FORTITUDE RE®

The Wheel Still Turns

Getting off it with AI

Heather Gordon | Fortitude Re | SEAC Annual Meeting 2026

Agenda

Our AI evolution, in five parts

The wheel

Seven manual steps, every model, every quarter



Where it lands

Proof, guardrails, and your first project

How it got that way

A sound design, run by hidden people

What we built

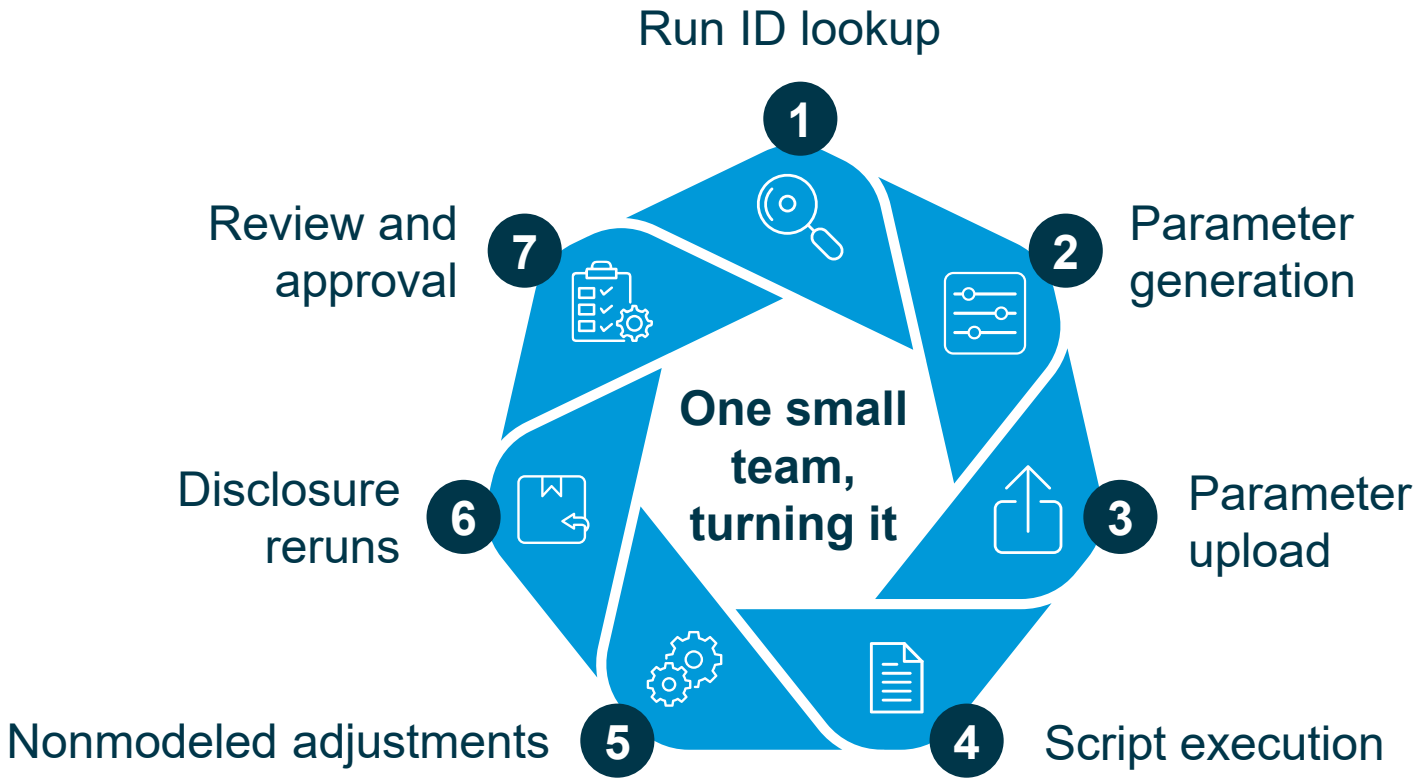
Three quarters that changed the slope of the line

The insight

Why a team of generalists could pull this off

Picture a quarter-end close

Ours, until recently



× Every actuarial model and line of business

× Every quarter

× Every rerun, when a late adjustment or a correction lands

All seven steps, by hand, for every model - every time.



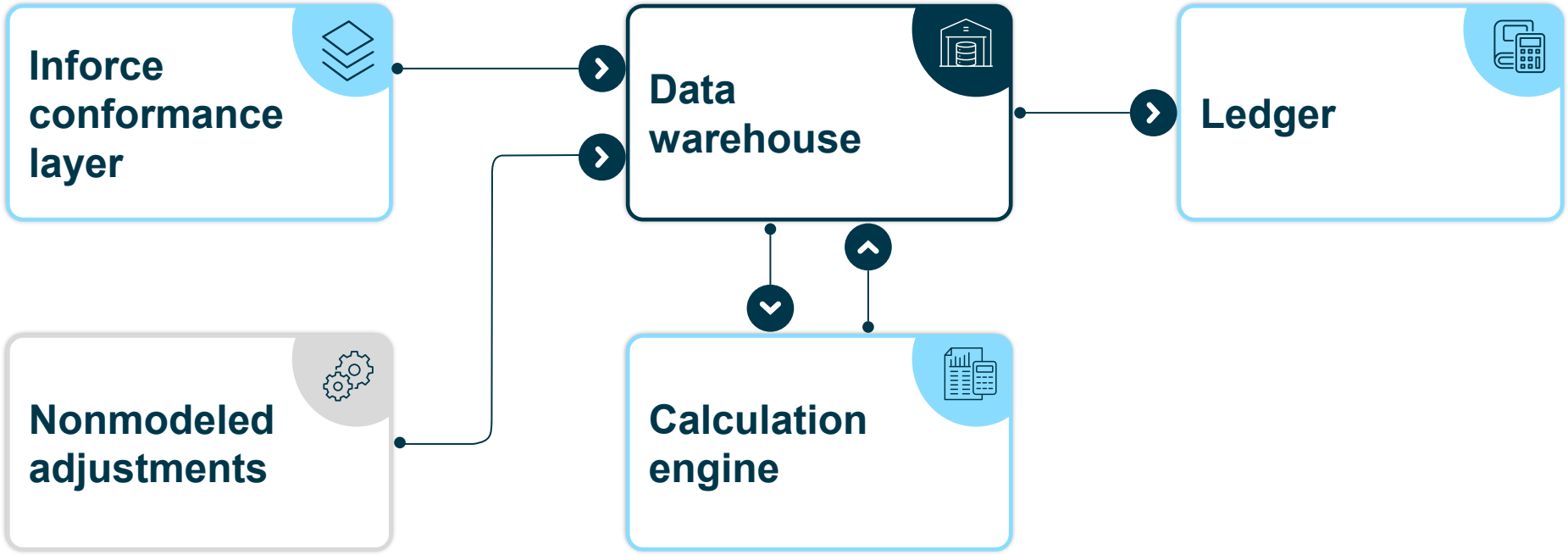
The question:

how does a team of actuaries get off this wheel - with no IT project and no new hires?

This talk is the true story of the answer.

The blueprint: nothing you have not seen before

A conformance layer, a warehouse, a calculation engine, a ledger



Nothing novel here.

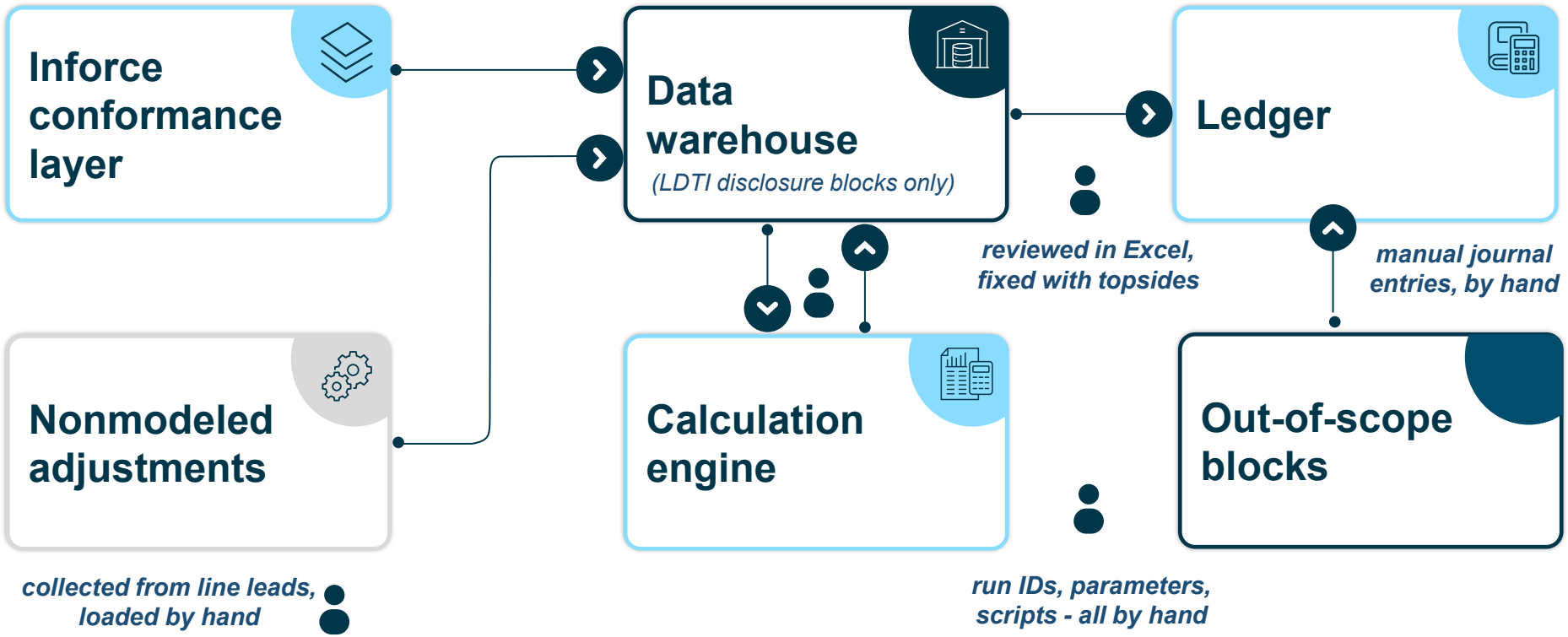
A controlled central repository feeding the ledger - with a channel for nonmodeled adjustments - is the industry pattern, and it is the right one.



The founding vision: *Consolidation owns the center - everything consolidated, controlled, and delivered to Finance.*

Our version with the people drawn in

It looked automated and complete. It was neither.



Hidden people at every hand-off.

One small team in the middle of both channels into the warehouse - and whole blocks outside the warehouse entirely.

What the wheel cost us

Not hypothetical risk - realized cost

Corrections reached the ledger	Missed exhibit reviews led to Excel topsides and manual journal entries - hard to track, and often carried forward into the next quarter's close
A control gap on nonmodeled data	With the tools of the day, it was hard for reviewers to see results immediately - verification lagged the adjustments
Consolidation was the bottleneck	Every line lead's submission, question, and rerun landed on one small team
Production reruns meant waiting	Scripts ran production-only, so every rerun pulled in IT and the data team

And the deepest cost - scalability:

Every new block of business meant the same team turning the same wheel more times. Growth arrived as workload, never as capacity.



And we tried. For a year.

Real progress, built inside the team - one win per quarter

2025

Q2

Distribute the work

- Excel tool so line leads could build their own nonmodeled files
- Adoption was partial - tech-savvy leads used it, others still relied on us

Q3

Stop the bleeding

- Ledger corrections mounted - and no way to rerun past periods
- Topside database built as a controlled safety valve for corrections

Q4

Build the plumbing

- Dev environment unlocked - test before production at last
- Drop zone auto-loads files; run-ID table + Power Query lookup

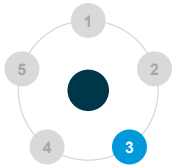


We knew exactly what to build.

Building it was the slow part - each win took a full quarter, and the hardest steps never moved.

The missing piece was never judgement

We knew what was wrong and what good looked like - we lacked the build depth



1 ***Saw the inefficiencies and the risks***	• Living the close every quarter means you know exactly where it hurts and what could go wrong.
2 ***Had built systems before***	• Historical knowledge of how systems fit together - data in, controls around, output out.
3 ***Did not know the modern tools***	• The warehouse internals, the Python patterns, the web tooling - the depth was the missing piece.

In Q1 2026, AI tools bridged exactly that gap - technical depth on demand, steered by someone who knew what the answer should look like.

Deep specialists ask narrow questions. Novices cannot spot a wrong answer. The generalist is the sweet spot.

The partnership: you supply judgement, AI supplies depth

Complementary by construction



You bring

- Context: the business, the data, the deadlines
- What the answer should look like
- Which question actually matters
- The authority to say “that's wrong”
- Accountability for what ships

AI brings

- Syntax in every language you will ever touch
- Patterns from a million codebases
- Patience for boilerplate and documentation
- Patience for rework - changing your mind costs nothing
- Breadth across technologies you have never used
- Speed on drafts one through ten

Anatomy of a working session

The loop that makes it fast and keeps it safe

Cheap: minutes per iteration

Non-negotiable: never skipped

- **Describe the context**
- **Let it draft**
- **Challenge the draft**
- **Test on dev data**
- **Verify numbers**
- **Ship with review**

The loop is fast because drafting is cheap.



The loop is safe because the last three steps are never optional.

Three quarters of compounding wins

Each quarter moved more steps than all of 2025

2026

Q1

Structure and control

- All nonmodeled blocks brought into the scripts
- Line leads fully self-sufficient - no more Consolidation in the middle
- Model output visible beside every adjustment - a long-standing control gap closed

Q2

The automation quarter

- Run-ID extraction fully automated from the source page
- A line lead's submission triggers every script, unattended
- Disclosure reruns fire themselves - the interruptions simply stopped

Q3

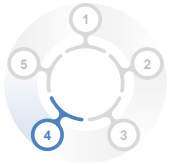
Review, modernized

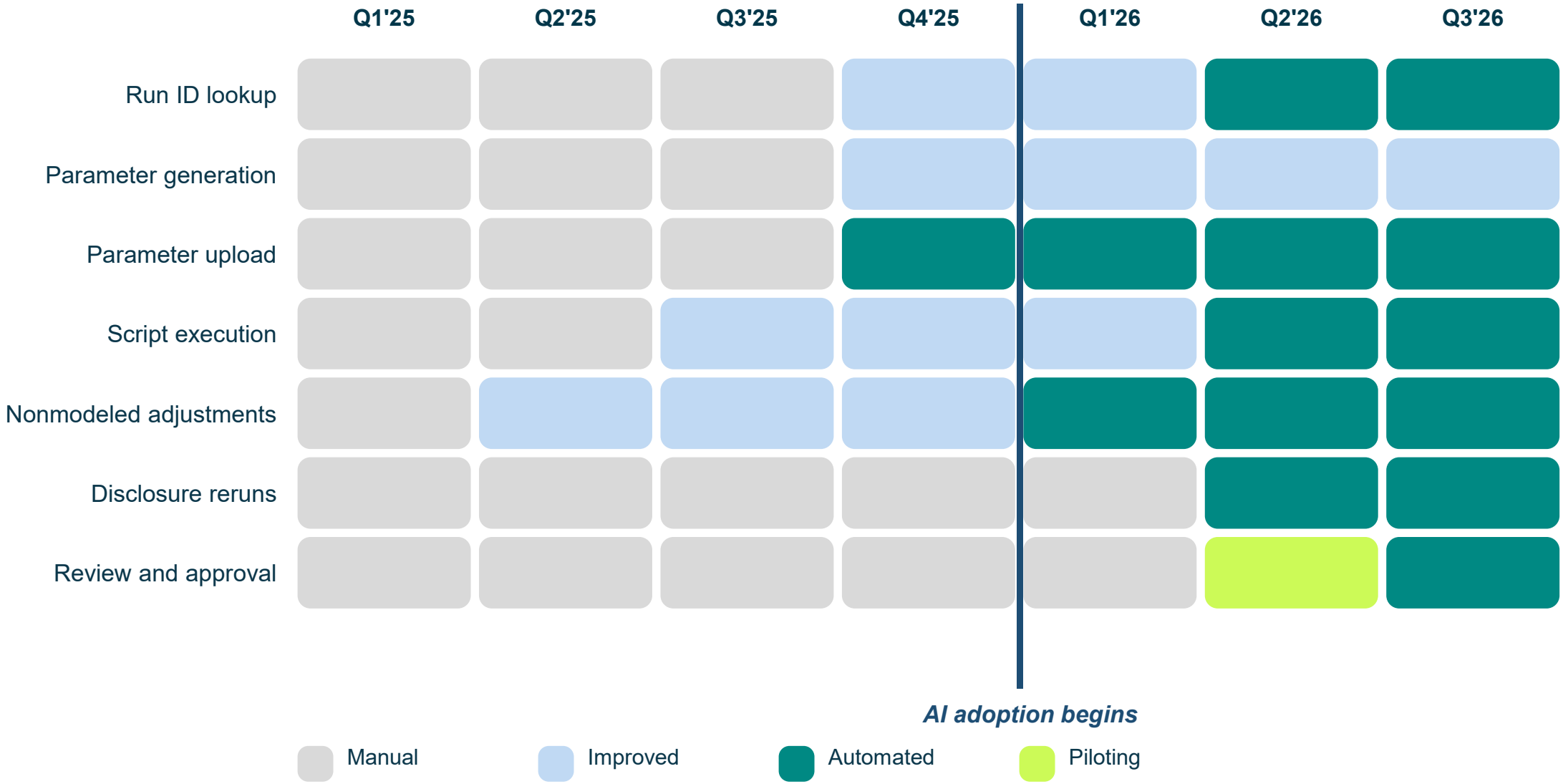
- One interactive approval page replaces two Excel files
- Automated checks run before approval unlocks
- Full audit trail - names and timestamps on every sign-off

Our own team built every one of these, with AI's help - still no IT project, still no new hires.

Seven steps, seven quarters, one page

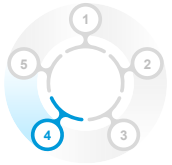
The same wheel, quarter by quarter





Getting off the wheel

Close night, then and now



Then

- Scripts kicked off one by one and babysat late into the night - and on weekends
- Every nonmodeled submission an interruption and a manual rerun
- Tracker workbooks updated by hand between runs
- Issues in ledger found from review - fixed with topsides

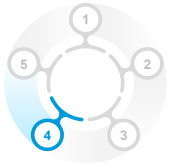
Now

- A line lead's nonmodeled submission is the trigger - scripts batch and rerun unattended
- Common cross-exhibit checks run first, flagging potential issues for the approver
- Approvals clicked on one page, with names, timestamps, and an audit trail
- The team works exceptions and review - not the loop

The wheel still turns. We just stopped being the ones turning it.

Rules we never break

Simple, boring, and the reason this works



1. No real test, no merge

Every change to a close process runs against real dev data before it ships - and the evidence goes in the review.

2. Verify numbers independently

Tie the output to a source you already trust before anyone relies on it.

3. AI drafts, humans approve

Sign-offs, judgement calls, and anything external stay with a person.

4. If you cannot explain it, do not ship it

Read the code. Make the AI walk you through it until you could defend it in a review.

Proof of concept, not the end state

The orchestration works. The tools are not the ideal ones.

What we proved

- End-to-end orchestration can be automated, with controls
- Reruns and restatements fall when the manual hand-offs disappear
- Enforced review beats after-the-fact topside corrections
- The blueprint from chapter two is sound

Why it is not the end state

- Out-of-scope blocks still post as manual journal entries
- The warehouse still covers LDTI disclosure blocks only
- Built on team-level tools - scripts, wiki pages, drop zones - not an enterprise platform
- Closing the gap is a larger IT and automation effort

The real win: We stopped being the hamsters turning the wheel 24/7 - the team runs leaner, and the time goes to our other core functions.

The case for the ideal state is now made with evidence, not theory - a working, controlled process to point at, not a slide.



Where to start in your own shop

Painful, verifiable, and low blast radius

Painful and repetitive

People should want it gone. Early wins need cheerleaders.

Verifiable output

The right answer is checkable against something you already trust.

Low blast radius

If it is wrong, nothing downstream breaks while you learn.



Then climb the ladder - earn trust in this order:

1. **Documentation and recons**
2. **Internal tooling**
3. **Process automation**
4. **Production code, with review**

Key Takeaways

Thank you - questions?



1

Getting off the wheel took depth on demand, steered by judgment already on staff.

2

Three AI-assisted quarters finished what four hand-built ones started - the proof of concept for the full build.

3

The generalist is the ideal operator: sees the risks, judges the output, borrows the depth.

4

Start painful, verifiable, and low-risk - and let the results recruit the skeptics.

Heather Gordon | heather.gordon@fortitude-re.com