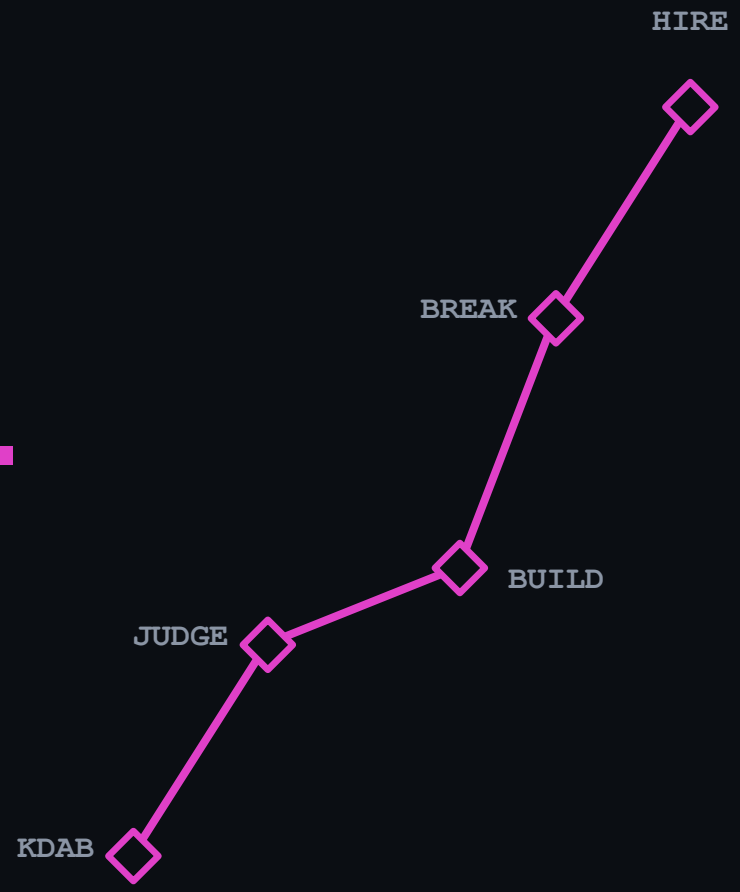
AISBA 2026 · PRACTICAL AI WORKSHOP

The Future of Flight Management Systems

Probabilistic tools.
Deterministic operations.



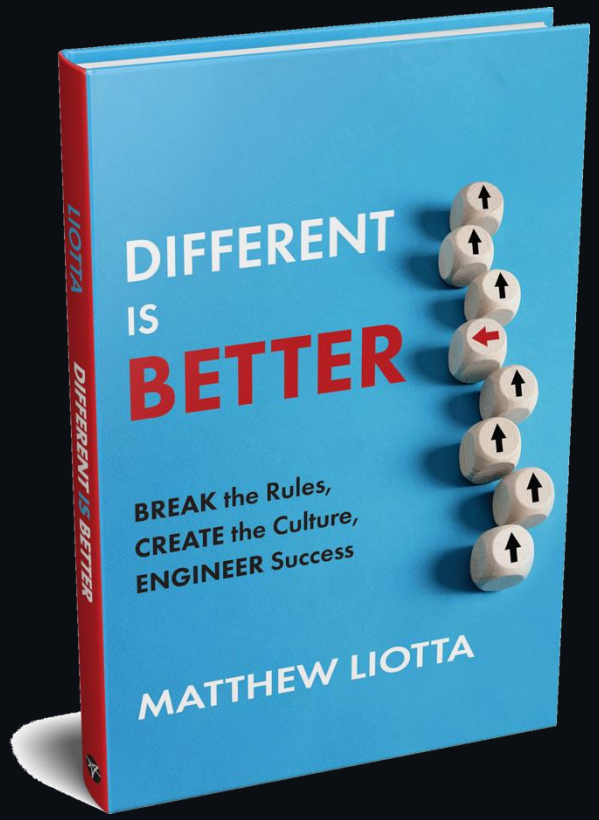
A hands-on workshop: build with AI, watch it fail, and learn how to engineer around it.

Matt Liotta · flyExclusive

Embry-Riddle Aeronautical University · Daytona Beach · September 16, 2026

WHY LISTEN TO ME

I wrote my first chatbot in 1997.



Advantage Books, 2026

- Computer science. Thirty years building software.
- Co-founded Volato. Our teams built its systems in-house.
- AI in a live Part 135 operation since 2023.
- Founded Vaunt. Integrates with several different FMSs.
- flyExclusive: Contrails, a multi-tenant FMS. Free to operators.

AN OLD PROBLEM IN A NEW PLACE

Aviation has been here before

1983

Ironies of Automation

Lisanne Bainbridge shows that automating the easy parts leaves the operator the hard parts, and strips away the practice needed to handle them.

1997

Children of the Magenta Line

Capt. Warren VanderBurgh teaches American Airlines crews what happens when pilots follow the flight path the box drew and lose the habit of commanding it.

2026

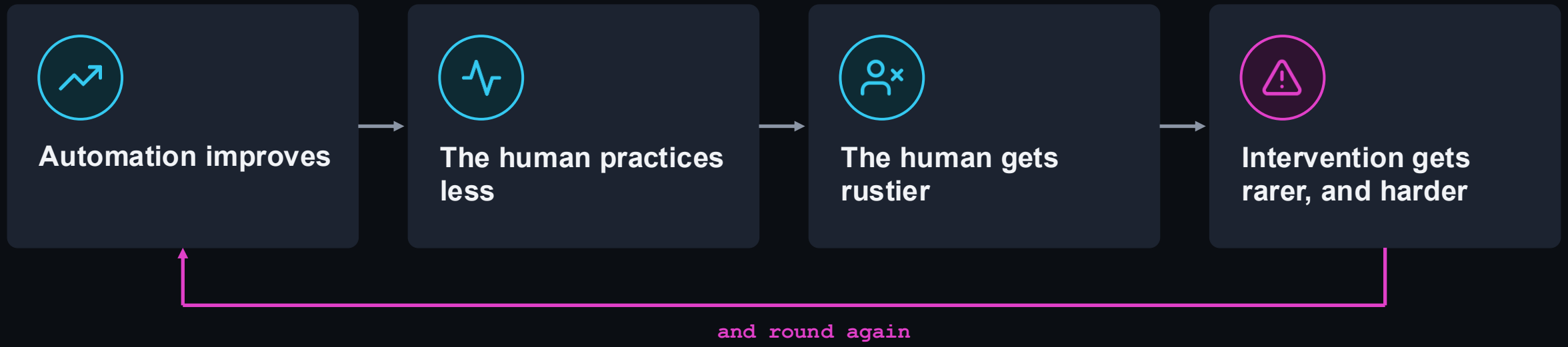
This room

Same problem, new automation. The line on the screen is now an answer on a screen.

Nothing about AI in operations is a new kind of problem. It is a very old one, arriving somewhere new.

1983 · BAINBRIDGE

The irony of automation



The better the automation, the stranger the situations in which a human is still required, and the less prepared that human is to meet them.

1997 · VANDERBURGH

"What is it doing now?"

The magenta line

On a glass-cockpit navigation display, the programmed route is drawn in magenta. VanderBurgh used it as shorthand for crews who had become superb at programming and monitoring the box, and less practiced at simply flying.

The warning sign is the question itself. When the crew is asking what the automation has decided, the crew is no longer commanding it.

The rule he taught

When the automation is making the problem harder, go down a level.

Not off. Down. Stop fighting the box, take a simpler mode, and if that is still not helping, fly the airplane.

The lecture is not an argument against automation. It is an argument for commanding it.

THE TRANSLATION

The magenta line in 2026 is AI

COCKPIT	OPERATION
FMS coupled, flying the programmed route	An agent runs the workflow end to end
Mode control: you set heading and altitude	The agent proposes, a human approves each action
Flight director: guidance only, you fly it	AI advises, the human does the work
Raw data	AI off

LESS AUTOMATION

When the agent is making the problem harder, go down a level.

THE COLLISION

An AI model can give you several plausible answers.

A crew is either legal to fly, or it is not.

Probabilistic

Built to produce what is likely. Same question, different answer. Sounds identical when it is right and when it is wrong.

Deterministic

The aircraft is in one place. The rest requirement is met or it is not. The charge was authorized or it was not.

Every real question in this workshop lives on that fault line.

WHY BOTHER

The prize is real. So is the downside.

Where the gains come from

Coordination	Hours a day spent moving information between systems and people
Utilization	Capacity that goes unsold because nobody could evaluate it in time
Fewer misses	Conditions caught while they are still cheap to fix
Attention	Expert judgment freed for customers and exceptions

And the downside

Done wrong you do not simply miss the gains. You get something worse:

- A system people trust until the day it is wrong
- A decision nobody can explain to a regulator
- An expert who has stopped checking the answers

The gap between those two outcomes is engineering, not ambition.

WHO'S HERE

Three people are in this room

The Builder

You're going to build your own FMS with AI. It will look done in a weekend.

Learn why it fails in month nine, and how to avoid it.

The Buyer

You're going to be sold AI that works beautifully in the demo.

Learn to tell a demo from a system.

The Leader

You're building a team where people and AI agents work together.

Learn what production AI actually requires.

Everyone leaves knowing how to use probabilistic AI to build deterministic systems you can trust.

HOW TODAY WORKS

Four labs. One fictional airline. No handouts.

LAB 1 · 8 MIN

AI as the judge

On your phone. Let AI decide which trips are legal.

LAB 2 · 15 MIN

AI as the builder

On the big screen. We build the thing that decides.

LAB 3 · 8 MIN

Break the build

At your table. Throw the real world at it.

LAB 4 · 8 MIN

Hire an AI teammate

At your table. Design an agent for a hybrid team.

Everything you need is on the screen.

Photograph the slides you want. If you have a phone with any AI assistant on it, you can do Lab 1. If you do not, turn to a neighbour who does. Labs 3 and 4 need nothing but your table.

The AI will be wrong today. That is the curriculum.

BEFORE WE BUILD

Using it well is not knowing what it does

Most of you are proficient

You write good prompts. You get useful work out of it. You have a sense for when to push back.

That is a skill, and it is real.

Fewer know the mechanism

What the model is actually computing, why it is fluent, and exactly where the error enters.

Without that, you cannot predict how it will fail, and you cannot design around it.

Ten minutes of fundamentals. Then we build.

FUNDAMENTALS · 2 MIN

Finish these out loud

"Peanut butter and _____"	jelly
"Roses are red, violets are _____"	blue
"Cleared for takeoff, runway _____"	(you hesitated)

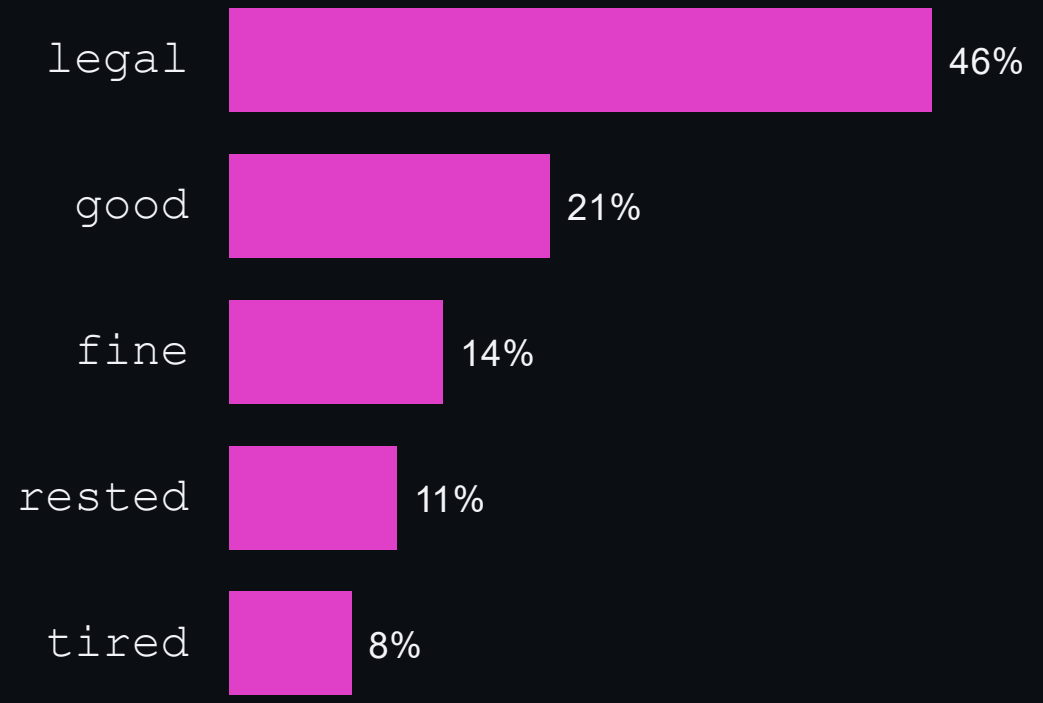
You did not look anything up. You predicted, from a lifetime of exposure.

The third one had no likely answer, so you stopped. A model does not stop. That is the whole difference.

LESSON · HOW IT WORKS

It's predicting the next word

The dispatcher checked the crew's rest and said they were _____



Illustrative probabilities

It says "legal" because "legal" is likely. Not because it checked.

One equation, read like a sentence

$$P(w_1 \dots w_n) = P(w_1) \times P(w_2 \mid w_1) \times P(w_3 \mid w_1\, w_2) \times \dots \times P(w_n \mid w_1 \dots w_{n-1})$$

The probability of a sentence is the probability of its first word, times the probability of each next word given everything already chosen.

The rule is exact

This half is provable mathematics. It never introduces a single unit of error.

The estimate is approximate

A neural network guesses each of those probabilities. Every error you will ever see enters right here.

You now know precisely where hallucinations live.

WARM-UP · 2 MIN

Ask your AI to pick a number

PROMPT

```
Pick a random number between
1 and 10. Reply with only the
number.
```

Tally the room

1	2	3	4	5
6	7	8	9	10

Is it random?

If a hundred people rolled real dice, you'd see a flat spread. Watch what the room gets instead.

Estimating is not choosing

Step one: estimate

The network scores every possible next word. This part is the same every time you ask.

Step two: choose

Something has to pick one word from that spread. That choice is where the variation lives.

Turning the dial down helps. It does not make the system deterministic.

Temperature

A setting controls how adventurously it picks. Low is conservative, high reaches for less likely words.

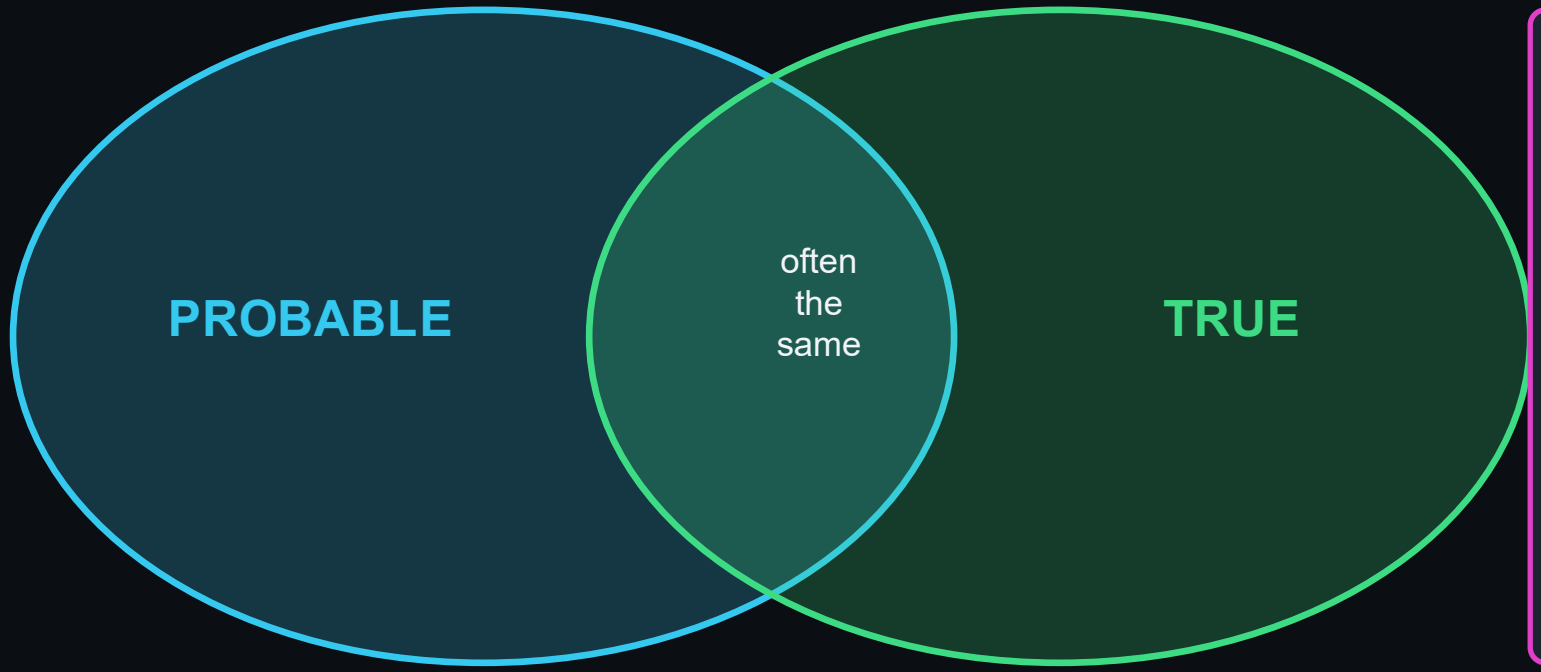
Wording

Change one word of your prompt and the whole spread shifts underneath you.

Everything else

Context length, model version, and hidden defaults all move the estimate.

It optimizes for probable, not for true



The gap

A fluent, confident, wrong answer is not a malfunction. It is the system doing exactly what it was built to do.

"The crew is legal" is a very probable sentence. Whether it is true is a different question entirely, and the model was never asked it.

Deterministic does not mean right

Deterministic: the same inputs produce the same output, every time, by construction.

It is not a claim about truth

A checker can be wrong. It will then be wrong the same way every single time.

Which is exactly why it is useful

A failure that reproduces can be written as a test, fixed, and kept fixed.

And why you can inspect it

The rule is written down. You can read it, argue with it, and version it.

Probabilistic systems fail unpredictably. Deterministic systems fail the same way twice.

Only one of those can be tested.

LESSON · WHAT FOLLOWS

AI isn't magic. It's probability.

Great at

- Language, ambiguity, and messy information
- Drafting, summarizing, and explaining
- Writing code and test cases
- Spotting patterns a person would miss

Unreliable at

- Applying many interacting rules exactly
- Giving the same answer every time
- Arithmetic across many rows
- Knowing what is true in your operation right now

Notice: "writing code and tests" is on the left. Remember that for Lab 2.

LAB SETUP

Meet Magenta Air

3	6	6	4
aircraft	pilots	trips tomorrow	rules

Magenta Air is fictional.

Its ops manual is a simplified training rule set invented for today. It is not 14 CFR, and it should never be used for a real operational decision.

Small enough to fit on your phone. Still enough to catch the AI out.

PHOTOGRAPH THIS SLIDE

The Magenta Air ops manual

TYPE	Both pilots must be rated on that aircraft type.
AOG	No departure while a grounding maintenance item is open. An item is open from its opened time until its closed time. Non-grounding items never restrict flight.
REST	A pilot needs 10:00 of rest before duty starts. Duty starts 60 minutes before their first departure of the day.
POSITION	An aircraft departs from where it last arrived, or from its starting airport.

All times UTC. Every trip must pass all four rules.

PHOTOGRAPH THIS SLIDE TOO

Tomorrow's operation

AIRCRAFT

N1MA	CL30	starts KDAB
N2MA	C56X	starts KTEB
N3MA	E55P	starts KPBI

MAINTENANCE

M1 N3MA
grounding: open 05:00, closed 12:00

M2 N1MA
grounding: open 20:00 yesterday, closed 11:00

PILOTS · rating · prev duty ended

Ali	CL30	23:00
Bev	CL30, C56X	20:00
Cruz	C56X	22:50
Diaz	E55P	21:00
Erin	E55P	20:00
Finn	C56X	18:00

previous duty all ended yesterday

TRIPS · route · times · crew

T1 N1MA KDAB→KTEB
12:00–14:30 Ali, Bev

T2 N1MA KTEB→KPBI
16:00–18:45 Ali, Bev

T3 N2MA KTEB→KBOS
09:30–10:45 Cruz, Bev

T4 N3MA KPBI→KMIA
10:00–11:00 Diaz, Erin

T5 N2MA KBOS→KTEB
13:30–14:45 Finn, Erin

T6 N3MA KORL→KDAB
15:30–16:30 Diaz, Erin

All times UTC, tomorrow, 17 September.

LAB 1 · AI AS THE JUDGE

Let the AI decide

8:00

PROMPT

```
You are a dispatcher for Magenta Air.
Using only the rules and the data in this
photo, list which of the six trips are
NOT legal to fly, with the rule broken
and a one-line reason.
```

1. Open any AI assistant on your phone.
2. Send it your two photos, or type the tables in.
3. Send the prompt. Write down what it says.
4. Ask it again in a brand-new chat. Compare.

No phone? Look over a neighbour’s shoulder. No AI app? A browser works.

Which trips does it flag? How sure does it sound? Do the two runs agree?

LAB 1 · REVEAL

It understood the rules. It still got it wrong.

T1

LEGAL

T2

LEGAL

T3

NOT LEGAL

REST

T4

NOT LEGAL

AOG

T5

NOT LEGAL

TYPE

T6

NOT LEGAL

POSITION

Three aircraft. Six trips. Four rules that fit on one slide. A capable reasoning model, reading them correctly, and still arriving at the wrong answer.

The key came from a deterministic checker, not from a model.

LAB 1 · THE TRAPS

Where the judge usually slips

T3

The sixty-minute duty start

Cruz looks like he has 10:40 of rest. Duty starts at 08:30, not 09:30, so he has 9:40.

T1

The item that closed

M2 was a grounding item, and it closed at 11:00. The trip departs at 12:00. Legal.

T4

The item that was still open

M1 closed at 12:00. T4 departs at 10:00. Not legal. Same rule, opposite answer.

T6

The aircraft that teleported

N3MA last arrived at KMIA. T6 departs from KORL. Easy to miss when you are reading a list.

Every one of these is arithmetic or bookkeeping. None of them is a reading-comprehension failure.

THE ARCHITECTURE

AI interprets. Deterministic systems decide.

Give the model the messy half

- Read a maintenance note written by a human
- Turn a broker email into a structured request
- Explain why two trips conflict
- Propose recovery options worth considering
- Talk to a dispatcher in plain language

Give the system the exact half

- Aircraft, crew, times, locations
- Qualifications and currency
- Rest, duty and maintenance status
- Every rule, as tested code
- The answer that gets acted on

Once the problem is reduced to aircraft, crew, times and rules, stop prompting. Compile it into state and execute.

The handoff point is the design decision. Everything else follows from it.

◆ THE LESSON

Right most of the time

is the most dangerous kind of wrong.

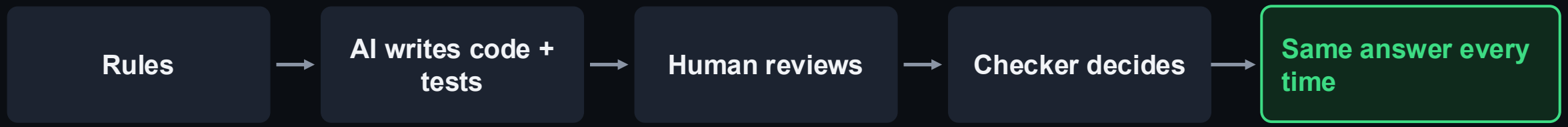
AI as the judge passes the demo, earns your trust, and then fails on the edge case nobody tested, usually after people depend on it.

THE BIG IDEA

Let AI write the rule. Never let AI be the rule.

AI as the judge

AI as the builder



The AI's mistakes move from runtime, where they hurt, to build time, where tests catch them.

LAB 2 · AI AS THE BUILDER

We build the judge, together

15:00

PROMPT

```
Do not decide which trips are legal. Build me
a tool that decides. Write a legality checker
for these four rules that I can run on the
six trips. Before you build it, write test
cases for every rule, including the boundary
cases. Then run the tests and show me which
pass.
```

1. I drive. It goes on the big screen.
2. You call out test cases. I add them.
3. We read the tests before we read the code.
4. We run it and compare with the answer key.
5. Any mismatch is a bug: new test, fix, rerun.

Nobody needs a laptop for this. Your job is to try to break it.

Shout out: exactly 10:00 of rest. An item that closes a minute before departure. A pilot rated on two types.

LAB 2 · DEBRIEF

Why the builder beats the judge

	AI as the judge	AI as the builder
Same input, same output?	No	Yes
When it’s wrong, can you find out why?	Rarely	Yes, and add a test
Can you prove it before you rely on it?	No	Run the tests
What does a human review?	Every answer, forever	The rules and tests, once

AI-written code can be wrong too. The difference: it's wrong the same way every time, so a test can catch it.

LAB 2 · THE SECRET

Your answer key is your most valuable asset

Today's answer key came from a checker that was tested against the rules. That's what let us grade the AI at all.

Without an answer key, you can’t tell a good AI from a confident one.

In your operation, the answer key is:

- Years of real trips and their outcomes
- Scenarios your Director of Operations has verified
- Every past mistake, turned into a test

LAB 3 · BREAK THE BUILD

Your checker meets the real world

10:00

MIDNIGHT

A leg departs 23:30 and lands 01:15.

LATE CHANGE

T02 moves 30 minutes later. What reruns?

AOG AT NOON

A grounding item opens mid-flight.

RACE

Two dispatchers assign Baker at once.

RULE CHANGE

Rest becomes 11:00 next month.

BAD DATA

A pilot's rating field is blank.

LOCAL TIME

"Depart Teterboro 8 a.m. Friday."

2 A.M.

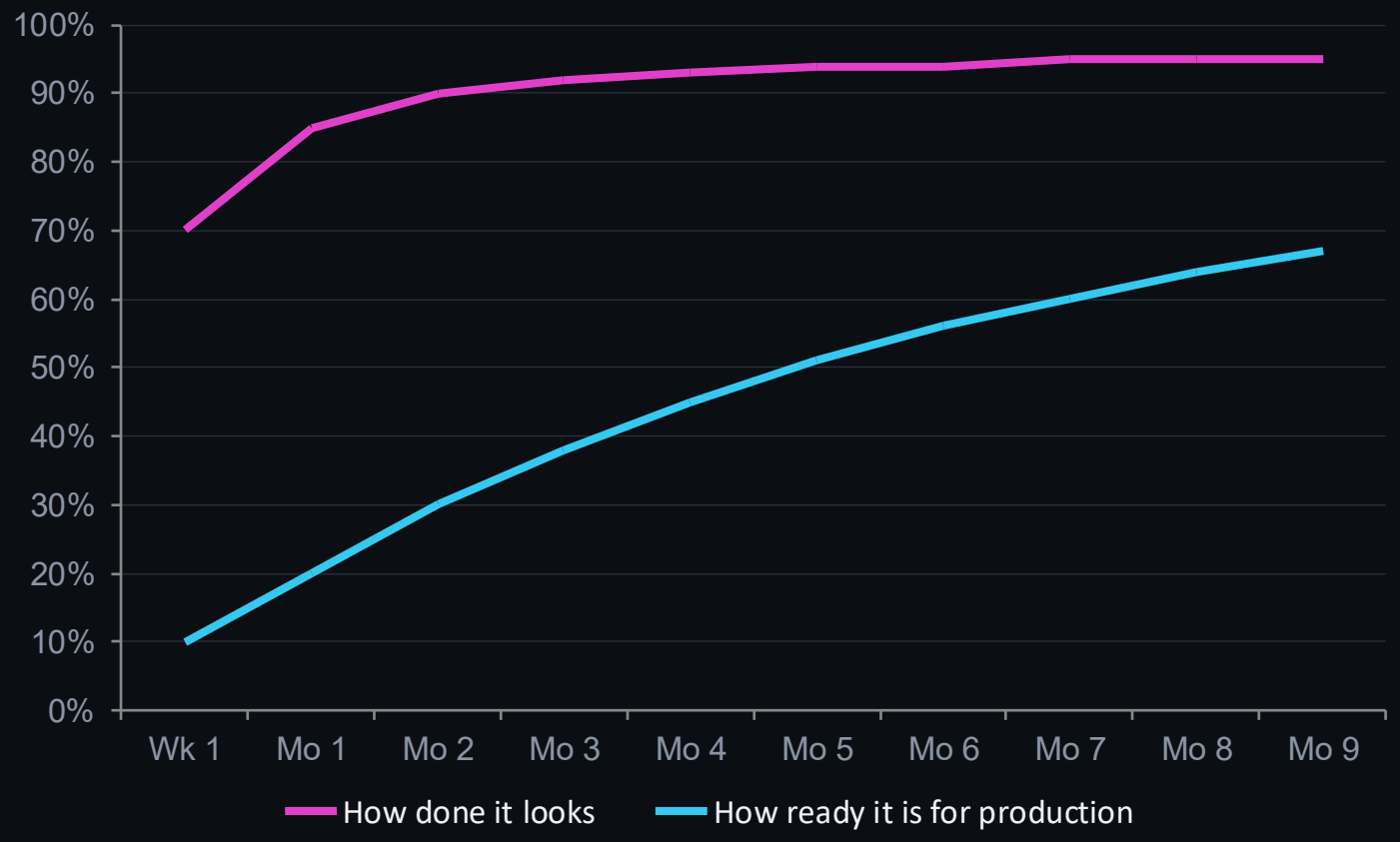
The checker is on a closed laptop.

AUDIT

Why was T04 approved? Prove it.

Score each card: handles it (2) · partly (1) · never thought of it (0)

The demo cliff



Illustrative

Homegrown systems rarely fail in the demo.

They fail after people depend on them.

The gap between the lines is time zones, concurrency, rule changes, bad data, audits, uptime, and the 2 a.m. phone call.

What production actually requires

EVERYTHING BELOW THE LINE

System of record	Edge cases and time zones	Concurrency	Rule versioning	Audit trail
Permissions	Integrations	Monitoring and uptime	Support at 2 a.m.	Regulatory acceptance

This is what's worth paying for, or building very carefully.

AN ASIDE ON ECONOMICS

Why the software itself is getting cheap

Building collapsed in cost

Work that took a vendor quarters can take a small team days.

Supply went up

More vendors, and more operators building their own.

Software became a complement

Marketplaces, payment platforms and operators can give it away to grow what they actually sell.

Disclosure: my employer gives its FMS away, so I have an interest in this observation. Judge it on the argument, and hold me to the scorecard at the end.

If the code is getting cheap, the value moves to everything the code sits on top of.

ANATOMY

An agent is a model inside a harness

The model

The next-word predictor from the last section. Stateless. It knows nothing about your operation and it does not change as you use it.

Rented.

THE HARNESS

Yours.

Tools

Named, permissioned actions against real systems. Your Lab 2 checker lives here.

Memory

What gets carried forward, and what gets looked up.

Skills

Your procedures, written down so the agent follows your way.

The loop

Assembles context, calls tools, retries, enforces limits, logs everything.

The model is the engine. The harness is the airframe, the avionics and the checklists. Only one of those is interchangeable.

Memory is not the model learning

Three different things people call memory

Context

What the model can see on this one call. Small, temporary, and rebuilt from scratch every single time.

Retrieval

The harness looks facts up in your systems and pastes them in. This is grounding, and it is where truth comes from.

Persistence

Notes the harness saved earlier and chooses to paste back in later. Useful, and the most likely to go stale.

None of these change the model. Its weights are frozen. Memory is text the harness decides to show it again.

Skills are the other half

Your procedures written down so the agent follows your operation, not the internet average of everyone else. Skills are how tribal knowledge stops being tribal.

Each part fails in its own way

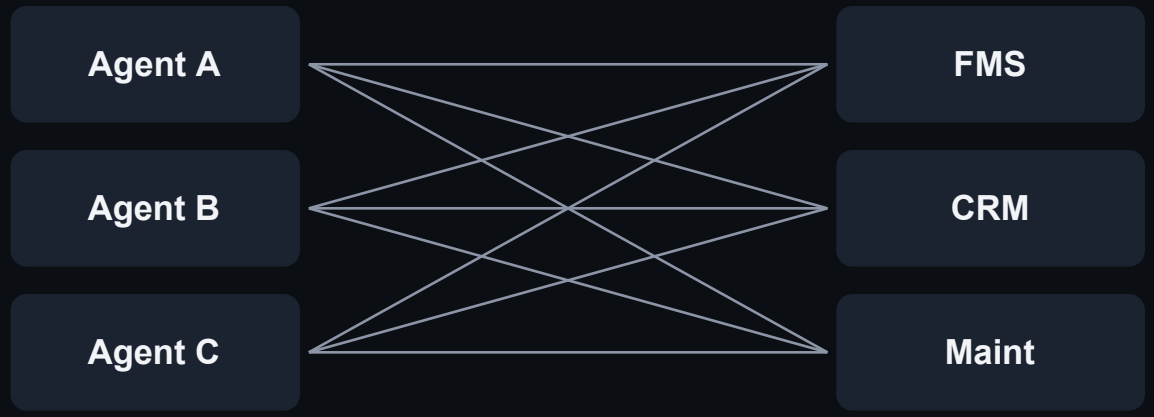
Model	Fluent, confident, wrong
Tools	Right answer from stale or wrong data, or too much authority
Memory	A fact that was true in March, re-shown with full confidence
Skills	Your outdated procedure, followed perfectly
Harness	The right context never reached the model, so the agent looked like it forgot

Four of these five look identical from the outside: a wrong answer, delivered confidently. Without a log you are guessing.

THE CONNECTION LAYER

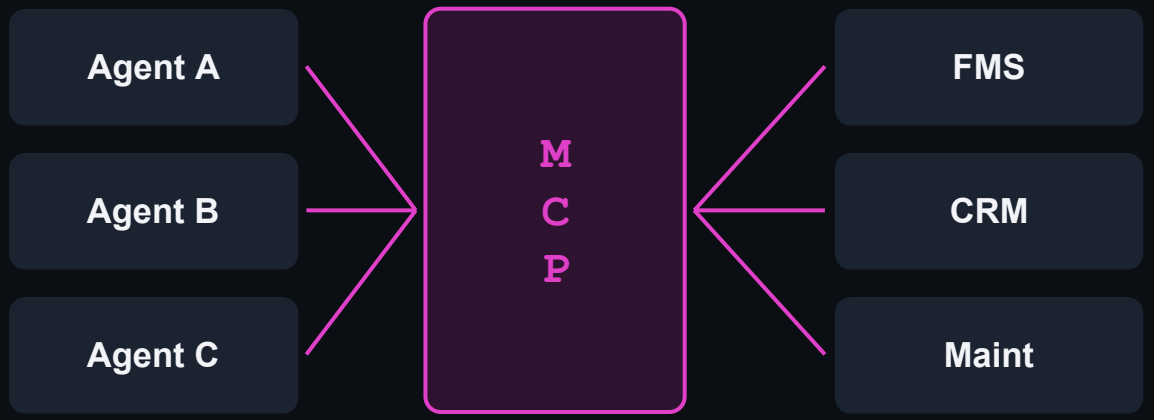
How an agent reaches your systems

WITHOUT A STANDARD



Nine custom integrations. Then a tenth system arrives.

WITH MCP



Each system publishes its tools once. Any agent can discover and call them.

The Model Context Protocol is an open standard for exposing tools, and the data they need, to any AI agent. A plug, not a product.

The question stops being "does your vendor have AI?" and becomes "can my agent call your system?"

THE ANSWER TO THE TITLE

The FMS serves two kinds of user now

People

get screens: schedule boards, dispatch views, the things a person clicks.

ONE SOURCE OF TRUTH

- The same ground truth
- The same rules, as tested code
- The same permissions
- The same audit trail

Agents

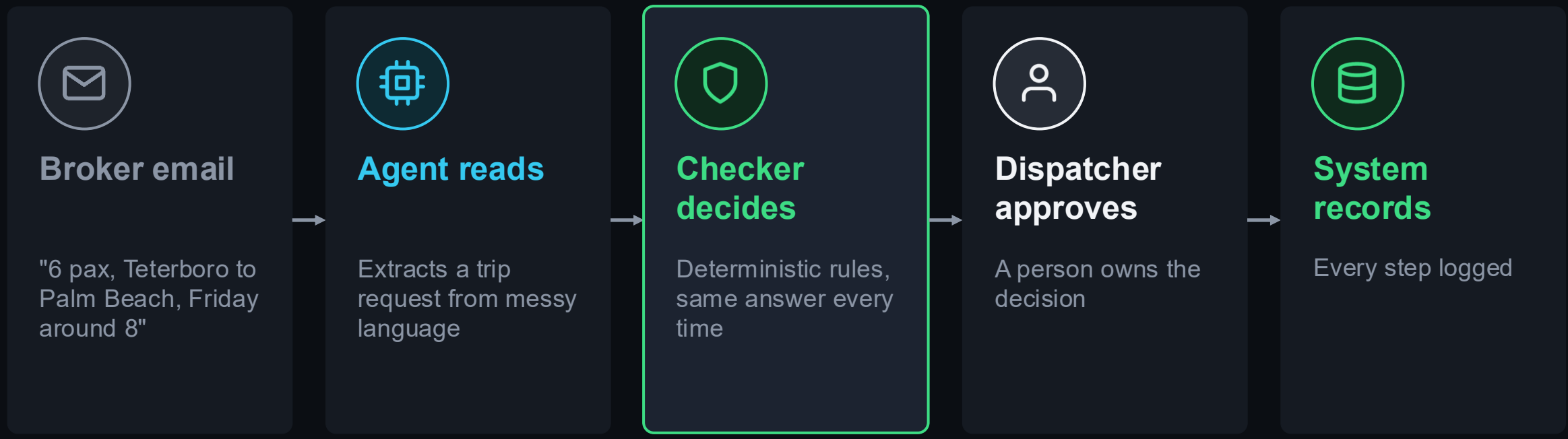
get tools: named, permissioned actions they can discover and call.

If a person and an agent get different answers, that is two systems, not one.

The future FMS is the trusted core that both halves of the team work from.

PUTTING AI BACK IN THE LOOP

Now use AI at runtime, the right way



Probabilistic where the world is messy. **Deterministic** where the answer must be right.

LIVE DEMO · A PRODUCTION SYSTEM

The same pattern at operating scale

TOOLS THE AGENT MAY CALL

`crew_coverage_gaps`

`trip_feasibility`

`fleet_availability`

`maintenance_alerts`

`propose_crew_assignment`
Note the verb on the last one.

AGENT

Proposes a crew assignment, confidently

SYSTEM

Rejects it: rest requirement not met

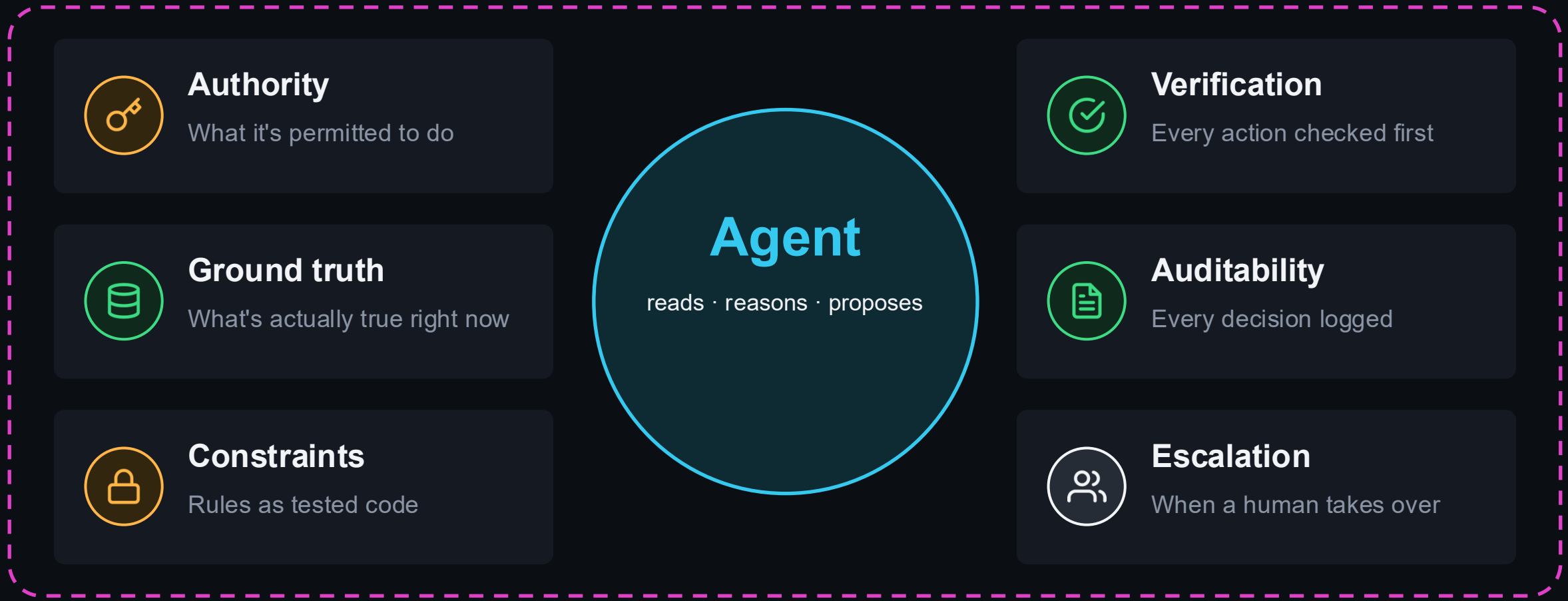
LOG

Proposal, rule and rejection recorded

The model was wrong. The system was not.

THE FRAMEWORK

The deterministic envelope



Probabilistic inside. Deterministic around it.

HOW MUCH AUTONOMY?

Four dials set the human's role

Consequence

What happens if it's wrong?

Confidence

Can we verify it's right?

Reversibility

Can we undo it?

Authority

Who is accountable?

MORE AUTONOMY

MORE HUMAN

Agent acts

Delay text to a passenger's assistant

Agent proposes, human approves

Crew reassignment after an AOG

Human decides, AI informs

Accepting a trip below cost

The goal isn't autonomous AI. It's operational leverage.

DOCTRINE

The dial is not set once. It moves during the event.

Normal Tuesday	Routine reassignment, hours of slack, easy to undo	**Agent acts**
Weather building	Several trips exposed, decisions interact	**Agent proposes, human approves**
AOG at 16:00 Friday	Cascading, expensive, watched by customers	**Human decides, AI informs**

Same workflow. Same agent. Three different rungs, chosen by the situation.

LAB 4 · HIRE AN AI TEAMMATE

Write the job description

8:00

THE INBOX

"Hi team, need a Challenger for 6 pax, Teterboro to Palm Beach, Friday around 8 in the morning, back Sunday afternoon. Client is flexible by an hour. Can you quote?"

Role: charter request agent

Reads
What data?

Calls
Which deterministic tools?

Acts alone on
Without approval

Proposes
For a human to approve

Never does
Hard boundaries

Reports to
The accountable human

Escalates when
Specific triggers

Reviewed by
Logs, audits, spot checks

Then set the four dials.

THE NEW TEAM

Hybrid teams: who owns what

People own

- Judgment
- Relationships
- Exceptions
- Accountability

Agents own

- Ambiguity
- Coordination
- Drafting
- Monitoring

Systems own

- Truth
- Rules
- Permissions
- The record

The future FMS is the system everyone on the team trusts, human or AI.

TAKE THIS HOME

What each of you should do Monday

Builders

Use AI to build deterministic systems, with tests, on a real system of record. Build the answer key first. Budget for month nine.

Buyers

Ask every vendor: is the AI the judge or the builder? Show me your tests. What happens when it is wrong?

Leaders

Design the team, not the tool: people, agents and systems, each doing the work they do best.

Get this right and the efficiency gains are large and durable. Get it wrong and you will not simply miss them.

BEFORE ANY OF THIS WORKS

Where is your software on this ladder?

0	**No way out**	Your data leaves as a PDF or a screenshot.
1	**Export only**	Reports and CSVs you pull by hand. Stale the moment you download them.
2	**A read API**	Documented, credentialed, and you can get your own operation out of it.
3	**A write API**	You can act on it, with permissions, a test environment and support behind it.
4	**Tools for agents**	Published so an agent can discover and call them, scoped per user, every call logged.

Most of this industry sits on rungs 0 to 2. A handful of platforms are climbing. Very few are at 4.

TAKE-HOME SCORECARD

Twelve questions worth asking

Ask about access

- [ ] Can I export everything I put in?
- [ ] Is there a documented API I can write to?
- [ ] Is there a test environment?
- [ ] What does access cost, and is it in my contract?

Ask about the AI

- [ ] Is a model deciding, or is tested code deciding?
- [ ] Show me your tests for the rules.
- [ ] Can I see which tools ran, and on what data?
- [ ] Who is accountable when the rule logic is wrong?

Ask about the deal

- [ ] How do you make money?
- [ ] What can you see about my customers and pricing?
- [ ] What happens to me if your strategy changes?
- [ ] Who else is in my data?

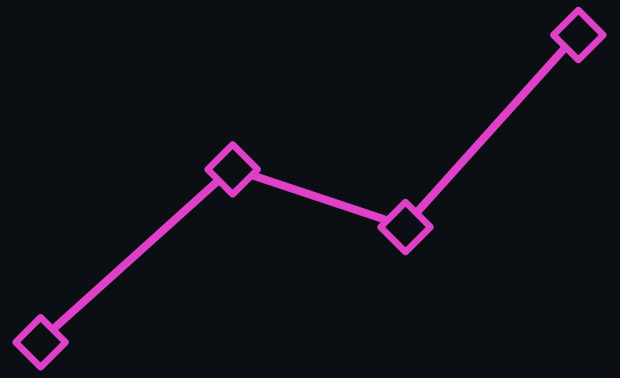
Ask us these too. A free FMS should survive the same questions as a paid one.

AISBA 2026

Let AI write the rule.

Never let AI be the rule.

Your AI will be wrong. Build systems where that's okay.



Matt Liotta · flyExclusive

Thank you.