A HITPOINT FRAMEWORK

The Lightest Mechanism Principle:

Most enterprise AI initiatives fail in the same way: leaders reach for the heaviest mechanism — autonomous agents — to solve problems that a well-designed rule or a one-line API call would handle more reliably, more cheaply, and more auditably. This paper presents a five-layer decision stack that operations leaders can use to choose the lightest mechanism that meets the requirement.

EXECUTIVE TAKEAWAY

For every workflow step, start at the bottom of the stack — **Rules** — and move up only when the layer below provably cannot do the job. The cost, latency, failure surface, and audit burden roughly **10× at each layer**. Agents are the right answer for ambiguity — and a wrong answer for almost everything else.

1 The Shift: why pilots stall at production

The pattern we see across SAP Concur, Salesforce, and Lark deployments in ANZ and Greater China is consistent. Pilots demo well; production exposes three failure modes.

~70%

PILOTS THAT STALL

of enterprise GenAI pilots fail to reach measurable production impact — driven primarily by misfit between mechanism and problem, not model quality.

Industry surveys, 2024–2025

10×

COST GAP PER LAYER

A determinstic rule costs ~1¢ per execution. An agent loop with tool calls and review can run 10–100× higher per case — before audit overhead.

Internal Hitpoint benchmarks

3×

AUDIT SURFACE

Each layer added above a workflow roughly triples the number of decision points an internal auditor must reason about during quarterly review.

Hitpoint client engagements

2 The Principle

For each workflow step, choose the lightest mechanism that meets the requirement. A mechanism is "lighter" when it is more deterministic, more inspectable, cheaper per execution, and faster to fix when it breaks. The counter-intuitive consequence: the most defensible AI architectures use the smallest amount of AI possible.

3 The Decision Stack

LAYER 1	LAYER 2	LAYER 3	LAYER 4	LAYER 5
Rules	**Automation**	**LLM Call**	**Agent**	**Human Review**
Deterministic. Policy-encoded.	Repeatable. Multi-step, no judgment.	Language-heavy. Single-turn judgment.	Ambiguous. Multi-step + tools.	Accountable. Risk gate.

← LIGHTER · CHEAPER · MORE INSPECTABLE → HEAVIER · MORE FLEXIBLE · HIGHER AUDIT COST →

Operating rule: Audit Trail is not a layer — it runs through every layer. If a step cannot be logged, replayed, and explained, it does not belong in production.

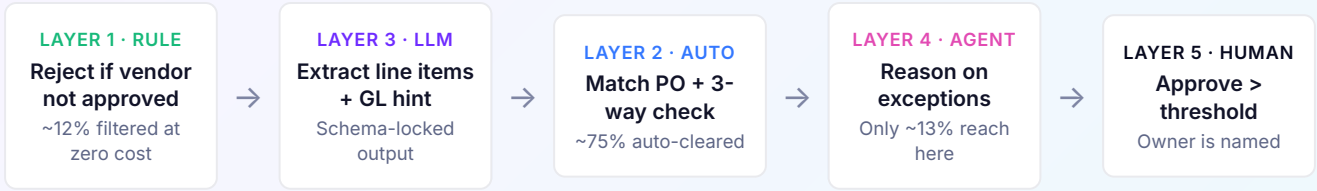
4 Applying the stack: fit, signals, and failure modes

For every workflow step, run three tests. **What does this layer do well? What signal tells me to use it? How does it fail when forced beyond its fit?**

LAYER	BEST FIT	SIGNAL TO USE IT	FAILURE MODE WHEN MISAPPLIED
Rules	Policy thresholds, eligibility, hard limits, format checks.	The decision can be written as an "if/then" a finance manager would sign off on.	*None — rules don't fail when in scope. They just feel "boring" and get over-engineered.*
Automation	Stitching across systems, scheduled jobs, retries, idempotent flows.	The sequence is fixed; only the data varies.	*Brittle on edge cases the original designer didn't anticipate.*
LLM Call	Extraction, classification, summarisation, translation, light reasoning.	Input is unstructured language; output shape is known.	*Hallucinated structure when prompted to "decide" rather than "extract."*
Agent	Genuinely ambiguous cases requiring tool use, multi-step planning, or context-fetching.	You cannot pre-specify the steps — only the goal and the boundaries.	*Runaway loops, opaque reasoning chains, costs that escape the budget.*
Human Review	Accountability points, high-blast-radius decisions, regulated approvals.	A regulator, auditor, or executive must be able to name the human owner.	*Used as a safety blanket on every step — kills throughput, trains reviewers to rubber-stamp.*

5 Worked example: invoice-to-approval, one workflow across the stack

A 30,000-invoice/year finance operation ANONYMISED CLIENT PATTERN



The agent layer handles only the residual ambiguity — not the whole pipeline. This is what makes the architecture defensible at audit and stable in production.

6 Five questions for your next AI workflow review

1. **What is the lightest layer** that could handle this step? Have we proven the lighter layer fails before moving up?
2. **Where does the policy live?** If it lives only in a prompt, it is not a policy — it is a wish.
3. **What is the audit story?** Can we replay any single decision and name the mechanism that produced it?
4. **Who is the human owner** at each risk gate, and what is the SLA before review becomes rubber-stamping?
5. **What is the unit cost** per case at each layer, and what is the blast radius when the layer is wrong?
- ★ **Bonus:** if we removed the AI layer entirely, what would the workflow still do correctly? That residual is the foundation.