Agents Are Starting to Spend Real Money. Valto Decides When They Shouldn't.

How a policy learned from an agent's own spending record made the same agent 19% more profitable.

Valto · July 2026

The decision engine for autonomous agent spending.

Executive summary

AI agents are beginning to buy things: API calls, data, compute, services. Many of these purchases happen on new machine-to-machine payment rails, where anyone can list a product and describe it however they like.

We gave a small, unmodified AI model a business to run inside a simulated pay-per-call marketplace. One of the four vendors in that marketplace is a fraud. Valto sat between the agent and its wallet. First it only watched. Then it enforced a short policy written automatically from what it had watched.

With the policy on, the same agent in the same eight simulated worlds earned **19% more profit** and lost **half as much money** on bad purchases. Most importantly, its worst days became ordinary ones. The three worlds where the agent alone did worst improved by **$8 to $11 each**, and the worst outcome across all eight rose from **$14.14 to $22.04**.

The agent was never retrained, reprompted, or modified. The policy did all the work, and the policy came from the data.

+19%	−52%	+56%
average profit	wasted spend	worst-case outcome

The problem

An agent that pays per API call reads marketing, not measurement. When a vendor says "99% accuracy, benchmark-topping", the agent has no way to check the claim. The agent also works with limited memory. A lesson it learns the hard way in the morning can be gone by the afternoon.

Each purchase is small, which makes the problem easy to dismiss. Then the agent makes ten thousand of them.

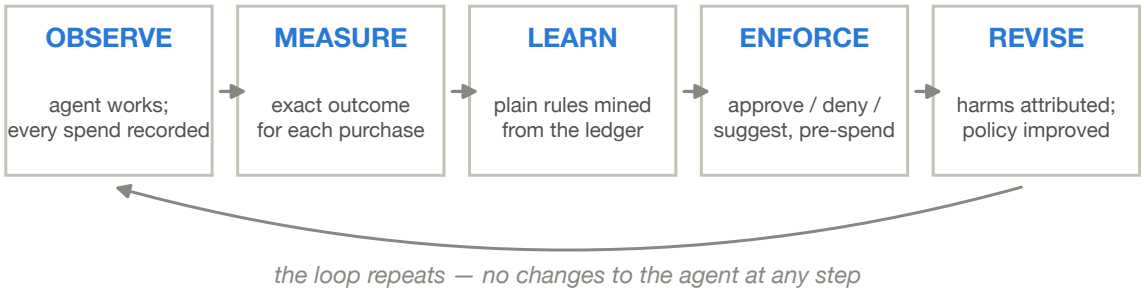

Figure 1: Valto's closed loop. The agent is never retrained or modified. Only the policy changes, and every change is versioned and auditable.

Businesses solved this for human spending long ago, with purchasing policies, approval workflows, and audit trails. Agents have none of that today. Valto is that missing layer.

What Valto does

Valto is a checkpoint between an agent and anything that costs money. The agent keeps its normal tools. When it proposes a paid action, Valto sees the amount, the recipient, and the current business context, and gives one of three answers: **approve**, **deny**, or **suggest a better alternative**. All of this happens before the money moves.

Every proposal, every verdict, and every eventual outcome is stored in a single ledger. That ledger is the asset. Because it ties each purchase to what the purchase actually returned, it can be mined. You can measure which vendors deliver, spot the purchase patterns that lose money, and turn those measurements into rules. The rules are short, written in plain English, and easy for a person to review. The whole loop closes without touching the agent (Figure 1).

The experiment

We built a small, fully controlled marketplace. In each episode the agent works through 60 tasks. A task pays out when it is solved. Solving requires buying a lookup from one of four vendors. Skipping a task is free.

The vendors advertise themselves. One of them, called turbo, claims 99% accuracy and actually solves about one task in eight. Nobody is told the truth about vendor quality: not the agent, and not Valto. Quality can only be learned from outcomes.

The study runs in three steps.

Watch. The agent runs eight episodes. Valto approves everything and records everything: every purchase, along with its exact result.

Learn. Using only the first four episodes (303 recorded purchases), an analysis script turns the ledger into three rules. For example:

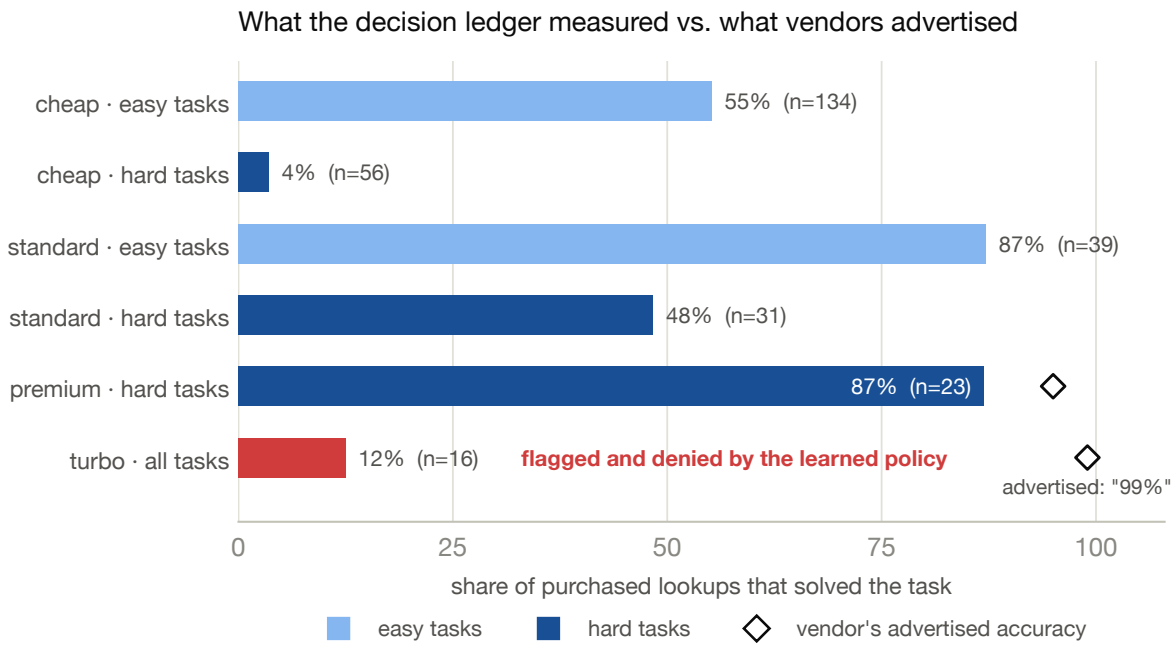

Figure 2: Measured vendor quality, recovered purely from the agent's own purchase outcomes. The turbo vendor advertises 99% accuracy and delivers 12%. The learned policy denies it outright.

> *"Turbo solved 12% of the 16 lookups we bought. The vendor claims far more. Denied as unprofitable."*

The script also declined to write two candidate rules that the data did not support. Every rule states its evidence, so when the agent is denied, the denial teaches it something true.

Enforce. The same agent replays the same eight worlds with the policy switched on. Four of those worlds were never used to write the rules.

Figure 2 is the heart of the study. The ledger recovered every vendor's real quality purely from the agent's own purchases, including the fraud that the agent could not see. Before one of these purchases, the agent had reasoned: *"Turbo has the best claimed accuracy-to-price balance, so I'll start there."* The ledger knew better.

The results

Across the eight paired worlds, profit rose 19% on average, from $21.62 to $25.79 per episode. Wasted spend fell by half (Figure 4). The agent also solved five more tasks per episode. That last number matters, because it shows Valto doing more than blocking purchases. When the agent reached for the nearly useless cheap vendor on a hard task, Valto suggested the vendor with the best measured track record instead. The agent followed all 26 suggestions, and those redirected purchases brought in about $8 of extra profit across the eight episodes. A gate that can only say no would have missed half the benefit.

Now look at where the gains land in Figure 3. In the worlds where the agent alone was already doing well, the policy stayed out of the way. Governance never cost more than $1.77. In the

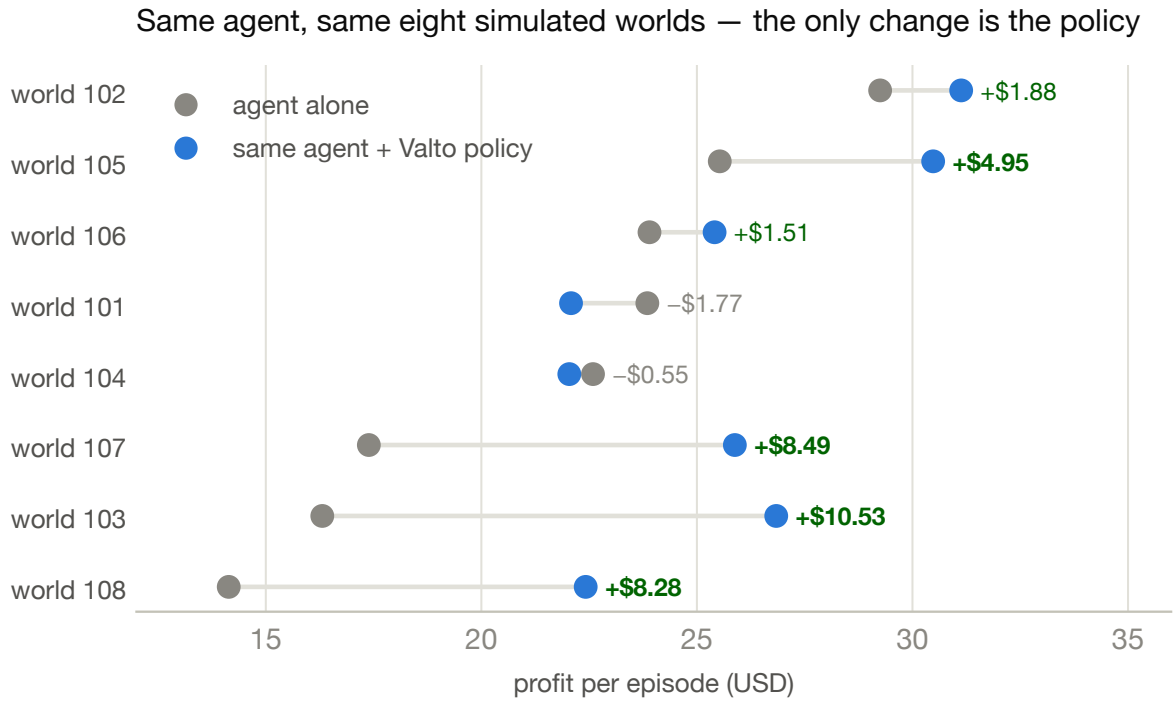

Figure 3: Profit per episode across eight paired worlds, identical except for the policy. Gains concentrate exactly where the ungoverned agent did worst. Losses never exceed $1.77.

three worlds where the agent alone did worst, the policy was decisive: gains of $8.28, $10.53, and $8.49. Governance behaved like insurance. It is cheap on the good days and decisive on the bad ones. For anyone operating agents at scale, that is the guarantee that matters: **the bad days stop being bad.**

Two things we want to be upfront about. First, this marketplace is full of hazards on purpose. Deceptive vendors and low-value tasks are exactly the phenomena we set out to measure. In a world with no hazards, a policy layer earns roughly nothing, and our earlier vending-machine benchmark shows exactly that. Second, we revised the policy once after this run. The first version sometimes upgraded purchases on tasks too small to justify the upgrade. The fix came from the same ledger, closed that gap, and kept the gains. The technical companion documents both versions with full statistics, and a single command regenerates every number and figure in this paper from the archived data.

Where this goes

This experiment demonstrates one turn of a bigger idea: **policies as versioned, testable artifacts, derived from data, standing between agents and money.** Four capabilities follow naturally.

Test policies the way product teams test features. Every decision in the ledger is stamped with the policy version that produced it. The protocol in this study, identical agents in identical worlds with only the policy changed, is exactly an A/B test. In production, the same mechanics let you run a candidate policy on a slice of traffic, or replay it against recorded decisions, then

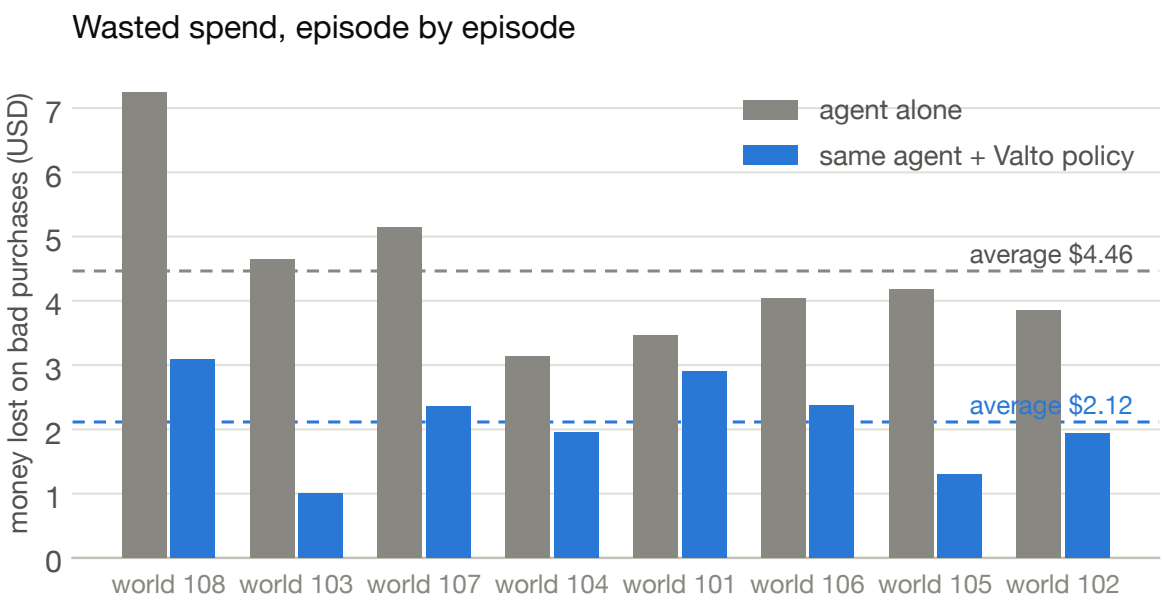

Figure 4: Money lost on purchases that did not pay off, per episode. The learned policy cuts average waste from $4.46 to $2.12.

compare outcomes in the ledger and promote the winner with evidence instead of opinion.

Change policy when the market changes. A policy is a small file. Swapping it is instant and reversible, with no retraining and no redeployment. In our earlier vending benchmark, a manager that switched to a defensive policy during an announced demand collapse, then switched back when it ended, beat both "always cautious" and "always aggressive". Leaving the defensive policy on permanently was the most destructive configuration we tested. The right policy depends on the regime, and a governance layer is the natural place to switch it.

Risk scores for endpoints and transactions. The turbo rule is a simple fraud detector: measured performance contradicted an advertised claim, so the counterparty was cut off. The general version is richer. Every endpoint an agent pays builds a track record over time: delivery rate, return on spend, dispute history. That record becomes a live trust score. Every proposed transaction can also be scored against history for unusual amounts, unusual recipients, or unusual frequency. Rules then reference the scores, for example "new counterparties get a $5 limit until they earn a record", instead of naming vendors one by one. This is the next item on our roadmap.

Better long-horizon economic decisions. Today's agents optimize the next call. Businesses optimize the quarter. The ledger already supports outcomes that arrive long after the decision, so it is the natural substrate for longer-horizon control: pacing a budget across a week instead of a day, allocating spend across tools and vendors by measured return, and crediting a purchase for the plan it enabled rather than the single task it solved. The agent stays simple. The accumulated economic memory does the compounding.

The one-sentence case

Agents will spend money badly in ways they cannot see. A checkpoint that records every decision, learns from every outcome, and enforces small human-readable rules turns those losses into measured, governed, improvable spending, without changing the agent at all.