FIRST STRATEGY · CASE STUDY

SaaS AI

A real engagement, anonymized to industry label. The full case study: the story, the deliverables we produced, and the plays that ran it.

The company that sold what it did not use

The CEO said it before we could. His company, about 120 people, built an AI platform. The product was selling. Customers were ramping. His sales team demonstrated agents doing the work of whole departments, every week, convincingly, and then closed their laptops and went back to a company that ran on project boards, shared docs, and the connective tissue of people retyping things between them. The ask, when it came through a relationship that already existed, was one sentence: figure out AI for our operations, not just our product.

It is a more unusual sentence than it sounds. Most companies come to us wanting AI and not owning any. This one shipped it daily and did not run on it. Everyone outside the building would have assumed otherwise, which is exactly why the CEO wanted it fixed before anyone outside the building noticed. Underneath the credibility itch sat a harder constraint: demand was growing faster than the company wanted to grow headcount, and the internal teams, customer service, the SDRs, operations, marketing, were carrying enough bureaucratic load that the growth had nowhere to land. He did not want more people. He wanted the people he had swimming in the same direction.

We proposed the standard entry: a fixed fee, a day inside the operation, a playbook within two weeks. The day is usually spent on a floor. This floor was made of screens.

The floor was made of screens

You walk a software company's floor by sitting next to the people who live in it, watching what they actually do, and counting. So we sat.

We sat with support while the queue moved. A senior support lead answered a configuration question in four minutes flat, not because the answer was simple but because she had written it before. She keeps a personal spreadsheet of answers she has already written, organized by topic, polished over years. She pasted, adjusted two product names, sent. It was the third time that week for that particular question. The spreadsheet was the best knowledge system in the company, and it belonged to one person, and the company did not know it existed.

We sat with a project manager in the services operation, the team that delivers implementations for larger customers. Friday morning, her share of the team's [dozen] live implementations open in tabs, building the weekly status report by hand: read the boards, chase the threads where the real decisions lived, reconcile the two, write the summary the customer expects. The better part of a day, every week, multiplied across every project manager on the team. "Most of my Friday is producing this," she said, about a report describing work she had not had time to do because she was describing it.

We sat in a standing sync whose entire function, timed, was reading a board aloud to people who could not trust that the board was current. We watched the marketing and sales teams present two versions of

the same pipeline. We found the partner motion living in a spreadsheet, fed by whatever attention was left over, which was not much.

And we kept noticing the same thing the CEO had noticed: every piece of it, the re-answered questions, the hand-assembled status, the data retyped between systems, was work this company's own product does for its customers. The platform was sufficient. The platform was, in fact, down the hall. Nobody owned pointing it at the building it was built in. The product organization built for customers; internal deployment was nobody's job. That is not a technology gap. It is an ownership gap, and it does not show up on any dashboard.

The loop nobody had named

The audit's sharpest finding was not the glue. It was a loop, and it started outside the building.

We traced one unit of work end to end: a larger customer, from signed contract to steady state. The handoffs lost context. The boards got rebuilt by hand. The status went out weekly, assembled the way we had watched. But the step that paid for everything downstream was the quietest one: when it came time for the customer's users to learn the product, they got a recorded webinar and a documentation link. They were, structurally, on their own, learning to operate an AI platform the way you learn a microwave.

Then the queue inherited whatever self-teaching missed, as tickets, indefinitely. When we pulled the queue's exports apart, [roughly half] of it was questions already answered before, the same how-to and configuration questions, asked by customers who had never been taught and answered by a team re-writing known answers from memory and one woman's spreadsheet. Re-answering consumed [most of two] full-time people. Ramps stretched to [a few months], most of it waiting on people and learning, not on software.

So the loop, named in the audit: customers learn alone, so they lean on support and services. Support and services stay buried. Buried teams have no capacity to fix how customers learn. Leadership had been reading the queue as weather, a fixed cost of having customers, and reading the capacity problem as a hiring problem. The floor said both readings were wrong. The glue work across the four teams added up to [several] full-time people who were already on payroll, spent on status and re-answering and retyping. The headcount they thought they needed to hire, they already employed.

The playbook we delivered two weeks later said: run the company on its own platform, operations first, then the queue. And then the move the trace had earned, the one that breaks the loop instead of servicing it. Teach the customers. Not with more documentation. With the operating knowledge the company was about to earn on its own floor.

Becoming their own customer

The first build went where the blast radius was private: the services operation. Digest agents, running on the company's own platform, reading the boards, the threads, and the calendars together, producing each morning the truth the project managers used to assemble by hand on Fridays. For weeks the old motion ran beside the new one on purpose, the hand-built report next to the digest, until the comparison got boring. Then the operations lead retired the Friday report and the first standing sync, by decision, logged, which is how you retire a motion without leaving half of it running out of habit. The hours came back exactly where the audit said they were. Board hygiene and handoffs followed, agents carrying information between the work management platform and the office suite so that people stopped living logged into either. The systems stayed. The logging-in went.

The queue was the harder promotion, because the words face customers. Before any of it went live, we had run the cheap test: agents drafting answers to the recognized questions from the documentation and the closed-ticket record, offline, judged blind by the senior support lead against what she would have written. The drafts were good. That was not the finding. The finding was the pattern in the misses: every so often the machine answered confidently, fluently, and correctly for the previous release. Nothing about the draft looked wrong. That catch, made by the one person whose spreadsheet had been keeping the company honest for years, could have been an argument against the whole idea. It became the architecture instead: the knowledge layer was rebuilt versioned against the product, her library was codified into it, and the rule was written before a single customer saw a machine-drafted word.

Then the gate went up, and she changed jobs without changing desks. Every customer-facing draft passed through human approval, every catch logged with its pattern, every pattern turned into a rule. The catch rate fell week over week until the recognized answers earned lighter review on evidence, the same way the company asks its own customers to graduate their agents. The known half of the queue now drafts by machine in minutes. The hard half, the escalations and the judgment calls, gets the senior hours the easy half used to consume.

Governed turned out to be a different state than live, and the difference announced itself quietly. Months in, a team restructured its boards, ordinary housekeeping, columns renamed, and the digest reading them kept producing, confidently, from the stale map. Nothing errored. The numbers just stopped moving, and the operations lead caught it on the weekly read because a project everyone knew was sprinting looked still. The fix was bigger than the bug: every reading agent got schema checks and a standing rule, halt and ask rather than produce from a stale picture. An agent that stops is annoying. An agent that confidently describes a world that moved is dangerous, and the company now had the scar to teach that with.

The floor became the course

Which mattered, because by then the engagement had turned the deployment inside out.

Everything the internal build had produced, the playbooks, the gate designs, the graduation rules, the versioning rule, the stale-board scar, was authored into a training program that teaches customers how to operate AI inside their own businesses. Not a feature tour. An operator's education, built from the company's own floor, piloted with [a few] customers, revised on their feedback, and made standard for new ones. The curriculum's authority is its provenance: every lesson in it is something the company demonstrably does, because the course was extracted from the doing.

The support lead teaches in it. The configuration question she once answered three times a week from a personal spreadsheet is a module now, taught once to people who stop needing to ask it. Her spreadsheet retired into the knowledge layer; she went from writing the answers to judging the machine's, and from judging to teaching, which is the job the spreadsheet had been trying to be all along.

The loop the audit named runs the other way now. Trained customers ramp faster and lean lighter. A lighter queue gives the senior people their judgment hours back. And the company that sells an AI platform runs on it, every day, in operations and in customer success, which means the sales demo is no longer a demo. It is a tour. Customers ramp faster because the firm has actually walked the floor it sells.

Day One Proposal

Day One

Prepared for the CEO.

What this is

A day inside your operation. Real work, not slides.

We spend it with the people doing the work, not in a conference room about the work. The support queue while it is being worked. The services projects while they are being managed. The pipeline while marketing and sales argue about it. The partner motion wherever it lives. In a software company the floor is made of screens, and we walk it the same way we walk a plant: sitting next to the person, watching what they actually do, counting what it actually costs.

We also spend part of the day with you and your leadership. What the company sells, in your own words. What the operation is supposed to do and what it does instead. Where you believe AI fits, because your belief is a data point, and the floor gets a vote on it.

You ship an AI platform. Part of the day's work is finding out why the company that builds it does not run on it, and whether that is a technology problem or something else. We come ready to listen and to think on our feet. No prepared deck.

What you walk away with

A playbook, within two weeks. Not a deck. Not a recommendation memo hiding in a PDF. A written read for operators.

The playbook answers three questions:

- Where AI fits in your operation, and where it does not.
- The highest-leverage moves we see, sequenced so you can act on them in order. A roadmap, not a list.
- What it would take to run the sequence: with your own team, with another firm, or with us.

The playbook is yours. Run it however makes sense.

What we need from you

- The day itself: access to the people doing the work, at their desks, doing it. Support, the services project managers, sales and marketing, whoever holds the partner relationships.
- Your leadership for part of the day: you, and whoever owns operations and customer success.
- A look at the queue and the boards as they really run, including the workarounds nobody documents. The workarounds are usually the map.

The terms

A flat fee of [flat fee] for the day and the playbook. Travel and expenses billed at cost, on top.

No retainer. No commitment beyond the day itself. If we are the right fit for what comes next, we will already have been talking about what that looks like. If we are not, the playbook is still yours to run.

What happens next

After the playbook, you decide. Run it with your own team, hand it to another firm, or build it with us. If the work points to a build we are right for, we will scope it in a separate proposal once the playbook has shown what is worth building.

Day One Audit

The one-line finding

You are your own best-fit customer, and the only one you have never onboarded. The platform you sell automates exactly the work your own teams drown in: answering known questions, assembling status, moving information between systems, keeping two departments looking at the same truth. The gap is not technology, you own the technology. The gap is ownership: the product organization builds for customers, and nobody owns deploying it on the company itself. And the gap feeds a loop. Your customers learn the product on their own, so they lean on your support and services teams, so those teams stay buried, so nobody has the capacity to fix how customers learn. Run the company on its own platform, and the operating knowledge that comes out of doing it becomes the thing your customers have needed from the start: someone to teach them how to operate AI, by a company that actually does.

How we looked, and how we measured

A day inside the operation, with the people doing the work. In a software company the floor is made of screens, so we walked it by sitting next to the people who live in them: the support queue while it was worked, a services implementation while its status was assembled, the pipeline review while marketing and sales each presented their own version of it, the partner tracker in its spreadsheet. Leadership had the same day told from their side, and the floor got its vote.

The counting came from the floor and was checked after it. Ticket exports, board history, and calendar time gave the frequencies; the people doing the work gave the durations, and their estimates were conservative in every case we could verify. Where a figure below is bracketed, the real number is held in the engagement record or stated as the band the evidence supports.

What runs the work

The systems leadership names are not the systems that run the work. The work runs on the glue between them.

System	What it officially holds	What actually happens
The company's own AI platform	The product. Sold, demonstrated, deployed for customers daily	Absent from the company's own operation
The work management platform	Projects, boards, the support queue, the pipeline	The boards are accurate the morning after someone spends hours making them accurate
The office suite	Documents, mail, meetings	Where the real state of the work lives: threads, comments, and meeting notes that never reach the boards
Spreadsheets on the side	Nothing, officially	Partner tracking, a support lead's personal answer library, list-building for outbound. The workarounds are load-bearing
Heads	Nothing, officially	The product knowledge that resolves the hard tickets, the project judgment that catches slipping work, the institutional memory of every renewal

The fifth row is the risk. The second through fourth rows are the tax. The first row is the unused fix for most of both.

Stakeholder map

Each leader holds a different piece of the problem. None is wrong. None alone is sufficient.

Role	What they own	Where the pain lands	Their definition of the problem
CEO	The company and the gap itself	Credibility and capacity: selling AI while running on glue	Figure out AI in our operations, not just our product
Head of customer success	The queue and the ramps	The same questions answered again and again; senior people pulled into tickets	The load is the problem; more demand means more people
Head of operations	The services delivered to larger customers	Project managers producing status instead of managing projects	The reporting is the problem; customers expect a cadence we assemble by hand
Sales and marketing leadership	The pipeline	Two versions of the same pipeline; leads cooling in the handoff	Alignment is the problem; the data never says one thing
Partner lead	The partner motion	Repeated onboarding, referral tracking by hand, co-marketing dates slipping	Attention is the problem; partners get the hours left over

The gap that matters is between the CEO's framing and the floor's reality. Leadership asked where AI fits, a technology question. The floor showed a technology already sufficient and a deployment that had never been anyone's job.

One unit of work, traced end to end

We traced the unit of work that touches every team: one larger customer, from signed contract to steady state.

1. The deal closes. Sales hands off in a meeting; the context that does not survive the meeting is lost to the next team. Error enters here.
2. Operations builds the implementation project by hand: the board, the plan, the dates, rebuilt fresh each time from the last project's copy.
3. Kickoff. The customer meets a new team and repeats what they already told sales.
4. The implementation runs. Real state lives in threads and calls; the board catches up when the project manager spends [the better part of a day each week] reconciling it.
5. Status goes to the customer on a weekly cadence, the product of that same hand-built day: the board, the threads, and memory, written up for the customer.
6. The customer's users get trained: a recorded webinar and a documentation link. They are, structurally, on their own. Most of what goes wrong downstream enters here.
7. Go-live. The support queue inherits whatever the self-teaching missed, as tickets, indefinitely.

8. Customer success intervenes on the accounts that struggle loudest. The quiet ones struggle unmeasured.
9. Steady state arrives in [a few months], later than the product requires, because most of the ramp was waiting on people and learning, not on software.

Steps 1 through 5 are the company's time. Steps 6 through 9 are the customer's experience. The trace says the expensive failure is step 6: a product that customers must operate, handed over without an operator's education. Everything after it pays for it.

The friction, quantified

What a day of watching and a week of exports support. Shares are of recoverable working time across the four teams.

Friction point	Frequency and cost	What it tells us
Known answers re-written in the queue	[Roughly half] the ticket volume is questions already answered before; re-answering consumes [most of two] full-time equivalents	The knowledge exists and is unreachable: closed tickets, threads, one lead's personal spreadsheet
Status assembled by hand	[A dozen] concurrent implementations; each project manager spends [the better part of a day each week] producing reports about the work instead of doing it	The information already exists in the boards and threads; assembling it is machine work
Meetings that exist to sync the boards	[A handful of] standing syncs a week whose function is reading state aloud to people who could not trust the boards	Coordination cost paid in senior hours, producing no decision
The marketing-to-sales seam	Leads waiting [days] for first touch; SDRs spending [a couple of hours] of research per qualified account; two pipelines that disagree	The seam is data glue, and it cools revenue while it waits
Partner operations on the side	Onboarding repeated by hand, referrals tracked in a spreadsheet, co-marketing slipping	A growth channel rationed by leftover attention

The cumulative read: the connective-tissue work across these teams adds up to [several] full-time equivalents. The capacity leadership assumed it would have to hire already exists on the payroll. It is spent producing status, re-answering answered questions, and moving information between systems by hand.

What the floor disproved

Three beliefs walked into Day One. None survived it.

- **"The problem is finding where AI fits."** The platform's own demo scenarios map one-to-one onto the floor's friction: drafting known answers, assembling status, watching boards, routing handoffs. The fit was never in question. Nobody owned the deployment.
- **"The support load is product weather."** The queue is treated as a fixed cost of having customers. [Roughly half] of it traces to customers operating a product nobody taught them to operate. The load is a symptom with a cause, and the cause is upstream, at step 6 of the trace.
- **"Scaling means hiring."** The glue work across the four teams already equals [several] full-time people. The headcount conversation was a capacity conversation, and the capacity is recoverable.

Where AI fits, and where it does not

- **Fits: the queue's known half.** Draft answers from the documentation and the closed-ticket record, with a human approving every customer-facing word. The platform does this for customers today.
- **Fits: status, assembled continuously.** Project digests built from the boards, the threads, and the calendar, read by the project manager instead of written by them.
- **Fits: the glue between systems.** Handoffs, board hygiene, data moving between the work management platform and the office suite without a person retyping it.
- **Fits: the revenue seam.** One pipeline read both teams trust, and account research done by machine before an SDR spends an hour on it.
- **Does not fit: the hard tickets.** The judgment calls, the angry customer, the edge-case diagnosis. That is what the recovered senior hours are for.
- **Does not fit: the relationships.** Customers, partners, renewals. People keep these; the machine clears the calendar for them.
- **Does not fit, yet: anything customer-facing without a human gate.** Trust is earned by catch rate, not granted by enthusiasm. The company should hold its own deployment to the standard it asks of its customers.

Risks and constraints we observed

- The credibility risk runs both directions. A platform company that automates its own operation owns the strongest story in its market. One that tries and quietly fails owns the worst. The internal deployment must be run like a customer deployment, with an owner, a baseline, and success criteria, not like a side project.
- The teams are not afraid of the machine; they are tired of the glue. The risk is not resistance. It is half-adoption: agents bolted on while the meetings and hand-built reports continue out of habit. Retire the old motion when the new one proves out, deliberately.
- The knowledge the queue depends on is concentrated in a few people, one personal spreadsheet among them. Codifying it is the move; losing one of those people first is the race.

- Replatforming is the seductive wrong move. The work management platform and the office suite are fine. The friction is between and around them. Keep the systems, remove the logging-in.

The signal we leave with

The strongest signal of the day: the company already owns everything it needs except the decision. The platform is sufficient. The friction is measured. The people who would govern the agents already exist, and the one who keeps a personal library of answers has effectively been training the system by hand for years, without the system.

The first move is internal, operations first, because the blast radius is private while trust is earned, and the density of recoverable hours is highest there. Customer-facing work follows once the catch rate proves out. And the move after that is the one that fixes the trace at its source: turn the operating knowledge the internal deployment produces into the education customers never got. The plan, sized by impact, is the Playbook and Delivery Proposal.

Playbook and Delivery Proposal

The playbook and the delivery proposal are one document because they are one act. The playbook says where AI fits and sizes the moves in order. The delivery proposal scopes the build for the moves you choose to start with. The first earns the second. Nothing past the first move is committed until the first move proves the approach in your business.

Part One: The Playbook

A written read for operators, not a deck. It answers three questions: where AI fits in your operation and where it does not, the highest-leverage moves in sequence, and what it takes to run them.

Where AI fits, and where it does not

It fits the glue. The Day One Audit measured [several] full-time equivalents of connective-tissue work across customer service, the services operation, sales and marketing, and the partner motion: known answers re-written, status assembled by hand, meetings that read boards aloud, data retyped between systems. All of it is work your own platform automates for customers today. It does not fit the judgment: the hard tickets, the relationships, the renewals, the calls that need a person. The point of removing the glue is to give the judgment its hours back.

One thing makes this engagement unusual, and the roadmap uses it twice. You own the machine. The build runs on your own platform, which means every internal deployment is also product evidence, and the knowledge of how to operate it is itself worth money to your customers. The roadmap's third move turns that knowledge into a product.

How to read the roadmap

The first two moves we diagnosed on Day One and sized against the floor's own counts. The third is diagnosed at its root, the audit's trace shows where the customer ramp breaks, and its full shape firms up as the first two moves produce the material it teaches from. The last two we saw the shape of and did not diagnose, and we say so rather than dress them up.

Each move is read across six dimensions: time, accuracy and quality, cost and recovered revenue, growth, employee experience, and risk. The first move earns the right to the next.

The roadmap at a glance

#	Move	Status	Leverage	Containment	Why it sits here
1	Run the company on its own platform: operations first	Diagnosed, sized	Highest	Internal only, one team at a time	Highest density of recoverable hours, private blast radius, and it makes the company its own reference customer
2	Customer success on the platform	Diagnosed, sized	High	Every customer-facing word human-approved	The queue's known half, drafted by machine, gated by people. Needs move 1's trust first
3	The training program: teach customers to operate AI	Root diagnosed	High	One curriculum, piloted with [a few] customers	Fixes the ramp at its source. Built from what moves 1 and 2 teach the company about operating its own product
4	One revenue motion: the marketing-to-sales seam	Candidate, partly diagnosed	Medium	One pipeline, agreed definitions	The seam cools leads and burns SDR hours. Earns its turn after the internal pattern proves
5	Partner operations on the same machinery	Candidate, not yet diagnosed	Medium	One partner cohort	Real friction, smallest measured stake on Day One. Sized when its turn comes

Move 1: Run the company on its own platform, operations first (start here)

Deploy your own agents on your own operation. The services PMO first: project digests assembled continuously from the boards, the threads, and the calendar, so status is read, not produced. Board hygiene and handoffs carried by agents between the work management platform and the office suite, so information moves without a person retyping it. The standing syncs that existed to read state aloud get retired as the digests earn trust, deliberately, not by attrition.

- **Time:** [the better part of a day each week] returned to each project manager; the sync hours returned to every team the syncs taxed. The audit's count puts the recoverable glue at [several] full-time equivalents across the four teams; this move recovers the densest share of it.
- **Accuracy and quality:** status that reflects the work as of this morning, not as of the last reconciliation. One version of the truth, read by everyone, argued by no one.
- **Cost and recovered revenue:** the capacity conversation changes. Growth gets absorbed by recovered hours instead of new hires, which was the constraint that brought us in.
- **Growth:** every internal deployment is product evidence. The sales team demos workflows the company runs on, live, not staged.

- **Employee experience:** project managers manage projects. The work people were hired for becomes the work they do. This is the swimming-in-the-same-direction the CEO asked for, made structural.
- **Risk:** lowest on the board. Internal audience, read-only digests before write actions, every automation reversible, and the old motion retired only when the new one proves out.

Move 2: Customer success on the platform

The queue's known half, drafted by machine. Agents draft answers to recognized questions from the documentation and the closed-ticket record; a person approves every customer-facing word; every catch is logged and the knowledge layer learns from the log. The personal answer library that one support lead built by hand becomes the seed of a governed knowledge layer, versioned against the product so the machine cannot confidently answer for the wrong release. Senior people stop being the queue's memory and become its judges.

- **Time:** [most of two] full-time equivalents of re-answering, recovered. First-response time on known questions drops from hours to minutes.
- **Accuracy and quality:** answers drawn from one governed source instead of from whoever is on shift. The catch log makes the machine measurably better every week, and the version-tagging makes it honest.
- **Cost and recovered revenue:** support absorbs customer growth without headcount growth. The senior hours recovered go to the accounts that need judgment, which is where renewals are saved.
- **Growth:** the deployment is the demo. A prospect asking whether the platform handles support gets shown the company's own queue.
- **Employee experience:** the people who answered the same question three times in a week stop doing that. The work left over is the work that needs them.
- **Risk:** moderate and gated. Customer-facing words ship only through a human gate while trust is earned, the same standard the company asks of its own customers. Graduation by catch rate, never by decree.

Move 3: The training program: teach customers to operate AI

The audit's trace found the expensive step: customers get a recorded webinar and a documentation link, then operate an AI platform on their own. [Roughly half] the queue and the slow ramps trace back to that step. The fix is not more documentation. It is an operator's education: a training program that teaches customers to run AI inside their businesses, built from the company's own deployment. Moves 1 and 2 produce the material: the playbooks, the governance patterns, the human-in-the-loop designs, the mistakes and their fixes, learned on the company's own floor. The curriculum teaches what the company now does, not what the product theoretically supports.

- **Time:** customer ramp to steady state compressed; the [few months] of waiting-on-people in the trace is the target. Tickets fall at the source instead of being answered faster.
- **Accuracy and quality:** customers operate the product the way the company operates it, on patterns proven internally, not on guesses.
- **Cost and recovered revenue:** the queue's known half shrinks structurally. Faster ramps pull revenue recognition and expansion forward; trained customers renew on results instead of on patience.
- **Growth:** this is the move that compounds. A trained customer base ramps faster, leans lighter, and expands sooner, and the program itself is a product the market was not offering.
- **Employee experience:** the support lead who kept the answer library becomes the person who teaches, which is the job her spreadsheet was always trying to be.
- **Risk:** low and honest. The curriculum can only teach what the internal deployment has proven, which is exactly why it sequences third, not first.

The later moves: named, not fully diagnosed

- **Move 4: One revenue motion.** One pipeline both teams read, definitions agreed once and enforced by the system, leads touched in [hours, not days], account research done by machine before an SDR spends an hour on it. Partly diagnosed: the seam's cost showed on Day One; the fix's design needs the deeper look it gets when its turn comes.
- **Move 5: Partner operations.** Partner onboarding, referral tracking, and co-marketing cadence on the same machinery. Real friction, smallest measured stake of the day. It earns its diagnosis after the moves ahead of it prove out.

Each is a contained bet with its own measurable result. None is committed now.

What it takes to run the moves

The discipline matters more than the technology, and in this engagement the technology is already owned.

- **Give the deployment an owner.** The gap was never capability; it was that internal deployment was nobody's job. It becomes someone's job, with a baseline and success criteria, run like a customer engagement.
- **Baseline before building.** The audit's counts are the before. Every move is measured against them, or it is an anecdote.
- **Keep a human on every customer-facing word.** Trust is earned by catch rate. The company holds itself to the standard it asks of its customers.
- **Retire the old motion deliberately.** A digest that nobody trusts yet changes nothing; a digest everyone trusts while the sync meeting continues changes half of nothing. Prove, then retire.

- **Write down what the deployment teaches.** Every pattern, catch, and fix feeds the knowledge layer and, through move 3, the curriculum. The operating knowledge is the second product.

The plays that run each canon come from our reusable plays library. The ones selected for this engagement are instantiated in the Charter. ## Who runs it

This can run with your own team, with another firm, or with us. It needs a few clear accountabilities: an executive sponsor who clears the way, an owner for the internal deployment, the support and operations leads as the gates on what ships in their domains, and builders who know the platform. You have the platform engineers and the judgment seats. The method, the operating discipline, and the outside eyes that walked the floor are the pieces you would bring in.

The recommended first move and the 90-day frame

Start inside, in operations. The first 90 days: the baselines confirmed from the queue and board history, the first agents live in the services PMO with digests read against the hand-built reports they replace, the first standing sync retired on evidence, and the support knowledge layer seeded from the closed-ticket record and the answer library, drafting offline while its catch rate is measured. Prove the pattern on the company's own floor, and the operation gains not just recovered hours but the standing to run move 2 on customers and the material to build move 3 for them.

Part Two: The Delivery Proposal

The proposal to build the playbook's first moves: the internal deployment in operations, then customer success under a human gate, with the training program built from what they prove. Scoped only after the playbook showed what is worth building.

What we understand

A 120-person SaaS company shipping an AI platform it does not run on. Demand growing faster than the company wants to grow headcount. [Several] full-time equivalents of glue work measured across four teams, and a customer ramp that breaks at an education step nobody owns. The technology in hand; the deployment never anyone's job.

What we will build

A company that runs on its own product, and the proof it happened. Agents on the services operation: continuous digests, board hygiene, handoffs without retyping. A governed knowledge layer for the queue, seeded from the closed-ticket record, versioned against the product, drafting under a human gate. The

measurement spine: the audit's baselines, read weekly against the live counts. Then the training program, built from what the deployment proved, piloted with [a few] customers and made standard for new ones.

How we will work

Four phases, mapped to the WISER canons. Each phase is independently valuable, priced on its own, and earns the next. The engagement can stop at any phase boundary with value already in hand.

Phase 1: Interrogate

Cheap tests before any production build.

- Tag [a week] of tickets to confirm the queue's repeat share and its top recurring questions.
- Draft answers offline against the closed-ticket record; measure the catch rate before anything faces a customer.
- Run a digest agent read-only on [two] live project boards; compare it to the hand-built report it would replace.
- Confirm the recoverable-hours baseline from exports and calendars.
- End of phase: the assumptions tested, the baselines hard, the wrong instincts ruled out for the cost of a few weeks.

Phase 2: Solve

The first working system, internal, in the services operation.

- Digests live for the PMO, read daily, measured against the baseline.
- Board hygiene and handoff agents carrying information between the systems people stop logging into.
- The first standing sync retired on evidence, deliberately.
- End of phase: operations running on the company's own platform, with measured hours recovered and the pattern proven.

Phase 3: Expand

The proven pattern, carried to the customer-facing work.

- The knowledge layer live in the queue: recognized questions drafted by machine, every word approved by a person, every catch logged.
- The answer library codified and versioned; senior support hours moved from re-answering to judgment.
- The same machinery extended across the remaining operations friction as each context earns it.

- End of phase: the queue's known half drafted by machine under a human gate, with the catch rate published internally.

Phase 4: Refine

Govern the system, and turn what it taught into the customer curriculum.

- Oversight tiers by risk, graduation by catch rate, drift watched on a cadence: the same governance the company will teach.
- The training program authored from the deployment's playbooks and catches, piloted with [a few] customers, then made standard for onboarding.
- The operating rhythm set: one weekly read of the counts that matter.
- End of phase: a governed internal deployment, and a curriculum that teaches customers to operate AI the way the company now demonstrably does.

What we need from you

- The CEO's mandate, visible, and an internal owner for the deployment.
- The support and operations leads as gates on what ships in their domains, with the hours to actually review.
- Your platform engineers' time for the build, because it runs on your product and that is the point.
- The queue exports, board history, and calendars that make the baselines hard.
- A weekly read, kept.

Infrastructure

You provide the platform, the seats, and access to the systems the agents work across. We provide the method, the build leadership, the governance design, and the curriculum's authoring.

Who is working on this

A senior practitioner who leads the engagement and owns the system design with your team. Your platform engineers build on their own product with us; your leads hold the gates. Small team, close to the work.

Investment

Phased. Each phase is priced on its own so the engagement can stop at any phase boundary with value already delivered. The fee basis and amounts are held in the private engagement record. We did not fabricate figures for this anonymized record.

Charter

What a Charter is

Not a project plan. Not a requirements document that executes once and collects dust. A Charter is the memory that survives the chaos. Its value is the decision log: when someone asks a year later why the deployment started in operations instead of the louder pain in the queue, or why the company kept its work management platform instead of replatforming onto its own product, the answer is here, with the alternatives that were weighed and the evidence that settled it. The Architect keeps it current, same-day.

Metadata

Field	Value
Project	Run the company on its own platform, and turn the operating knowledge into the customer training program
Client	The SaaS AI platform company (anonymized)
Charter Keeper	The Architect
Dates	Held in the private engagement record; relative markers used here
Current canon	Refine. Operations and customer success run on the platform; the training program is live with customers; the wider system leadership envisioned continues to build
Version	Running state

Positions

The work was held together by clear accountabilities, not an org chart.

Position	Who held it	Tension owned
Sponsor	The CEO	Authority. Owned the why, named the gap, cleared the way
Guide	First Strategy senior practitioner	Translation. Carried the method and kept the Charter honest
Architect	First Strategy	Curiosity and stewardship. System design and Charter Keeper
Sage	The head of customer success and the senior support lead	Context. What the queue actually contains and what resolves it
Scout	[A few] customers in the training pilot	Empathy. Validated whether the curriculum actually changes how a customer operates
Builder	The client's platform engineers, with First Strategy	Execution. The agents, the knowledge layer, the digests, built on the company's own product
Operations gate	The head of operations	Safety. Approved every internal automation and chose when an old motion retired
Customer gate	The head of customer success	Integrity. Approved every customer-facing word before it shipped

On a small team one person holds several Positions. As the system proved reliable, a Position could be augmented by an AI agent inside documented constraints, with the human shifting from doing to directing and reviewing.

Objectives and constraints

The build specification: what the project set out to do and the lines it would not cross.

Scope

In scope: the company's internal operation, the services PMO, the support queue, the glue between the work management platform and the office suite, and the training program built from the deployment. The agents run on the company's own platform; the build is also product evidence. Out of scope: the product's roadmap, the customer-facing platform itself, and the sales and partner motions until their moves earn their turn.

Objective and success criteria

Run the company on its own platform, recover the glue hours, and turn the operating knowledge into the education customers never had.

Measure	Baseline	Target	Result
The company on its own product	Absent from its own operation	Operations and customer success running on the platform daily	Live. The company runs on its own product every day
Glue hours	[Several] full-time equivalents across four teams	Recovered to judgment work; growth absorbed without matching hires	Recovered in operations and the queue; stated directionally pending cleared figures
Queue re-answering	[Roughly half] the volume, [most of two] FTEs	Known questions drafted by machine under a human gate	Live; the known half drafts by machine, catch rate published internally
Status production	[The better part of a day each week] per project manager	Status read, not produced	Digests live; the hand-built weekly report retired
Customer ramp	Steady state in [a few months], self-taught	Trained customers, faster ramps	The training program is live; customers ramp faster

Constraints

- Every customer-facing word passes a human gate while trust is earned, and graduates only on catch rate. The company holds itself to the standard it asks of its customers.
- Keep the systems, remove the logging-in. No internal replatforming; the agents work across what already runs.
- No old motion retires until its replacement proves out. Half-adoption is the named failure mode.
- The internal deployment runs like a customer engagement: an owner, a baseline, success criteria, a cadence.
- Headcount absorbs growth only where judgment requires it, never to patch glue.

Architecture and human-in-the-loop design

The company's own platform sits at the top, running the agents. Below it, three working layers. The digest layer reads the boards, the threads, and the calendars, and produces the operational truth people used to assemble by hand. The knowledge layer holds what the queue knows: the documentation and the closed-ticket record, codified from the heads and the side spreadsheets that held it, versioned against the product so an answer cannot ship for the wrong release. The glue layer moves information between the work management platform and the office suite so people stop retyping it.

Above all of it, two human gates. The operations gate approves internal automations and decides when an old motion retires. The customer gate reads every customer-facing word against the knowledge layer

before it ships. Every catch is logged with its pattern; the knowledge layer learns from the log; graduation to lighter review happens on catch-rate evidence, never by decree.

Current state at the start

Carried from the Day One Audit. A company of about 120 people shipping an AI platform and running itself on hand-glue: [roughly half] the queue answered from memory and one lead's personal spreadsheet, project status assembled by hand [the better part of a day each week] per project manager, standing meetings that read boards aloud, leads cooling [days] in the marketing-to-sales seam, partners managed from attention left over. Customers learning the product alone from a webinar recording, and the ramp paying for it. The technology sufficient and owned. The deployment nobody's job.

Decision log

The decisions that shaped the build, each with the alternatives weighed and the evidence that settled it. This is the part of the Charter that answers "why did we do it this way."

When	Decision	Alternatives rejected	Rationale	Evidence
Interrogate, wk 1	Run the company on its own platform before telling the story anywhere	A marketing-led "we run on our product" narrative first	Credibility runs both directions; a told story without a run floor is the worst outcome in the market	The audit's credibility-risk read; leadership concurrence
Interrogate, wk 1	Start in operations, not in the queue	Start with support, the loudest pain	The queue touches customers; trust is earned on a private blast radius first, and the PMO held the densest recoverable hours	The friction counts; the gate design not yet proven
Interrogate, wk 2	Keep the work management platform and the office suite; automate across them	Replatform the internal operation onto the company's own product	The friction was the glue between systems, not the systems; replatforming spends months for no customer value	The Day One trace; the floor's own counts
Interrogate, wk 2	Baseline from exports before building	Build first, measure impressions later	Without the before, every result is an anecdote	Queue exports, board history, calendars
Interrogate, wk 3	Seed the knowledge layer from the closed-ticket record and the answer library	Author a knowledge base by hand	The knowledge already existed and was unreachable; authoring fresh discards years of answered questions	The support lead's spreadsheet; the repeat-share tag
Solve, wk 1	Digests read-only until trusted, then write actions	Full automation from day one	Trust is earned by parity with the hand-built reports, then by catches	The digest-versus-report comparison
Solve, wk 3	Retire the first standing sync on evidence, by decision	Let meetings fade by attrition	Half-adoption is the failure mode; the old motion ends when the new one provably carries it	The operations gate's call, logged
Expand, wk 1	Version the knowledge layer against the product before customer-facing drafting scales	Ship drafts from the unversioned corpus	The gate caught confident drafts answering for the wrong release	The catch log's first pattern
Expand, wk 2	Every customer-facing word through the customer gate;	Auto-send the known answers	The standard the company asks of its customers, applied to itself	The gate's catch rate, read weekly

When	Decision	Alternatives rejected	Rationale	Evidence
	graduation by catch rate			
Refine	Turn the deployment's playbooks into the customer curriculum	Keep training as billable services hours; more documentation	The trace broke at self-teaching; documentation was already losing; operating knowledge was the scarce thing	The ramp evidence; the pilot cohort's read
Refine	Teach customers to operate, not just to use	A feature-tour curriculum	Customers fail at governance and method, not at buttons; the company now had a proven method to teach	The pilot feedback; the queue's question mix

The decision and experiment record

The supporting narrative behind the log. The project ran the full WISER method. Witness was Day One; the project picked up at Interrogate and ran through Refine.

Interrogate

The first weeks were spent making the audit's counts hard and the instincts testable. A week of tickets was tagged by hand: the repeat share held at [roughly half], and the top recurring questions were fewer and more concentrated than anyone expected, which made the knowledge layer's first cut smaller than feared. The drafting test ran offline: agents drafted answers to the recognized questions from the documentation and the closed-ticket record, and the senior support lead judged them against what she would have written. The drafts were good, and one pattern in the misses mattered more than the hits: the machine would answer confidently from material that belonged to an older release. That catch, before a single customer saw a draft, set the versioning rule the knowledge layer was built on.

The digest test ran read-only on [two] live project boards. The comparison was the project manager's own Friday report, and the first digests lost: they could see the boards but not the decisions living in mail threads. Wiring the office suite into the read fixed it, and the fixed digests matched the hand-built reports closely enough that the project manager said the quiet part out loud: most of my Friday is producing this.

Solve

The first working system went live in the services PMO. Digests, daily, assembled from the boards, the threads, and the calendars, read by the people who used to write them. Board hygiene and handoff agents followed, moving information between the systems without a person retyping it. For weeks the old motion ran alongside the new one on purpose, the hand-built report beside the digest, until the parity was

boring. Then the operations gate retired the Friday report and the first standing sync, by decision, logged. The hours came back exactly where the audit said they were.

Expand

The proven pattern moved to the queue, where the stakes change: the words face customers. The knowledge layer went live drafting answers to recognized questions, every word read by the customer gate before it shipped. The senior support lead's spreadsheet, the library she had built by hand for years, was codified into the layer and versioned against the product, and she moved from writing the answers to judging them. The catch log did its job: each miss became a pattern, each pattern a rule, and the catch rate fell week over week. The known half of the queue now drafts by machine; the hard half gets the senior hours the easy half used to consume.

Refine

Live and governed are different states, and the difference is the cadence. Oversight tiers were set by risk, graduation thresholds written down, and the weekly read established: the counts against the baselines, the catch rates by content type, the drift checks on the layers underneath.

Then the engagement turned the deployment inside out. Everything the internal build had produced, the playbooks, the gate designs, the graduation rules, the catches and their fixes, was authored into a training program that teaches customers to operate AI inside their own businesses, the way the company now operates it inside its own. It was piloted with [a few] customers, revised on their feedback, and made standard for new customers. The curriculum's authority is its provenance: it teaches what the company does, demonstrably, on its own floor.

Hierarchy of Agency

Three tiers of human oversight on what the machine produces, by risk. The gates own the tiers; the tier governs how much of the gate's attention each output type gets.

Tier	Oversight	Applies to
1: Light review	Spot checks on a cadence; the output ships without a per-item gate	Internal digests and board hygiene after parity proved; recognized-answer drafts whose catch rate stayed near zero across a full cycle
2: Full review	Every item read before it ships	Customer-facing drafts on newer question types; status that leaves the building; changes to the knowledge layer
3: Human-led	The machine assists; a person authors and decides	The hard tickets, escalations, anything touching pricing, contracts, renewals, or a partner commitment

An output type moves to a lighter tier only on evidence: a full review cycle in which the gate's catch rate on that type stays near zero. Recognized-answer drafts earned Tier 1 that way. Escalations never will, by

design. If a tier drifts, it falls back to heavier review. A human is accountable for every customer-facing word, at every tier.

Risk register

Risk	Mitigation	Status
The credibility story runs backwards: an internal deployment that quietly fails	Run like a customer engagement: owner, baseline, success criteria, weekly read	Held; the deployment is the demo now
Half-adoption: agents live while the old meetings and reports continue	No motion retires until its replacement proves; retirement is a logged decision	Held; the Friday report and the first sync are gone, on evidence
The machine answers from the wrong product version	The knowledge layer versioned against releases; version checks in the gate's read	Realized once, offline, in Interrogate; the versioning rule exists because of it
Queue knowledge concentrated in a few heads and a spreadsheet	Codified into the governed knowledge layer; the owner became its judge and teacher	Resolved; the spreadsheet retired into the layer
Customer-facing drafts erode trust	Every word gated while trust is earned; graduation by catch rate only	Active control; catch rate falling by type
The digests go quietly stale as boards change shape	Schema checks, a trigger list, and the weekly drift read	Realized once, post-launch; caught and fixed; see the drift record
The curriculum drifts from what the floor actually does	The program teaches the live deployment; revised on the same cadence the deployment evolves	Active control; pilot feedback wired in

Drift and incident record

The system's one post-launch incident earns its place in the record because of how quietly it arrived. A team restructured its boards, renamed columns, split a group, ordinary housekeeping, and the digest that read those boards kept producing, confidently, from the stale mapping. Nothing errored. The numbers just stopped moving. The operations gate caught it on the weekly read: a project everyone knew was sprinting showed a digest that had not changed in days.

The response:

Action	Detail
Contain	The affected digests flagged and read manually while the mapping was rebuilt
Fix	Schema checks added: a digest that reads a changed board structure stops and says so instead of producing from the stale map
Generalize	A trigger list written: the conditions that force an agent to halt and ask, board restructures and release changes first among them
Teach	The incident, its catch, and its fix went into the training program, because customers' boards get restructured too

The lesson logged: an agent that stops is annoying, and an agent that confidently produces from a stale picture is dangerous. Build them to stop. Watch the aggregates for the numbers that stop moving, because silence is a signal.

Evolution history

How the oversight posture changed over time, and why.

When	Change	Trigger
Solve	Digests read-only, every output compared to the hand-built report	Trust not yet earned
Solve, late	The Friday report and the first standing sync retired	Parity proved and boring; the gate's logged decision
Expand	Customer-facing drafts at full review, every word gated	New stakes; no track record on customer-facing words
Expand, late	Recognized-answer drafts graduated to light review	A full cycle with the catch rate near zero on the type
Refine	Schema checks and the trigger list on every reading agent	The stale-digest incident
Refine	The deployment's playbooks authored into the customer curriculum, revised on pilot feedback	The training pilot's read: customers fail at method, not buttons

Current status and what transfers

The company runs on its own product every day. Operations runs on digests people read instead of reports people write; the known half of the queue drafts by machine under a gate whose catch rate is published internally; the glue between the work management platform and the office suite is carried by

agents, and the need to live logged into those systems all day is gone. The training program is live: built from the company's own deployment, piloted with [a few] customers, standard for new ones. Customers ramp faster because the company has actually walked the floor it sells.

The judgment seats are the client's and always were. The gates are theirs, the knowledge layer is theirs, the curriculum is theirs, and the decision log is theirs. The work continues toward the full system leadership envisioned, with the same discipline this Charter records.

Outcomes

- The company runs on its own platform, daily, in operations and customer success. The deployment is the demo.
- The glue hours recovered: status read instead of produced, known answers drafted by machine, information moving between systems without retyping. Stated directionally pending cleared figures.
- The hand-built Friday report and the first standing syncs retired on evidence, not attrition.
- The queue's known half drafts by machine under a human gate; senior support hours moved to the tickets that need judgment.
- The knowledge that lived in heads and a personal spreadsheet now lives in a governed, versioned knowledge layer.
- The training program live with customers: an operator's education built from the company's own floor. Customers ramp faster.
- Growth absorbed by recovered capacity rather than matching headcount, which was the constraint that brought us in.

Plays

The WISER plays this engagement ran, instantiated with the client's specifics. This is the index and what each produced. The high-value plays are held as standalone documents; the rest were applied inline in this Charter. | Canon | Play | What it produced | Source | |-------|------|------------------|--------| | Witness | Friction Mapping | The internal friction map, counted from the queue, the boards, and the calendars | Standalone play | | Witness | User Flow Mapping | The signed-contract-to-steady-state trace and its broken education step | Inline in the Day One Audit | | Witness | Documenting Current State | The what-runs-the-work read, workarounds included | Inline in the Day One Audit | | Interrogate | Assumption Auditing | The register of beliefs tested: hiring, replatforming, the queue as weather | Standalone play | | Interrogate | Experiment Selection, Logging | The ticket tag, the offline drafting test, the read-only digest trial | Standalone play | | Solve | Human-in-the-Loop Design | The two-gate system and the catch log | Standalone play | | Solve | Quality Objective Setting, Value Validation | The baselines and the parity standard for retiring old motions | Inline above | | Expand | Expansion Sequencing, Context Fit | Operations to queue to curriculum, each context earning the next | Inline above | | Refine | Drift

Monitoring, Incident Response | The stale-digest catch, the schema checks, the trigger list | Standalone play | | Refine | Hierarchy of Agency Design, Graduation | The tiers and the catch-rate thresholds that move a type between them | Inline above |

The deployment is built, and it was never only a deployment. The company that could not find time to onboard itself now teaches its customers from the experience of having done it. The next moves are named in the playbook, the revenue seam and the partner motion, and the machine that earns them is already running.

The plays

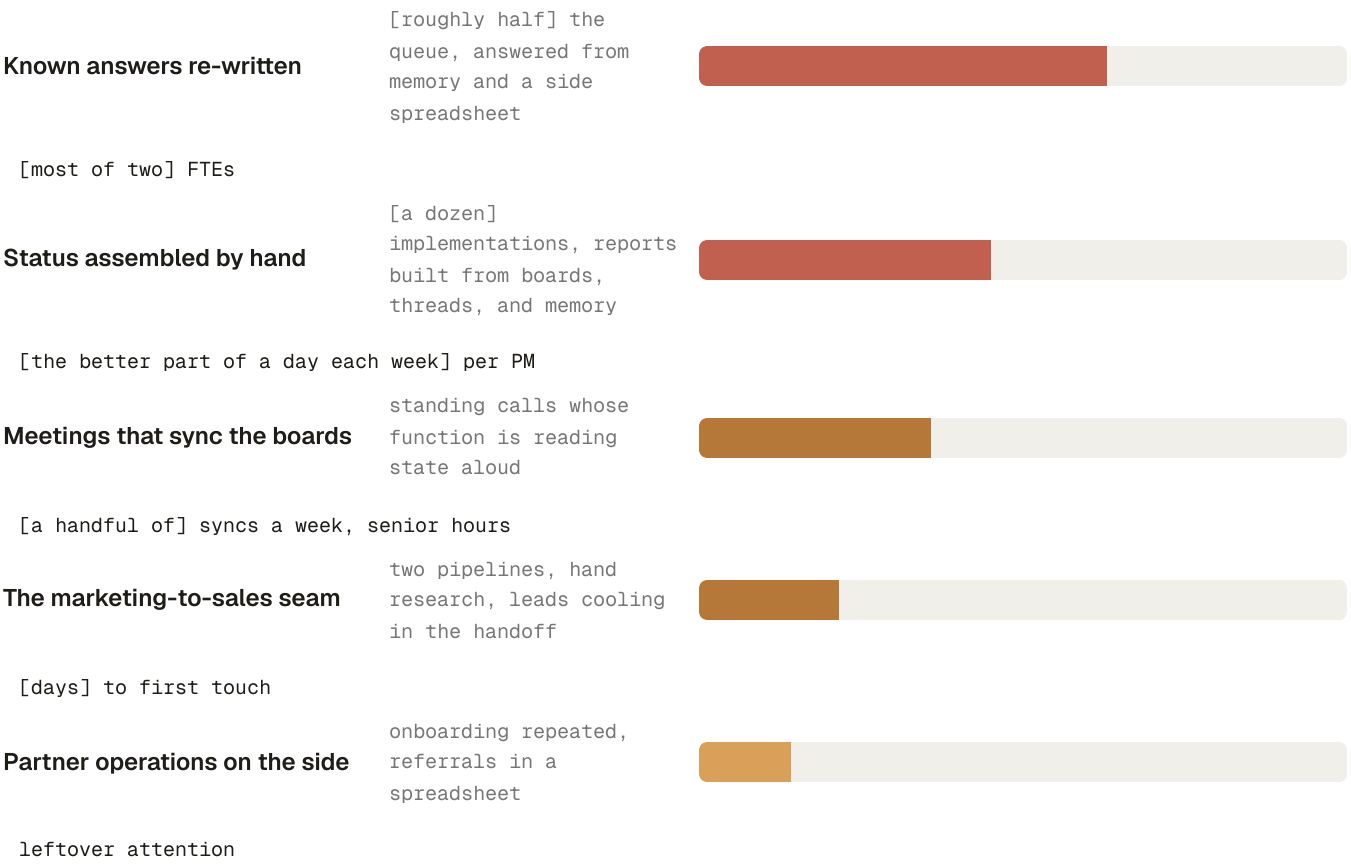
The WISER plays this engagement ran, instantiated with the client's specifics, ordered by canon.

WITNESS PLAY

Friction Map

Witness play, instantiated for the SaaS AI platform engagement. Purpose: locate and quantify where the work breaks. The floor here is made of screens, so the map was counted from the queue's exports, the boards' history, and the calendars, checked against a day of sitting beside the people who do the work. Shares are of recoverable working time across the four teams the day covered.

The friction, at a glance



The friction table

Friction point	Frequency and cost	What the people doing the work said	What it tells us
Known answers re-written	[Roughly half] the ticket volume; [most of two] full-time equivalents	"I have answered this one three times this week." The senior lead keeps her own spreadsheet of answers	The knowledge exists and is unreachable; the queue runs on memory
Status assembled by hand	[The better part of a day each week] per project manager, every week	"Most of my Friday is producing this"	The information already lives in the boards and threads; assembly is machine work
Meetings that sync the boards	[A handful of] standing syncs weekly across teams	Attendees pre-write updates that restate the board	Coordination bought with senior hours because nobody trusts the boards' freshness
The marketing-to-sales seam	Leads waiting [days]; [a couple of hours] of research per qualified account	Each team presents its own pipeline version	The seam is data glue, and it cools revenue while it waits
Partner operations	Onboarding re-built per partner; referral tracking by hand	"Partners get whatever attention is left"	A growth channel rationed by leftover hours

The root-cause read

Every row above is work the company's own platform automates for its customers. The friction is not a missing capability; the capability is the company's product. The root cause is ownership: the product organization builds for customers, and deploying the platform on the company itself was never anyone's job. The map's second finding is the loop behind the biggest row: customers learn the product on their own, so they lean on the queue and the services teams, so those teams stay buried, so nobody has the capacity to fix how customers learn. The map says start where the hours are densest and the blast radius is private, operations, and fix the loop at its source once the floor is proven.

INTERROGATE PLAY

Assumption Register

Interrogate play, instantiated for the SaaS AI platform engagement. Purpose: surface the beliefs the build would otherwise inherit unexamined, including the client's own diagnosis, and test each one before

money follows it. Every assumption below was named in writing during Interrogate and settled by evidence, not argument.

The verdicts

KILLED

Scaling means hiring

The glue work across four teams already equaled [several] full-time people. The capacity was on payroll, spent on status, re-answering, and retyping.

KILLED

Internal AI adoption means new tools and a migration

The agents worked across the work management platform and the office suite as they stood. Keep the systems, remove the logging-in.

REFRAMED

The problem is finding where AI fits

The CEO's framing was a technology question. The platform was sufficient and owned; the missing piece was ownership. Reframed: the problem is that internal deployment was nobody's job.

REFRAMED

The support load is product weather

[Roughly half] the queue traced to customers operating a product nobody taught them to operate. A training problem wearing a support costume.

KILLED

The teams will resist automation of their work

The teams pulled the agents in. Nobody defends the glue. The real risk was half-adoption, agents live while the old meetings continued, and it was named and managed instead.

CONFIRMED

Drafted answers can match the senior lead's

Offline drafting against the closed-ticket record held up under her judgment, with one pattern in the misses: version confusion. The confirmation came with the rule that made it safe.

The register

Assumption	Held by	Test	Verdict	What it changed
Scaling to demand means hiring to match	Leadership, implicitly	Count the glue from exports and calendars	Killed	The build targets recovered hours; headcount goes only where judgment requires it
Internal adoption requires replatforming	Unspoken, inherited from how vendors sell	Run agents across the existing systems read-only	Killed	The systems stayed; the agents carry the glue between them
The problem is finding where AI fits	The CEO, at intake	A day on the floor against the platform's own demo scenarios	Reframed	The first decision: internal deployment becomes a named, owned project
The support load is fixed weather	Customer success, by habit	Tag [a week] of tickets by type and origin	Reframed	The training program became a roadmap move, not a services upsell
The teams will resist	Conventional wisdom	Watch the pilot teams' behavior	Killed	The half-adoption risk replaced it in the register, with retirement-by-decision as the control
Machine drafts can hold the senior bar	First Strategy, to be earned	Offline drafting judged by the senior support lead	Confirmed, with a catch	The knowledge layer was versioned against releases before anything faced a customer

What the register protected

The two killed assumptions were each worth real money. Hiring into the glue would have added cost that the recovered hours made unnecessary. Replatforming would have spent months of engineering for no customer value. The reframed pair mattered more: they moved the engagement's center of gravity from a technology search to an ownership decision, and from answering tickets faster to making half of them unnecessary. The register is also now a lesson in the customer curriculum, because every customer arrives carrying the same first three assumptions.

INTERROGATE PLAY

Experiment Log

Interrogate play, instantiated for the SaaS AI platform engagement. Purpose: record each experiment, its result, and what it changed. The rule the log enforces: the cheapest test that can kill a hypothesis runs before any build that assumes it. Every experiment here ran in days, not weeks, and none required production access to a customer.

The sequence

1 The ticket tag CONFIRMED

[A week] of tickets tagged by hand for type and origin. The repeat share held at [roughly half], and the top recurring questions were concentrated enough that the knowledge layer's first cut could be small.

2 Offline drafting against the closed-ticket record REFRAMED

Agents drafted answers to the recognized questions; the senior support lead judged them blind against her own. Quality held. The misses shared one pattern: confident answers from an older release's material. The experiment confirmed drafting and surfaced the versioning rule.

3 The read-only digest trial REFRAMED

A digest agent ran read-only on [two] live project boards, compared weekly to the PM's hand-built Friday report. The first digests lost: the decisions lived in mail threads the agent could not see. Wiring the office suite into the read produced parity.

4 The replatforming check KILLED

The assumption that internal adoption required migrating off the existing systems, tested by running experiments 2 and 3 across those systems as they stood. It did not. The migration nobody wanted was crossed off in writing.

5 Machine research on a sample account list PROMISING

Account research drafted by machine on a sample list, judged against an SDR's [couple of hours] per account. Parity in a fraction of the time. Logged and parked: the revenue seam is move 4, and it waits its turn.

The log

#	Hypothesis	Test	Cost	Result	What it changed
1	The queue's repeat share is [roughly half]	Hand-tag [a week] of tickets	Hours	Confirmed; recurring questions concentrated	Sized the knowledge layer's first cut; hardened the audit's count
2	Machine drafts can hold the senior bar	Draft offline from docs and closed tickets; blind judgment	Days	Held, with version confusion in the misses	The versioning rule, written before any customer saw a draft
3	Digests can replace the hand-built report	Read-only on [two] boards versus the Friday report	Days	Parity only after the office suite joined the read	The digest architecture reads boards, threads, and calendars together
4	Internal adoption requires replatforming	Run 2 and 3 on the systems as they stood	Free, by design	Killed	Months of migration removed from every plan
5	Machine research matches SDR hand research	Sample list, judged by the SDR	Hours	Promising	Parked as evidence for move 4, the revenue seam

What the log bought

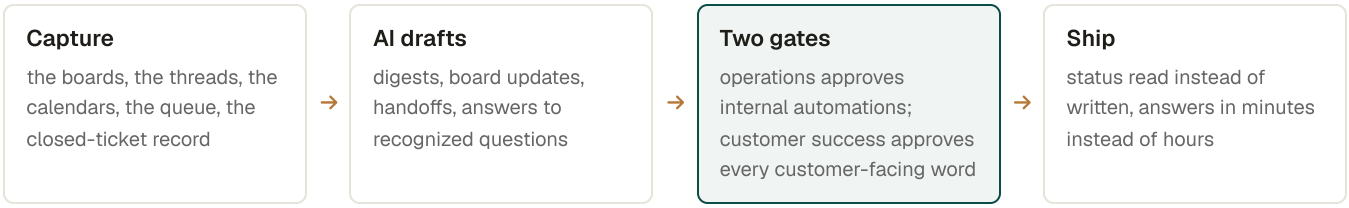
Two weeks of cheap tests settled what a quarter of confident building would otherwise have assumed. The expensive failures that did not happen: a knowledge layer shipping wrong-version answers to customers, a digest system blind to where decisions actually live, and an internal migration with no customer value. Experiment 2's catch became the build's most important rule, and experiment 3's miss became its architecture. The log, like the rest of the engagement's working record, now teaches in the customer curriculum: run the cheap test first, and let the misses design the system.

SOLVE PLAY

Human-in-the-Loop Design

Solve play, instantiated for the SaaS AI platform engagement. Purpose: define who reviews AI output, how, and what gets logged. This engagement has two distinct stakes, internal truth and customer-facing words, so the loop runs through two gates instead of one.

The flow



↺ Every catch is logged with its pattern. The knowledge layer learns from the log, the versioning rule keeps it honest against releases, and graduation to lighter review happens on catch-rate evidence, never by decree.

The design

Element	Design
Who reviews	Two gates, by stakes. The head of operations gates internal automations and decides when an old motion retires. The head of customer success gates every customer-facing word
What they check	Internal: parity with the truth the team would have assembled by hand, and freshness. Customer-facing: correctness against the versioned knowledge layer first, then tone, then whether the question should have escalated to a person instead
What the AI drafts	Project digests, board hygiene, handoffs between the systems, and answers to recognized questions from the documentation and the closed-ticket record
What the AI never decides	Escalations, the hard tickets, pricing, contracts, renewals, partner commitments, and when an old motion retires. Retirement is a human decision, logged
What gets logged	Every catch, with its pattern: the wrong-version answer, the stale read, the question that needed a human and almost did not get one
Where the log goes	Into the knowledge layer and the rules. Each pattern becomes a check, so the machine stops making the mistake category-wide instead of ticket by ticket

How the grip loosens

Both gates started reading everything. Output types earn lighter review only on evidence, under the tiers in the Charter's Hierarchy of Agency: a full review cycle in which the gate's catch rate on that type stays near zero. Internal digests earned it first, after weeks of running beside the hand-built reports they replaced. Recognized-answer drafts earned it later, on a published catch rate. Escalations and anything touching money never will, by design. A type that drifts falls back to heavier review, and the stale-digest incident in the Charter's drift record is why the fallback is not theoretical.

Why this matters

The company asks its customers to deploy AI with humans in the loop; the engagement held the company to its own standard. That symmetry is not cosmetic. The two-gate design is what made it safe for a 120-person company to put machine-drafted words in front of customers and machine-assembled truth in front of its own decisions, and the catch log is what made the machine improve instead of merely operate. The design now teaches in the customer training program, which means the support lead who judged the first drafts has had the unusual experience of seeing her catches become a curriculum.

REFINE PLAY

Drift Monitoring

Refine play, instantiated for the SaaS AI platform engagement. Purpose: watch for drift, including drift hidden in aggregates. The engagement's one post-launch incident is the reason this play is standalone: the failure arrived silently, produced confidently, and was caught by a human noticing that numbers had stopped moving.

The incident

Steady state
Digests live for the services PMO, trusted, the hand-built Friday report retired on parity evidence.

Ordinary housekeeping
A team restructures its boards: columns renamed, a group split. Nobody thinks to mention it, because why would they.

Confident staleness
The digest keeps producing from the stale mapping. Nothing errors. The numbers just stop moving.

The human catch
The operations gate reads the weekly counts: a project everyone knows is sprinting shows a digest that has not changed in days.

Contain, fix, generalize
Affected digests flagged and read manually. Schema checks added: an agent that reads a changed structure stops and says so. A trigger list written for every reading agent.

The monitoring plan

Watch	How	Cadence
Output that stops changing	The weekly read flags any digest or count that has not moved against a board that has	Weekly
Structure under the readers	Schema checks on every reading agent: a changed board structure halts the agent instead of letting it produce from a stale map	Continuous
The knowledge layer against the product	Reconciliation of the layer's material against the current release, so answers cannot quietly age	Each release
Gate catch rates by type	A rising catch rate on a graduated type pulls it back to heavier review	Weekly
The curriculum against the floor	The training program teaches the live deployment; when the deployment evolves, the curriculum is revised on the same cadence	Each revision

The trigger list

The conditions that force an agent to halt and ask rather than produce. Written after the incident, applied to every reading agent since.

- A board or document structure changed since the agent's mapping was built.
- A product release that touches material the knowledge layer draws on.
- A source that has produced nothing in a period when its siblings produced normally.
- An output that would contradict the previous run by more than the run-over-run norm.

The lesson, kept

An agent that stops is annoying. An agent that confidently produces from a stale picture is dangerous, because confident output is exactly what trust was trained on. Build the readers to stop. Watch the aggregates for silence as much as for noise, because in a system that produces on schedule, the number that does not move is the alarm. The incident, the catch, and the trigger list all teach in the customer training program now, because customers restructure their boards too, and none of them think to mention it either.