

# 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.”*



Tools

Fast ▾



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.”*



Tools

Fast ▾



*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."*



Tools

Fast ▾



*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.*”

+ Tools

Fast ▾



Invisible Ink: *How* – rules & 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.*”

+ Tools

Fast ▾



“Please don’t be *buggy.*”

Invisible Ink: *How* – rules & 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*)

- Sign up & check out
- Quarterly report
- Automating customer service
- Analyzing large datasets to find trends

Stories have rules: (*Invisible Ink*)

- ● Minimize user effort & build trust.
- ● Requirements from regulatory bodies.
- ● A seamless, omnichannel experience.
- ● 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*)

- Princesses' journey to independence
- Protects humanity and fights for justice
- Hope comes from the most unlikely places
- Terminator; from destroyer to protector

Stories have rules: (*Invisible Ink*)

- • In Disney, animals talk to princesses.
- • In DC – Superman can fly.
- • The one ring is evil and hobbits are pure.
- • 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.*



Tools

Fast ▾



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



Tools

Fast ▾



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

+ Tools

The Feature: *The Story/Goal*

The Translation: *The AI Alignment*

Fast ▾



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

+ Tools

*"Please don't be buggy."*

Fast ▾



# 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).

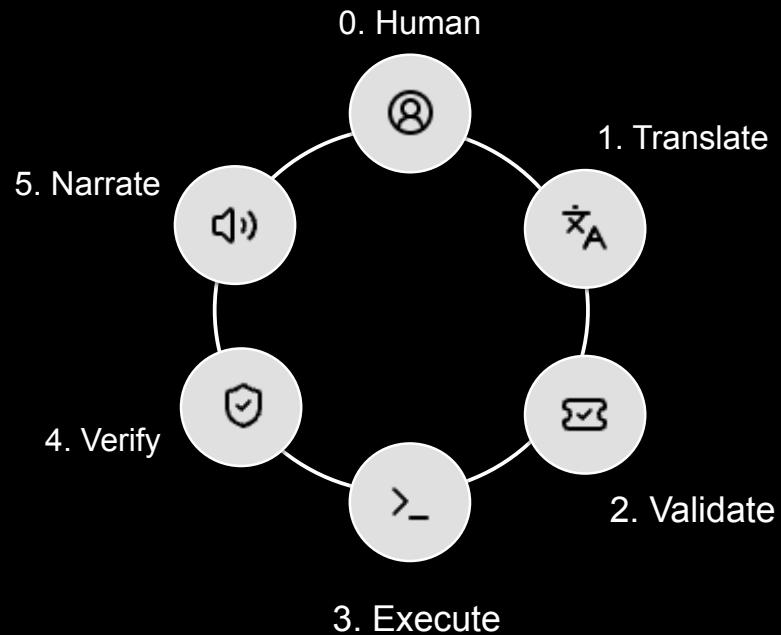
# Act 2

## Better boundaries

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

## Steps – Protect the Intent/Narrative

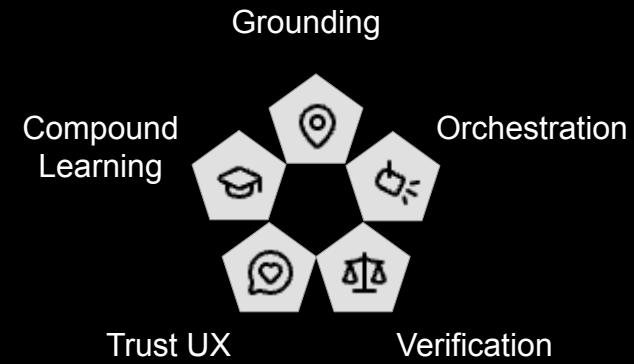
- ✓ Keep the human in the loop
- ✓ Align AI/human Intent
- ✓ Keep AI in the middle
- ✓ Enable agile AI



# 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

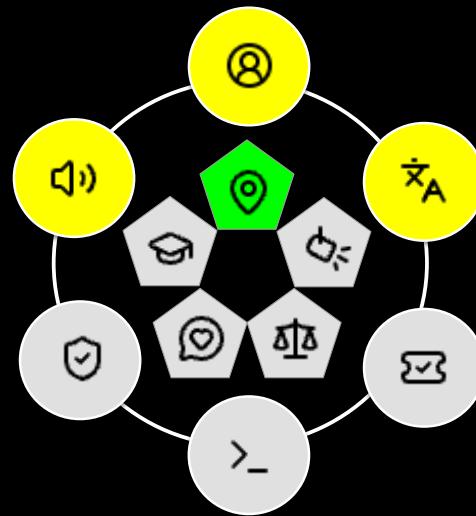
WHY: Avoid invented facts and GIGO

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

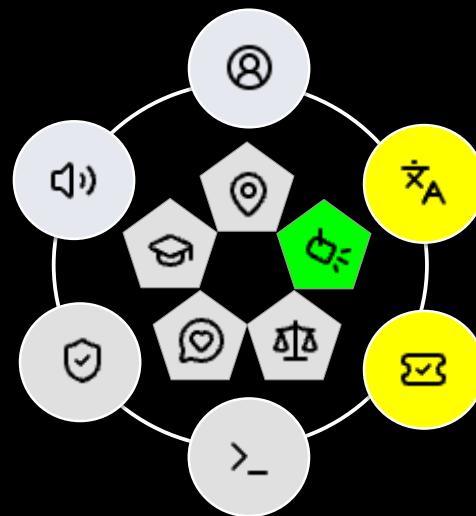
WHAT: Stepwise tool workflow

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

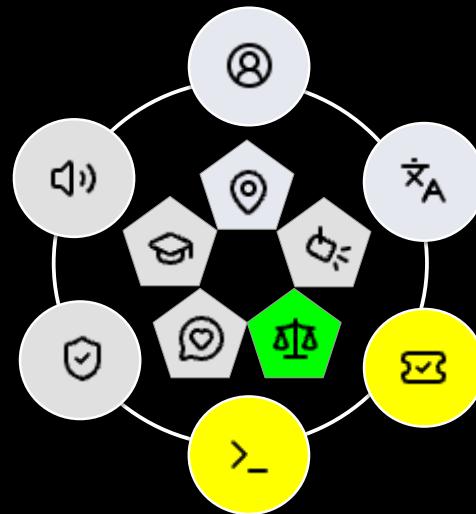
WHAT: Tests decide

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

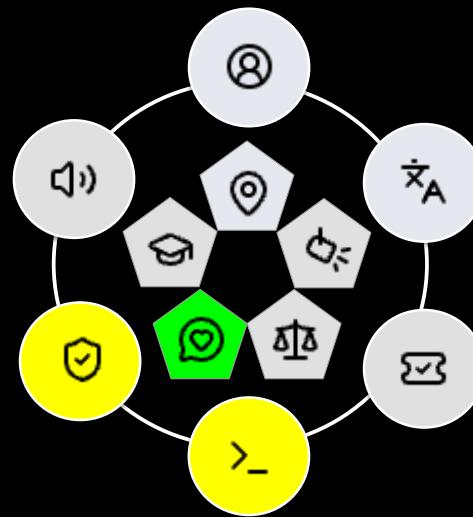
WHAT: Make uncertainty explicit

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

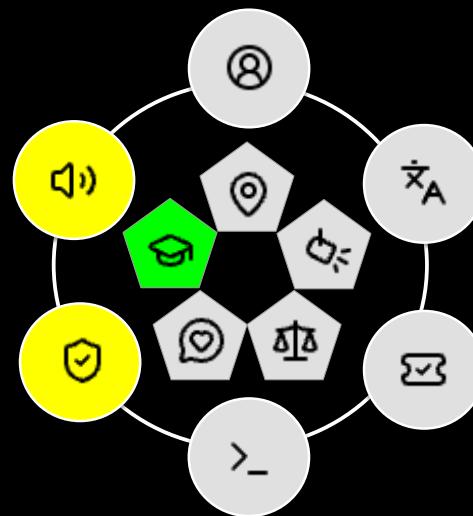
WHAT: Improve without regressions

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
<b>3. <i>Execution truth</i></b>	<b>Classic Software</b>	<b><i>Protects the goal</i></b>
4. Verify	Verification	Protects execution truth
5. Narrate	Narrate	Protects narrative truth
<b>6. <i>Receipts</i></b>	<b>Compound Learning</b>	<b><i>Protects progress</i></b>

# 👍 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*

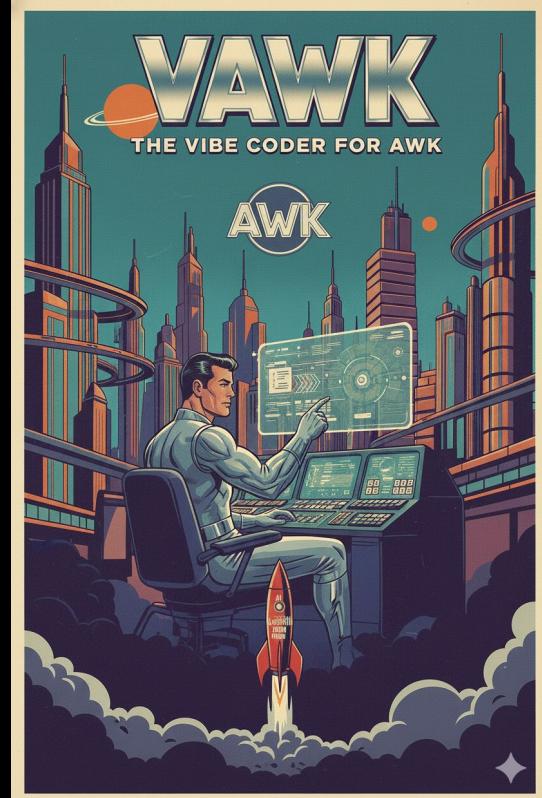
# Demo 1: VAWK – AWK vibe coding

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

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



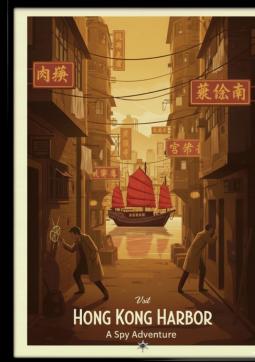
<https://github.com/dwellman/vawk>



# Demo 2: A BUUI Adventure

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

- 1. Grounding:** World anchored in state transition
- 2. Orchestration:** Dungeon Master, one command per tick
- 3. Verification:** Game engine rules decide; RAG retrieves the rulebook/state anchors.
- 4. Trust UX:** State change with receipts
- 5. Learning:** Scenario replay with eval



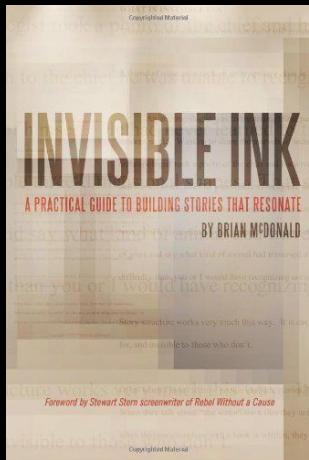
<https://github.com/dwellman/adventure>

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

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

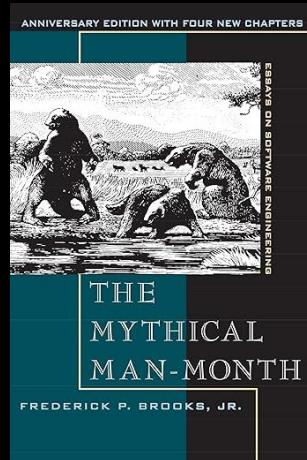
## Invisible Ink:

- Brian McDonald



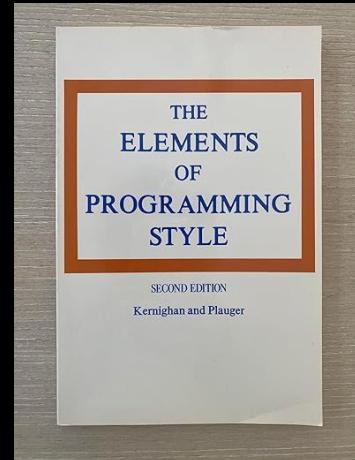
## The Mythical Man-Month:

- Frederick Brooks Jr.



## The Elements of Programming Style

- Brian W. Kernighan, P. J. Plauger



<https://a.co/d/8zIOQIJ>



<https://a.co/d/4Rc07fg>



<https://a.co/d/35IA5ZW>

# Q&A

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

Building the boundary and injecting the 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](#)]

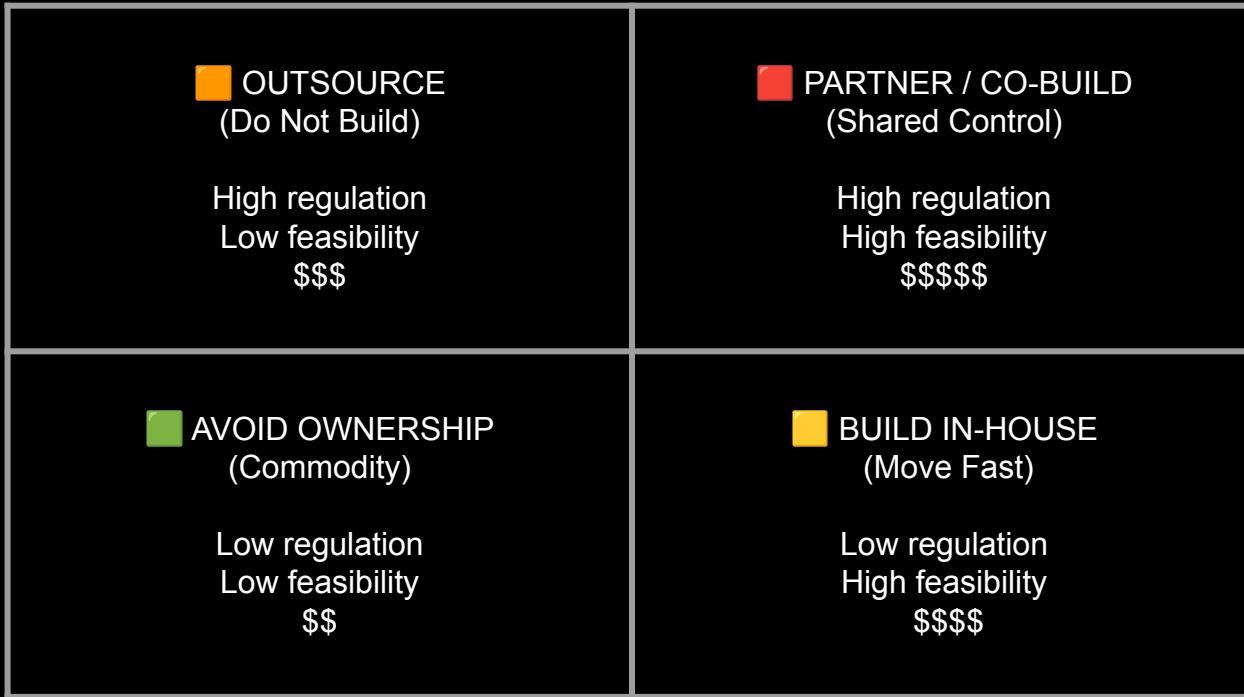
## Presentation Review

- **Thesis** The AI's job is plausibility, not truth.
- **Keystone** AI says it compiles. The compiler says no.
- **Patterns** 1. Grounding, 2. Orchestration, 3. Verification, 4. Trust UX, 5. 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>

# Should I fine-tune?



# Model Selection Matrix

High Determinism / Low Latency

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

Cognitive / Linguistic Complexity