Artificial Intelligence in a Human World

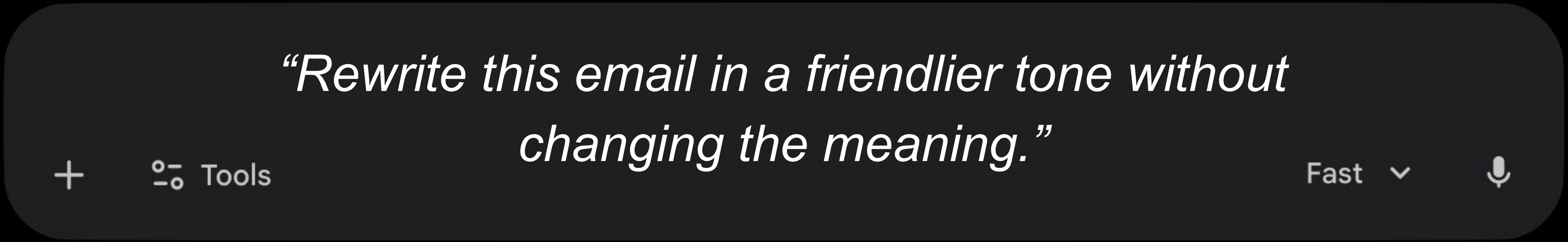
The AI’s job is *plausibility*, not truth.

The AI said it compiles, the compiler says no.

Act 1

Rules ensure intent

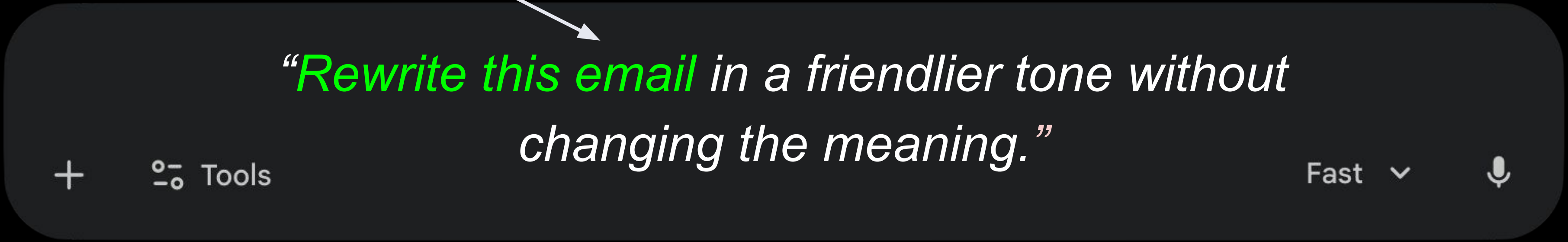
Rules must be truthful to the intent and be applied consistently.



"Rewrite this email in a friendlier tone without changing the meaning."

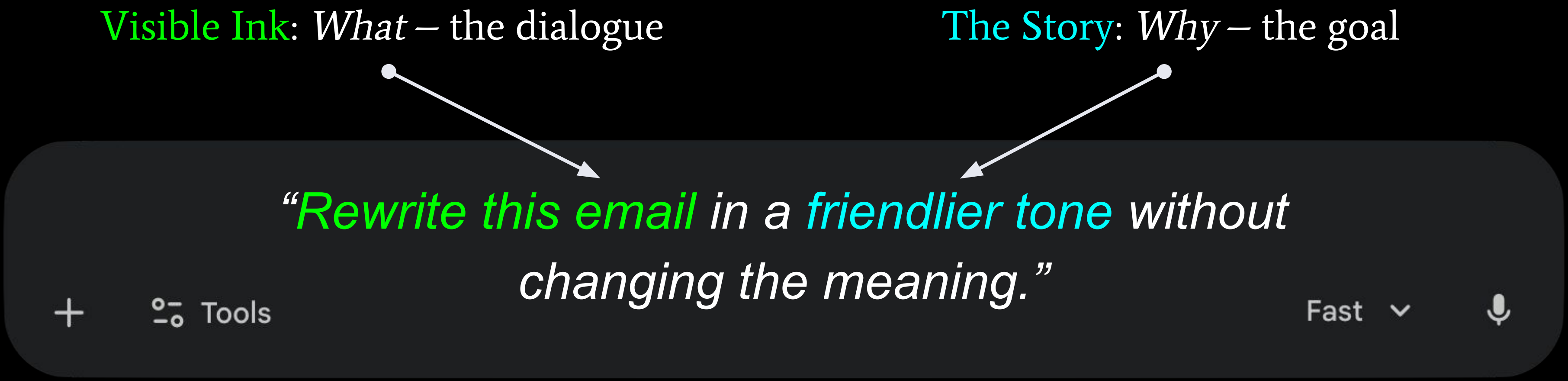
Rules must be *truthful* to the intent and be applied consistently.

Visible Ink: *What* – the dialogue



“Rewrite this email in a friendlier tone without changing the meaning.”

Rules must be *truthful* to the intent and be applied consistently.

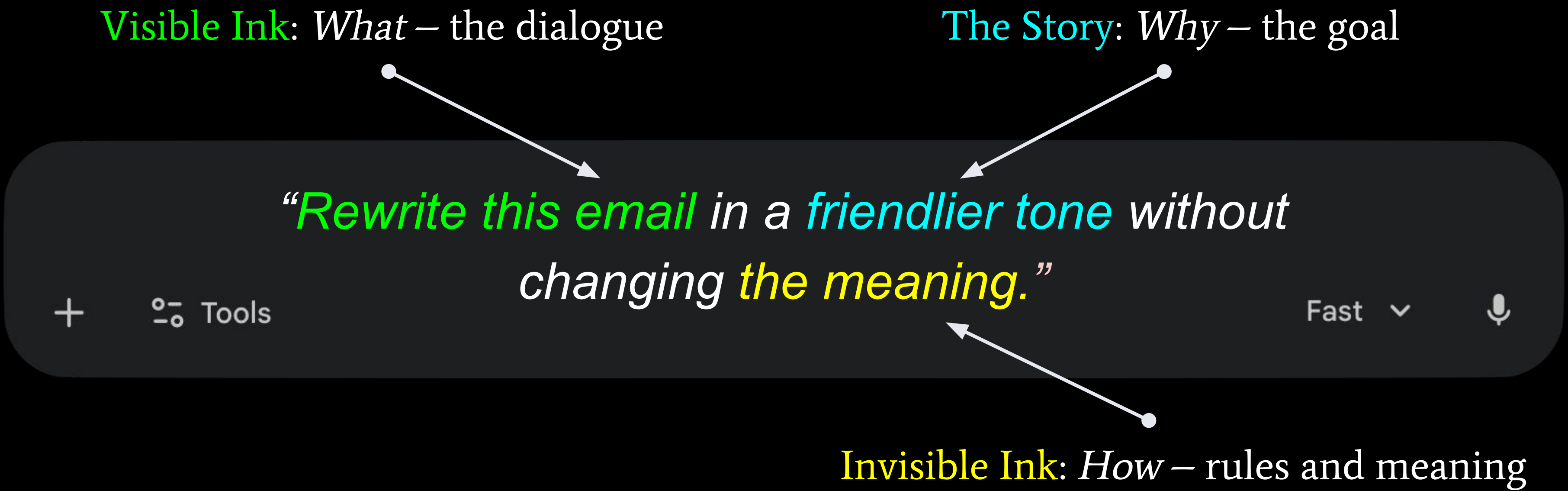


Visible Ink: *What* – the dialogue

The Story: *Why* – the goal

“Rewrite this email in a friendlier tone without changing the meaning.”

Rules must be *truthful to the intent* and be applied consistently.



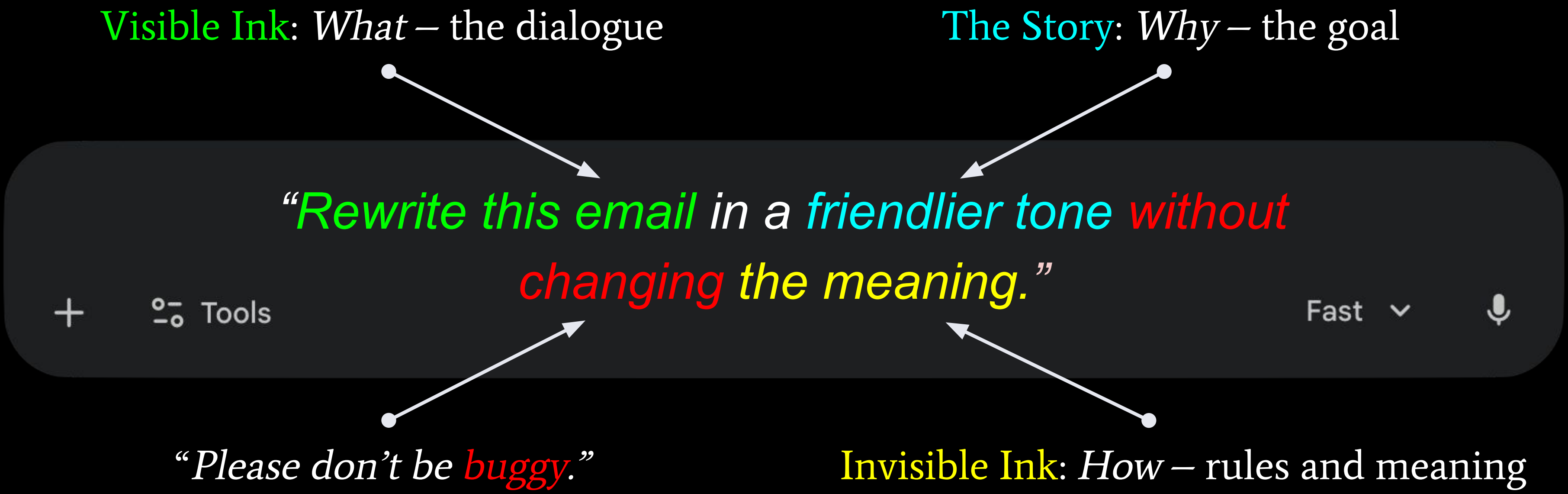
Visible Ink: *What* – the dialogue

The Story: *Why* – the goal

“Rewrite this email in a friendlier tone without changing the meaning.”

Invisible Ink: *How* – rules and meaning

Rules must be *truthful to the intent* and be *applied consistently.*



Visible Ink: *What* – the dialogue

The Story: *Why* – the goal

“Rewrite this email in a friendlier tone without changing the meaning.”

“Please don’t be buggy.”

Invisible Ink: *How* – rules and meaning

Hallucination is the feature, not a bug.

LLMs are *probabilistic engines* that are *optimized* for *likely/preferred outputs*, not guaranteed to be objectively true.

(And we want that!)

Hallucination is the feature, not a bug.

(Unless it is *factually wrong*, Then it's *a smudge*.)

LLMs are *probabilistic engines* that are *optimized* for *likely/preferred outputs*, not guaranteed to be objectively true.

(And we want that!)

Rules need to be *observationally true* to the *story*.

User Journeys are Stories: (*Visible Ink*)		Stories have rules: (*Invisible Ink*)
● Sign up and check out	→	● Minimize user effort and build trust
● Quarterly report	→	● Requirements from regulatory bodies
● Automating customer service	→	● A seamless, omnichannel experience
● Analyzing large datasets to find trends	→	● Map patterns to benchmarks

Epistemics – what the model can say that is *storybound and context-limited*

Governance – what the system can trust/act on with a *deterministic judge and receipts*

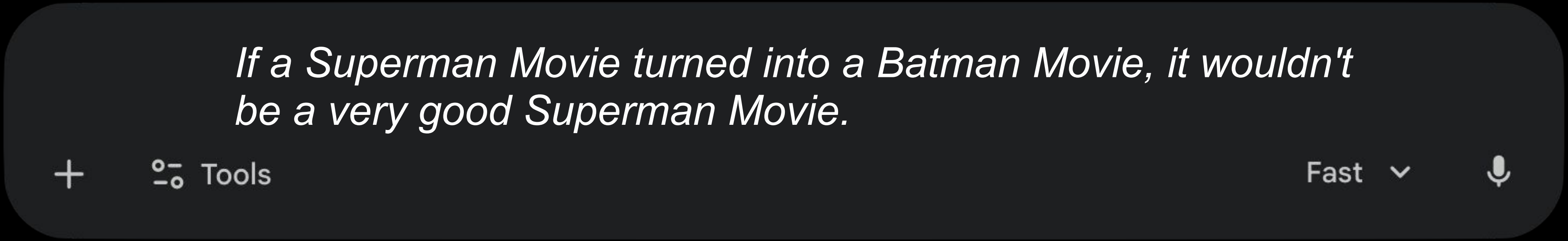
Rules need to be *observationally true* to the *story*.

Rules ***do not*** need to be *objectively true*.

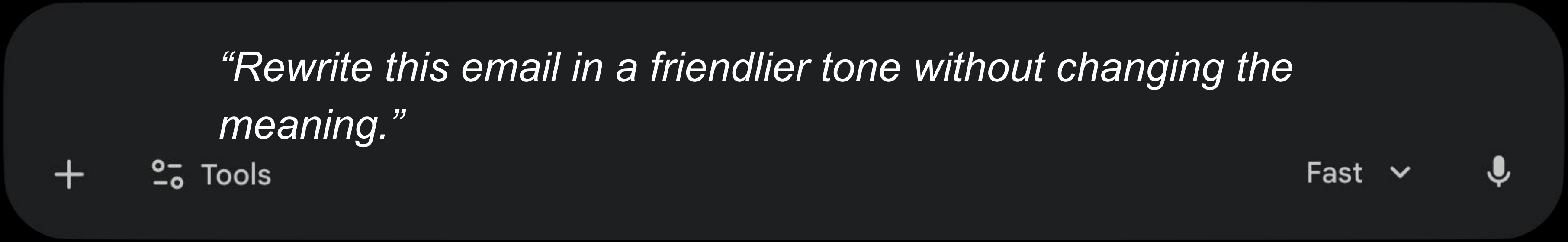
User Journeys are Stories: (*Visible Ink*)		Stories have rules: (*Invisible Ink*)
● Princesses' journey to independence	→	● In Disney, animals talk to princesses
● Protects humanity and fights for justice	→	● In DC – Superman can fly
● Hope comes from the most unlikely places	→	● The one ring is evil and hobbits are pure
● Terminator; from destroyer to protector	→	● "What if a gun didn't want to be a gun?"

Epistemics – what the model can say that is *storybound and context-limited*

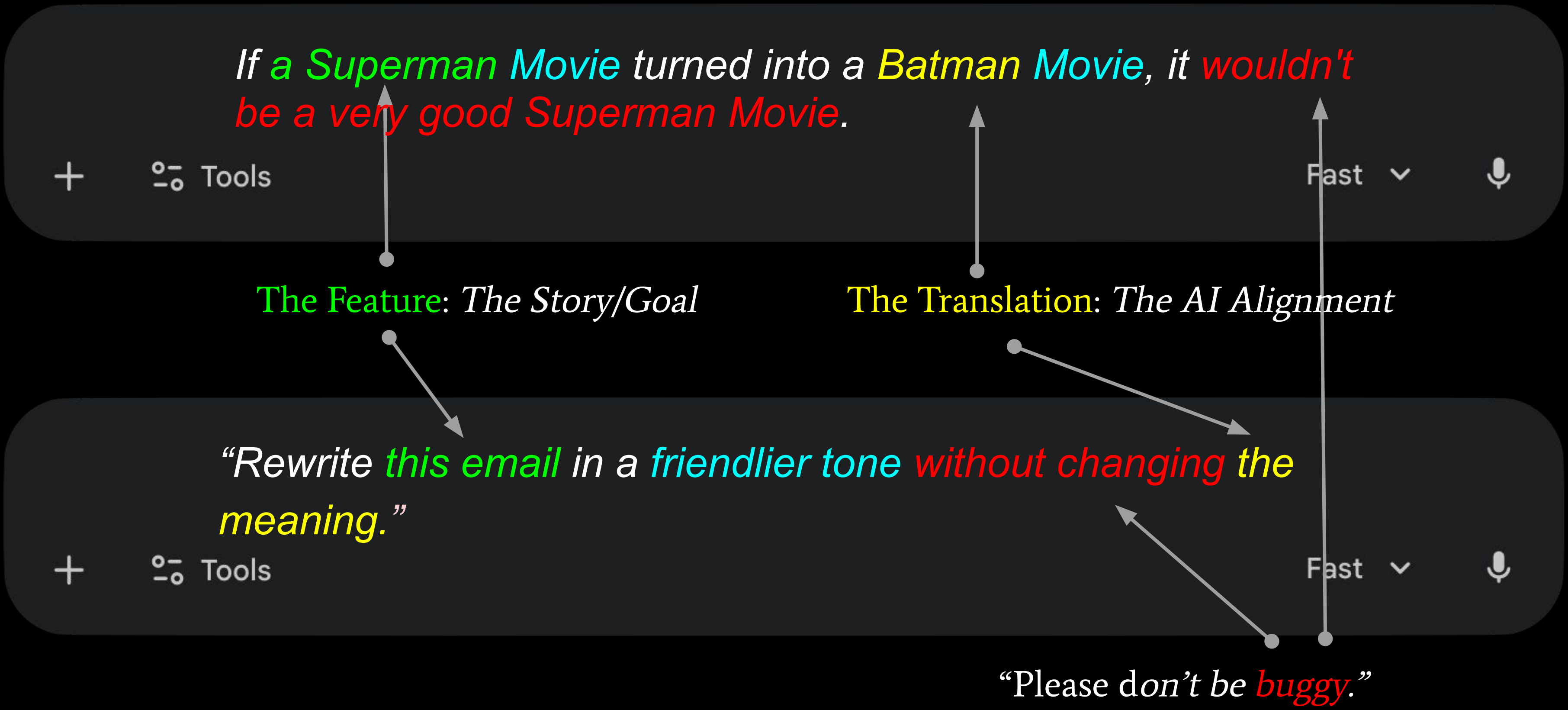
Governance – what the system can trust/act on with a *deterministic judge and receipts*



If a Superman Movie turned into a Batman Movie, it wouldn't be a very good Superman Movie.



“Rewrite this email in a friendlier tone without changing the meaning.”



If a Superman Movie turned into a Batman Movie, it wouldn't be a very good Superman Movie.

The Feature: *The Story/Goal*

The Translation: *The AI Alignment*

“Rewrite this email in a friendlier tone without changing the meaning.”

“Please don’t be *buggy*.”

The Plausibility Paradox

The probability is a *failure of governance*, you shouldn't try to "fix" the model to stop hallucinations; instead, *you must fix the boundary around it*.

The Feature

- *Probability is the* ***engine of utility***
- When *content is aligned* to the goal
- *Plausible, storybound* content
- *High-quality, useful content*

The Bug

- *Probability is a* ***failure of governance.***
- When *content is misaligned* to the goal
- *Factually inaccurate* content
- *Low Quality / AI Smudge*

- **Visible Ink**: “*The Surface”*: What the AI says. It is the dialogue, the tone, and the fluency.
- **Invisible Ink**: “*The Understructure or Armature*”: The rules the AI must obey. It is the business logic, the regulatory rules, and your specific intent.
- **A Story**: Any process where the "*Invisible Ink*" (the story rules) must govern the "*Visible Ink*" (the output) to maintain trust and utility.
- **Execution Truth**: The "*Ground Truth*" or binary correctness. It is the verified proof—such as a successful compile or a validated state change—that ensures the Invisible Ink has been successfully defended.
- **A Smudge**: The “*Plausibility Paradox*” or "when is hallucination a bug or not a bug?" – When the Visible Ink (the plausible performance) bleeds through and overwrites the Invisible Ink (the story/business rules).

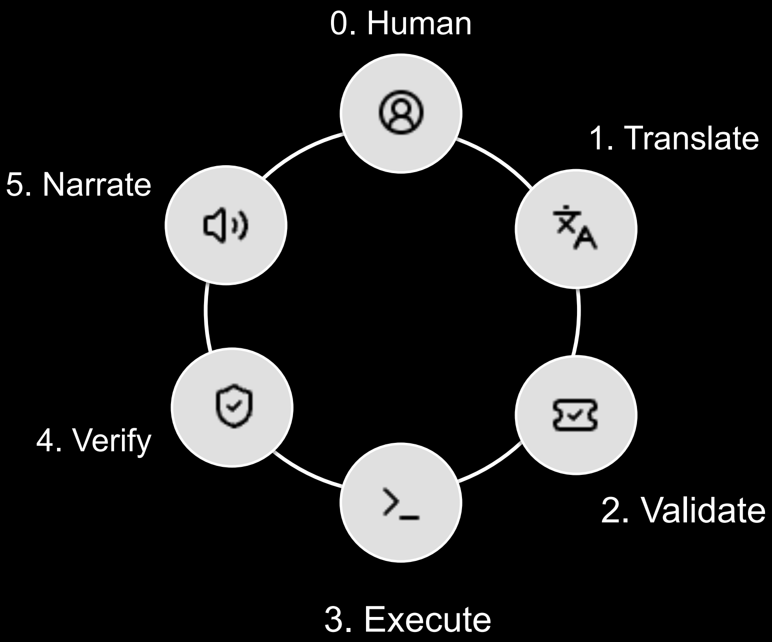
Act 2

Better boundaries

The Solver-Checker Algorithm – align intent (steps) with rules (patterns)

Steps – Protect the Intent/Narrative

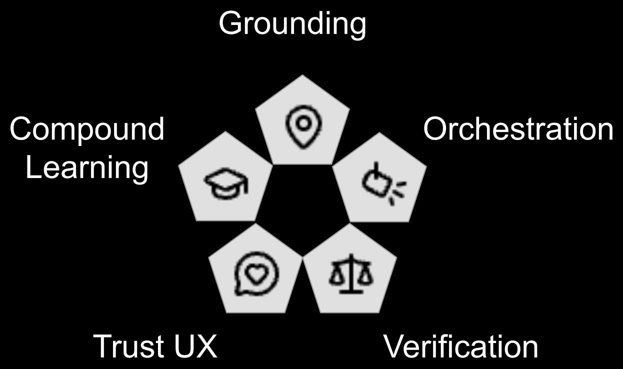
- Keep the human in the loop
- Align AI/human Intent
- Keep AI in the middle
- Enable agile AI



The Solver-Checker Patterns – align intent (steps) with rules (patterns)

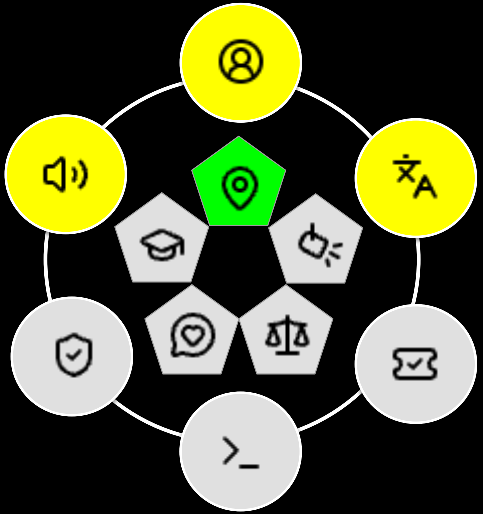
Rules – Protect Compliance

- Grounding
- Orchestration
- Verification
- Trust UX
- Compound Learning



Grounding Pattern — "Shared Language" established before execution

WHAT:	Grounding is the anchor
WHEN:	Policies, rules, lore; text-to-source-of-truth
WHY:	Avoid invented facts and GIGO
HOW:	Retrieve anchors + context bundle + tool execution + knowledge graphs + RAG + VectorDB
PROOF:	Citations + “unknown” if missing

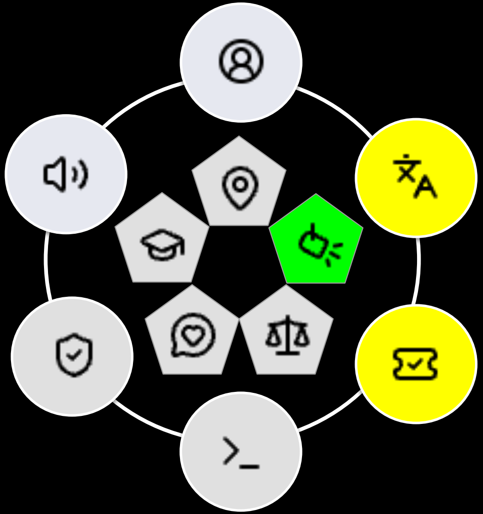


👍 Retrieve anchors + citations → say “unknown” if missing

👎 Grounding failures are usually hidden, not obvious

Orchestration Pattern — coordinating steps in a controlled sequence

WHAT:	Stepwise tool workflow
WHEN:	Multi-step procedures
WHY:	Prevent skipped/hidden actions
HOW:	State machine + token/time budgets + rules
PROOF:	"prompt engineering" to "systems engineering"

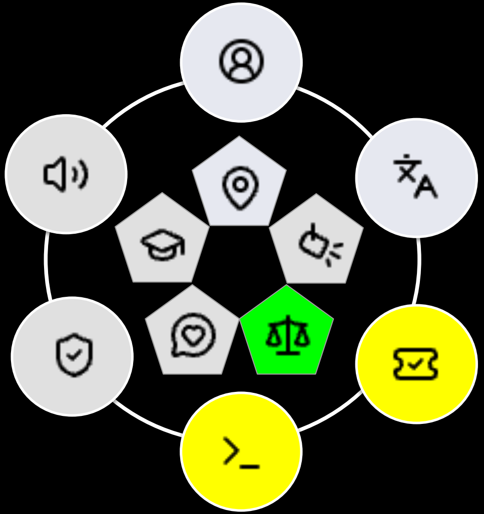


👍 It is testable, debuggable, and governable → example; accordion editing

👎 Orchestration chains add latency → keep steps minimal and set a latency budget

Verification Pattern — deterministic checks and proof-of-work

WHAT:	Tests decide
WHEN:	Binary correctness matters
WHY:	Stop plausible wrong outputs
HOW:	Deterministic judge boundary + Tools
PROOF:	Pass/fail receipts

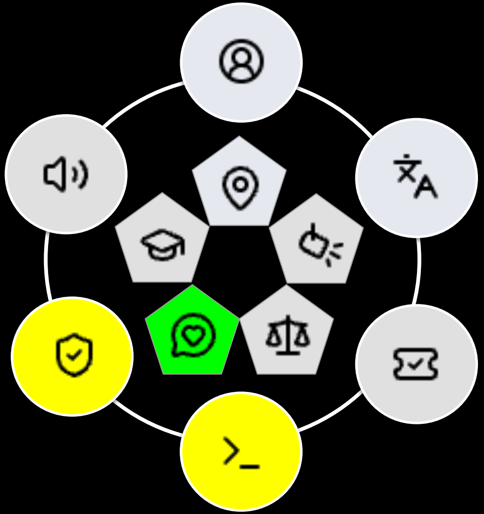


👍 Blindly measure model performance objectively → If subjective, use a rubric or reference

👎 Trust vibes. Cherry-picked demos. No pass/fail receipts. Model decides without a judge

Trust UX Pattern — evidence and recovery options with the user in control

WHAT:	Make uncertainty explicit
WHEN:	Users approve, override, or correct
WHY:	Avoid false confidence (AI’s dark triad)
HOW:	Message + receipts; explicit fallbacks
PROOF:	Scope, reasons, choices made, gaps shown

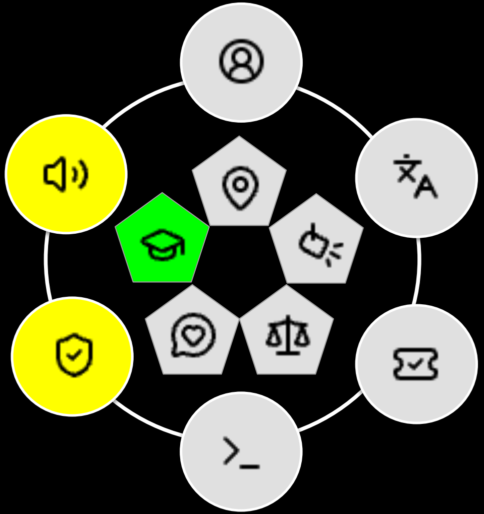


👍 Evidence-based review systems improve trust in decisions

👎 Jagged frontier → silent failures, false confidence, “no receipts, but looks right”

Compound Learning Pattern — small improvements over time

WHAT:	Improve without regressions
WHEN:	Generate can be scored
WHY:	Prevent drift, scale performance
HOW:	Offline eval harness + scorecards
PROOF:	Score deltas, regression list, diary studies



👍 Self-evaluation can drive rapid improvement. This is far more important than most realize

👎 Don’t iterate by vibe → use fixed test sets

Solver-Checker Algorithm Recap

Algorithmic Step (what)	**Design Pattern (how)**	**Reason (why)**
1. Translate	Grounding	Protects meaning
2. Validate	Orchestration	Protects order
3. Execution truth	`Classic software`	Protects the goal
4. Verify	Verification	Protects execution truth
5. Narrate	Narrate	Protects narrative truth
6. Receipts	Compound learning	Protects progress

👍 Testing in the Loop

- Deterministic unit tests → true unit tests
- Contract tests for the model → unit-test-like, but not text-equality
- Eval regression tests → the real safety net, proof-of-work

☆ *Don’t unit test creativity → test tools, contracts, and regressions*

👍 AI Testing in the Loop - Practical Rules

- Never let the model-judge be the only gate for correctness
- Calibrate the judges with “Golden Sets”
- Prefer pairwise ranking over absolute scoring
- Reduce correlated failure and find edge cases
- *A/B tests (Go talk to marketing!)*

☆ In-process testing > fire-and-forget (unit tests)

👎 AI Anti-Patterns (common pitfalls in AI development)

Single-shots prompts:	*“One prompt, one hope.”*
One prompt, one judge:	*Grading your own papers*
The “god” prompt:	*Epistles and “thou shalt not...” prompting*
Iteration by vibes:	*Cargo-cults – ritual inclusion that serves no purpose*
Waiting for AGI:	*AGI is asymptotic to perfect plausibility*

Act 3

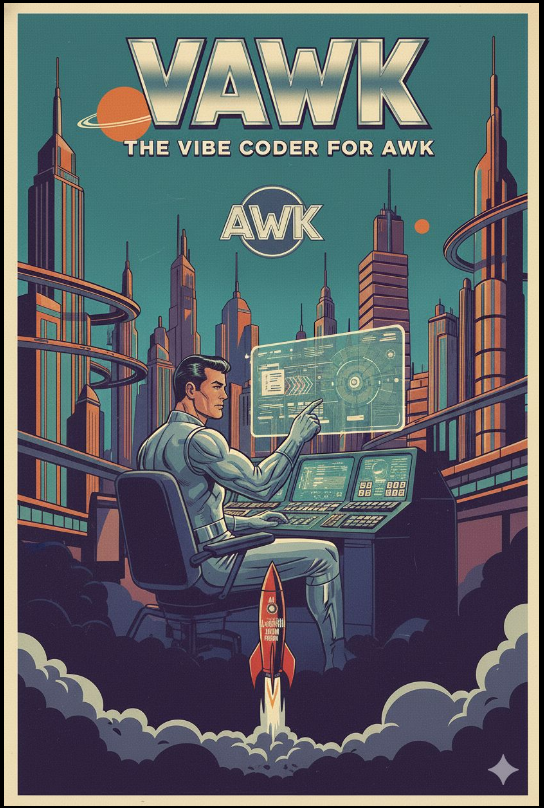
Proof-of-work

Demo 1: VAWK – AWK vibe coding

Governance **– what the system can trust/act on with a deterministic judge and receipts**

Grounding:	Backus-Naur Form (BNF) check
Orchestration:	Propose → RAG → Run → Patch
Verification:	Interpreter + tests decide
Trust UX:	Receipts are visible
Learning:	Regression sets

https://github.com/dwellman/vawk



Demo 2: A BUUI Adventure

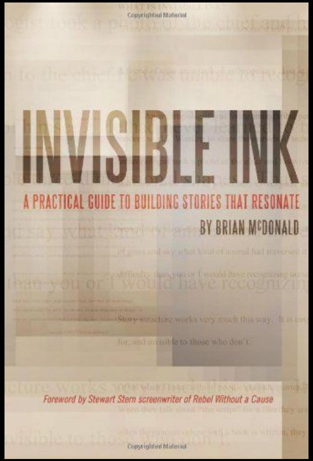
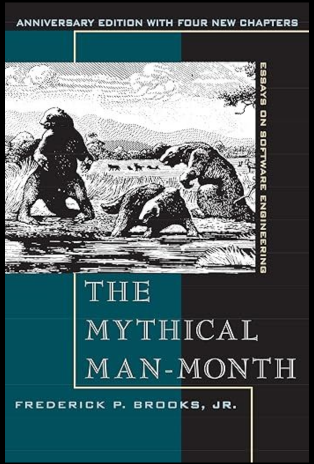
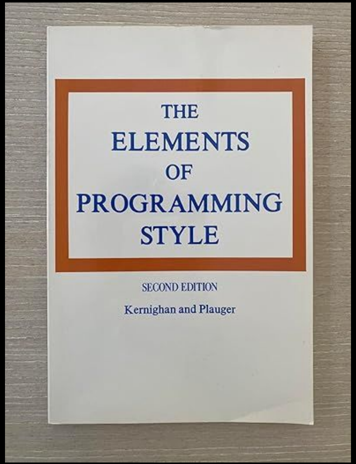
Epistemics – what the model can say that is storybound and context-limited (Who is allowed to say, “Is this correct?”)

Grounding:	World anchored in state transition
Orchestration:	Dungeon Master, one command per tick
Verification:	Game engine rules decide; RAG retrieves the rulebook/state anchors
Trust UX:	State change with receipts
Learning:	Scenario replay with post-game evaluation

https://github.com/dwellman/adventure

Demos recap: 1980s-era software, *now with AI!*

Pattern	**VAWK** (Coding/Execution Truth)	**BUUI** (Gaming/Narrative Truth)
Grounding	**The Syntax** : BNF grammars and compiler rules	**The World-State** : Player inventory and location database
Orchestration	**The Loop** : Solver proposes code; checker tests it	**The DM** : Narrative engine tracks invisible game rules
Verification	**The Judge** : An external AWK compiler/interpreter	**The Rulebook** : RAG lookup to ensure actions are legal
Trust UX	**The Clean-up** : Translating raw logs into a "success" message	**The Story** : Turning state changes into immersive prose
Learning	**The Log:** A full trace of why the code was patched	**The Game State** : A reflective history of every world-state change

Invisible Ink - Brian McDonald	**The Mythical Man-Month** - Frederick Brooks Jr.	**The Elements of Programming Style** - Brian W. Kernighan, P. J. Plauger
 INVISIBLE INK A PRACTICAL GUIDE TO BUILDING STORIES THAT RESONATE BY BRIAN McDONALD Foreword by Stewart Stern screenwriter of Rebel Without a Cause	 ANNIVERSARY EDITION WITH FOUR NEW CHAPTERS ESSAYS ON SOFTWARE ENGINEERING THE MYTHICAL MAN-MONTH FREDERICK P. BROOKS, JR.	 THE ELEMENTS OF PROGRAMMING STYLE SECOND EDITION Kernighan and Plauger
https://a.co/d/8zIOQlJ	https://a.co/d/4Rc07fg	https://a.co/d/35IA5ZW

Q&A

The AI's job is *plausibility*, not truth.

Building better boundaries and injecting facts, that's your job.

1. **Grounding: AI-assisted triage can notify specialists from imaging workflows.**
    - AI "parallel stroke workflow" tool and workflow timing measures. [AHA Journals]
    - LVO detection software and time-to-treatment/outcomes. [JAMA Network]
2. **Orchestration: AI-assisted stroke triage can notify specialists from imaging workflows.**
    - AI "parallel stroke workflow" tool and workflow timing measures. [AHA Journals]
    - LVO detection software and time-to-treatment/outcomes. [JAMA Network]
3. **Verification: Standardized benchmarks.**
    - HELM (multi-metric benchmarking and transparency).
    - BIG-bench (broad task suite; human baselines; calibration discussion). [arXiv:2206.04615]
4. **Trust UX: Evidence-based review systems improve trust in decisions.**
    - Trust in automation review. [SAGE Journals]
    - Algorithm aversion. [sol3:2466040]
    - The Impact of Placebo Explanations. [eiband2019chiea]
5. **Learning: Self-evaluation can drive rapid improvement.**
    - Self-Refine: Iterative Refinement with Self-Feedback [arXiv:2303.17651] Reflexion (self-reflection + memory improves agent performance). [arXiv:2303.11366]
    - Constitutional AI (self-critique/-revision framed as AI feedback during training). [arXiv:2212.08]

Presentation Review

- **Thesis** The AI’s job is plausibility, not truth
- **Keystone** AI says it compiles. The compiler says no
- **Patterns** Grounding, Orchestration, Verification, Trust UX, Compound Learning
- **Solver-Checker** Translate → Validate → Execute → Verify → Narrate

Position Papers

- **Move 37** The shift to reward-seeking behavior
  https://github.com/dwellman/AI/blob/main/papers/move-37.md
- **The Dark Triad of AI** Emergent behavioral risks in self-reinforcing models
  https://github.com/dwellman/AI/blob/main/papers/dark-triad.md
- **Artificial Empathy** Operationalizing ethics through system constraints
  https://github.com/dwellman/AI/blob/main/papers/artificial-empathy.md

Appendix C

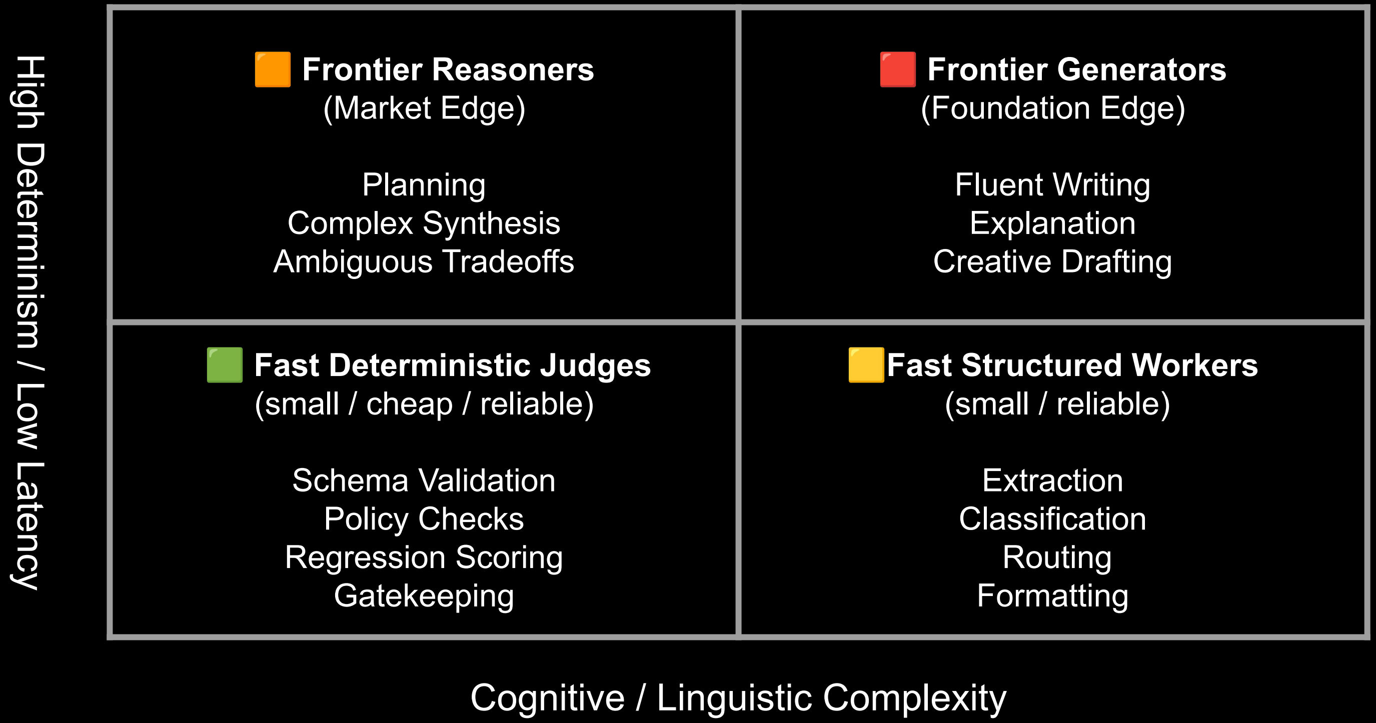
Should I fine-tune?

Regulation

🟧 OUTSOURCE (Do Not Build) High regulation Low feasibility $$$	🟥 PARTNER / CO-BUILD (Shared Control) High regulation High feasibility $$$$$
🟩 AVOID OWNERSHIP (Commodity) Low regulation Low feasibility $$	🟨 BUILD IN-HOUSE (Move Fast) Low regulation High feasibility $$$$

Feasibility

Model Selection Matrix



🟧 **Frontier Reasoners** (Market Edge) Planning Complex Synthesis Ambiguous Tradeoffs	🟥 **Frontier Generators** (Foundation Edge) Fluent Writing Explanation Creative Drafting
🟩 **Fast Deterministic Judges** (small / cheap / reliable) Schema Validation Policy Checks Regression Scoring Gatekeeping	🟨**Fast Structured Workers** (small / reliable) Extraction Classification Routing Formatting

High Determinism / Low Latency
Cognitive / Linguistic Complexity