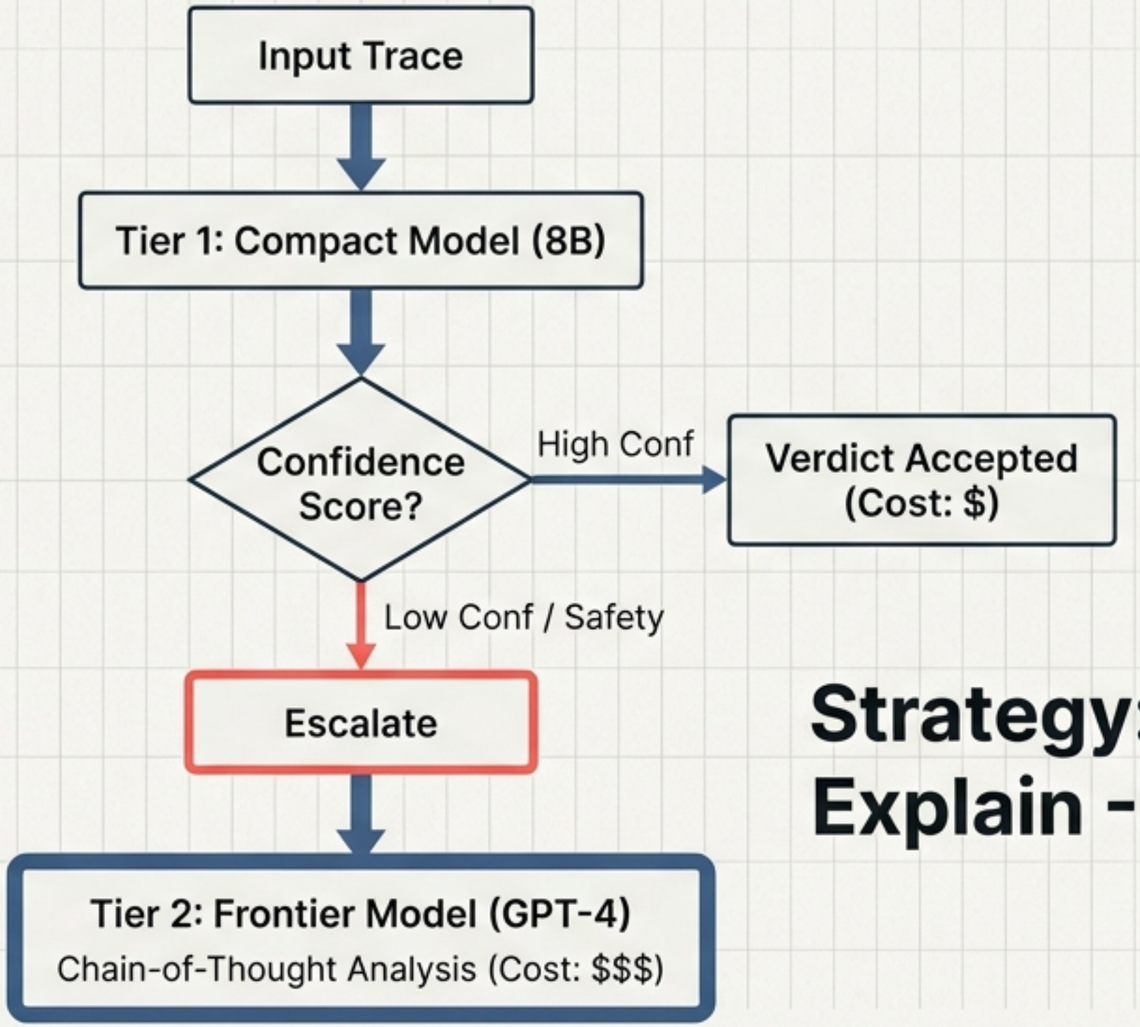
Ability (MT-Bench-101)	Definition	The Right Test
Perceptivity (Memory)	Context retention, anaphora resolution (“it” = “the movie”).	**N-1 Prompt Sampling**
Adaptability (Reasoning)	Handling goal shifts and multi-turn reasoning chains.	**Session Pass/Fail** & **Perturbations**
Interactivity (Clarification)	Proactive questioning when instructions are ambiguous.	**Turn-Level Inspection**

Frameworks: TD-EVAL & MT-Bench-101

Production Scale: The Monitor-Escalate Pipeline

High-fidelity eval is expensive. Use a tiered defense.

54% Cost Reduction



Strategy: Detect -> Explain -> Escalate

Adapting the Framework to Your Domain

Define your Session Goal first. The principles stay the same.

Goal: Resolution.

Failure Mode: Citing wrong policy.

Metric: Retrieval Accuracy.

Goal: Implementation.

Failure Mode: Losing codebase context.

Metric: Execution Success.

Goal: Diagnosis Suggestion.

Failure Mode: Missing connected symptoms.

Metric: Consistency (Coherence).

Boundary Conditions: When This Breaks

Safe Zone (Works)

- Task-oriented conversations.
- Binary success criteria.
- Context < `100k tokens`.
- Observable behaviors.

Know the limits of the framework.

Danger Zone (Fails)

- Open-ended social chat.
- **Infinite context** sessions.
- **Subjective/Implicit** goals.

The Engineering Checklist

☐	**Define Session Success:** Create a binary Pass/Fail metric based on user goals.
☐	**Prune the Trace:** Attempt Single-Turn Reduction before debugging context.
☐	**Test Memory:** Implement N-1 Prompt Sampling for perceptivity bugs.
☐	**Stress Test:** Run Perturbations for goal-shift robustness.
☐	**Optimize Cost:** Deploy Monitor-Escalate pipeline for production.

Key Takeaways & Axioms

01 **Session-Level is Primary.**
Optimize for user success.

02 **Reduction First.**
Multi-turn bugs are often single-turn bugs.

03 **Benchmarks Lie.**
They do not predict application capability.

04 **Granularity is for Debugging.**
Don’t put turn-level metrics on a dashboard.

05 **Scale Smart.**
Use tiered models to save >50% on cost.

“The goal of evaluation is not to generate a score, but to identify the specific primitive (**Memory, Retrieval, Instruction**) that needs fixing.”