THE WORKING-OPERATOR'S VERSION OF BOTH SIDES OF TECH

The Six-Lever Diagnostic

Find your most expensive gap — and the exact next move for each lever

Ryan Richardson

How to use this

There are six levers. Pull them well and a small technical team compounds your output. Pull them badly — or ignore them — and the gap between the business side and the technical side quietly taxes everything you build.

This is not a document to read once. It is a diagnostic. Each lever has the same shape: **the gap** (what goes wrong, and why it stays invisible until it is expensive), **diagnose it** (yes/no signals and honest questions), **the advice** (how to pull it), **what it costs to get wrong** (the real failure), and **your next move** (three things to do this week).

Run the diagnostic honestly. The boxes you tick are your map. If more than one lever lights up, that is the normal result — not a failing grade.

The order to fix in, if everything feels broken: start with Levers 1 and 2. Unclear context and the wrong work structure make every other lever harder to pull. Fix those and 3–6 get easier.

01

Context & Communication

> **The translation gap opens here first: both sides think they are aligned, and neither has tested that assumption.**

The gap — what goes wrong

The operations lead wrote a comprehensive brief. He documented everything — options, frameworks, the manual process in detail. He thought he was making the technical team's job easier. The technical team thought the same thing. They built everything in the brief, exactly as written.

What the operations lead actually wanted — what he never said out loud because it felt too obvious to say — was a light-touch tool he could update every few weeks. Simple. Flexible. Something his team would trust immediately. He wanted speed and simplicity. He did not want everything he had written built exactly as written.

The context conversation that would have resolved this took thirty minutes. The scope dropped by ninety percent. A solution that had been six months in progress was rebuilt in a day.

This is the first and most expensive form of context failure: both sides optimising for different things without realising it.

The business side documents context to help. The technical side treats documentation as a specification. Neither names the assumption out loud.

The second failure lives one level deeper. Features get built without understanding the system those features need to live inside. The result is technical debt — not from laziness, but from building specific answers to specific questions when what was needed was a flexible framework that could generate many different answers from the same underlying logic.

Diagnose it — is this lever costing you?

Symptom signals — yes or no:

- ☐ You have handed over a brief that felt complete and received something that missed the point
- ☐ A technical project has taken significantly longer than expected, and nobody flagged a scope issue early
- ☐ You have approved scope without the technical team pushing back on what was actually necessary versus nice-to-have
- ☐ The same problem keeps getting rebuilt as a feature rather than solved as a system
- ☐ You have felt "probably clear" on a requirement and later discovered the expensive assumption hiding underneath it
- ☐ A technical person decided on a solution while you were still describing the problem
- ☐ You have received something technically correct that was commercially useless

Diagnostic questions from the chapter and workbook:

1. Think about a project that went sideways. At what point did the context gap actually open — and who was responsible for closing it?
2. What is a requirement or brief you have recently given or received that felt complete but probably wasn't? What assumption was hiding underneath it?
3. When you are working through a problem with your technical team, what signals tell you the picture is complete enough to start building?
4. What is the equivalent of the "systems versus features" question in your current environment — what keeps getting built around instead of solved?
5. Name the assumption you are most likely carrying into your next brief without testing it.

The advice — how to pull this lever

Find the Venn diagram. There are two circles. The technical circle: what can be built, how it works, what the constraints are. The user or business circle: the real problem, how the work gets done today, what good looks like to the person who has to live with the outcome. Your job is to find the middle.

Stay at the level of the problem longer than feels comfortable. The natural instinct — for technical people especially — is to go down rabbit holes. A detail surfaces, it seems important, and suddenly everyone is two hours deep into an edge case that may not even matter. The discipline of context mapping is the discipline of not doing that.

Ideate, question, ideate again. Put a rough picture of the problem on the table, even if it is partially wrong. Question why it is wrong. Revise. Keep going until the picture is complete — not perfect, complete. The test: can you describe the problem clearly enough that someone not in the room could understand it without asking a clarifying question?

Ask "what am I missing here?" — repeatedly, in different words. Not once. At different points in the conversation, from different angles. It is less like a checklist and more like navigating a map where parts are still in shadow.

Treat the first pass as a throwaway. Building a light version and seeing where it breaks is often the fastest way to surface context gaps that conversation alone did not find.

Build systems, not features. A system-oriented approach inverts the cost curve. The initial build takes longer because you are designing for flexibility. But the cost of change stays flat — or decreases — because changes are surface level only. The dynamic system question: instead of building ten survey questions each with their own data flow and maintenance overhead, can you write the underlying logic once and let the specific questions sit on top as a surface layer?

Use process owners. Non-technical people who live inside the problem every day are often better at context mapping than technical people, once you give them the right frame. Get them to stop describing a solution and just describe their experience of the problem — what is hard, what is slow, what breaks. The context you get is usually far more useful than anything a requirements document produces.

The right question closes the gap. Not a comprehensive interrogation — one specific question, asked at the right

moment, that surfaces the assumption that was about to become an expensive mistake.

What it costs to get wrong

A predictive modelling project. The ask was specific. It was in the budget, in the business plan, they had a demo of a competitor's version. Scope was approved. Work started.

Somewhere in the middle of it: they had no idea what predictive modelling actually was. Not a partial understanding — no understanding. Nobody had ever asked if they knew what it meant. The ask was so specific, the signals so convincing, that the most basic question was never asked: do you actually understand what this produces and what you'd do with it?

That question takes thirty seconds. Not asking it cost weeks.

Your next move

This week:

1. Before your next technical brief or requirements conversation, fill in the two circles honestly: what can be built (technical), and what the real problem is (business). Write down the overlap before anyone starts building.
2. Find the assumption you are most likely to carry untested into your next project. Say it out loud to the technical team and ask if it changes anything.
3. Look at one thing currently being built as a feature. Ask whether it should have been designed as a system —

something that changes frequently at a surface level while the underlying logic stays stable. If the answer is yes, have the thirty-minute conversation now rather than six months in.

02

Work Structure

> **The leader is either involved in decisions that belong to the technical team, or absent from ones that need business judgment — and both failures widen the gap.**

The gap — what goes wrong

Work structure is where most leaders think they are helping and are actually doing damage. The failure mode is not negligence — it is overreach. You weigh in on technical decisions that should sit with the team. You ask for updates on things that do not require your input. You review, redirect, and second-guess — not because the work is wrong, but because the invisibility of technical work is uncomfortable, and stepping in makes you feel less exposed.

The other failure is the mirror image: being absent from decisions that actually do require business judgment. When a developer tells you that cutting load time by fifty percent would cost a month of engineering, the technical call is theirs. Whether that month is the right investment right now is yours. When both sides of that equation collapse into one person — or neither does it — the work goes sideways.

Your role has three parts that are distinctly yours and that nobody else on the team can fill as well: context (why the work matters and how it connects to everything else), application (translating technical information into business decisions), and direction (which features, in which order, at what cost). Everything else is your team's call.

Diagnose it — is this lever costing you?

Symptom signals — answer honestly:

- ☐ You regularly weigh in on technical decisions — implementation approach, architecture, tooling choices — without being asked
- ☐ Your team brings questions to you that they could resolve themselves with a clearer brief
- ☐ The best work happens when you are not closely involved, and you know it
- ☐ You have noticed team members waiting for direction rather than generating it
- ☐ The quality of questions from your team has become lower-effort — confirmation-seeking rather than judgment-exercising
- ☐ You have ended up in conversations with clients or stakeholders that should have been handled by someone else first
- ☐ Your team has absorbed pressure or bad behaviour from outside — a demanding client, an unreasonable deadline — without flagging it to you

Diagnostic questions:

- Where in your current work do you step in when you probably should not? What is that costing the person you are stepping in for?
- What are the decisions your team is currently waiting on you for that they could make themselves with a clearer brief?
- If you had to describe your role in one sentence to a new team member — not your title, your actual function — what would you say?
- Am I involved in this because my involvement is genuinely necessary, or because it makes me feel less exposed?
- Is the team still exercising genuine ownership — or have they quietly slipped into execution mode, waiting for direction rather than generating it?

The advice — how to pull this lever

Start with the person, not the process. Before any project framework, have the conversation about how you are going to work together. What they are trying to get out of the engagement. What else is happening in their life. Make the rules explicit from the start — escalation paths, communication norms, what urgent actually means.

Work from long to short on goals. Big goal: what does success look like at the end of this phase? Near-term goal: what needs to be true in the next two weeks? Stretch goal: what does better than expected look like? Then let the domain lead generate the tasks underneath those goals themselves. The act of generating the tasks is how you verify that they have the same picture of the

goal you do. This should take five minutes — it is an alignment check, not a planning session.

Make your role explicit early. Tell them directly: you are better than me at your specific thing, I hired you because of that, I want to hear it when you think I am wrong. My job is to give you context and direction, not to tell you how to do your work. If I start doing that, call me on it.

Monitor ownership, not output. The signal that something has gone wrong is usually in the quality of questions the team asks. When those questions become confirmation-seeking rather than judgment-exercising, something upstream has changed — a deadline got tighter, you got more involved than you should have. Step back and reestablish the conditions that made the ownership possible.

Cadence over intervention. The fix is not checking in when something feels wrong. It is checking in on a rhythm so things do not get far enough wrong to feel that way. Short, regular, low-stakes. Not glamorous. Just the thing that works.

Communication stack that works: scheduled recurring meetings (weekly or fortnightly) for brainstorming, deep dives, and check-ins; screen capture video for anything that needs context but not a live conversation; Slack for coordination; phone call or WhatsApp for genuinely urgent matters, which you try to make rare.

What it costs to get wrong

The clearest version of this failure is not dramatic — it is slow and invisible.

A client under pressure from their own organisation starts bypassing working norms. Late-night messages. Deadlines set without consultation. A communication style toward the offshore team that crosses a line. The team says nothing directly. They absorb it. They work harder, stay later, say yes to things they should have pushed back on.

The first visible signal is not a complaint. It is a drop in the quality of the work. Flat energy in team communication. By the time you dig in and find out what has been happening, the cost — to trust, to output, to the relationship — is already significant.

The fix, once found, was to end the engagement. Not immediately, but within two weeks. And to be explicit with the team about why — not to make a point, but because they needed to see that the rules set at the start were real. The energy came back. The quality came back.

The smaller version of this failure happens constantly: not being across team dynamics until something has been brewing for weeks; a handoff that lacked context nobody checked; a client conversation handled directly that should have been escalated. Neither side acted in bad faith. Both made a completely reasonable individual decision. Together they created a mess. Every time.

There is also the failure of holding back when you should have challenged. A senior person — widely respected, very well paid — with fundamental flaws in their process that were creating problems for every team that touched the system. The doubt crept in: do I really know this better? What if I'm missing something? That doubt is not irrational. The problem is using it as a reason to wait rather than to investigate. By the time the situation was undeniable, the cost was already significant and the conversation was harder than it needed to be.

Your next move

1. **Name one decision your team is waiting on you for** that they could make with a clearer brief. Write the brief this week and hand it over.

1. **Run a five-minute goal alignment check** with each active domain lead: big goal, near-term goal, stretch goal — then ask them to generate the task list underneath. If their list surprises you, that is your context gap, not their problem.

1. **Set or reinstate a standing cadence** — weekly or fortnightly, short and low-stakes — so you are checking in on rhythm, not reacting to signals that something has gone wrong.

03

The Right People

> **Hiring for technical skill alone, without the ability to translate between business and technical worlds, is where the capability gap compounds into a people problem.**

The gap — what goes wrong

Most hiring processes optimise for the wrong signal. The CV becomes the primary filter, technical credentials become the primary criterion, and the thing that actually determines whether someone compounds your output — their ability to move between the technical and business worlds — never gets tested.

The result is a team that can build but cannot translate. They interpret requirements in ways that make sense from a coding perspective but make no sense from a business one. They optimise for the wrong success criteria, defined entirely by the people doing the building rather than the people who need to use the output. The work is technically correct and commercially useless.

The right people for a high-output technical team share a specific characteristic: genuine curiosity about the other side. Not tolerance of it. Curiosity. That curiosity is not a personality

trait you either have or don't. It is a decision. And it is the decision that determines whether someone becomes a strategic asset or stays a cost centre.

Diagnose it — is this lever costing you?

Symptom signals — answer honestly:

- ☐ Your developers deliver work that is technically accurate but analytically meaningless or practically unusable.
- ☐ When a project goes sideways, the post-mortem usually surfaces a requirement that was misinterpreted, not a skill gap.
- ☐ You have hired people who said the right things in interviews and produced nothing in the first two weeks.
- ☐ Your technical team optimises for elegance, accuracy, or architecture — without asking what the output needs to do for the person using it.
- ☐ You invest heavily in correction and coaching on underperforming hires, hoping to make it work, rather than exiting cleanly.
- ☐ You rely on credentials and CVs to predict capability in fast-moving domains.
- ☐ Nobody in your team can explain a technical deliverable in terms of the outcome it creates, without being asked to.

Diagnostic and screening questions:

- *In the interview:* Bring a real problem you are currently working on — not a hypothetical. Talk about it with them. Do they lean in? Do they ask good questions? Do they start

building on ideas in real time? The conversation quality is the signal.

- *After the interview:* Did they follow up with actual thoughts? Not a polished thank-you — thoughts about the problem. The ones who are still thinking about it after the conversation ended are the ones worth backing.
- *On translation:* Can they describe a technical decision in terms of what it does for the person using it — without being asked to dumb it down? Not the features. The outcome. Not the accuracy. The decision it makes easier.
- *On values alignment:* Do they hold themselves to the same standard you do? Do they say something when they think something is wrong, or do they nod and quietly do what they were told?
- *On domain curiosity:* Look for evidence of genuine interest pursued independently. Something built. Something figured out. Something hard that did not have to be done but was done anyway. That signal is rarer than a degree and worth more than any of them.

The advice — how to pull this lever

Treat the CV as binary, nothing more. Do they have semi-relevant experience in the general area? Yes or no. If yes, they can probably do the job. If no, you're done. The CV is a history document. You are hiring for now.

Use Fiverr to understand the capability landscape before you hire. Browse services not to find the cheapest option but to understand what the capability landscape looks like. What skills keep appearing? What tools are popular? What adjacent services tell you where the real complexity sits? When you enter a

domain you don't know well, this orientation matters more than the job description.

Run searches in parallel, not sequentially. Ask offshore leads if they know anyone. Post a LinkedIn ad — deliberately specific, describing what you're building, where you're stuck, and what your goal is. Put your email on it. Test a specific capability on Fiverr with a small paid engagement. The people who respond with genuine enthusiasm about the specific problem are the ones to talk to first.

Run a two-week trial on real work, structured so they don't have to change their situation. Not a test task — actual work toward an actual goal. They keep their existing role. You pay them part-time. Worst case they made some extra money and learned something. Your maximum exposure is two weeks of part-time pay. The structure of the engagement is the most important risk management decision you make in hiring. Not the interview process. The structure.

Hyperspecialists are almost never what you need yet. Hyperspecialists excel at the final one percent of a problem — the optimisation, the edge case, the marginal gain that matters when everything else is already working. You need to do the ninety-nine percent first. And the ninety-nine percent requires someone curious, capable, and genuinely invested in figuring it out.

Back your judgment. Keep the exposure small. Move on cleanly. Every interview framework is ultimately trying to protect against the downside. They all fail at roughly the same rate. The difference is not in the sophistication of the process — it is in whether you back your judgment when you feel it, and whether you protect against the downside structurally rather than procedurally.

What it costs to get wrong

The expensive version of a hiring mistake is the traditional one: six-month probation, genuine coaching investment, trying hard to make it work. The longer you invest in trying to correct something that is not working, the more it compounds.

When it goes legally — offshore in particular, where the laws are different, your onshore lead can be exposed, and the person on the other side may have quit a stable job and is now in a genuinely desperate position — the pattern is consistent. The work does not materialise, the story expands to explain why, and when the engagement ends it shifts quickly from hardship framing to sustained pressure designed to make you pay more to make it stop.

The honest signal is this: the hires that didn't work out usually had a moment in the first conversation where something didn't quite land. Something noticed and then immediately talked out of because wanting it to work felt more urgent than trusting the read.

The two-week trial doesn't solve the moment before — where something is off and you proceed anyway. What it does is mean backing someone costs two weeks of part-time pay if it doesn't work, not six months of salary, a legal process, and the kind of sustained stress that bleeds into everything else.

Your next move

1. **For the next person you need to bring in:** Before posting, spend two hours on Fiverr understanding the capability landscape. What skills appear consistently? What tools keep surfacing? Then write the ad around the specific problem, not the job title — and put your email on it.

1. **Design a two-week trial structure now, before you need it.** What would you run? What would count as evidence it's working? Having this ready means you can move fast when the right person appears, and exit cleanly when they don't.

1. **For anyone currently not performing:** Ask honestly — when did you first sense something was off, and what did you do with that? If you've been investing in correction rather than exiting cleanly, name the real cost of that: not just money, but the decisions you're deferring and the energy it's absorbing.

04

Location Arbitrage

> **Accessing global talent pools is powerful when you do it deliberately — and it has burned a lot of people who didn't.**

The gap — what goes wrong

Location arbitrage sits at the intersection of two bad assumptions. The first: that offshore means cheaper, and cheaper is the goal. The second: that hiring globally is riskier than hiring locally, so you default to whoever is available nearby.

The real gap is where "assumptions about quality and culture replace actual evaluation." You either write off global talent based on one bad experience (or none at all), or you sprint toward the cheapest rate you can find and wonder why the work falls apart. Neither is a strategy. Both are reactions.

The compliance and structure realities compound this. Most people who write about offshore hiring skip the legal infrastructure entirely — because it's unglamorous. But the legal owner of the employment contract is typically your local lead. If you stop paying, they carry the exposure. Understanding this explains why good leads are hard to find and why the ones who exist are cautious about who they work with.

The offshore-at-scale trap: you discover global talent is genuinely good and lean in hard. Six months later, reports are technically correct but analytically useless. Then you start cutting rates, squeezing cost. Quality drops. Attrition rises. The rework costs exceed what a reasonably priced local team would have charged.

Diagnose it — is this lever right for you (and are you doing it well)?

Yes/no signals:

- Do you know specifically what capability gap you're trying to fill — or are you just looking to reduce cost?
- If you went offshore before and it went badly, have you diagnosed *why*, or have you written off the whole model?
- Do you have a brief clear enough that someone in another timezone could execute it without a daily check-in?
- Do you have a trusted local lead — someone accountable for outcomes, not just activity?
- Are you prepared to be honest about commercial uncertainty with people you hire? (If a project might run out of runway in three months, say it upfront.)
- Are you optimising for the right person for the role, or the cheapest person in a lower-cost market?

Diagnostic questions:

- What is your honest current position on offshore — and what is that based on? Direct experience, or something you heard?
- If you were to build one offshore relationship in the next six months, what specific capability gap would you fill — and what would the right person need to look like?

- Is this high-output flexible work (where global often wins regardless of cost) — or finding a senior specialist with narrow domain expertise (where local signal-to-noise is easier to manage)?
- Is this even the right lever right now? If your biggest constraint is unclear requirements or wrong people in wrong roles, fix those first — they'll make this lever work better when you pull it.

The advice — how to pull this lever (carefully)

The lead is everything. Not the rate. Not the platform. Not the contract structure. The lead. Find someone accountable for outcomes, not just activity, with enough standing in their local market to be useful when things go wrong. Keep that person genuinely engaged — not just financially.

What you're actually buying is access. Not cheaper time — access to a much larger talent pool and a flexibility in engagement structure that's nearly impossible to replicate onshore. A developer running ten focused hours a week will outperform a full-time local hire productive for three of their eight daily hours. The impact ratio is what you're optimising for.

The most underrated model: a senior specialist for whom this is a second engagement — someone already operating at a high level who runs hard for you in a contained, well-defined scope because the work is interesting and the arrangement works for their life.

Understand the legal reality before anything else. To engage someone offshore compliantly you need either a legal or payroll presence in their country, or to work with someone who already has that structure. Employer of Record services exist — none

give you the certainty you want, and all cost more than people expect. Don't skip this step.

Be transparent about economics. On fixed-cost client work, a workable structure is around forty percent going to the team covering their rate, taxes, and any overhead the lead carries. Transparent ownership of the economics, shared wins when they happen.

On culture: Some countries have strong hierarchical structures where challenging your superior is culturally dangerous — but this doesn't map cleanly to geography. I've seen it more in certain US contexts than in Bangladesh. Build an environment where disagreement is explicitly welcomed and output is the measure — then wait. The people worth keeping will find their footing in it.

The quality of output is almost entirely a function of the quality of the process. How clearly you define what you need, how carefully you evaluate who you hire, how much care you put into the relationship. That doesn't change when you cross a border.

What it costs to get wrong

The common failure modes: bait and switch (you assess an impressive portfolio; someone else does the work). High turnover with no control over your resources because a middle layer owns the relationship. Agencies squeezing margin at every point. Unpaid interns doing paid client work. Direct hiring through a trusted lead is the most reliable protection against all of these.

My honest summary: I've played the offshore model conservatively and haven't had a catastrophic failure. The worst

outcome is just nothing — no meaningful work. Containable if the structure is right. The structure being right is the whole game.

The race to the bottom is the failure mode hardest to see coming. It looks like optimisation. It ends with rework costs that exceed what a reasonable local team would have charged.

Your next move

1. **Name the gap before you name a location.** Write down what you need — the role, what great looks like, where that person most likely exists. The offshore conversation follows that question; it's not the starting point.

1. **Find your lead, not your first hire.** The first investment is finding one trusted local lead in a market that fits your need. This takes time. It is the work. Everything else builds from it.

1. **Run the brief test.** Take the next piece of work you want to offshore and ask: could someone in another timezone execute this without a daily check-in? If no — fix the brief first.

05

Tools & Automation

> **Tools & Automation is where the technical side optimises for capability and the business side optimises for familiarity — and neither gets what they actually need.**

The gap — what goes wrong

The technical team is drawn to what is possible. The business side is drawn to what is already familiar. Together they produce a tooling environment that is neither cutting-edge nor stable.

The subtler problem is that tool-tinkering is one of the most convincing forms of avoidance available. Mapping workflows, evaluating platforms, rebuilding systems — it feels like productive work. It often is not. The elaborate system is not a productivity tool. It is a very convincing way to avoid the hard part.

Two categories where genuine friction lives:

Alignment vs optimisation. Every tool change has a cost that never appears in the evaluation: cognitive overhead, reduced productivity while habits reform, resistance from people who were finally comfortable. These costs compound across a team

and are almost never factored into the business case for the switch. Using a slightly worse tool that requires no change is a better outcome than an optimised toolkit that requires meaningful adaptation.

Premature automation. The temptation to automate immediately is strong and usually premature. Build the automation before you understand the tool, and six weeks later you realise it does not do what you needed. Now you have a migration problem on top of the original problem. The cost is not the tool cost — it is the rework cost and the team's trust in the next tool decision.

Diagnose it — is this lever costing you?

Symptom signals — yes or no:

- ☐ Monthly subscriptions running that nobody is actively using or measuring
- ☐ A new tool adopted in the last six months that required significant onboarding before producing anything
- ☐ Automation built around a tool before you fully understood its limitations
- ☐ A tool in current use out of habit rather than because it is genuinely the best option now
- ☐ All tool decisions flow through the leader — the team has no sanctioned way to test in their own domain
- ☐ Something has been in "setup" or "optimisation" mode for weeks and has not shipped anything concrete

- [ ] Significant AI investment pointed at a critical, high-stakes use case before proving value on low-stakes, high-volume work

Diagnostic questions:

- What is one task your team does regularly that, if it disappeared entirely, nobody would miss — and could it be automated?
- What tool are you using out of habit rather than because it is genuinely the best option now?
- What AI use case in your specific work sits in the "close enough at scale" category — where ninety percent accuracy at ten times the volume would be genuinely valuable?
- Is the system you are currently building going to ship something — or is it what you are building *instead of* the thing that needs shipping?

The real leverage test: Does this tool eliminate the task, or just make it easier? Elimination is a fundamentally different outcome to reduction.

The advice — how to pull this lever

AI is estimation — frame it that way. Anytime you need "close enough at scale," AI is the answer. Anytime you need exactly right, or the stakes of being wrong are high — AI is not the answer yet. The failure mode is pointing your biggest AI investment at your most critical use case, because that is exactly where it is hardest to deploy and the failure rate is highest.

Removal first, challenge second. The best AI tools completely eliminate something you used to think about. Not reduced —

gone. Gone is a fundamentally different outcome. Second best is challenge: brainstorming, assumption testing, first-pass thinking on problems not yet fully formed. You are not looking for the right answer. You are looking for ten angles, three of which are interesting.

Hold the structural tools boring and stable. Project management, documentation, communication — keep these as stable as possible. Nobody ever got a competitive advantage from switching PM tools. The switching cost compounds across a team in ways almost never recovered in the productivity gain.

Five-minute test, not a formal evaluation. Drive straight in and try the tool on something real. If you cannot get value quickly — sometimes five minutes, sometimes a day — cancel it and move on. A tool that requires three hours of setup before it does anything useful requires three hours of setup every time someone new joins your team.

Use it manually first. Automate later. Do not build deep process around a tool until you have used it long enough to know it is staying. Get the result first. Then make the result repeatable.

Distribute the search. A leader-driven approach to tooling is bounded by the leader's own work. Give the team a small discretionary budget with no approval process. What it communicates is that their judgment is trusted and their curiosity is sanctioned. The result is a team staying ahead of what is possible rather than waiting for you to notice it first.

Build for change, not for correctness. In AI specifically, the goal is not to build the right thing. It is to build something you can swap out quickly when the right thing becomes something different — which in AI it will, and soon.

What it costs to get wrong

The worst tool decisions are not the ones where you picked the wrong tool. They are the ones where you picked the right tool at the wrong time and built process around it before you understood it well enough.

A mining company invested close to half a million dollars building a central LLM platform. By the time it was finished, the underlying technology had moved on and the platform did not work properly with the current models. Months of startup work went into a custom reasoning loop for an LLM feature. By the time it was working, the model provider had shipped a native capability that did the same thing a hundred times better out of the box.

The closer-to-home version: trying to scale content. The bottleneck is not a tool problem. The solution is simple — pick up the camera and film something. Instead: mapped workflows, evaluated scheduling platforms, started and restarted the system three times. Filmed almost nothing. Optimising the process is comfortable. Filming is uncomfortable. The elaborate system is not a productivity tool. It is a very convincing way to avoid the hard part.

Your next move

1. **Name the avoidance.** Identify one thing in "setup" or "optimisation" mode for more than two weeks. Ask whether you are building the system or building *instead of* the hard part. If it is avoidance, stop and do the thing.

1. **Run the five-minute test on one tool you are paying for but rarely use.** If you cannot get value out of it in five minutes, cancel it. The time you would spend "learning it properly" is time you will not recover.

1. **Give one person on your team a small discretionary budget with no approval process.** Let them test one tool in their domain this week. What they surface will almost certainly be more relevant than what you would have found yourself.

06

Retention & Culture

> **The accumulated cost of every other lever gap shows up here — in people who leave, disengage, or stop bringing their full capability — and it is the one discipline that does not emerge naturally.**

The gap — what goes wrong

Most of what falls under retention and culture does emerge naturally if you get context, structure, the right people, and a working model right. But not all of it. The part that does not emerge naturally is the part that costs you something personally every time you do it right.

The specific failure is a distortion of awareness. Leaders who are otherwise doing most things right are acutely aware of their own decisions and strategic calls. They are much less aware of what the team is doing in the background to make those outcomes possible. Over time that asymmetry creates a picture where the leader's contribution looms large and the team's contribution becomes ambient — expected rather than noticed.

The number one place this breaks down is money. When things go well, do you actually follow through on what you implied or promised? The bonus you mentioned when the

project was under pressure — does it arrive? The rate you agreed to when you needed someone urgently — does it get reviewed when the engagement stabilises? Your team is asking these questions even when they are not asking them out loud. Intent does not pay anyone's rent. The follow-through is the signal.

The second place it breaks down is avoidance. The quick cut when someone is struggling. The conflict swept under the rug because addressing it is uncomfortable. The salary conversation that never happens because you are not sure how it is going to go. The culture is not what you intend. It is what you demonstrate under pressure.

Diagnose it — is this lever costing you?

Symptom signals — yes or no:

- ☐ When things have gone well, you have meant to revisit someone's rate or bonus and something else came up instead.
- ☐ You have managed around a performance problem — restructured the work, hoped it would resolve — rather than naming it directly.
- ☐ You have made the quick cut when someone was struggling rather than picking up the phone.
- ☐ Your check-ins, personal investment, and follow-through apply to your local team but are largely absent with contractors or offshore.
- ☐ You have treated a resignation like a betrayal — cutting the person out, managing them down aggressively — rather than making the exit clean.

- ☐ The last time something went wrong on your team, your first instinct was to point at the offshore team, the time zones, or the communication differences rather than the brief.
- ☐ No one on your team pushes back on your ideas. When you float something, everyone agrees.

Diagnostic questions:

- Think about the last time a team member left or disengaged. Looking back honestly — what were the signals you saw and did not act on?
- What is a commitment you have made to someone on your team — explicit or implied — that you have not followed through on? What is that doing to the relationship?
- Who on your team receives the least personal investment from you right now? What would change if you reversed that for thirty days?
- Before you assess whether someone on your team made a mistake, have you asked yourself whether the target was clear enough, whether they had what they needed, and whether you were providing enough coaching to make the standard obvious before the error happened?
- When did someone last genuinely disagree with you in a working session — not rudely, just honestly? If you cannot remember, something has shifted.

The advice — how to pull this lever

Start with genuine awareness. Your people are what make you good. The product looks good because they made it look good. Remembering that — genuinely remembering it, not just saying it — is the foundation of everything else.

Follow through on money without being asked. When someone does something that warrants a pay rise or a bonus, do it. Do not intend to. Do not wait for them to raise it. The moment passes. The person noticed and they will not forget. When you do it without being asked, you tell them something important about how you see the relationship.

Have the hard conversation directly. Name the problem, acknowledge what is good, move immediately to what happens next. Not a formal review. Not a documented warning. Something like: *"Honest version — your communication style with your juniors is hurting more than helping. You're doing great work and I get that they need to step up, but that approach isn't working. Can we figure out a plan together?"* Directness is not unkind. Avoiding the conversation is unkind — to the person, to the people affected, and to you.

Apply the same standard to everyone on the team. The check-ins, the personal investment, the protection from bad client behaviour — these cannot apply to some of the team and not others. If you want the same quality of output, the same ownership, the same willingness to run hard and say something when something is wrong, you have to create the same conditions for contractors and offshore as you do for your local team.

When something urgent breaks the norms, own it specifically. Acknowledge it is outside your normal norms. Make clear it is an exception and not a new expectation. Then explain what you are personally changing so it does not happen again — not "the client caused this" but "I missed a planning step that caused this, here is what I am changing." The acknowledgment without the specific personal accountability is just an apology. The accountability is what makes it mean something.

Create conditions for honesty. The signal that it is working is when someone pushes back on something you have proposed. Not rudely — genuinely. That means they believe their opinion is worth having in the room. When that stops happening, something has shifted and you need to figure out what.

When accountability goes wrong, trace it to its root. Most performance problems, when you trace them back honestly, have a leader decision somewhere near the root cause.

What it costs to get wrong

At one consultancy, resignations were treated like betrayal. The moment someone handed in notice they were cut out, talked about, managed out aggressively. Everyone who stayed watched. The turnover was enormous and nobody could figure out why.

The hardest version of losing someone is not the dramatic one. It is the one where nothing went wrong — a capable, motivated person who simply wanted something different from their work and their life. You cannot fix that. What you can do is make the exit good — be supportive, help them land somewhere better.

How you treat people on the way out is one of the clearest signals your culture sends. Everyone who stays is watching.

The more expensive version: one salaried team member let go from a client contract became, under the terms of the arrangement, a six-month problem on payroll — not working, with legal and contractual leverage, using it. Six months of salary, significant legal time, and the kind of ongoing stress that makes everything else harder. The probation period that is supposed to protect you rarely does in practice; by the time you

have enough evidence to act on, you are usually past the point where acting is clean.

Your next move

1. **Identify the follow-through gap this week.** Think of one commitment — explicit or implied — you have not followed through on with someone on your team. Act on it before anything else.
1. **Have the conversation you have been avoiding.** Name the problem directly, acknowledge what is good, and move immediately to what happens next. Give yourself a deadline of this week.
1. **Check your coverage.** List everyone who does work for you — local, contractor, offshore. Circle the ones who receive the least personal investment. Pick one and change that for the next thirty days.