# Building AI That Actually Gets Used

## How to Move From AI Demos to Real Business Value

# About the Author

I'm not writing this as a theorist or outside commentator. I'm writing it as someone who has been in the room where generative AI gets scoped, built, and deployed for real business outcomes - not just demoed in slides.

As a founder and former Chief Operating Officer at Stellar, an AI and data practice that helps companies move from AI curiosity to AI implementation. We've worked with over 50 customers across 15+ industries - from pre-seed startups to multi-billion dollar enterprises - with a focus on pragmatic outcomes, not hype.

Before Stellar, I spent 15 years in operational leadership at high-growth tech companies. I was on the founding team at Canvas, which was acquired by Jobvite. Before that, I led US operations at Virtusa after they acquired Apparatus, where I'd helped build the data practice. I've been through two successful exits and learned what actually scales versus what just looks good in a pitch deck.

# What This Experience Has Taught Me

I've talked about this work on the Indianapolis Business Journal podcast, written about AI in healthcare for ERP Today, and spoken at industry events about what's actually working in the field. But the insights in this book don't come from stages or articles - they come from sitting in kickoff meetings, watching pilots fail, debugging why adoption stalled, and figuring out what actually moves the needle.

Here's what that experience has taught me,

> I've seen the same failure patterns over and over - pilots that look great but never ship, strategy work that never ties back to business outcomes, and organizational resistance that kills momentum before value can be realized.

> I've also seen where AI actually creates leverage - and what it takes to get there - prioritization, scope discipline, stakeholder alignment, and clear execution pathways instead of vague ambition.

This book is a distillation of that pattern recognition - from what works, what doesn't, and why most businesses stall long before they win with AI.

If you're tired of the hype and want to make real progress - without wasting budget, time, or credibility - this book will help you start where most organizations actually need to start.

CHAPTER 1

# AI Hype vs. AI Reality

# What People Think AI Projects Look Like

A lot of dreaming goes on when people talk about AI. Fast, easy, cheap, potentially magical - that's the pitch. And because most people are accustomed to SaaS, they think AI looks like an implementation. Plug it in, turn it on.

But most companies haven't built custom software in years. They've been buying SaaS. AI projects require real development work, and that's a muscle most organizations have let atrophy.

Here's what happens - someone uses ChatGPT to draft an email or summarize a document. It blows their mind. Now they want to bring that magic into their business. The problem? That mind-blowing moment is one step out of ten. They're ignoring the other nine.

> **AI is not magic.**
>
> **But it is magical.**

That magical part? It's only about 10% of the entire process. The other 90% is integration points, front end, back end, database, authentication, security and compliance, multiple stakeholders, public versus private facing sources, what data you're going to feed it - all kinds of decisions that can balloon a project if you let it get too big.

People think AI projects look easy, fast, transformative. You just have to be really good at identifying the key goal, figuring out how to solve it, and getting it done in the fastest, cheapest, best possible way. Then move on to the next thing.

# What AI Projects Actually Look Like in Real Companies

Every project starts with excitement. You go through the sales process, get general agreement on an SOW, scope out work that's loose but reasonably defined. Then the real discussions begin.

Almost immediately - especially with larger companies - you're in a kickoff call with representatives from everywhere. IT, security, the business side. We're often dealing with C-level executives, including the CEO. We've done work orders for less than $100,000 where the CEO has direct involvement. Kind of mind-boggling, especially at larger enterprises.

But at larger companies? You start with that kickoff meeting - a room full of people, a pile of opinions, everyone wanting to extract value. And it gets too big, too fast.

> **You need to isolate it down to one simple thing.** Because even with group agreement, you still have to build something.

Infrastructure, most likely. Existing tools or new ones? Environments to provision. Tickets to create. Sign-offs to collect. Two environments? Three? Four? CI/CD pipelines to figure out. Data access to negotiate - is it in Databricks or Snowflake where you can't give unfettered access? Restrictions to implement. Other teams to loop in.

All of those concerns are real, and they'll need to be addressed eventually. But right now? We just want to prove whether AI can actually solve the problem. Chunk up your data, do it outside the system, do it in a sandbox environment. This doesn't need to be production ready out of the gate. Prove the concept works, then figure out how to put it back into the development pipeline.

# The Difference Between Demos and Production

### Demos

Demos are meant to be fast, easy, and prove value. Nothing more. A demo can be disconnected from live data. It doesn't need secure constraints or authentication. The UI can be a little sloppy, chat-based, focused on one use case. The question is simple. Can we get the math to work? Is this a worthwhile effort?

### Production

Production is the robust thing that actually needs to work. Consistent. Predictable. Deterministic - or at least with as many deterministic paths as possible.

The only way to get something into production is to keep it super skinny. Otherwise you hit a thousand roadblocks from a thousand departments and a thousand worried people. It'll kill the project. Painfully.

Work with a team that has the skid super greased - people who will make this thing happen. You don't want to do it unsafely, but start with something super small that you can isolate to one area. Get it running. Build confidence around it. Then replicate it a few times and expand.

# Why Cool Demos Die in 30 Days

Demos typically die in 30 days. Because that's what they are - cool demos.

One of our team members built a really cool app using multimodal services. You could find a pizza place, type in natural language what you wanted to order, and the voice system would call the restaurant and make the order on your behalf. You could say "I want to order pizza for 12 adults" and it would pick out the best options and quantities. Really impressive. The room loved it.

**But it's not a practical use case.**

A lot of cool demos die because they don't have real application in the business world. Keep that in the front of your mind. Building cool stuff is fun - it's a great way to get people excited about AI. But you also need to tie it directionally to real outcomes. Don't do something just because it's cool - it'll be expensive, time consuming, and potentially break other things. Be thoughtful about where you're going.

# Why Leadership Expectations Are Usually Wrong

Most pilots never turn into real tools. The reason? Leaders think it's going to be fast. They're hearing stories about companies removing entire customer service teams - though in several cases, those companies reversed course. But leaders hear the sound bites - fast, easy, cheap, revolutionizing. All of that can be true, but it doesn't come out of the box that way.

Leaders need to dig in and understand the actual problem they want to solve and how to solve it. But don't overthink it into paralysis. The key is to jump in and do something. Pick a problem - it doesn't matter which one. Pick one that's center cut for AI and do whatever you can to make it happen fast.

### Domain Expertise

You need people familiar with the tools. If you want to do image generation, who knows the right prompts and models? If you want data classification, you need someone who knows which use cases apply.

### Tool Expertise

If you want playbooks or battle cards, you need someone with expertise to turn that into an executable agent.

You need domain expertise and tool expertise. Pull those people into the right room and problem solve.

# Why Most Pilots Never Turn Into Real Tools

You get through the cool demo, it turns into a pilot, and then the rollout slows to a crawl.

We've done demos where we rolled out to one marketing team, expanded to a few more members, with the goal of reaching the entire sales team. The problem is every new person has an opinion. "This is great, but I'd rather say it this way" or "This didn't seem quite right." We're measuring against a level of perfection, which is insane. That's not the right way to evaluate these things.

I call it the automated car problem. About 40,000 traffic deaths happen in the United States every year. Automated cars could dramatically reduce that. But if there's one accident with an automated car, people throw up their hands - "We can't rely on machines, they're not safe."

> **But here's the math, machines only need 39,999 deaths per year to be better than humans.** If we're better than humans, that's all we need to aim for. A little bit better.

Yes, that's still a lot of lives lost, but we're on a trajectory toward many, many fewer. So let's not stop. Let's figure out how to make it safer and safer.

The same logic applies to AI in business. You can't be paralyzed by fear of where it might go.

Everyone has a perspective. Everyone thinks they know the right answer. The reality is the right answer doesn't exist in most cases. More right or less right, sure - but true right doesn't exist outside of math or physics.

When pilots expand to more people, those people might not be the right audience. Focus on early adopters, first movers, innovators. They'll accept the pros and cons. You want people who have used AI before. If someone's never used AI, this isn't the time to introduce them. Those are the laggards. Get to them later. Don't try to impress them - they're not going to get it.

We're in the buffering stage, like video was 10 or 20 years ago. Watch for 10 seconds, buffer for four, watch for five more, buffer again. That's where AI is right now. It's not that bad - we're seeing amazing things. But if you can't deal with the buffering stage, knowing that amazing awesomeness is coming, you're probably not the best person to be part of a pilot.

# The Biggest Misconception Executives Have About AI

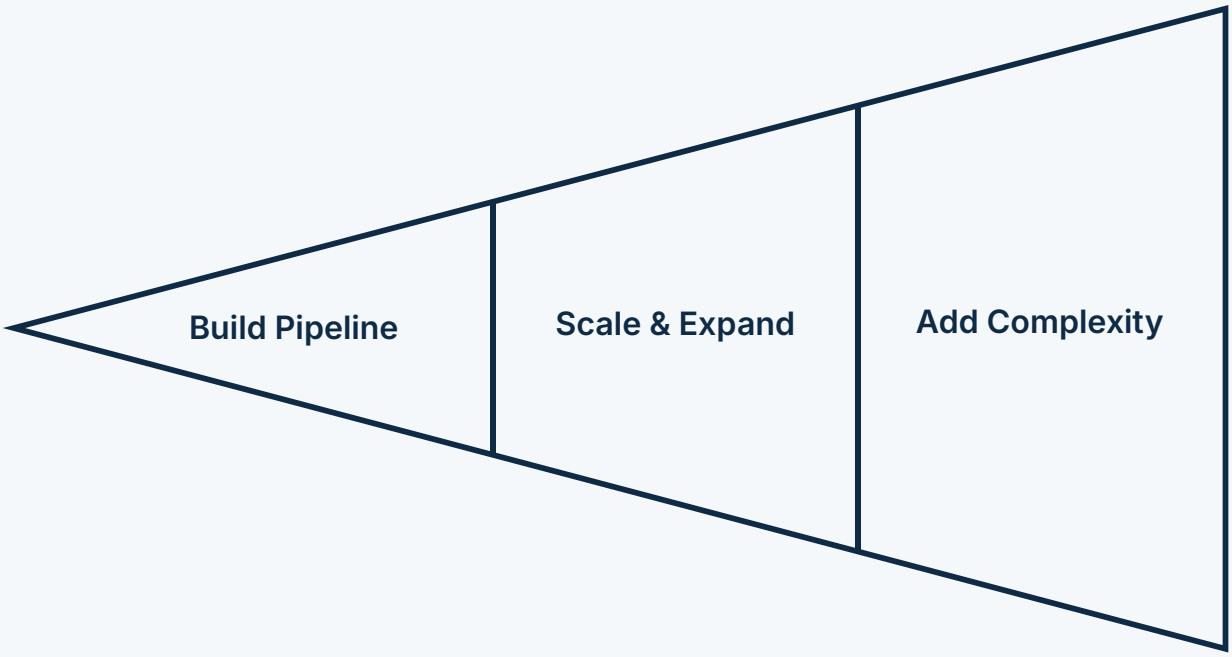
Executives think AI will deliver big gains, fast, with minimal input.

Be careful with that assumption. It doesn't have to be expensive or ultra time-consuming - that's not my point. But you need the right people inside your organization - capability, domain expertise, technical know-how. Get those people lined up, and they can deploy something awesome - demo to pilot to production.

# What Almost Always Goes Wrong in the First 60 Days

There's a ton of excitement up front. Then you get into the reality of the use case, and the scope keeps shrinking. Sometimes people feel like it's been whittled down to nothing - no value left.

**But you've got to prove you can get it done.**




Build the pipeline first. Get something through from input to output. Once that pipeline exists - that's scalable, that's extensible. Then you can add more inputs, make the path more complex, add more layers. But first you need that base layer done. Inputs here, outputs here, process in between.

Once the pipeline is flowing, you can expand. Scale stops being the problem. Now you can speed it up and scale it out.

# Scars & Lessons

### Scars

- We've built demos that impressed everyone in the room - and went nowhere.
- We've watched pilots die not because the AI failed, but because scope and stakeholders exploded too early.
- We've underestimated how quickly excitement turns into friction once real infrastructure, security, and process show up.
- We've seen projects stall for months while teams debated perfection instead of proving value.

### Lessons

- A demo only needs to answer one question, "Is this worth doing at all?"
- The fastest way to kill momentum is to make something "enterprise-ready" too early.
- Skinny wins build trust. Trust buys you room to expand.
- If you can't ship something small quickly, you won't ship something big later.

# What This Chapter Is Really About

This chapter isn't meant to discourage you. It's meant to prepare you.

The gap between what people think AI projects look like and what they actually look like is where most failures happen. Not because AI doesn't work - it does. But because expectations are wrong, timelines are wrong, and teams aren't set up to move fast.

The companies that understand this reality - that treat AI as **10% magic and 90% execution** - are going to ship while everyone else is still in kickoff meetings.

> Every week you spend in planning purgatory is a week your competitors might be shipping.

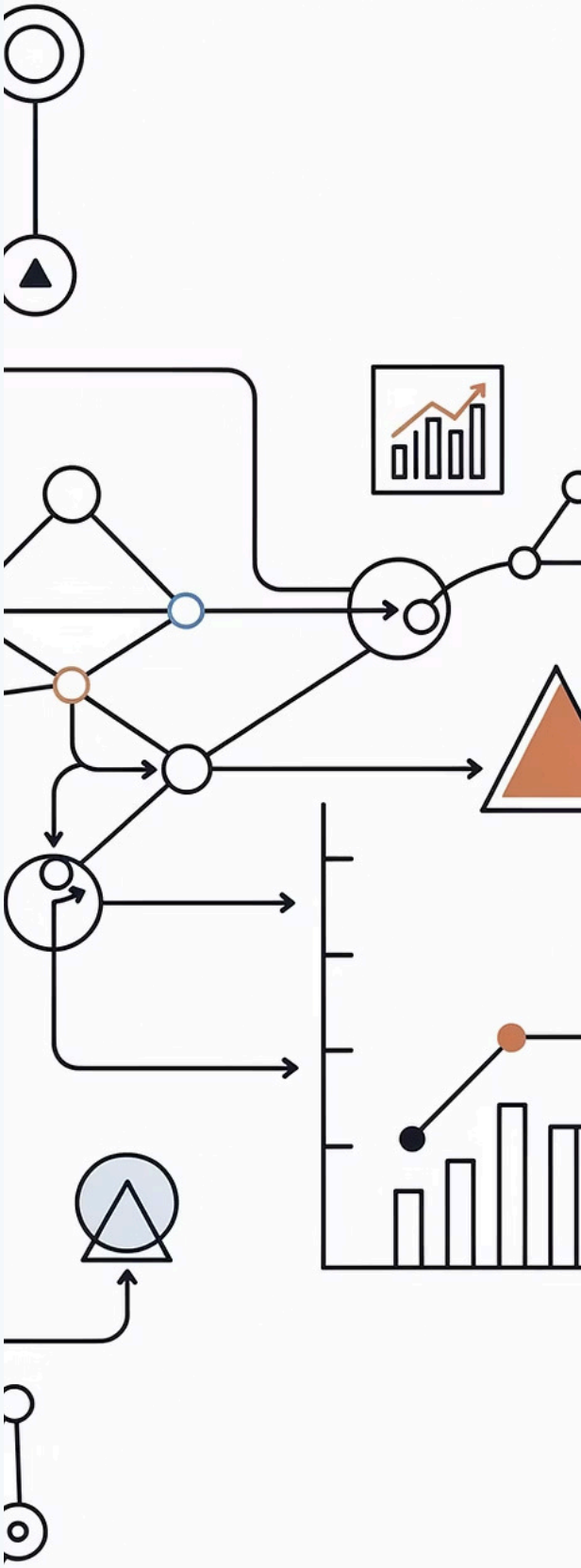
> Every month you spend chasing perfection is a month of leverage you're leaving on the table.

The good news? The path forward isn't complicated. It's just disciplined. Start small. Prove value fast. Build the pipeline. Then expand.

That's the game. Now let's talk about the hard truths nobody wants to hear.

CHAPTER 2

# The 10 Hard Truths About AI in Real Business



# Hard Truth #1 - AI Is Mostly a Data and Process Problem

Garbage in, garbage out. If you don't have good data coming into the system, you're never going to have good data going out. But don't let that paralyze you. You don't need perfect data to get started.

We spent the last 10 or 20 years trying to centralize everything in data warehouses. Dashboard tools everywhere. Great debates about architecture. And it probably hasn't transformed businesses as much as we thought it would. We were living in a utopian dream - get one dashboard for the whole business, one place to see all the pros and cons, and we'll beat the competition and sell a lot more.

I don't think that really happened. We got a lot of data, but we found out data is messy. There are 10 different ways to count things, and none of them are necessarily right or wrong.

> At one of my last jobs, if you asked how many customers we had, you'd get three different answers. Sales had a number. Finance had a different number. Operations had a very different number. All of them had good reasons.

But if you grabbed the wrong number and used it as a numerator or denominator - because you just assumed whatever you were given was right - all your other calculations stop making sense. You're starting from the wrong spot.

It's also a process problem. A lot of processes are tribal knowledge living in people's heads. It doesn't feel like a process because nobody's called it one. But there are processes for almost everything, sales orders, new customer onboarding, complaints, invoices, escalations, code reviews. Some documented, some not. All of them complex. All of them done differently depending on who you ask.

If multiple teams do the same process, they all do it differently.

## So how can AI help?

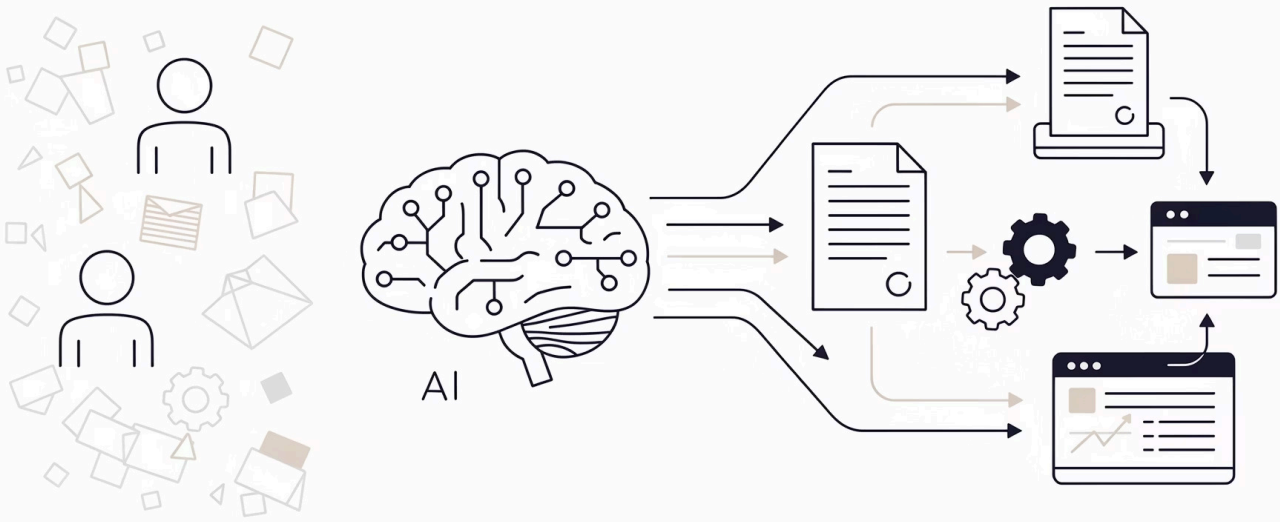
Start by asking, do you even have the data to make these decisions? Are you working off low-quality data? If so, move on - find a different use case. If you're working on a process that hasn't been identified yet, figure out the exact steps and exact parties involved first. If that's clear, you can plug AI into it. If it's not, you have to do that work before anything else.

That's why things take so long. The process isn't identified. Or it's not written down. Or you ask five people what the process is and get five different answers. That means it's not a good project for automation. If they could have automated it already, they probably would have.

Here's the move - if a process is documented, start by implementing AI in just one knowledge-based decision point. Do that well, then figure out how to insert AI in more steps. You'll accelerate time-to-completion without boiling the ocean.

But when you hit a wall, step back further. Ask why the process exists the way it does. Usually the answer is humans have real constraints in time and capacity. We designed the process around those constraints. If you remove those constraints, new solutions open up.

Humans overcomplicate things. We solve problems one step at a time because that's what we can handle. But AI doesn't have those limits. Give it the inputs, the desired outputs, and the parts in between - you might get a completely different process. More efficient. Faster. More scalable.



# Hard Truth #2 - Model Choice Almost Never Matters

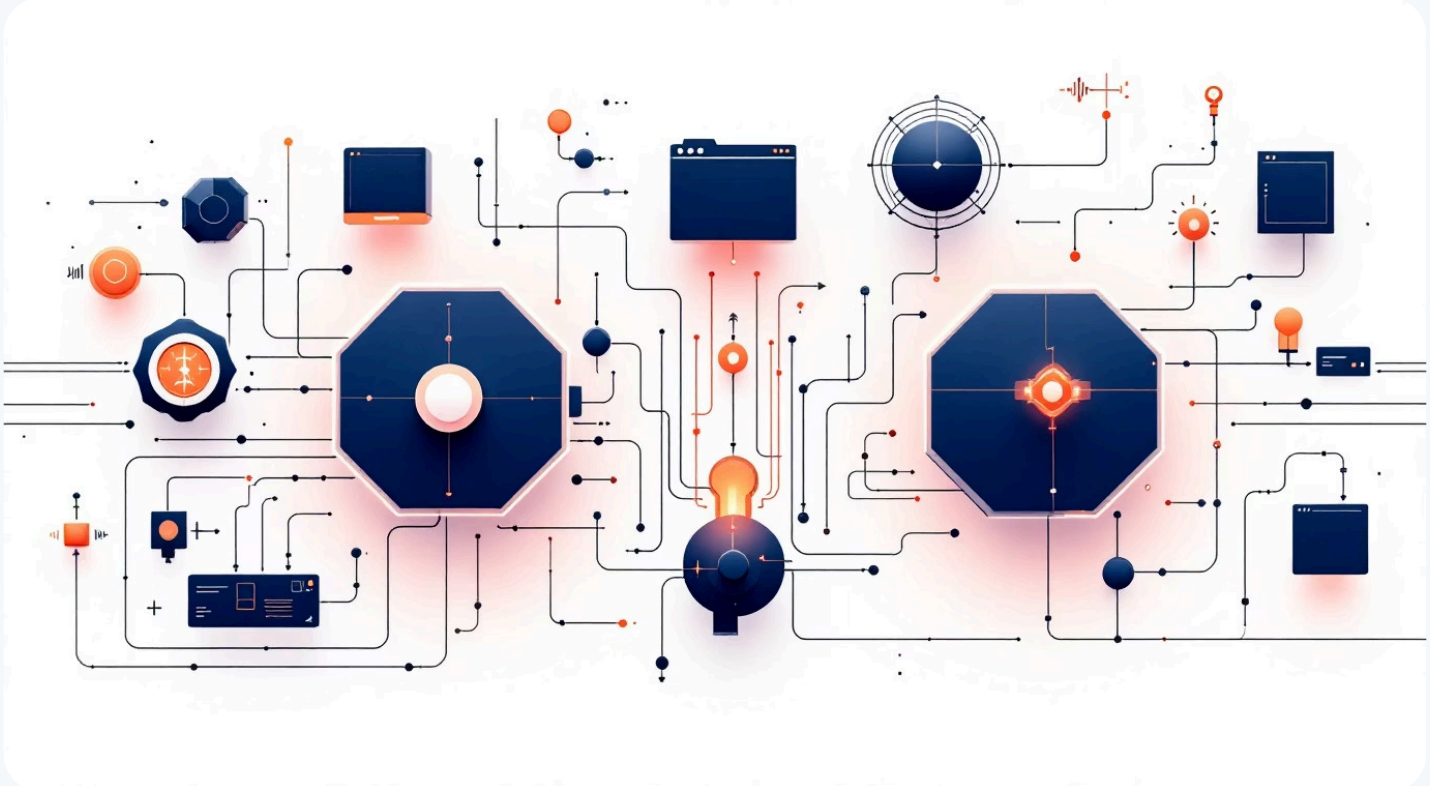
There's always going to be a new model. Right now they're dropping every three to six months. ChatGPT, Gemini, Claude - the frontier keeps moving. Hard to tell if the pace will speed up or slow down.

**Here's what matters - abstract the model from the process.** Think of AI as the intelligence layer. Independent of which model you're using, there's a step in your process that calls out to AI - make a decision, do a task, use tools. Define that step clearly, pick whatever model works today, and make sure you can swap it tomorrow.

GPT-4 models are being decommissioned. If that's hard-coded anywhere in your application, you need to upgrade and rip it out. Your system has to handle new models without breaking.

Different models excel at different things. One's better at image creation, another at code generation. ChatGPT was the best at everything for a while, but others are catching up and surpassing it in specific areas. That's going to keep happening.

Your job is to build something robust and durable that withstands any model change.



# Hard Truth #3 - Accuracy Expectations Are Delusional

Accuracy sounds like a straightforward goal. It's not.

Outside of math and physics, there are no universal truths. Accuracy can be boiled down to math, sure - but then there's interpretation. Is this the right answer or the wrong answer?

When you're evaluating a chunk of text, determining right versus wrong is surprisingly hard. Some people say an answer was too verbose. Others say it wasn't verbose enough. You'll never satisfy everyone's personal preferences. You can configure AI to respond in different tones, depths, levels of detail - all through prompting. People will still call it inaccurate.

Maybe it doesn't source things how they wanted. Maybe it referenced an older document. Maybe the documents in the system are wrong and the AI is actually right. That's a human problem, not an AI problem.

> Traditional software is deterministic. Black and white. If X happens, I expect Y. You can trace defects to root causes. With LLMs, you're black-boxing it. You're saying give me an answer based on what you know and what I've fed you. Some control is lost.

We're moving from a deterministic world to a non-deterministic world. That doesn't mean you sacrifice accuracy. It means you reframe what accuracy even means.

Think about human-level accuracy. If a machine hits human-level accuracy, you're already far ahead - because you also get machine speed and machine scale. As fast as you want. As broad as you want. The human stops being the bottleneck.

Customer support teams, for example - often undertrained, underpaid. Quality is "good enough" right now. Assigned qualitatively, not measured.

When people ask me about accuracy, I ask back, what's your current accuracy? Exactly zero times has anyone given me a number. They say it's important. They believe it's important. But they're not measuring it today. So how do you measure it tomorrow? What's the baseline?

## A Real Example

We worked with a global medical device manufacturer doing data classification - cases came in and got classified into a four-tiered hierarchy. Could be classified as 1, or 1-2, or 1-2-1-7, depending on how you were trained and what you decided to do.

We worked with one individual, and the AI scored 74% against his results. He graded each output - good, not good, good. 74% felt solid for a first pass.

Then another person graded it. We dropped to 50%.

Why? The training data came from 10 to 15 people who tracked classifications over time. What we didn't consider was the massive variability in how humans did it. Some always picked the highest level. Some went to the deepest level. Some were basically random.

So we ran an experiment. Took one person's output, gave it to another person to grade. Grades between individuals were wildly different. The AI's grade was no better or worse than human grades. We couldn't tell which was AI-graded versus human-graded. Some humans even graded their own past work as wrong.

Too much variability. Accuracy becomes a red herring.

But here's what mattered - they felt current quality was good enough. If they could replicate that quality while improving speed - getting back to customers faster - and solve the scale problem without hiring more humans when volume spiked, we'd solved it at current quality.

**Here's how that played out.** This same company had a compliance backlog of over 100,000 unreported device incidents. They were assembling a 6-8 person team to manually classify incidents and produce reports in government-mandated formats - a process expected to take a full year.

We deployed AI-powered classification and summarization. The entire backlog - 100,000+ cases - was cleared in a single afternoon.

**Human-level accuracy plus machine speed. That's the math that matters.**

# Hard Truth #4 - "We'll Just Use ChatGPT" Fails

Someone sees something impressive in ChatGPT and thinks, that's going to be my business process.

If you're a one-person operation, maybe that works. But if you want something truly transformational for your business, just pointing people at ChatGPT isn't going to cut it.

All the process limitations we talked about - tribal knowledge, people using it differently, data inputs varying person to person - that creates different outcomes. Different outcomes create accuracy problems. You're back where you started.

ChatGPT as a single shared tool might be fine for proof of concept. But if you want your whole company to benefit in a consistent, standardized, predictable way - the things that actually matter for a business - you need to build a process around it. Make it more deterministic. Remove variability at every step.

| 1 | 2 | 3 |
| --- | --- | --- |
| **Define the Process** | **Wrapper It** | **See Real Gains** |
| Have AI play its role in the right part | Right outcomes, right data in, right data out | Consistent results at scale |

Anybody can go to ChatGPT and get one great answer. If you want that answer a thousand times across your company, you need to think about it completely differently.

# Hard Truth #5 - Security and Legal Always Show Up Late

Security and legal always kill momentum. They're necessary - but this is where demos and pilots go to die.

You have to get them bought in early. Get their heads wrapped around what you're doing. What you don't want is to spend real money building something, only to have it squashed at the end because you're not compliant or legal decides to shut it down.

Start with internal use cases. Less exposure, more control. If you want to get AI moving in your business, that's the path. Teams will be much more willing to proceed versus doing something customer-facing right out of the gate - especially when you don't know what variability might pop up.

Get security and legal involved early. You're not going around them. Get them bought in, scope it small. Remember, this is 10% AI, 90% normal software development. Legal and security play a role in that 90% anyway. Account for it, get ahead of it, push them to move quickly.

# Hard Truth #6 - Adoption Is Always Harder Than Building

Building is the fun part. Avoid the build trap, and you can crank out cool stuff all day.

Adoption is hard because it's change. You're asking people to do something a new way. You've defined a process. Even if it was loosely defined before, now you're enforcing something stricter. Gates. Measurements. Metrics. If someone's not following the process, they're having a conversation about it. They feel like their autonomy got reduced or removed.

How do you keep it open-ended? How do you make it not feel strict and restrictive?

Make it exciting. Show real value. Don't load it up with requirements. Just get people using it.

But implementing that in current processes, with current tools, current expectations? People default to, "I know how to do it the other way, I'll just keep doing it that way until I'm forced to change."

That's not where you want to be. And those aren't the people to start with. Find the people who want change. Let them be the drivers. The rest of the organization will follow.

# Hard Truth #7 - Internal Politics Kills More AI Than Tech Ever Will

Lots of stakeholders. Lots of opinions. Everyone thinks they have the company's best interests in mind. Everyone also thinks they have ownership over their piece.

Here's the dynamic. SaaS over the last 20 years took a lot of responsibility out of IT's hands. IT wasn't owning the infrastructure these programs ran on anymore. Marketing or finance could unilaterally go buy software. But that software still needed SSO integration, authentication, database connections, ties to other internal applications. That created a ton of work for IT. Most of it unplanned. That caused a lot of frustration.

Now AI is coming. There are AI SaaS companies, but it's so new - not many of them, or they're small point solutions. AI is also showing up inside Microsoft Copilot, Salesforce AgentForce, Gemini Gems. Built into native platforms, which helps. But it still requires IT to enable access or give people the capabilities to use what's already there. Or it requires IT to build new infrastructure in Azure, GCP, AWS.

**Power is shifting back to IT.** If you have to work with IT to get things built, they're the gatekeeper. For good reason - they need oversight. But it slows the business down and causes infighting.

Best advice - be open-minded about solving problems collectively. Don't create a pile of unplanned work for IT that shuts everything down.

There are also people who want credit, who want things to be their idea. Security concerns, legal concerns, privacy fears, hallucination worries - people will use those to shut things down. Maybe valid, maybe not. Take all the contrarian viewpoints and still craft a solution that works.

# Hard Truth #8 - AI Strategy Decks Are Mostly Useless

Just like most consulting decks are mostly useless. Here's what we're going to do. Here's how we'll do it. Here's how we'll govern it. Everyone says the same things. No real differentiation. It's all important - you need those pieces. But everyone's deck looks the same.

What's your cloud strategy? In an ideal world, one cloud provider, everyone knows how it works. Reality? An acquisition put you on a different provider. You're already multi-cloud. One application has different requirements. Multi-cloud happens fast whether you planned it or not.

Same thing with AI. Some things will run on ChatGPT, some on Claude, some on Gemini, some on whatever's next. That's just how it's going to be. No unified strategy. Some will be point solutions. Some in your native cloud. Some self-hosted locally.

Strategy decks lay all this out. But ask yourself, what would your strategy deck have said last year about where you'd be today? If it wouldn't have been mostly right, why write a deck this year that's going to be mostly wrong next year?

Think differently. Find a way to accomplish where you want to go and how you want to get there - while staying nimble and flexible.

**Strategy emerges from wins. Not from PowerPoint.**

# Hard Truth #9 - Small Use Cases Beat Big Vision

Big vision is great. That's where you want to go. But you never really achieve your vision if it's a good one - by the time you're approaching it, it's already emerged into something else. That's how you keep expanding toward bigger and bigger vision.

Small use cases prove that vision is even possible. Data classification. Small chatbots. Document creation.

Knowledge creation has dropped to the price of zero. We have standard structures for everything - SOWs, contracts, marketing decks, sales pitches, QBRs, management reviews, board meetings. There's mostly structure to all of it. What we should do is firm that up and codify it. Instead of copying and pasting from the last template, turn it into something reusable.

> **Turn your business into code.**

When you have new content for an existing structure, it happens automatically. Saves time, effort, resources - because knowledge creation is free. Build a brain for your business with all your data sourced, and knowledge creation pulls from that brain into the formats you've defined. Instantaneous. Always updated. You become the reviewer of documents, the editor - not the creator.

That's the vision. How do you start small?

01

### Pick one document type

SOWs. Scan all your existing SOWs. Have AI identify the different formats and use cases - enterprise SOWs, SMB SOWs, managed service SOWs, product SOWs. Now you have format and structure. Small project, no real impact on anyone. Just classifying, categorizing, creating structure around what already exists.

02

### Turn notes into content

Take notes and transcripts from customer sales calls. Turn that into content. Throw that content into the predefined SOW structure that best fits. Now you've got a ready-to-go SOW pulling content from internal calls, external calls with the customer, notes from those calls, plus follow-up internal meetings. All merged into structure.

That dramatically accelerates SOW creation. That's leverage.

# Hard Truth #10 - Most Teams Aren't Actually Ready for AI Yet

First movers, innovators, early adopters - they're out there. But not many. A lot of people use ChatGPT the way they used to use Google. They haven't realized the full value. They don't know how to build a good, thoughtful, clean prompt.

Think about how I'm using it right now. AI voice system - Granola - recording all my thoughts. I'll take this transcript, push it into an LLM with a predefined agenda and scope for the content I'm writing. It's going to work beautifully.

But I know how to use AI. Three years of doing this. Building it for customers. Learning the tips and tricks is what enables you to actually take advantage of the tools.

Teams need to be nimble. Willing to take risks. Willing to fail. We've tried fine-tuning models - more times than not, we turned the model into a dumber version than what we started with. It takes a lot of time. RAG plus prompting on foundational models gets you about 90% there. Fine-tuning takes attention, work, effort - and if you want to add new data, you have to retrain. A lot of effort that could be bypassed.

Most teams don't know this stuff. That's fine - everyone will learn over time. It'll get easier. But this is why most teams aren't ready. They've got the excitement, the desire - but they don't know how to execute because they're not used to building custom software.

If you build something internally and roll it out to the whole organization, you're going to get feedback. What about this? What about that? I have three great ideas. This doesn't work the way I expected. I think this is a bug. I can't log in.

All of that is someone's problem. If you don't have that person internally - and you don't have budget for that person to exist - you'll roll this out and adoption will be painfully slow. Maybe nonexistent. People won't be happy. They'll have recommendations. You'll have no money to act on them. Excitement wears off. They stop using the tool.

If you don't have solutions to those problems, go back to the drawing board.

# Scars & Lessons

### Scars

- We've chased accuracy targets that turned out to be human preference debates.
- We've fine-tuned models and made them worse, not better.
- We've watched legal, security, and compliance stop projects - not because they were wrong, but because they were brought in too late.

### Lessons

- Human-level accuracy plus machine speed is already transformative.
- Model choice matters far less than process design.
- Most AI problems are data and workflow problems wearing a technical costume.
- Getting buy-in early beats arguing for exceptions later.
- Progress comes from disciplined execution, not perfect answers.

---

# What These Hard Truths Add Up To

None of these truths are meant to discourage you. They're meant to equip you.

The teams that internalize these realities are the ones who ship. They don't get stuck in strategy purgatory. They don't spend six months on accuracy debates. They don't let politics kill momentum. They don't wait for perfect data or perfect processes.

**They start small. They move fast. They build leverage.**

Every week you spend learning these lessons the hard way is a week your competitors might be shipping.

You've been warned. Now let's talk about what actually works.

CHAPTER 3

# AI Hype vs. AI Reality

## These Are Patterns, Not Products

Before we get into the specifics, understand this - these are not product pitches. These are patterns - things we've built multiple times that actually delivered value.

The goal is not for you to copy exactly what we did. It's for you to see the underlying shape of a good AI use case and recognize similar opportunities in your own business.

# Pattern #1 - Internal Knowledge Search (The Company Brain)

If you believe AI is going to be core and critical to your business in two to five years, and you believe your business is going to have automated processes and agents running things and assisting humans, then you should also believe that those agents need to be plugged into a central brain. One that knows something about your business, your personality, and your desires - versus just what the frontier models can tell them.

Without that, you just have normal agents running around. For some tasks, that might be okay. But for a lot of tasks, it's not good enough.

What are your internal policies and procedures? What's the process from quote to cash? What's the process for escalations? Those things are important. If you don't have them in the brain, you're just using general knowledge. That could be okay, but it's going to have faults.

You need to start building this internal brain of your business. Not all the documents, just the critical ones. The right ones.

# The Shift From Creation to Curation

Knowledge generation has gone down to the cost of zero. As people worry about where jobs are going, the shift will be from knowledge creation to knowledge curation. We're going to have people almost like librarians, making sure the content feeding the AI brain is the good, right, and correct content.

This is not about determining which information is appropriate from a political perspective. It's about identifying the critical pieces of information you would use to train an employee. Imagine how you would train a salesperson. A customer support person. A finance person. Take all that information - which, let's be honest, most of those training programs aren't very good right now anyway - and put it into the system.

As you update the information, rip out the old and drop the new in. That way everyone accessing the brain knows exactly what the latest, correct, most accurate information is.

> Then you can ask questions of your business. It's like you've got the smartest employee ever. The brain now knows everything about sales, finance, all your customers, all your processes, all your people, their skill sets, their talents, how long your business has been going, your historical sales, and your future projections.

I've done this. It's a lot of fun. I really enjoy chatting with AI and having it create ideas and thoughts I hadn't considered, then taking those down the path and figuring out if they can be implemented.

# The Build

We've built this many times. It's nothing more than a standard RAG model. Easy to build and very impactful.

Break it up into different groups and start small. It would be great if you could integrate into Google Drive or SharePoint or OneDrive, but don't start with that. Just have each department create their own folder and upload information to it. You can chat with those things and get all the information you need.

**Security is a key concern.** Don't put any information in there that can't be accessed by everyone unless you have a good security protocol in place - which you probably don't because you're building your first version. Future versions? You can definitely do that.

# Pattern #2 - Document Processing and Summarization

This is what AI is awesome at.

I saw a 60-page report from Anthropic the other day. Most people probably took that report, put it into Claude, and said "summarize this for me." Which is exactly what I did. Because it's really good at taking complex, wordy, lengthy content and turning it into something much more consumable.

We live in TikTok world. We're all used to 10-second videos. Reading 60 pages is not something most people do.

This is great for SOWs and contracts. You can summarize PowerPoints, Excel sheets, almost anything. It's a great way to get a lot of content summarized very quickly.

# Pattern #3 - Customer Support Deflection and Triage

### The Real Example

We were dealing with a complex medical device. The end administrator of the device didn't know how it worked on the inside. The customer support person didn't either. They had manuals they'd been lightly trained on, but it was a tough, complex machine.

If something broke, they'd call customer support, and the team wouldn't know exactly how to handle it. They'd classify it. If they could identify the issue from the existing knowledge base, great. But a lot of times, they just didn't know.

# What We Built

We built an AI solution that listened to the call transcript, or read the email. It determined what the issue was. It highlighted the relevant section in the manual or internal knowledge docs to help the support person identify what could be solved. And it highlighted what it thought was the best classification.

Once the machine had the classification, the case could move to the next step - either the factory or the engineer for evaluation.

That classification process normally took a long time. Lots of phone calls, lots of checking with other agents, lots of checking with other people and knowledge sources. It was greatly extending how long it took to get back to the customer and for the case to move along.

For cases where something was truly broken and needed a technician, that time was even more extended because it had to be verified by management or an engineer. By quickly getting to classification, we kept the process moving so technicians could be deployed much sooner.

# The Real ROI

Customer satisfaction increased. But the most important thing was that they were expecting a large increase in volume because of new product releases. They needed to hire people to handle the volume, but they didn't have budget.

This tool enabled everybody to get through cases much more quickly. They were able to handle the additional load and volume without hiring additional resources, all while maintaining better satisfaction on both the customer and employee side.

> **Win-win for all parties. That's leverage.**

# Pattern #4 - Sales Enablement Copilots

This was a global sales team with people all over the world speaking different languages.

The challenge was that if they had questions about what they were selling - and these were complex devices - they would reach out to the marketing team. How can I sell this? For what indications? The competitors are saying this, so how do I compete?

The marketing team was based in the United States with about eight people. When questions came in through the global email distribution, well, we've all dealt with email distributions a thousand times. Sometimes nobody answers. Sometimes three people answer with three different answers. Sometimes one person answers but it takes them two days.

Just a really outdated way of communicating.

## What We Built

We built a sales enablement chatbot. We put all the marketing information in - battle cards, competitive information, matrices around competitor products, internal and competitor websites, price sheets, industry analysis, internal analysis.

When the sales team had questions, they could go to the chatbot, ask a question, get the answer immediately, and then go back to the customer with high confidence and high-quality output.

## The Time Savings

### Response time went from 7-9 days to under one minute.

That's not an estimate. That's measured. Before, they'd ask a question, and it would take 12 hours to get a response. The response was kind of good but not all the way there. There would be follow-up questions. It was all asynchronous. It just took a long time and that became the norm.

Think about that. 7-9 days not getting back to the customer. 7-9 days opening a door for a competitor to swoop in. Not a good look.

Now sales reps can review information quickly, ask the chatbot any questions, and go right back to the customer with marketing-approved language and content. Everything is linked and sourced so they can find and send files very quickly.

# The ROI

One additional sale of these complex devices would have more than covered the entire build and implementation expense. The project wasn't expensive - lightweight, with less than 1,000 total documents and URLs included. Very straightforward build and the rollout was done well.

Within two months, the company transformed its document delivery process from sluggish to near-instant. Marketing staff were freed up to focus on high-value projects instead of fielding repetitive requests.

# The Surprise Benefit

## The Training Tool Nobody Expected

This became a great training tool. Sales reps in their off time were chatting with the bot.

- "How are competitors positioning against this?"
- "I've got a specific customer looking for this, so how does my product best meet that?"
- "Give me marketing pitches to beat this competitor who's been embedded for five years."
- "Give me creative ways to reach out to a customer who hasn't been responding."

All of that interaction created a sticky product that enabled the entire sales team to change how internal processes work.

# Pattern #5 - Operations Automation (Where RPA Failed)

This was another company in healthcare. They deal with a lot of purchase orders from various vendors. Each PO is slightly different and can change at any time.

What they actually did - POs would come through email, they'd open the file, switch to the ERP, and manually type in the SKU, quantity, price point - all of it. They were getting the purchase order and then manually entering it into another system using humans.

## Why RPA Failed

They tried to automate in the past. The challenge was that using RPA, any small change in the PO form would break the system. They'd have good success for a while, and then there would be a minor change that could potentially break the whole thing. They'd have to rework it. It just wasn't worth it. Not consistent and robust enough.

## What We Did Differently

### The AI Solution

We put AI around it. We passed through the PDF in two forms - turning it into a text file and also putting the PDF in raw. By passing both, we had near 100% accuracy.

Now, I know I said accuracy was delusional before. But that's only when looking at a non-deterministic case. This is deterministic. It's definitely true that this number says 900 or this number is 12. If you have a golden truth, you should measure against it with accuracy.

We did, and it was 99-plus percent. In fact, I think we only saw one error the entire time. We weren't entirely sure if the AI misread it or if the source document itself had an error. But either way, we're almost certain it was more accurate than the humans had been.

So now we remove the human from the process. The email comes in, it's immediately picked up by the system, and the data is dropped in.

It did take an integration with the ERP, but it was relatively easy. Pretty open API and we were only doing lightweight number drop-ins.

# Pattern #6 - Intake and Triage Systems

This was a program that took top college students, soon to be graduates, and connected them with businesses looking for highly entrepreneurial, highly motivated, highly ambitious students. They run them through a process to find the best of the best, then place them at top companies in the local area with mentoring to help them grow.

They get thousands of applications, narrow down to a couple hundred, and put them through final selection where everyone interviews with partner companies. Then they're matched up based on how the interviews go. A little like speed dating.

## The Problem

There were a lot of applications, and it took a long time for existing members to review every application, every resume, and see if they met standards. There were essays written for each student.

## What We Built

### Natural Language Search for Candidates

We took the rubric they created and inserted it into the intake process. We also created a chatbot around each candidate.

Want to know more about someone? Where did you go to school? Looking for everyone that went to a particular school and played a particular sport? All of that was available through natural language search.

We took thousands of candidates on intake. We classified them, organized them, and sorted them based on responses. We graded against the rubric. Capabilities were pulled out through interview and essay analysis, plus predictive index scoring. All of it was combined into a natural language searching mechanism.

When partner companies were looking for people, they could easily search, sort, and filter down to exactly who they wanted without sorting through thousands of people. Faster, more pleasant experience that allowed them to find the right candidates with the right skills.

# No Bias Added

We weren't eliminating anyone from the solution. We were just organizing and categorizing information in a way that was easily found, without a bunch of filters and parameters and checkboxes.

Once you got to a candidate, you could also chat with the resume and essay. Ask questions. Get answers.

# Pattern #7 - Content Generation That Actually Gets Used

We built the backend for several startups doing awesome, interesting things.

One of them had a lot of good content - pictures, text, document uploads. Great material going directly into the application. End users used it. But it was never aggregated.

## What We Built

### The Automated Weekly Email

We built an email that would naturally pull from all the content generated in the past week.

Here are all the pictures. Here are all the points of interest. Here's the aggregated and summarized data of what happened inside the application based on user activity. It was an activity-based application. All of that was consolidated into one generated email.

That email became the weekly email sent to all customers.

Just by collecting and storing that information in a transactional system, the AI scanned through and determined what would make an interesting email, what stories to tell, what hooks to use, what would be a great subject line. Combine it all, create an email, send it out. Basically zero effort from administration.

> **That's how content generation actually works.**

# Pattern #8 - Analytics Copilots

## Analytics Copilots

At first, I didn't think AI tools were very good at analytics. Okay at best. But they're getting much, much better.

### The Real Build

A customer wanted to connect different parts of their information. Contacts from their database, deals. Connect that to data enriching opportunities. What do we know about these contacts? These businesses? What services are they buying? Then run that against customer satisfaction.

They had three or four different data sources that were disparately stored with no data warehouse.

We collected all this information and built a data warehouse where we piped data in raw. Then we trained the AI with queries, showing it the types of questions that could be asked and the kind of query you would run. We also imported the entire schema, including primary and foreign keys and connections to all the tables.

Once it knew that, we only needed to train it a little bit. Because of its deep expertise in databases, it was writing really good queries very quickly.

All of this was behind the scenes. A natural language question comes in, it processes it. We set up an agent to go out to the databases and execute the query. It would run it, come back, and the LLM would wrap it with natural language back to the user.

Natively in the application, you could ask for a chart or graph. Trim this over time. Extend the data. Change where clauses. All of that wasn't available in one tool before because each source had its own dashboarding system. There was no centralized data warehouse connecting them. No way to get it.

# The Meeting Problem Solved

How many times have we been in a meeting where analysis is presented, somebody asks a question, and the answer is "I don't know, but I'll have to get back to you"? That normally takes days, if not weeks.

We were able to do it in the moment. Ask and answer that question in those meetings, all via chatbots connected to data they'd never used together before.

Fantastic use case. We're going to see a lot more of that.

# Why These Patterns Work

### Scoped Small

One clear problem, one clear outcome.

### Built on Existing Processes

We didn't invent new workflows; we improved existing ones.

### Measurable

Even if qualitatively, you could tell if it was working.

### Valuable Without Perfection

90% accuracy on a process that was taking forever is still a massive win.

The use cases that work are not the flashy demos. They're the boring, repeatable processes where you can actually measure before and after.

# What You Should Do With This Chapter

Don't try to copy these exactly. Your business is not the same as theirs.

Instead, ask yourself,

- Where are my people doing repetitive knowledge work that could be accelerated?
- Where am I losing time to "I don't know, let me get back to you"?
- Where is data scattered that should be connected?
- Where are we manually moving information from one system to another?

Those are the patterns. The specifics will be yours.

> **That's leverage sitting on the table. The question isn't whether these patterns apply to you. The question is how long you're going to leave that leverage uncaptured.**

Now let's talk about the patterns that don't work - so you can avoid the expensive mistakes.

CHAPTER 4

# The Use Cases That Sound Great and Usually Fail

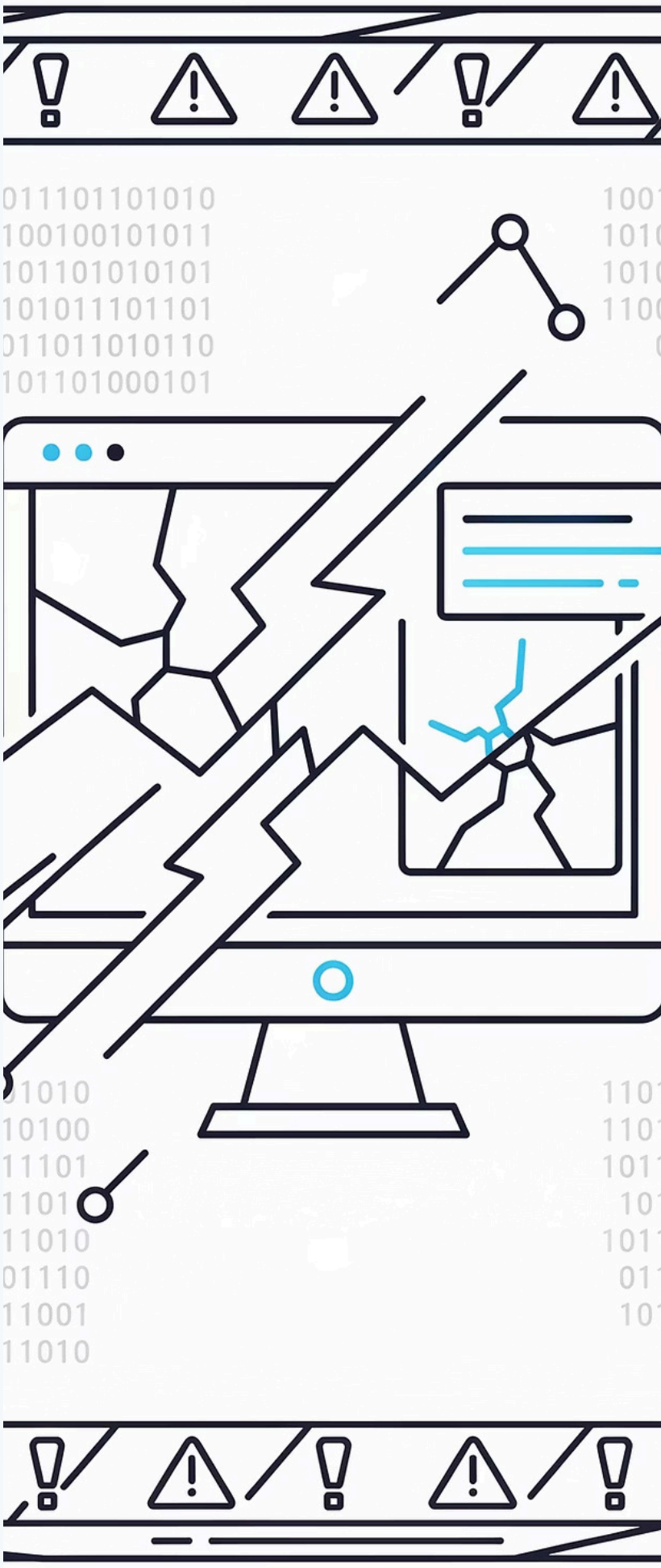
If Chapter 3 was about what reliably works, this chapter is about what reliably hurts.

Not because the ideas are bad. Not because the technology isn't capable. But because the surrounding systems, incentives, and expectations aren't ready for what people are actually trying to do.

Almost every failed AI project I've seen started with a genuinely good idea. And almost all of them failed for the same few reasons.

This chapter is here to help you recognize when you're trying to do something too early, too broadly, or with the wrong expectations.

Most of these ideas eventually become possible. They just become possible much later than people think.



# Pattern #1 - The Fully Autonomous AI Agent

This is the most common fantasy.

"We want an AI agent that just does the whole job."

Books orders. Runs payroll. Handles customer support. Manages scheduling. Qualifies leads. Onboards customers. Closes tickets.

It sounds incredible. And someday, pieces of this will work.

**But right now, this almost always fails in production.**

Why?

Because real jobs are not single workflows. They're bundles of decisions, exceptions, judgment calls, tribal knowledge, political context, and risk tradeoffs.

What people call "a job" is actually 50 tiny workflows stitched together.

When you try to automate all of that at once, three things happen. The workflow explodes in complexity. The number of edge cases goes to infinity. And nobody agrees on what "done correctly" means.

The result is chaos.

> Think about agents at the smallest, smallest level instead. I want this agent to do a daily search on this list of competitors and return any news articles that have come out in the last 24 hours. A standard search. This is really no different than what old RSS feeds used to be, but it's a great way to have a tool that does one thing and one thing only. That's an agent.

The beautiful part is once you get that agent out there, you start to understand how it works and how to integrate with it. Then you can expand. Add configuration. Add more sources. Make it more robust.

But start with the very basic, most simple thing. Once you have that, you can plug it into a myriad of operations in many different ways.

# Pattern #2 - "Let's Replace This Whole Role With AI"

This one usually comes from leadership.

"If AI can do X, why do we still need people to do Y?"

It sounds logical. It's also almost always wrong.

Humans are not a great way to classify automation. Humans are just a collection of tasks. Each job and role has a different set of tasks. Some are single step, some are multi-stepped. Most of them are multi-stepped, multimodal, and multidisciplinary. They're very complex.

So don't try to replace a human. Try to catalog and inventory what the humans do. Break those down into small, bite-sized chunks and then see if you can automate those pieces. That's where you should start. Small.

Replacing humans backfires because you think you know what the humans do, when in practice you actually don't. There are a lot of things that happen that aren't recorded, aren't known to management, especially executive management. Those people do a lot of things they're not accounting for. They say this is a pretty fast and easy process, when in reality it's not.

This can be tough, especially when humans are thinking they're going to get replaced by robots. But it could also be a really fun exercise. I really enjoy documenting all the things I'm doing and how I might be able to do them differently.

The point is that even when AI can technically perform many of the tasks in a role, removing the human usually breaks the system in subtle ways. What actually works is using AI to remove the worst parts of people's jobs. Free up time. Increase leverage. Reduce burnout. That's where real ROI lives.

# Pattern #3 - Unstructured Data Everywhere

This one kills timelines.

"We have tons of data. Let's just feed it all into the model."

The data usually looks like PDFs, scanned documents, Word files, email threads, sloppy spreadsheets, and notes. It's inconsistent. It's messy. It's unlabeled. It contradicts itself.

Before AI can do anything useful with this, someone has to clean it, normalize it, label it, version it, and decide what's authoritative. That work always takes 10x longer than people expect.

Unstructured data could be anything - Word docs, Excel sheets, images, PDFs, anything that's not a structured data point that fits into a structured table. You can still fit unstructured data into a structured table. Here's the document name, here's the created date, and then here's the JSON blob of everything in that document. There are ways to put structure around unstructured data.

But it's the internal parts of that unstructured piece that make every bit unique and dissimilar from all other data points. You want to have a way to search and find that content, and unstructured data can make this harder.

> This doesn't mean you can't use unstructured data. There's a lot of opportunity there. But the key point is this - **unstructured data projects are data projects first and AI projects second.**

# Pattern #4 - Fine-Tuning First

This one is subtle and very common.

Teams jump straight to fine-tuning. Not because they need to. But because it sounds advanced. It sounds serious. It sounds like "real AI."

In practice, this almost always backfires.

As I mentioned earlier, we've tried fine-tuning models multiple times - more often than not, we ended up with a dumber model than what we started with.

Fine-tuning is expensive, time-consuming, brittle, hard to maintain, and hard to update. It locks you into one model family.

In most real use cases, RAG plus prompting gets you 90% of the value with 10% of the pain.

Fine-tuning only makes sense when the domain is extremely stable, you have a large labeled dataset, you need extremely consistent outputs, and you understand exactly what you're optimizing for.

Otherwise, it's a distraction. Fine-tuning is a last-mile optimization, not a starting point.

# Pattern #5 - The Eternal POC

This one deserves its own warning label.

A team builds a POC. It works. Everyone loves it. So they keep going.

They add features. They add users. They add workflows. Accuracy expectations rise. Infrastructure hardens.

All under the banner of "We're still in POC mode."

This is how POCs quietly turn into accidental products. With no new scope. No new budget. No new timeline. No new stakeholders. No new expectations.

> **The moment the POC worked, you should have stopped and started a new project.**

What should happen - the POC is a disposable proof artifact, not a proto-product. If it works, kill it. Reset scope. Reset expectations. Launch Phase 2 properly.

Because here's what happens if you don't. Scope explodes. Accuracy expectations rise. Stakeholders multiply. Infrastructure requirements appear. And you accidentally build a product with no proper design, no real architecture, no clear scope, and no reset expectations.

This is how teams burn political capital.

The rule is simple - if your POC works, kill it. Start Phase 2 as a new project.

# Pattern #6 - Big Vision, No Owner

This one looks like strategy.

"We're launching an AI initiative."

There's a deck. There's a roadmap. There's a task force.

And there's no actual owner.

Nobody wakes up every morning responsible for shipping it, fixing it, driving adoption, fighting politics, and getting budget approved.

So it drifts.

What actually works is one product owner, one accountable executive, one narrow mandate.

Everything else is theater.

# Pattern #7 - Over-Indexing on Accuracy

I covered accuracy expectations in detail in Chapter 2 - including the story about the 74% accuracy score that dropped to 50% when a different person graded it, and the compliance backlog that was cleared in an afternoon once the team stopped chasing perfection.

The operational implication is this. Accuracy anxiety kills velocity.

Early demos work. Then someone says "This needs to be perfect." Expectations rise. Scope rises. Edge cases pile up. Velocity dies.

> **Don't let accuracy anxiety stall your project. Human-level accuracy plus machine speed equals leverage. That's the math that matters.**

# Pattern #8 - Building a Platform Before You Have a Use Case

This one is very seductive.

"We need an AI platform."

So teams build vector databases, ingestion pipelines, model abstractions, agent frameworks, and prompt management systems.

Before they've shipped a single real use case.

This is infrastructure cosplay.

The right order - ship one skinny use case, learn what actually breaks, then build infrastructure.

Platforms emerge from use cases. Not the other way around.

If you start with the platform, you're going to spend months building something that you think is awesome that nobody in the market wants. And you're going to have this big monstrosity. You show it to somebody and they see a product going 18 different ways, and it's really tough for them to understand. Where's that small little thing, that niche, that wedge where I can find value?

You have to focus on that first. Get something working. Get it in the hands of real users. Then expand.

# What All of These Failures Have in Common

- They start too big
- They assume clarity that doesn't exist
- They skip workflow design
- They ignore adoption
- They raise expectations too fast

AI didn't fail in these cases. Project design did.

# The Real Cost of These Mistakes

Every one of these failure patterns has a price tag. And it's not just money.

The eternal POC burns political capital. The platform-first approach burns months. The accuracy obsession burns momentum. The "replace the human" fantasy burns trust.

That's the real cost. Not just what you spent - but what you didn't build while you were stuck.

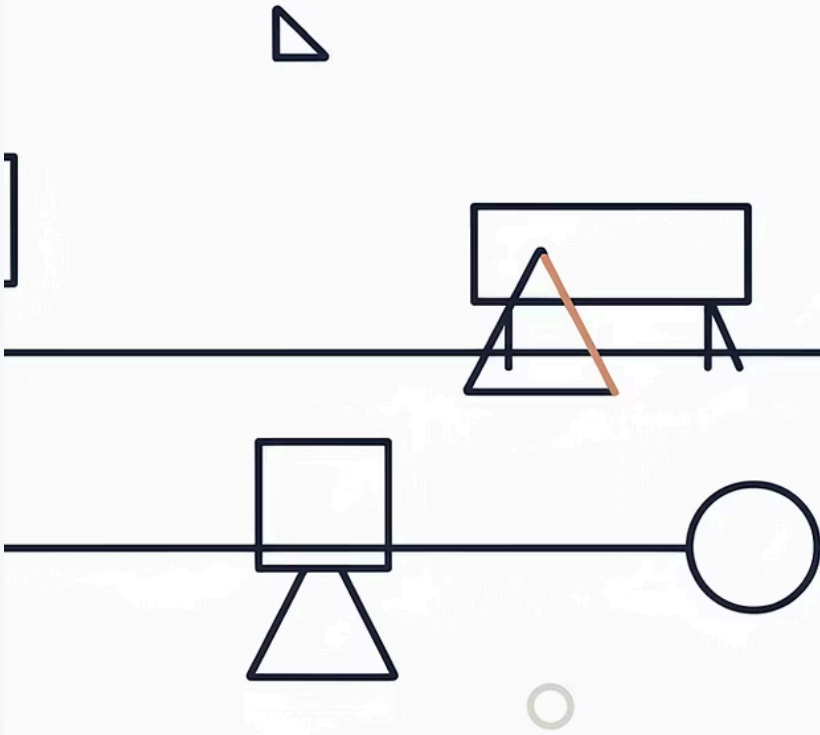
Most of these things will work someday. They just work much better after you've built muscle memory, after you've shipped skinny wins, after you've earned trust, after you've clarified workflows.

Ambition is good. Just don't start there.

Now let's talk about the real bottlenecks that kill AI projects - and none of them are the technology.

CHAPTER 5

# The Real Bottlenecks Nobody Talks About



If you've read the first four chapters, you might be thinking, "Okay. I get it. AI is powerful. It's not magic. There are traps. There are patterns that work."

At this point, most people still think the hard part is technical.

It isn't.

Chapter 2 was about the hard truths - the mindset shifts you need to make. This chapter is about the operational blockers - the specific things that will slow you down or stop you cold once you actually start building.

Once you can see these clearly, you stop being surprised when things slow down. And you stop blaming the wrong things.

# Bottleneck #1 - Data Readiness

Data readiness is a perpetual task. What data is your company producing? What data is your company integrating with? Where is it all going? How's it all connected?

Here's what most people miss. A lot of people think they can't implement AI until they've got their data perfect. I have the complete opposite perspective. Your data can be in any state. The real goal is to start with one idea, with AI or automation.

Say you want to create a chat inside of Slack that you can ask about your bookings for the last month. You need to build a chatbot. You need to connect it to the CRM. You need that CRM object connected - probably a deals object or an opportunity object.

Go make that one and only one connection. Connect to the CRM. Connect to that one object. Don't worry about doing a full integration. Don't worry about this massive amount of data you've got to pull in. Pull in that one thing, and you're going to deliver real quick value.

> Data readiness doesn't require some large, enormous exercise where everything else shuts down. "We've got to figure out this multi-year plan on how to get our data ready" - by the time you've done that, you're two years later with two more years of data that wasn't accounted for.

You need something much more flexible. Much more malleable. Get the data in and get the end users accessing and querying that data as fast as possible.

# Bottleneck #2 - Process Ambiguity

There's a ton of process ambiguity. It's cross-departmental and people don't know how to properly interact with the other groups. People haven't written things down. I don't know why they don't write things down. It's kind of baffling to me that if this is your job, you'd want to document it.

A lot of people say that things are documented. I've got firsthand experience where I was looking for a process a couple jobs ago, and I found the wiki. There were literally seven different processes for the exact same task. Which one's the right one? They're written by seven different people. I'm sure each of those seven people thought theirs was the right way. Maybe some were identical. I have no idea. But no one even checked. Or if they did, they just wrote their own documentation anyway.

We have to get out of "I'm going to do it my way" and start thinking about "I'm going to do it the company way." What's the company way? It's whatever you define it as, but it's the way that the entire group has a process around the process.

> **If the process isn't measurable in any way, then it's not a process.** It's got no value. It could be very broad or qualitative to start. But you need some measurement. This started on this day and ended on this day. These people were involved. This person approved it. All those things have to be part of a process.

If those things don't exist, there is no process. It's ad hoc, ad libbing life and business. And that's not where you want to be.

# Bottleneck #3 - No Clear Ownership

This goes back to the same issues with process. Process isn't well-defined because probably nobody owns it.

Management has a lot of fear about giving ownership to certain things because they're afraid people aren't going to be here. They're afraid that if that person's there, they're going to do it one way and then might need to hand it off to somebody else. There's a lot of change. It's hard for managers to define specific things they want their employees to do.

I've been on both the product side and the services side. I've seen a lot of managers be very strict and apply pretty good management tactics against vendors, but they can't do that against their own employees. It's very easy to hold a vendor accountable. They go back to the contract. This was completed in the SOW, this was not. Why wasn't that completed? I want a discount or I want extra time.

But that rarely happens in an employee-manager relationship. That's process ambiguity driving it. You haven't been able to drive that ambiguity down to virtually zero with your team.

> **That's why process has to start from the top. You have to demand process. You have to demand metrics. You have to demand vital information to understand the core parts of your business.** If you don't have that now, and if you can't get that done with humans, you're not going to be able to get this done with automation.

# Bottleneck #4 - No Budget for Iteration

This one surprises people.

They fund a POC. They fund a pilot. They ship something useful. And then... nothing.

No budget for fixing edge cases. No budget for improving prompts. No budget for adding logging. No budget for tightening UX. No budget for handling feedback.

AI systems are not one-and-done. They are living systems.

If you don't fund iteration, quality decays and trust erodes.

It's really hard to get definitive ROI sometimes. Sometimes it's really easy. We've done a lot of projects where a particular task took multiple hours and we reduced it to minutes. Very easily reportable savings.

But a lot of tasks like chatbots are like, "I just feel this is better. I feel like I've got my information much faster." Because there wasn't a process before, you can't measure it. So there's no reason to recreate history and define how long it took before versus now.

That's a lot of qualitative. That's okay. Just call it what it is. And if you're feeling like you're moving in the right direction, sometimes that's good enough instead of forcing yourself into tracking ROI when it might not be helpful.

But you still need budget to keep improving the thing. Build that into your planning from day one.

# Bottleneck #5 - Tool Sprawl and Vendor Chaos

Every solution I've seen for the last two years is touting all their AI capabilities. It's probably true, but it's also very complex.

All the old tools that have been around forever are saying they're going to be AI-first, which is probably somewhat accurate. But then you've got all these new tools that don't have all the history, don't have all the experience, and they're built around AI. That sounds appealing because it seems fast and easy to turn on. But can it work for your particular business's use cases? Maybe yes, maybe no.

Everyone is moving so fast and trying to grab so much land that it's tough to determine who can really do what.

There's not a best way other than to get started, learn, get some information, update your hypothesis, and move on from there.

# Bottleneck #6 - Security, Legal, and Compliance

There's all kinds of state laws right now saying conflicting things. Colorado has some pretty strict laws. New York and California were thinking about following suit. There's a lot of discussion at the federal level about guidelines that supersede state and local rules. That's what's going to be needed here, because otherwise it's going to be incredibly hard to develop and deploy AI solutions if you've got to look at each state law and regulation.

Some of these are saying things like you can't use a solution that has any bias in it at all. From a design perspective, politically that seems like it makes sense. But actually, this is exactly what AI is doing. You're not wanting a biased result that injures or hurts a human, but you are wanting to include certain pieces of information and say, give me more of this or summarize in this tone, in this voice. Depending on your definition of bias, it could definitely fall into that.

Security and legal teams are not villains. They're doing their jobs. The problem is timing. They often show up after demos exist, users are excited, and leaders are sold. And then they say, "You can't do this."

Not because it's evil. But because data is flowing incorrectly, permissions are wrong, logging doesn't exist, or vendors aren't approved.

This feels like sabotage. It isn't. It's mis-sequencing.

# Bottleneck #7 - Adoption Friction

I covered adoption in Hard Truth #6 - the core insight is that adoption is change, and change triggers resistance even when the tool is good.

The operational reality - don't start with skeptics. Start with the people who want change and let them be the drivers. Make it exciting. Show real value. Don't load it up with requirements. Just get people using it.

The rest of the organization will follow once they see it working.

# Bottleneck #8 - Accuracy Anxiety

I covered accuracy expectations in detail in Chapter 2 - including the story about the 74% accuracy score that dropped to 50% when a different person graded it. The lesson - human variability is often higher than you think, and "accuracy" is a red herring when there's no baseline.

The operational implication - accuracy anxiety kills velocity. Don't let it stall your project. Human-level accuracy plus machine speed equals leverage.

# Bottleneck #9 - Lack of Operational Muscle Memory

If you believe AI is going to be a critical and core component of your business in the next two to five years, you need to be doing something today. But it's not just using point solutions and Granola notes and Zoom transcripts and Copilot.

You've got to be building AI native in your business processes. Building it native in your applications. Building it native to how you onboard and train employees.

What we did with a lot of our customers is we would encourage some employees to work along with us. That way, you're not just adding AI to your process. You're adding knowledgeable, talented people who have domain expertise and company expertise inside your company and giving them additional training.

This is a great way to upskill your workforce. We're in the buffering stage - just like the internet was 10, 20 years ago. There's a lot of new. A lot of learning. No one has a ton of experience because it's also new.

Put the people who are motivated and innovators, give them the incentive to do this, give them opportunity to learn a new skill, and then your business is not only improved with AI - it has a team of people who are AI literate and can take you to the next step.

> Here's what that looks like when it works. We helped a digital healthcare company build an AI-powered operational stack - RPA, agentic workflows, automated customer processes. In two years, they scaled from a handful of states to 21, achieved 4x growth, and did it all without increasing headcount. That's what happens when you build operational muscle memory and fund iteration properly.

# Bottleneck #10 - Expectation Drift

This one ties everything together.

AI looks magical. So expectations rise. Scope creeps. More stakeholders show up. Accuracy standards float upward. Timelines slip.

Everyone feels disappointed. Not because the project failed. But because nobody froze expectations early.

The biggest and most important thing here is drawing that line in the sand. It doesn't really matter if that line is close to where it ends up being. The important part is if the line is there, you've got something to measure against.

Start with that. Start measuring. You might find you're wildly faster or slower than you need to be, in either the positive or negative way. Then start moving in the direction you want to go.

> **But if you don't draw that line, if you don't have a good North Star, you're going to be treading water and never make the progress you want.**

# What This Chapter Is Really About

Every company has these bottlenecks. Including very good ones.

The difference between teams that win with AI and teams that stall is not intelligence. It's realism.

The winning teams expect these bottlenecks. They name them early. They sequence around them. They fund iteration. They assign real ownership. They start small. They reset expectations deliberately.

None of these bottlenecks are technical. They're organizational habits. And habits can change. Slowly. But reliably.

> **These bottlenecks are predictable. Which means they're avoidable - if you plan for them.**

Now that you know what kills AI projects, let's talk about what a good one actually looks like.

CHAPTER 6

# What a "Good" AI Project Actually Looks Like

Up to this point, this book has mostly been about mental models, patterns, and traps.

This chapter is different.

This is the "okay, but what do I actually do?" chapter.

If you only read one chapter in this entire book and tried to apply it directly, it should probably be this one. Because this chapter is not about theory. It's about what a sane, low-risk, high-probability AI project actually looks like in the real world.

Not the version in vendor decks. Not the version in conference talks. The version that quietly works.

# The Big Idea - Treat AI Like a Product, Not a Feature

This is the single framing that makes everything else make sense.

Most teams treat AI like a feature. "Let's just add AI to this thing." That almost always goes badly.

**The teams that succeed treat AI like a product.** Even if it's a tiny one.

That means it has a clear user, a clear job, a clear owner, a clear workflow, a feedback loop, and a roadmap.

This doesn't have to be heavy. It just has to be intentional.

# Step 1 - Start With One Skinny Use Case

Everything starts here.

Not with a strategy deck. Not with a platform. Not with a roadmap. With one very boring, very specific problem.

You're looking for something with high pain, high frequency, clear ownership, narrow scope, reasonable data quality, and obvious ROI.

### Good examples

- "Classify inbound support tickets."
- "Summarize long documents for execs."
- "Route HR requests."
- "Answer internal policy questions."
- "Draft sales follow-ups."

### Bad examples

- "Transform our entire business with AI."
- "Build a universal AI agent."
- "Automate all of customer support."

You want something you can build in weeks, not months.

This has to be small. Super tiny. "Project" might even be a misnomer - these things could be tasks. Start with a hackathon mentality. Put your team together for a day or two. Have them research what's possible, see what other people are doing. The goal is coming away with a quick win.

It could be something as small as creating an agent that helps someone generate a document or an email template. Something that saves time. That's not really a project. It's a micro-project.

But once you have some awareness and familiarity with that, teams will get really excited. I've seen this happen many times. They'll talk and talk and have a ton of ideas.

The question becomes, how do you vet those ideas and progress them without stifling the innovation?

### Real-World Success Story

We worked with a global manufacturer that had accumulated over 10,000 CAD drawings across years of projects but had no way to search or reuse them. Their 7-person engineering team was constantly recreating parts that already existed. We deployed Vision AI to parse those legacy files, extract parts and dimensions, and build a searchable catalog. Result - 40% faster design cycles and 10,000 drawings converted from dead weight into active assets. That started as one skinny use case - "help engineers find existing parts."

Once they had that searchable catalog working, they kept extending it - adding automated part numbering, custom component matching, and expanding to new product lines. One skinny use case became a proprietary platform that increased their competitive edge. That's what happens when you find a vein and mine it.

# A Quick Warning About AI Councils

I've seen several customers put in AI leadership councils with multiple stakeholders from the business side, the IT side, the executive side. It all sounds good. Make sure you've got people marching in the same direction.

**But here's what actually happened.** We went from doing three and four projects at a time - small dollar projects, $25,000 to $50,000 over four to six weeks - to basically nothing once the AI council was created.

There had to be an approval. There was an intake form. Everyone had to go through Copilot training before they could even receive a license. All these friction points were put in and it dramatically slowed the excitement and output.

The organization was too big. Too much politics. Everyone was looking to mitigate risk, and nobody was looking to maximize opportunity. They believed it would help them move faster and safer. Instead, it made them crawl to a halt.

> **We are not at the point where we should be optimizing for AI. We're in the period where we need to test, test, test and trial, trial, trial. That's how we're going to find success. Not from a perfectly planned project up front.**

# Step 2 - Build a Demo First (On Purpose)

This part surprises people.

**I am pro-demo.**

Demos are not lying. They're just incomplete.

A good demo should answer one question, "Is this idea even worth pursuing?"

At this stage, hard-coding is fine. Prompts can be messy. UX can be ugly. Security can be fake. Data can be partial.

You're not building production yet. You're proving that the workflow makes sense and the outputs are useful.

This stage should be fast. Days or a couple of weeks. Not months.

### Speed Matters

If you're looking at the lowest hanging fruit, you can do that in a day or two. We've done many workshops where we show people live activities, have them go create an agent - this is all possible in a day. And then you can introduce it into your work routine very quickly.

### Vibe Coding

Here's what I actually do. I vibe code it myself. I'm constantly building websites and applications to show what I think the UI and UX should be. I don't even necessarily turn them into fully functioning applications - I leave them in the interactive prototype phase. But it's a highly functional experience that I can hand to a developer and say, "This is exactly what I'm thinking."

### Natural Language Power

I can move the buttons around. I can create new pages. I can do all of that with natural language. Once you know these concepts, you can move at pretty fast speeds. And you become less and less reliant on other developers.

# Step 3 - Turn the Demo Into a POC (But Declare It Disposable)

This is where most teams mess up.

**A POC is not a baby product. It is a proof artifact.**

The job of a POC is to answer slightly harder questions, Will users actually use this? Is the output good enough? Does this save real time? Where does it break?

At this stage, you add logging. You stabilize prompts. You clean inputs a bit. You add minimal UX. You add basic guardrails.

But you explicitly declare, **"This POC will be thrown away."**

That single sentence prevents 50 future problems.

### Critical Declaration

"This POC will be thrown away."

# Step 4 - If the POC Works, Kill It

I covered this in detail in Chapter 4 (Pattern #5 - The Eternal POC). The short version - the moment your POC works, stop. Don't keep adding features under the banner of "we're still in POC mode."

The right move is, **"Great. This worked. Now let's design Phase 2 properly."**

# Step 5 - Reset Expectations and Launch Phase 2

Phase 2 is a new project. New scope. New budget. New timeline. New stakeholders.

This is where you decide. Is this a real internal product now? Who owns it? How reliable does it need to be? How many users will it have? What workflows does it touch?

This is where the 90% starts to show up. And that's okay.

> **For first projects, you really can make good improvements and a full production-ready product if you have buy-in from your IT team, your business, and all your internal stakeholders. You can go from first discovery meeting to something deployed in production in 30 days. That's definitely possible.**

# Step 6 - Build a Deterministic Backbone With AI Inserted

This is the technical heart of good AI systems.

**You do not build "an AI that does everything."**

**You build "a normal workflow with one or two AI steps inside it."**

1. Ingest input
2. Validate input
3. Call AI to classify
4. Apply business rules
5. Route result
6. Log outcome
7. Let a human override

AI is just one step. Not the whole system.

This dramatically improves reliability, debuggability, trust, and maintainability.

### The Math That Most People Don't Think About

Let's say you get pretty good qualitative accuracy - call it 90%. But the tool has to make multiple calls to get your answer, just like thinking mode or reasoning mode in any of the LLMs today.

If there are four tool calls, and you get 90% on average in each of those, you get 0.9 × 0.9 × 0.9 × 0.9. That's 0.66.

You were normally getting a 90% answer with one tool call. But with four tool calls, there's degradation at each step. You can very quickly get down to a coin flip of accuracy.

**Because of that, keep scope super tight. Get the thing working really well, then expand.**

# Step 7 - Target Human-Level Accuracy, Not Perfection

This rule saves enormous time and pain.

**You're not trying to beat your best employee. You're trying to match your average employee.**

In most workflows, human-level accuracy plus machine speed equals massive leverage.

Trying to get to 99.9% too early will stall you forever.

Ship at "good enough." Iterate from there.

# Step 8 - Start With Humans in the Loop, Then Graduate Out

This is not a permanent state. It's a launch strategy.

Early systems should let humans review outputs, override decisions, and correct mistakes. This builds trust with stakeholders. It catches errors before they compound. It creates training data you can use to improve the system.

But the goal is to progressively remove human involvement as confidence grows.

Think of it as a graduated autonomy model. Week one, humans review 100% of outputs. Month two, humans spot-check 20%. Month six, humans only see exceptions and edge cases. Eventually, humans are managing the system, not doing the work.

> **If humans are still reviewing every output six months in, you haven't built an AI system. You've built a suggestion box with extra steps.**

The whole point is leverage. You start with guardrails so you can remove them later.

# Step 9 - Fund Iteration Like a First-Class Activity

This is where most teams quietly fail.

They ship something useful. Then they stop funding it.

**AI systems decay if you don't maintain them.**

Iteration is not optional. It's the product.

Build iteration budget into your planning from day one. Start super small with integrations - only connect the one object you need. Once the integration pipeline is built, adding more is simple. Get confidence, get adoption, then extend.

# Step 10 - Once You Find a Vein, Mine It Aggressively

Once one use case works, adoption is real, trust exists, and ownership is clear - don't tiptoe. **Scale hard.**

Add new workflows that touch the same data. Add new users in adjacent teams. Add new capabilities that extend what you've built. Push the pipeline you've created as far as it can go.

This is how platforms emerge naturally - not from grand architecture, but from one working thing that keeps getting extended because it keeps delivering value.

The caution in this book has been about *starting* - don't start with a platform, don't start with a massive scope, don't start with perfection as the goal.

But once you've proven something works? That's the moment to go after it again and again and again. Every week you don't expand a working system is leverage left on the table.

> **The companies that win aren't the ones who found one vein and admired it. They're the ones who found one vein and mined every ounce of value out of it before moving to the next.**

# The Whole Thing in One Page

If you forget everything else, remember this,

1. **Pick one skinny use case**
2. **Build a fast demo**
3. **Turn it into a disposable POC**
4. **If it works, kill it**
5. **Launch Phase 2 properly**
6. **Build a deterministic workflow**
7. **Target human-level accuracy**
8. **Start with humans in the loop, then graduate out**
9. **Fund iteration**
10. **Once you find a vein, mine it aggressively**

That sequence works. Over and over.

# What This Chapter Is Really About

This chapter isn't here to make AI feel heavy. It's here to make it feel safe.

You do not need a platform. You do not need a strategy deck. You do not need a giant budget. You do not need a massive team. You need one problem, one owner, one small budget, and a willingness to iterate. That's it.

But here's the thing - this playbook only works if you actually run it. The companies that read this chapter and nod along but don't start anything next week are going to be in the same place six months from now. The companies that pick one skinny use case tomorrow and ship a demo by Friday are going to have momentum, learnings, and leverage that compounds.

**The sequence is simple. The execution is what separates winners from watchers. You know what to do now. The question is whether you're going to do it.**

CHAPTER 7

# A Simple AI Maturity Model (That You'll Actually Use)

Most AI maturity models are useless.

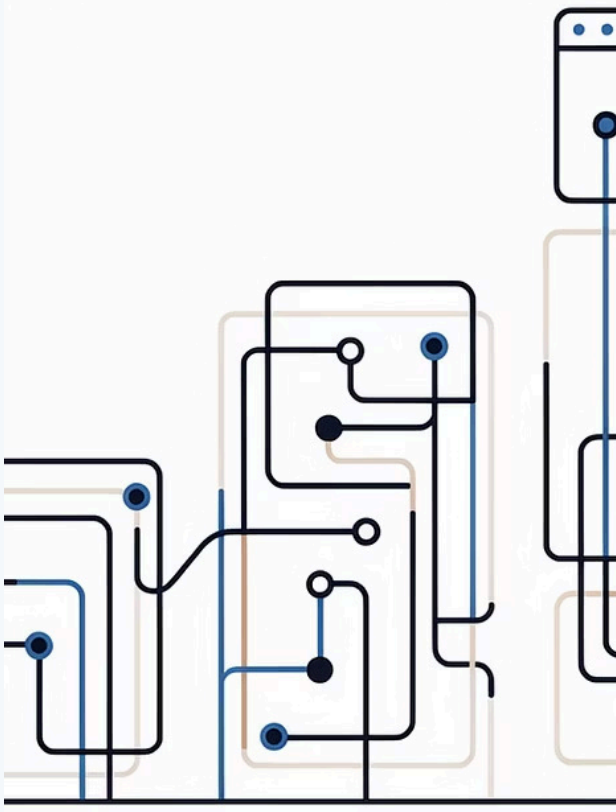
They look impressive. They have lots of boxes. They use fancy language.

And they don't help you decide what to do next.

This chapter exists to fix that.

Not with a framework you hang on a wall. But with a simple mental model you can actually use to diagnose where you really are, decide what to build next, avoid jumping steps, set realistic expectations, and sequence ambition correctly.

Here's what I want you to walk away with - a clear picture of where you are, where the leverage is, and exactly what to do next to capture it.



# The Big Idea - AI Maturity Is About Workflow, Not Intelligence

Most maturity models rank companies by how advanced their models are, how much data they have, or how sophisticated their tooling is.

**That's the wrong axis.**

**The real maturity curve is about how deeply AI is embedded into real workflows that people actually depend on.**

That's it. Everything else is secondary.

Nothing has to be precise here. Nothing has to be exact. You just need a good ballpark of where you are on this spectrum. And honestly? If you're reading this, you're thinking about it, you're using AI - you're already ahead of most.

We talk to a lot of people who are worried they're not doing enough. What I tell them is they're doing the exact right thing right now. You can't go back and change anything. You can only look forward.

But here's what I also tell them - every week you wait to move up this curve, you're leaving leverage on the table. Real leverage. The kind that compounds.

So let's figure out where you are and what's next.

# Level 0 - No AI (Or Quietly Ignoring It)

This is where most companies still are.

They might be curious about AI, talking about AI, sending people to conferences, watching demos on YouTube.

But nothing real exists inside the business yet. No workflows. No tools. No usage.

If we expand this to include automation more broadly, almost no one truly falls into this bucket. There's always some kind of automation - email marketing, workflow triggers inside existing tools. So you're probably not at zero.

But if AI isn't touching any real work yet, you're sitting on a massive opportunity. Not because you're behind. Because the leverage available at the next level is enormous, and it's easier to capture than you think.

# Level 1 - Ad Hoc AI (The ChatGPT Phase)

This is the first real step.

People start using ChatGPT, Claude, Copilot - whatever tool their company hasn't blocked yet.

They paste things in. They get useful outputs. They quietly get more productive.

This is probably where most companies are right now. Using AI for one-off processes - email summarization, helping construct emails, writing documents, content generation, light image generation. Basic one-off use cases. Very much ad hoc. No process around it. No workflow.

This phase is good. It creates awareness, excitement, and internal champions.

It's also chaos. No standard prompts, no shared workflows, no auditability, no consistency. This is shadow IT.

And that's okay. This is how every real adoption curve starts. This is the crawl phase of crawl, walk, run.

### Why This Matters

You need to figure out how AI actually works. How do you write a good prompt? Why does prompt quality matter? If you don't understand these things, you'll struggle at every level above this.

It's all natural language. There's no reason you can't understand it. It's not written in code. If you can understand the prompt and why you're getting the output you're getting, you're going to be in really good shape.

But if you stay at Level 1 too long, you're watching other companies build real leverage while your people are copy-pasting into ChatGPT one conversation at a time. The jump to Level 2 is where things get interesting.

# Level 2 - Internal Copilots and Tools

This is where things start to get real. Instead of everyone doing their own thing, the company builds one or two internal tools with standardized prompts, controlled data, logging, and minimal UX.

**Examples** - internal knowledge bot, support triage tool, document summarizer, sales copilot. These tools are skinny, narrow, owned by one team, and used by real people.

**This is the most important maturity jump. Most ROI lives here. And most companies haven't made it yet.**

The natural evolution is what we call internal copilots. In our workshops, we have people build small agential workflows or single-action agents. That's the next step. What that leads to is chatbots that actually know about your business.

You preload it with your critical processes for your marketing department. Now everyone in marketing has access. You can expand that to other teams. Someone wants to add HR information - policies, vacation schedules, 401k benefits.

But here's my advice. Don't connect them all too much at the beginning. You'll be promising too much to everybody. If you tell people they can get all their HR information and it doesn't have it, who do they go to? Who's running this project? What if changes get made that other departments don't like? That's too complex. Start with these internal copilots. Keep them focused. Put a knowledge base behind them full of documents.

### Start Manual

Don't worry about fancy integrations that automatically sync to root files. There's real critical value in knowledge curation. If you're managing that in your copilot, you're going to get excellent outputs.

### Monitor Usage

You're going to see what people are asking and how they're responding. Are they asking the same questions again and again? Are they looking at source documents? Are they looking for information that isn't there?

### Curate Knowledge

That's the knowledge curation part. It takes work. It takes effort. But that's how you get a good product at the end.

Level 2 is where you start building your company's internal brain. Not a chatbot. Not a wiki. A living system that knows how your company works. Every week you delay this, you're losing knowledge that walks out the door, gets buried in email threads, and never compounds.

# Level 3 - Embedded AI Workflows

This is where AI stops being "a tool" and starts being "how work gets done."

AI steps are embedded into existing software, existing processes, existing workflows.

Tickets are auto-classified before humans see them. Requests are auto-routed. Drafts are auto-generated. Data is auto-enriched.

At this level, people stop thinking about "using AI." They just do their jobs faster. This is where adoption friction disappears.

### Document Creation Example

Think about document creation. Typically you start from a template or copy the last version. You do a find and replace. You change the dates, change the names, update sections. It's haphazard. It's error prone.

What embedded workflows do is formalize that. Instead of chatting with the LLM, you have predefined UI that asks for or pre-populates information at each step. Once you've approved that step, you move on. At each step, the AI passes through everything that's been done before, everything it knows about you, everything it knows about this document and project.

Each step gets better. The final output feels exactly like something you would have done on your own - in about a tenth of the time.

**This is where your processes become software. And once your processes become software, they can scale. They can improve. They can compound.**

# Level 4 - AI-Native Processes

This is where things get interesting.

At this level, you start designing workflows that only make sense because AI exists.

Continuous document triage. Real-time analytics copilots. Dynamic routing systems. Always-on internal brains.

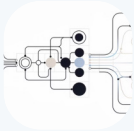
These workflows would be insane without AI. Now they're obvious.

### Continuous Scanning

You can scan data continuously and have AI generate weekly summaries for customers - here's what's happening, here's how users are engaging.

### Automated Briefings

You can create daily briefings for admins to understand what's going on without anyone manually compiling reports.



### Dynamic Systems

A lot of really interesting things become possible when you build AI-native.

This level is rare today. But it's where serious leverage shows up. The companies that get here first are going to operate in ways that feel unfair to everyone else.

# Level 5 - AI as Core Infrastructure

This is the endgame.

AI is no longer "a project." It's just part of the company's operating system.

Like databases. APIs. Identity. Email.

It powers knowledge, operations, sales, support, analytics.

At this point, new workflows assume AI exists. New hires assume AI exists. The company is structurally different.

Almost nobody is here yet. That's fine.

> **But here's what's important. The companies that will be here in three years are the ones making the Level 2 and Level 3 investments right now. They're building internal brains. They're embedding AI into workflows. They're creating the foundation.**

**The gap between companies that start now and companies that wait is going to be enormous. Not because of the technology. Because of the compounding.**

# The Most Important Rule - Don't Skip Levels

This is where teams get into trouble.

They try to jump from Level 1 to Level 4, or Level 2 to Level 5. Because that's what sounds impressive.

**That almost always fails.**

Each level builds muscle memory, trust, data quality, and operational maturity. Skipping levels is how you create brittle systems, political backlash, unrealistic expectations, and burned credibility.

The good news? You don't need to skip levels. The leverage at each level is real. You can capture enormous value just moving from Level 1 to Level 2, or Level 2 to Level 3.

# How to Use This Model in Real Life

This model is here to guide sequencing.

Ask yourself, "What level are we actually at right now? What would one small step up look like? What use case would move us one level forward?"

That's it.

Don't design for Level 5. Design for Level 2 to Level 3. That's where the leverage is. That's where you should be focused.

# A Quiet Truth Most Leaders Miss

Here's something nobody says out loud.

Most companies will get enormous value from Level 2 and Level 3. And never need to go further.

You do not need to become an "AI-native company" to win.

You just need a few great workflows, a few great tools, and a few real wins.

That alone is transformative.

**But you have to start. And you have to start now. Because the leverage compounds. Every month you wait is a month of compounding you don't get back.**

# What This Chapter Is Really About

This chapter isn't here to make you feel behind. You're not behind. You're exactly where most smart companies are.

But I do want you to feel the weight of the opportunity in front of you.

**The internal brain you don't build**

is knowledge that never compounds.

**The workflows you don't embed**

are processes that stay manual.

**The wins you don't stack**

are leverage you never capture.

You're early. You're smart. You're paying attention.

## Now it's time to move.

CHAPTER 8

# How I'd Do This If It Were My Company

This is the chapter I wish someone had handed me three years ago.

Not a strategy deck. Not a vision document. Not a list of tools.

Just a concrete, realistic plan for how to start, what to prioritize, and what to ignore.

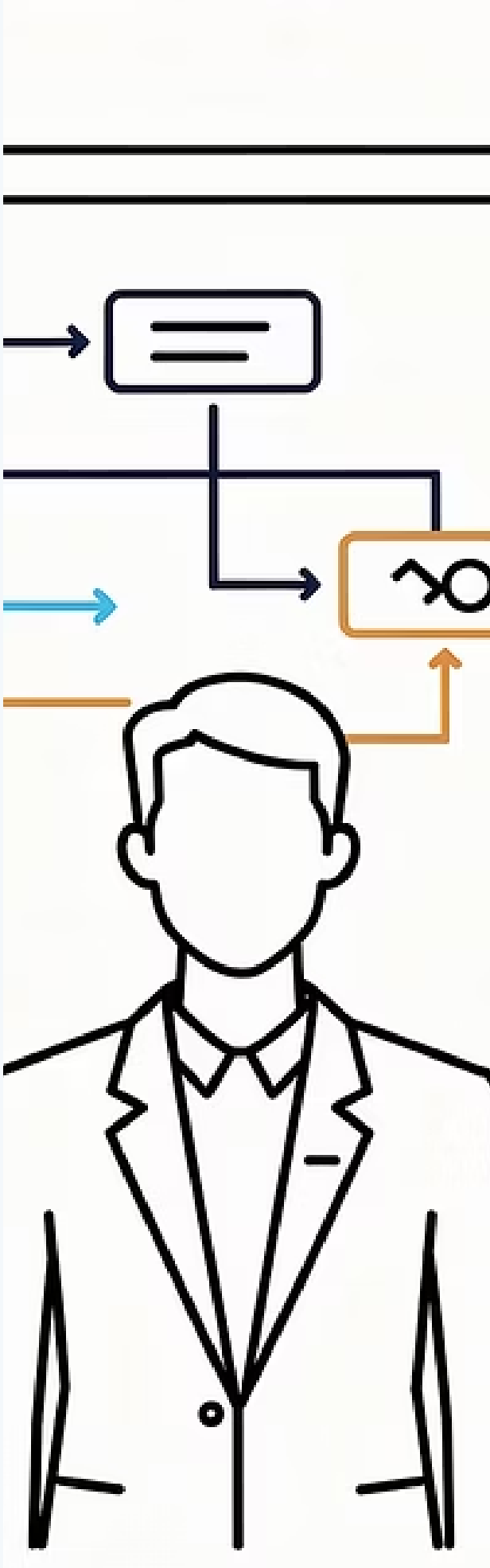
So that's what this is.

If I were dropped into a normal SMB or mid-sized company tomorrow and told, "We want to use AI. But we don't want to light money on fire."

Here's exactly how I would do it.

The good news is I am doing this with my own companies right now. I did this with an AI services consultancy I started about three years ago. And I'm doing it again with a SaaS company I'm building now - essentially an AI brain for small and medium businesses to help them document all the knowledge of their business.

So this isn't theory. This is the playbook.



# The Ground Rules

Before we talk about timelines, there are three rules I would lock in on day one.

### Rule #1

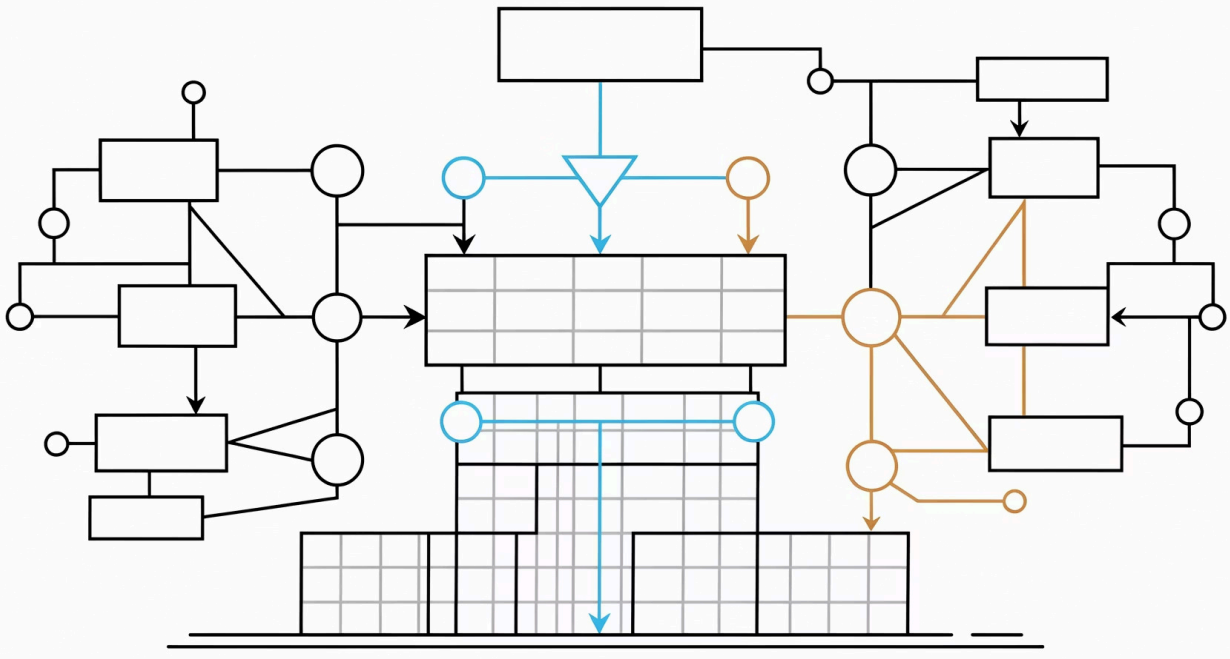
**No platforms. No grand architecture.** We are building one useful thing.

### Rule #2

**No perfection.** We are targeting human-level accuracy and real time savings.

### Rule #3

**Start where the pain is obvious.** Not where the vision is big.



# Month 1 - Find One Real Win

The entire goal of Month 1 is simple. Ship one skinny AI tool that real people actually use.

Nothing else matters yet.

## Step 1 - Find the Right Use Case (Week 1)

I would spend the first week doing one thing - talking to operators.

Not executives. Not innovation teams. The people doing the work.

I would ask, "What is the most annoying, repetitive thing you do every week?" "What do you hate doing?" "What takes forever and feels stupid?" "What do you keep re-answering for people?"

Then I would look for something with high pain, high frequency, narrow scope, clear owner, and reasonable data quality.

And I would pick one.

## Step 2 - Build a Demo Fast (Week 2)

Then I would build the ugliest demo imaginable.

Hard-coded. Messy prompts. No auth. No logging. No UX.

Just enough to answer one question, "Is this idea even useful?"

This should take days. Not weeks.

## Step 3 - Turn the Demo Into a Disposable POC (Week 3-4)

If the demo feels promising, I would add minimal UX, add basic logging, stabilize prompts, and clean inputs slightly.

And I would say out loud to everyone involved, "This is a disposable POC. We will throw this away if it works."

That sentence prevents future pain.

Where you should be at the end of Month 1- a usable product you can start showing people. From a marketing and sales perspective, you want something you can pull up, click around, and give a demo of. Sure, it's going to be demoware. That's okay. At this stage, everyone expects that. But you have to have it.

# Month 2-3 - Turn One Win Into a Real Tool

Now things get real. This is where most teams get impatient. Don't.

## Step 4 - Kill the POC and Reset Expectations

If the POC worked, I would stop. Completely.

And say, "Great. This worked. Now we're starting Phase 2."

This is where the 90% starts to show up.

## Step 5 - Build It Properly (But Still Skinny)

Phase 2 is where I would design the workflow intentionally, add auth and permissions, add proper logging, add error handling, add versioning, and add minimal monitoring.

Still no platform. Still no abstraction layers. Just one good internal product.

## Step 6 - Drive Real Adoption

This part matters more than the tech.

I would train users live, sit with them while they use it, watch where they get confused, fix the UX fast, and collect feedback weekly.

And I would make sure a real manager cares, a real executive endorses it, and a real team depends on it.

By month three, you should have an MVP that you can get your first one or two customers on. These are your design partners - people who are very aware this is early stage development. They know there's going to be things that don't work. There's going to be hand-holding and a lot of feedback. But if you find the right partners, you can build something pretty special.

Because you can vibe code, because you can churn out a lot more code, you don't want to build in a vacuum. You could end up in the build trap - producing a bunch of things very quickly that you think are awesome that nobody in the market wants. And now you've got this big monstrosity going 18 different directions that's tough for anyone to understand.

**Focus on that small wedge. That niche. That one thing.**

# Month 4-6 - Expand and Mine the Vein

Only now do you earn the right to expand.

## Step 7 - Add One More Use Case

Not five. One.

Preferably adjacent to the first.

Reuse the same patterns, the same infrastructure, the same ownership model.

This is how platforms quietly emerge.

Your subsequent ideas should happen much faster because you've already built the infrastructure. You've already got the pipeline. You've already got people doing different things.

Think about agents. You could build agents immediately. That's something you can get out there and do right away. What I would probably do is have somebody who built one of these go talk to other departments and show them how they could build agents too. Have some fun internal workshops. It's basically free - just takes the time of the person who knows how to do this. They probably love it anyway. Have them help set up agents, take notes on the processes being replaced, and create mini case studies.

## Step 8 - Start Building Muscle Memory

At this point, I would start formalizing a release cadence, a bug triage process, a feedback loop, and a tiny roadmap.

Nothing heavy. Just enough to make this feel like a real product.

By month six, you should have multiple AI applications running in multiple ways. There should be some workflow automation. There should be some chatbot or copilot products. There should be data pipelining and movement. There should be some analysis.

All lightweight. If you're trying to build this too big, it's going to take forever. But if you build each of those little components, you can move very quickly.

# What This Actually Looks Like When It Works

Here's what actually happens when this works.

After the first win, people start bringing you ideas. Leaders start paying attention. Budget gets easier. Momentum builds.

AI stops being scary. It just becomes useful.

And here's the thing that should light a fire under you. I've seen projects where an activity that took three or four people six to eight weeks to compile every single year - a federal reporting requirement - got automated using AI. The interpretation, the steps to approve, the classification. All of it.

> **It took less than five hours. For a total cost of less than $500.**

Those things can happen. Those things will happen. That's the leverage sitting in your business right now.

# What I Would Explicitly Not Do

This is just as important.

### I Would Not Buy 10 Tools

One model API. One vector store (maybe). One workflow engine.

That's it.

Tool sprawl kills velocity.

Don't buy 50 licenses. Buy one or two at a time and let people test and trial. Maybe there's a way you can take content and have an agent built inside a copilot or ChatGPT. Have the content turn into a slide deck using a tool like Gamma. That can be done in seconds, look clean, have all your branding, and save 30 minutes of somebody's day every single day. That's a huge time savings. And Gamma doesn't cost that much.

You're not looking for the right answer. You're looking for any right answer. Sometimes that includes using small, cheap tools.

### I Would Not Hire a Big AI Team

You do not need one.

One product-minded engineer and one business owner beats five AI researchers.

You've got to find someone who loves this stuff, who's playing with it all the time, who's already built a bunch of things on their own, who already knows how to vibe code and has had a lot of success with it. Those are the people you want. Let them run. Give them appropriate constraints, environments, tools, and access to other teams. Let them build things and prove them out. Then require them to report on their successes.

# What I Would Explicitly Not Do (Continued)

### I Would Not Build an AI Council

Covered this in Chapter 6. Councils optimize for risk mitigation when you should be optimizing for learning. Test, test, test. Trial, trial, trial.

### I Would Not Write an AI Strategy Deck

Strategy emerges from wins. Not PowerPoint.

### I Would Not Promise Anything Big

No big timelines. No big claims.

Only, "We're running small experiments and scaling what works."

### I Would Not Invest in a Big, Long Project

I wouldn't want to spend a couple hundred grand on something that's promising to unify all my company's data. That's way too big. If it sounds too good to be true, it probably is.

Start small. Integrate one system at a time. Create a chatbot that helps sales analytics. Then move into financial analysis. Then document writing for marketing. You don't start with the unified data platform - that's the thing that never gets done.

# What This Chapter Is Really About

This chapter isn't here to make this feel complicated. It's here to make it feel urgent.

You do not need millions of dollars, a massive team, or a moonshot vision.

**You need one problem, one owner, one small budget, and a willingness to iterate.**

That's it.

But you have to start. And you have to start now.

The federal reporting example - six to eight weeks down to five hours - that's not unusual. That's the kind of leverage sitting untouched in almost every business I've seen. Manual processes eating weeks of time. Knowledge trapped in people's heads. Workflows that exist because "that's how we've always done it."

The companies that move now are going to look back in two years and realize they built something that compounds. The companies that wait are going to be playing catch-up.

**You're smart. You're early. You know what to do.**

**Now go do it.**

CHAPTER 9

# The Leverage Doctrine

This chapter exists for one reason.

Everything you've read so far was governed by a small set of principles.

They weren't presented as rules. They weren't framed as doctrine. They were embedded into the stories, the scars, the patterns, and the advice.

But they were always there.

This chapter names them.

Not as best practices. Not as tips. Not as AI guidance.

But as **the operating system I now believe modern businesses must adopt if they want to build durable internal leverage in the AI era.**

These are not theoretical ideas. They are the principles that quietly governed every project that worked, and every project that failed, over the last three years.

If you internalize these seven ideas, you will see AI - and your business - very differently from this point forward.

# Principle 1 - Knowledge Is Infrastructure

Knowledge is not exhaust.

It's not documentation. It's not a byproduct of work. It's not something you deal with "when you have time."

**It is core operating infrastructure.**

Every modern business already runs on knowledge - how things are done, why decisions were made, what rules actually matter, what exceptions exist, what customers actually need, what the business has already learned the hard way.

Most of this knowledge lives in people's heads, in Slack threads, in emails, in decks, in meeting notes, in tribal memory.

Which means it's fragile, inconsistent, impossible to scale, and lost every time someone leaves.

**AI changes that permanently.**

It's now economically possible to continuously capture knowledge as work happens, distill it into usable artifacts, organize it, reason over it, and turn it into a real system.

This is the deepest idea in this entire book.

Once you have a corpus about your business built, you can really interact with LLMs in a powerful way. Whether it's through fine-tuning or RAG and prompt combinations, once you've dialed in the LLM to know about you and your business specifically, and you can spread that knowledge across your entire company - everybody has access to the same repository and knowledge store.

That's super powerful. Now when you're working with other applications or LLMs, you're not starting from scratch. You're plugging your brain in, and it immediately has complete content and context of what your business is doing, what you've done historically, and what you want to do going forward.

> **Implication** - The companies that win in the next decade will not be the ones with better tools. They will be the ones that deliberately build and maintain better internal knowledge systems.

# Principle 2 - AI Is a Workflow, Not a Feature

AI only creates leverage when it's embedded into real workflows.

Good AI systems are not magical brains. They are deterministic systems with one or two AI steps inserted, with humans in the loop, with logging, overrides, and guardrails.

**AI should almost never be the entire system. It should be one step inside a normal, boring, reliable workflow.**

When teams treat AI like a feature - "Let's just add AI to this thing" - they get chaos. Inconsistent outputs. Broken trust. Impossible debugging. Unclear ownership.

When teams treat AI like a workflow - "Let's design a normal system and insert AI where it creates leverage" - they get reliability.

This is the single most important technical worldview shift in the book.

AI is not magic, but it is magical. It's 90% normal software development, 10% AI - that magical part. All those other normal things don't go away. You need people who know how to work inside of that world. But then you also need people who know how to use AI to shortcut all those things so you can focus on that central AI component.

**Implication** - If your AI idea cannot be drawn as a deterministic flowchart with one or two AI steps inside it, you're not ready to build it yet.

# Principle 3 - Start Skinny or Don't Start at All

Leverage compounds. Big visions kill momentum.

Every successful AI project I've seen started as one narrow use case, one workflow, one owner, one measurable win.

Every failed one started as, "Let's transform the business with AI." "Let's build a platform." "Let's automate everything."

Ambition is not the problem. Sequencing is.

You do not earn the right to build big things. You earn it by shipping small wins that compound into trust, data, and momentum.

**Implication** - If you cannot build and ship your first AI use case in weeks, not months, your scope is wrong.

# Principle 4 - Adoption Beats Intelligence

A mediocre model inside a real workflow beats a brilliant model nobody uses.

The real work in AI projects is not intelligence. It's UX, trust, ownership, training, workflow integration, and habit formation.

The best model in the world is worthless if nobody changes how they work because of it.

**Demos lie. Adoption tells the truth.**

If it doesn't save time in one week, it's dead. Move on. Try something else.

And we shouldn't be looking for 10% gains or 5% gains. We should be looking for 2x, 3x, 5x gains. I've seen this happen many times. That's the leverage waiting in your business right now.

**Implication** - You should be optimizing first for usage, not accuracy.

# Principle 5 - Human-Level Accuracy Is Enough

Velocity beats perfection.

In most real workflows, human-level accuracy plus machine speed equals massive leverage.

Trying to get to 99.9% accuracy too early kills momentum, ROI, and trust. Because edge cases explode, scope creeps, timelines slip, and expectations drift.

Perfection is a Phase 3 problem. Shipping "good enough" is a Phase 1 requirement.

I use an analogy from Harry Potter. A lot of people want magic, and with AI, we've kind of given them magic. But then I get frustrated when people just want the magic to work without really trying. They want it to read their mind.

In the world of Harry Potter, they have legit real magic. But the only way you can do magic is you've got to go to Hogwarts. You've got to spend your entire adolescence learning how to say the spells, recite the spells, use your magic wand, use the right motions. Learn all these different aspects of magic. If you don't know that, the magic is not going to work.

So in a dream world like Harry Potter, where people have been given the gift of magic, even those people have to work for it. Why would we expect to not have to work for it?

In the very real world of AI, you've got to learn how to prompt. You've got to know how to interface with the different LLMs. You've got to know how to use it as a thought partner. It's not one and done. You can't just ask one vague question and expect to get a very specific answer.

This is just like working with a human. If you can't teach a human how to do something, you're not going to be able to teach AI how to do something.

> **Implication** - If your AI system is as accurate as your average employee and 10x faster, you already won.

# Principle 6 - Internal Brains Beat Tools

Buying tools gives you features. Building internal brains gives you leverage.

Your real moat is not your model, your vendor, or your software stack.

**It's your knowledge, your workflows, your context, your decisions, and your patterns.**

An internal brain is not a chatbot. It's a living system that knows how your company works, knows what rules actually matter, knows what decisions have already been made, and knows how things are supposed to be done.

Tools come and go. Models change.

**Your internal brain compounds.**

You need people who love and lean into AI at every step of the process. Some of the people on our team are constantly trying every move they can because they want to see if they can get a little bit of lift. One of our developers wrote 300,000 lines of code overnight using AI coding tools. Obviously hardly any of those lines went unchanged, but through constructing a very defined set of rules and expectations, once you put the edges around it and define exactly how you want things to work, you can move at incredible speed.

AI-native thinkers are going to be critical to moving your business in the future.

You need to be using all these tools yourself. I love Excel. I think Excel is the best business tool ever created. But I don't want to miss out and not know how to use Cursor. I don't want to not know how to vibe code. You need to know how to do this yourself, independent of what your role is.

If you don't know that, you're going to miss the AI wave. Five years from now, this is going to be a natural skill set that everyone has. And if you don't have it, you're going to find yourself severely behind and not able to understand how these parts and pieces connect together.

> **Implication** - If your AI strategy depends on a single vendor or product, you do not have a strategy.

# Principle 7 - Small Wins Compound Into Strategy

Strategy is not designed. It emerges from shipped leverage.

One skinny win → one trusted workflow → one internal brain → one platform.

Not the other way around.

Most teams try to design a strategy first and then execute it. The teams that win do the opposite. They ship leverage, then name the strategy that emerged.

You have to have a product owner. You've got to have somebody who's going to watch and babysit this thing. Someone who's defining the roadmap, someone who's driving adoption, someone who's interfacing with your internal customers.

If that role does not exist, your software is not going to work. But if you have that ownership, if you keep shipping, if you keep improving - strategy emerges. You look up one day and realize you've built something that compounds.

**Implication** - If your AI strategy exists primarily in a PowerPoint deck, you are not ready to execute yet.

# What This Chapter Is Really Saying

These seven principles are not AI advice.

**They are leverage doctrine.**

They are the rules of the game for building modern businesses in the AI era.

If you adopt them, you will build internal leverage faster, avoid most AI traps, compound knowledge instead of losing it, turn workflows into assets, and turn experience into infrastructure.

If you ignore them, you will chase tools, burn money, build fragile systems, stall politically, and move slower than you should.

The gap between companies that adopt these principles and companies that don't is going to widen quickly.

# The Quiet Realization

If you zoom out, something uncomfortable becomes obvious.

None of these principles are actually about AI.

They are about how businesses should be built, how knowledge should be treated, and how leverage should be compounded.

AI just happens to be the missing tool that finally makes this operating model practical.

The question isn't whether you believe in AI.

The question is whether you're going to capture the leverage while it's still available - while you can still build an internal brain before your competitors do, while you can still compound knowledge before it walks out the door, while you can still turn your workflows into assets that appreciate over time.

The companies that internalize these seven principles are going to operate in ways that feel unfair to everyone else.

That's the leverage doctrine.

**Now the only question is, are you going to adopt it?**

CHAPTER 10

# What This Means for the Next 3-5 Years

Up to this point, this book has mostly been about the present.

What works. What fails. How to start. How to build safely. How to avoid lighting money on fire.

This final chapter is about zooming out.

Not into sci-fi. Not into hype.

But into the very real structural changes that are already underway.

Because once you internalize everything in this book, one thing becomes obvious.

**AI is not just a new tool category. It's a new layer of infrastructure for how work happens.**

And that has consequences. Big ones.

Peter Diamandis has a quote that says there are two types of businesses that will exist in the next five years - those who fully embrace AI and those who are out of business. I used to think this was a bit dramatic. But now that I've seen it - seen which businesses are using AI and how they're using it, seen how quickly they're evolving versus those who aren't - I believe this is directionally very true.

The key is to get started. You have to start moving this way. And that involves getting started now.

# The First Big Shift

## Knowledge Creation Is Becoming Free

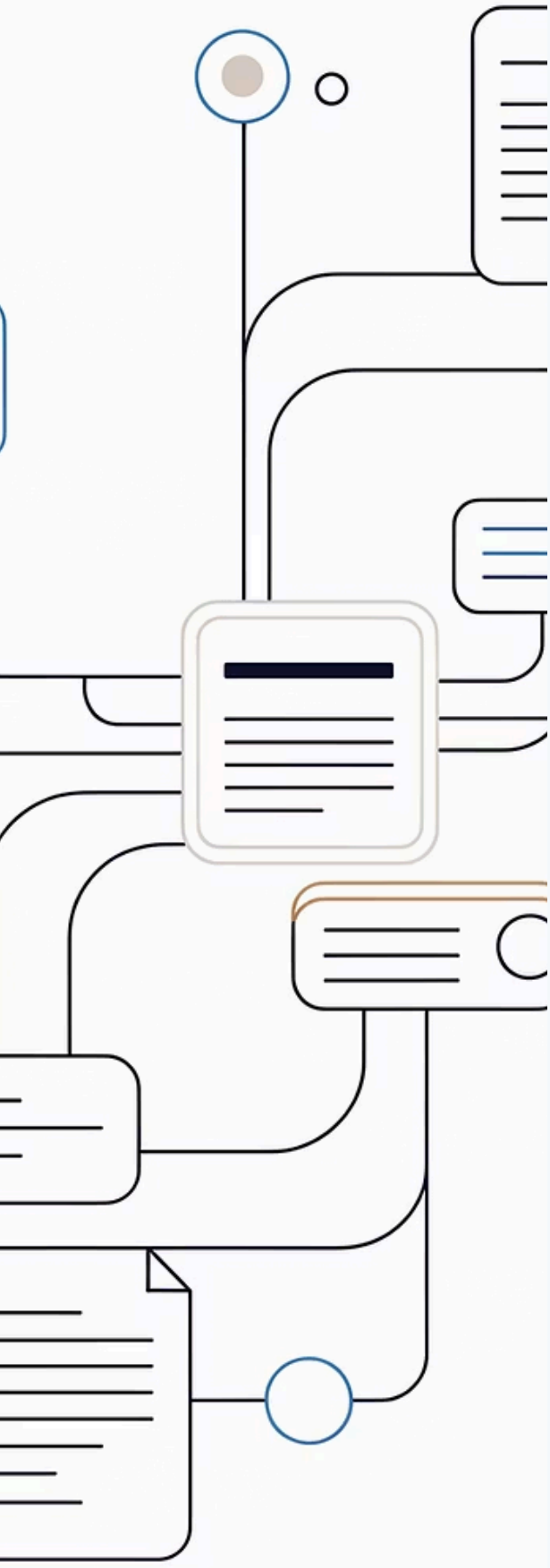
For most of history, knowledge was expensive to create. Writing, summarizing, documenting - it all took time. So most knowledge stayed in people's heads.

**AI changes that permanently. At near-zero marginal cost.**

The bottleneck is no longer "Can we afford to create knowledge artifacts?"

It becomes "Can we curate, organize, and trust the knowledge we're creating?"

That's a completely different problem. And a much better one to have.



# The Second Big Shift

## Every Company Will Build an Internal Brain

This is the inevitable conclusion of everything you've read so far.

Not a chatbot. Not a wiki. Not a search engine.

**A living system that knows how your company works, knows what your policies are, knows what decisions you've made, and knows how things are supposed to be done.**

This brain becomes the first place people go for answers, the first place new hires go to learn, the connective tissue between teams.

This is not futuristic. This is already happening quietly in good organizations.

I do this myself. I sit around on Saturdays and Sundays watching football, and I chat with my business. I've built my own internal brain about our company, and I'm able to ask a wide variety of questions. Instead of bothering employees on the weekend, I'm chatting with the AI to help me understand things and think through things.

**If you have the brain, it opens up so many doors. The brain is going to be more important than the tools.**

# The Third Big Shift

## Knowledge Will Become a First-Class Asset

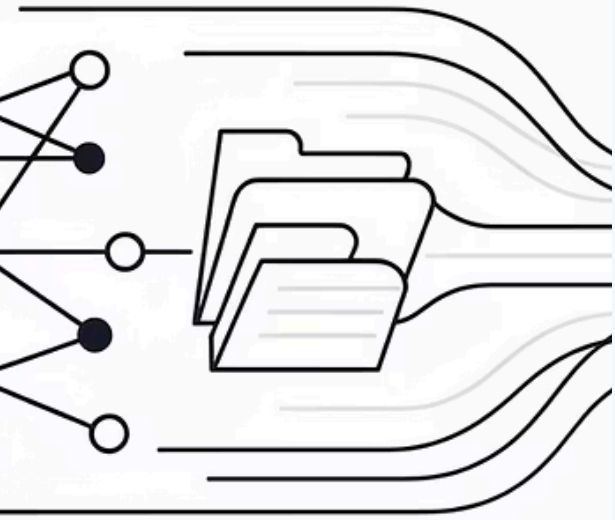
Right now, most companies treat knowledge like exhaust. Something that happens incidentally as people do work.

That will flip.

When your internal brain becomes core infrastructure, suddenly knowledge quality matters, knowledge freshness matters, and knowledge ownership matters.

Companies will start asking, "Who owns this knowledge?" "Is this up to date?" "Can we trust this answer?" "What decisions is this powering?"

Knowledge will become something you deliberately manage. Like data. Like code. Like infrastructure.



# The Fourth Big Shift

## Everyone Gets a Digital Clone of the Business

This one sounds wild, but it's very real.

As internal brains get better, every employee will effectively have a personalized digital interface to the company's collective intelligence.

They'll be able to ask, "How do I do this?" "What's the right answer here?" "What's the latest policy?" "Who should I talk to?" "What's the context behind this decision?"

And get answers instantly.

That changes training, onboarding, decision-making, and autonomy.

It collapses the distance between "knowing who to ask" and "just knowing."

# The Fifth Big Shift

## The Real Divide Won't Be AI vs Non-AI Companies

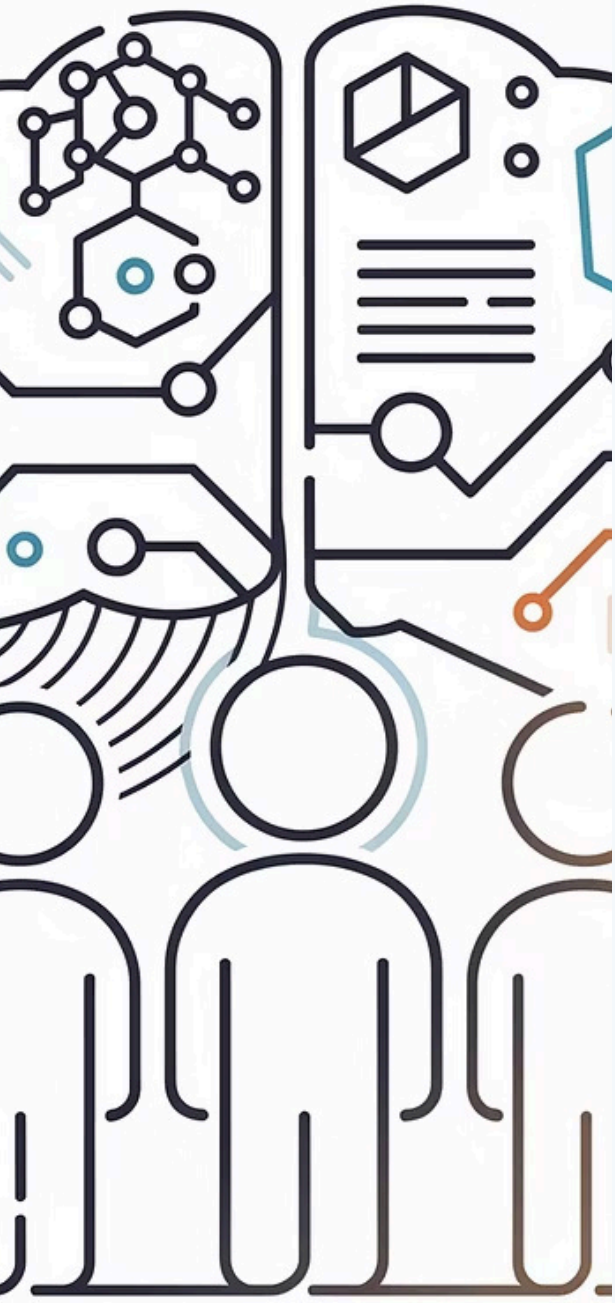
Everyone will "use AI."

That's not the divide.

**The real divide will be between companies that built internal brains early, captured their knowledge, and embedded AI into workflows - and companies that didn't.**

The gap will show up as speed, consistency, decision quality, onboarding time, and operational leverage.

It will feel unfair. Because it will be.



# The Sixth Big Shift

## Small Teams Will Become Unfairly Powerful

This is the part most people underestimate.

When AI is embedded properly, one person can do the work of five. A tiny team can run operations that used to require departments. Founders can stay lean much longer. SMBs can punch far above their weight.

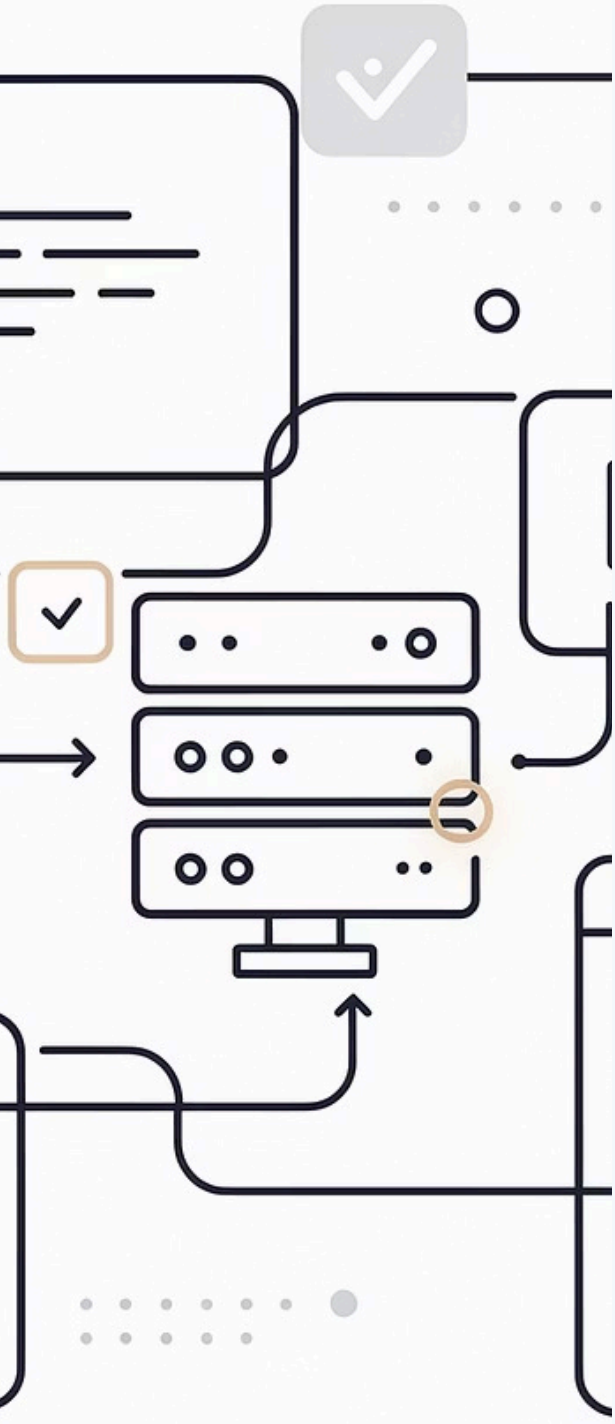
Not because AI replaces humans. But because it removes the worst parts of work and amplifies the best parts.

I think we're going to have a ton of small companies rise up and provide great livings and opportunities for people who would never have had that before. They can now get expertise from lawyers, accountants, HR, strategic consultants - all in the form of an LLM.

There's talk of the one-person unicorn. The billion-dollar valuation built by a single founder. Are we there yet? No. Are we close? I'm sure there are several being worked on right now, and I think we'll see one very soon.

Micro SaaS is going to be huge. Really small products that charge from free to $2.99 to $9.99 to $12.99. Products that only do one thing, do it really well, and are probably built by a one-person startup.

**This is going to reshape competition at the low and mid-market in a huge way.**



# The Seventh Big Shift

## "Software" and "Operations" Will Blur

This one is subtle but profound.

As AI gets embedded into workflows, the line between a business process and a software system will disappear.

**Processes will become software. Software will become process.**

I call this "business as code." The more codified your business is, the more easily you'll be able to update it. Before, if you wanted a process change, you probably had to have a staff meeting, explain to everybody why it's going to change, put together some PowerPoint decks, walk through the old process and the new process and how you're going to measure it. Really hard. Really time consuming. Probably never happens.

But if your business is codified - if people are plugging into the step of the process that the code is presenting to them - then that change happens immediately. That gives real control and power to the executive team to command and control the business.

Static playbooks are dead. We need to codify all the playbooks. Once they're codified, we can have AI help evaluate them at every step and tell us when they need to change. If it's codified, we can handle everything upstream and downstream, just like we do with code review and a code push.

That changes how companies think about ops, IT, product, data, and knowledge. They all collapse into one thing - the company's operating system.

# What This Means Practically

The companies that win in the next 3-5 years will not be the ones with the fanciest models, the biggest AI budgets, or the flashiest demos.

**They'll be the ones who started early, built internal brains, embedded AI into real workflows, and treated AI like infrastructure.**

Agents are going to be everywhere. Not just thousands - millions, if not billions. So many that we don't even think about them. A lot of them will be so minor and task-oriented that you won't even know how many agents are being called in a particular activity. One prompt might call out to a thousand different processes.

We're going to buy agents just like we buy humans. Instead of creating a full-time employment engagement, we'll have arrangements where we're buying an agent for a period of time or working on a consumption model. Not replacing humans in whole - humans are too complex to replace right now. But replacing a lot of tasks.

Take all your humans and write all the tasks down. Then start applying those tasks to agents. One at a time. The next thing you know, you haven't replaced a human in whole, but you've replaced the parts and pieces many times over.

# The Quiet Irony

Here's the ironic part.

The future I just described does not require huge teams, massive budgets, moonshot projects, or revolutionary breakthroughs.

It requires starting small, shipping boring wins, iterating quietly, and building internal leverage over time.

The exact opposite of what most people think AI transformation looks like.

# The Real Point of This Book

This book was never about AI.

Not really.

**It's about leverage.**

It's about turning knowledge into infrastructure, turning workflows into software, and turning experience into assets.

AI just happens to be the tool that finally makes this practical.

# The Encouraging Truth

**You are not late. You are not early. You are exactly on time.**

The winners are not being decided this year. They'll be decided slowly, quietly, over the next few years.

By teams that start small, stay consistent, build internal leverage, and keep compounding wins.

That game is still wide open.

There's going to be a talent war for AI expertise - companies are paying crazy sums of money, like athlete money, to some of these people. Most companies can't afford that. So what you need to do is upskill your team. You've probably got a great team. Make sure they continue to be great and stay up on the curve.

There's also huge opportunity with younger, less experienced people who are more AI-native thinkers. They're coming out of college into a tough job market because larger companies believe they don't need as many junior developers. That might be true, but there's a huge amount of talent that could be plugged into your organization at really effective price points.

I look at this like the introduction of the PC or the introduction of the Internet - like the geek squad that used to exist, where young kids who were native to computers and the Internet would come help businesses get established. They created websites, created portals, trained people on tools. We're in that kind of state right now.

# A Final Thought

If you take nothing else from this book, take this.

> **You do not need to build an AI company. You need to build a company that uses AI well.**

Those are very different things.

One is hype.

The other is durable advantage.

The companies that move now - that start capturing knowledge, building internal brains, embedding AI into workflows - are going to look back in three years and realize they built something that compounds. Something that gives them speed, consistency, and leverage their competitors can't match.

The companies that wait are going to wonder why everything feels so hard.

You've read this book. You know what to do.

## Now go build your brain.