

A guide for forward-thinking AI platform leaders

Build vs Buy

What it takes to build an agent trust platform yourself, and what it costs compared with buying one.

Contents

1	Executive summary	3
	The headline numbers, and the company we used to model them	
2	Build first, buy when it breaks	4
	When building your own makes sense, and the point where it usually stops	
3	The requirements of an agent trust platform	5
	The three parts of the platform, and where each one gets hard to build	
4	What it takes to build one yourself	7
	Where AI coding tools help, the build timeline and the first-year budget	
5	Weighing the final decision	10
	Three-year costs side by side, what teams who built their own tell us, and how to make the call	

Executive summary

This guide estimates what it costs to build an agent trust platform in house, and compares that with buying one.

- Your team now answers for two things in production: the data in your warehouse and the agents reading from it.
- Your agents read that data, act on it and hand work off to other agents. An agent trust platform covers the whole chain: the data, what each agent does with it, and platform-native agents that find and fix problems in both and improve how your agents work, so nobody writes rules by hand.
- AI coding tools speed up year one, because year one is mostly code. They do little for year two, which goes to tuning baselines, cutting alert noise and building the agents that fix things.
- Tuning runs on a record of which alerts your team acted on, and only time builds that record.

Methodology

Every figure in this guide is modeled on a composite company, not a specific customer. All roles are costed at fully loaded rates. Your own costs will vary with salaries, table count and agent volume.

\$6B	15,000	20	30,000	Mid-sized
Annual revenue	Employees	Data engineers	Tables	Agent program

The numbers at a glance

\$5.14M to build, over three years	~\$0.98M to buy, over the same three years	4 weeks until you're live, with agents triaging and fixing issues
~24 months for a build to match what you can buy	13+ years of full-time engineering effort to get there	30-40% of the build cost, every year, to keep it running

Build first, buy when it breaks

Most engineering teams approach large technology projects the same way: build first, then buy when it breaks. For many infrastructure decisions, that's a sensible approach. You learn what you need by building, and early problems rarely justify a vendor relationship.

AI coding tools have made building look even easier. But building, deploying and maintaining specialized software is harder than it looks, and running checks on general-purpose models gets expensive as your agent program grows.

When building makes sense

- You have one or two agents in production, or you're still experimenting
- You're iterating on prompts, tools and architecture, and want to learn what you need
- You can start right away, with no procurement process
- Token volumes are small, so the running cost stays manageable
- Open-source tracing tools give you enough visibility for now

When it stops making sense

- Token costs start showing up in budget reviews
- Maintaining the tooling competes with shipping new agents and products
- Gaps in coverage cause incidents that take too long to investigate
- You don't have agent engineering expertise in house
- Agents start writing to production systems or acting on their own, so a failure costs more

~5 agents

the usual tipping point

In the teams we work with, the problems on the right tend to arrive together, at around the fifth agent in production. By the time a team notices all of them, it has usually spent months on a system it is about to replace.

The rest of this guide looks at what happens past that point: what an agent trust platform needs to do, what it costs to build one, and how that compares with buying.

The requirements of an agent trust platform

Before you can price a build, you need to know what you're building. Here's what the platform has to cover, and why it's a bigger job than it was two years ago.

How the build has changed since 2025

01

Agents act on your data now

They read it, act on it and hand work to other agents. Your team answers for all of it.

02

AI writes the code faster

Detection code ships faster than it used to. Tuning it to your data still takes months.

03

Every check now has a token bill

Checks built on general-purpose models charge you every time they run.

The three parts of an agent trust platform

An agent trust platform doesn't stop at spotting problems. It fixes them and improves your agents over time, working inside the stack you already build on.

Agentic operations

Agents that find, fix and improve

They profile your tables, set up checks, triage alerts and fix issues without anyone writing a rule.

Monitoring

Setup

Discovery

Triage

Performance

Troubleshooting

Remediation

Helpdesk

Context layer

Your data

- Freshness, volume and schema changes
- Column-level lineage
- A connector for every source
- Alert routing and ownership

Agent layer

Your agents

- Whether the agent got the right context
- What each step cost and how long it took
- Whether it used the right tools in the right order
- Whether the answer was any good

Covering data and agents takes different skills

The data side needs data engineers.

The agent side needs AI and ML engineers.

And each is hard to build for different reasons.

Your data

Breadth is the hard part

You only cover what you built

Every new table or column change needs a rule, and someone has to keep it working.

Connectors break every quarter

Snowflake, dbt and the rest ship changes. You patch each one by hand.

Thresholds need years of corrections

They improve by learning which alerts people acted on, and you only have your own history to learn from.

Your agents

Running cost is the hard part

Every check pays for the whole conversation

A chat that starts at 5,000 tokens is past 40,000 by turn ten.

So you check less

Most teams end up evaluating about one call in ten.

Nothing lines up when it breaks

Agents call other agents, so piecing together one incident takes 10 to 15 minutes.

Agentic operations is the hardest part to build

Agentic operations takes reactive work off your team, so it does the most for day-to-day operations. It's also the last thing a build gets to, and the one most data teams have the least experience with.

01

What it does

- Looks at every table and works out which ones matter
- Sets up checks without anyone writing a rule
- Handles the small alerts and passes the big ones on
- Knows when to stop and ask a person

02

Why builds stall

- It's agent engineering, and most data teams haven't built agents before
- There's no reference design to copy
- It comes last, so every earlier delay lands on it
- Without it, you have a dashboard and still fix incidents by hand

03

A second system to run

- An agent fleet running your operations is another AI system in production
- It can break in the same ways as your other agents
- So those agents need finding, fixing and improving, too
- Teams are only starting to budget for this

What it takes to build one yourself

With the scope defined, the next question is effort: how long the build takes, who you need on it, and what the first year costs.

Where AI coding tools help, and where they don't

AI coding tools speed up year one, because year one is mostly code. Year two is mostly tuning, and tuning runs on your own operating history.

What they speed up

Writing the detectors

Building the lineage graph

Scaffolding the connectors

Dashboards, alerts and ticket routing

What they don't

Tuning

Baselines that know your seasonality, instead of a threshold someone picked

Sensitivity per table

Learned from the alerts people acted on

Turning the noise down

Before the team stops looking

Orchestration

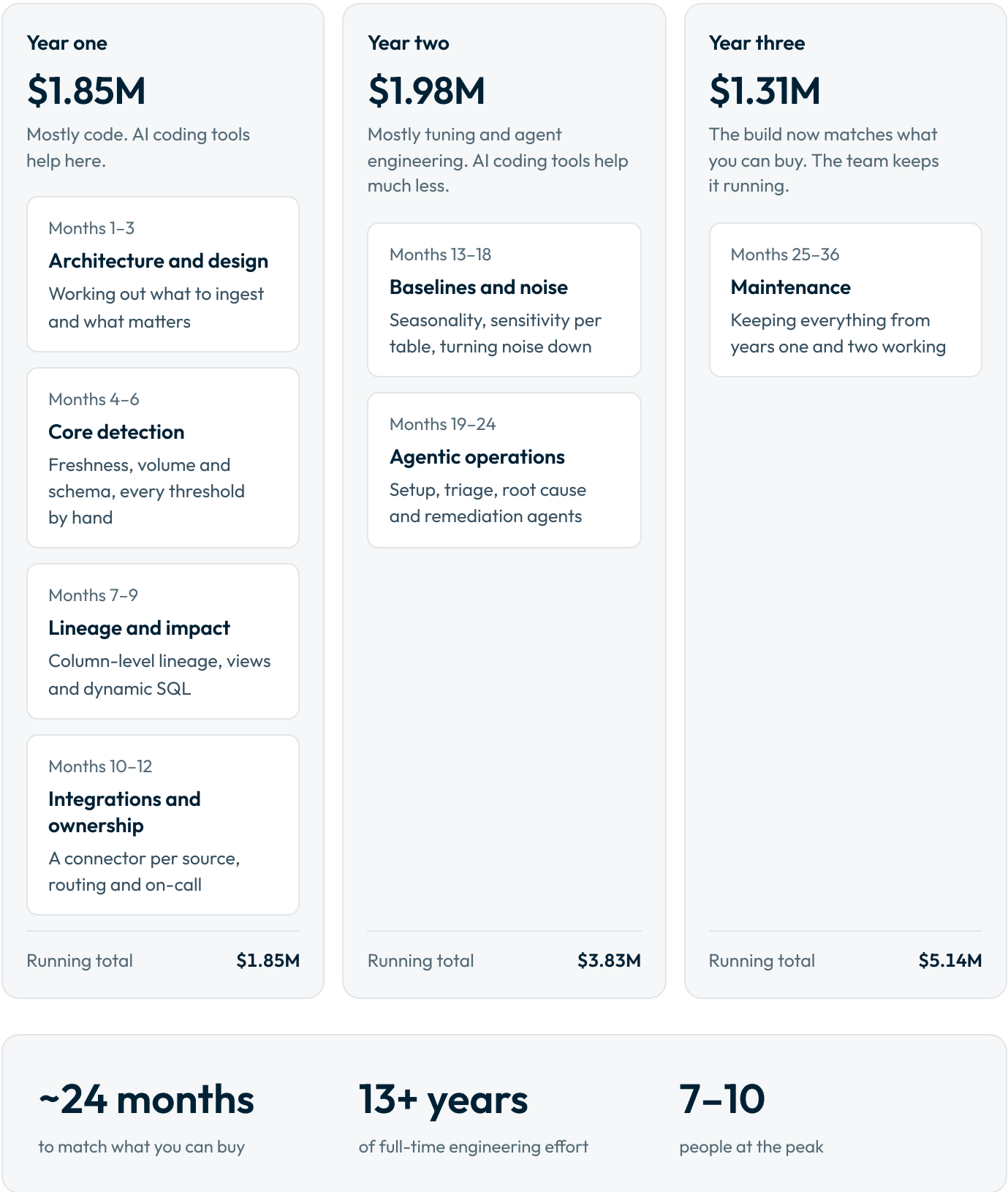
When to escalate, and when to ask a person

Why tuning takes time

Tuning needs a record of which alerts your team acted on. No AI tool can write that record for you.

The build timeline, year by year

Matching what you can buy takes about 24 months.
After that, keeping it running costs 30 to 40% of the original build cost every year.



The first-year budget

The first year costs about \$1.85M, and none of it goes to agentic operations yet. The figures use the composite company described in the methodology on page 3.

Line item	Scope	Cost
4 senior data engineers	Detection, lineage and connectors. \$180K each	\$720,000
2 ML and platform engineers	Agent traces and the evaluation harness. \$190K each	\$380,000
1 data architect	Half the year. \$200K	\$100,000
Product oversight	Half a person. \$150K	\$75,000
Infrastructure and tooling	Cloud, CI/CD, logging, trace storage	\$110,000
QA, documentation and rework	25% on top of staffing costs	\$319,000
Tokens	Four to six times what the tool would cost	\$150,000
Total year one		\$1.85M

Year two

Year two adds \$1.98M. Agentic operations gets built while the team keeps tuning and maintaining everything from year one.

Figures are modeled estimates for the composite company described on page 3. Your costs will vary with salary rates, table count and agent volume.

Weighing the final decision

With the build costed, here's how it compares with buying over three years, and what that looks like for teams who have already built their own.

Three-year cost of building vs buying

Over three years, building costs about five times as much as buying. Once it works, keeping it running costs 30 to 40% of the build every year.

Build it yourself

Year one	\$1.85M
Year two	\$1.98M
Year three	\$1.31M

Three-year total

\$5.14M

7-10 people at the peak, and two years before it matches what you can buy.

Buy Monte Carlo

~\$0.98M

over the same three years

You spend ~\$4.16M less
than building it yourself

You're live in 4 weeks
with agents triaging and fixing issues

You sign one contract
for the full agent trust platform

What you have after two years of building

After two years of building, most teams are still behind where they'd be four weeks after buying. The biggest gaps are agentic operations and alert noise.

	Build, month 24	Monte Carlo, week 4
Your data	Manual thresholds, lineage for one warehouse	Learned baselines, lineage across the warehouse, dbt and BI
Your agents	Traces in a dashboard, one call in ten checked	Context, cost, behavior and answers, all of it
Agentic operations	Starting, at month 19	Running from week one
Noise	Fixed thresholds, more alerts, nobody looking	Sensitivity per table, learned from real teams
Audit evidence	Another project to staff	Lineage and audit trails, as a by-product

What we hear from teams who built their own

Across the hundreds of enterprise data and AI teams we work with, we meet many who set out to build this themselves. Most started for good reasons. They wanted to move quickly, avoid a procurement cycle and learn what they actually needed.

The challenges they describe are consistent. Hand-written rules pile up faster than anyone can maintain them. One custom tool gets replaced by another. Engineers hired to build products end up on call for the tooling instead. The build was only the start, and maintenance took over.

“I want to be in the business of putting out more agents, not in the business of building out observability frameworks.”

CTO, Healthcare

“This custom solution was created five years ago, and then we created another custom solution two years ago... When we want to expand and also maintain this, it is very much manual work.”

Head of Data and ML, Fortune 100 Insurer

“The question goes back to maintenance. How many data quality engines do we need to maintain? In house already we have, I think, four or five.”

Data Engineering Lead, Fortune 100 Insurer

What it costs them

1,100

rules written by hand, for 12 tables

2.5 people

working full time maintaining checks

6 hours

a day, per developer, debugging pipelines

20–30%

of the team on reactive work

Making the call

Building an agent trust platform yourself is possible, and with one or two agents it can be the right call. The question is what happens as your agent program grows. Past a handful of agents, token bills climb, the tooling needs its own team, and agentic operations is still close to two years away.

You could build it. It would cost \$4.16M more over three years and take about two years to catch up.

	Build	Monte Carlo
Cost	\$5.14M over three years	~\$0.98M over three years
Time	24 months to match it	You're live in 4 weeks
Scope	Agentic operations starts at month 19	The full agent trust platform, one contract
Risk	Nothing is watched while you build	Watched from week one
People	7-10 of your engineers, for two years	Your engineers stay on your products

The numbers point the same way. Building costs more, takes longer and leaves your agents unwatched while it happens. It also ties up 7 to 10 engineers for two years who could be building your products instead. Buying gives you the full platform in four weeks, so your team can spend that time shipping agents.

Trust your agents in production.

Weighing a build? Book a chat and we'll run these numbers against your team, tables and agents.

Book a chat

montecarlo.ai

