Introduction

In the fall of 1979, I sat at a terminal connected by phone line to a mainframe I couldn't see, keying in real estate data one number at a time, waiting to find out where a restaurant chain should build next. There was no software for this in any sense you'd recognize today — just a rented database, a stack of variables I'd talked my way into adding, and hours of typing that felt more like data entry than analysis. When I finally had enough loaded to run the model, I typed the command and watched dollar signs scroll down the screen for what felt like half an hour — the machine's way of telling me it was working, and billing my company for every second of it. A week later my boss called me into his office with a printout and a number on it: fourteen hundred dollars, for one model run. That machine could answer a question. It had no opinion whatsoever about which question was worth asking. Nobody in that building mistook the two.

I've spent the forty-some years since watching that distinction slowly come apart.

This book makes one argument, and it makes it from as many angles as a forty-year career in data, analytics, and technology consulting can supply: the technology built to help leaders make decisions has gotten faster, cheaper, and dramatically more fluent, generation after generation, while the actual decision — what question matters, what a number means, what to do about it — has never once moved from the human being in the room. Every era I've worked through mistook a better version of the asking for progress on the deciding. Mainframe time-sharing did it. Data warehousing did it. Predictive analytics did it. Big data did it. Cloud infrastructure did it. And generative AI, the most fluent and convincing

version of this promise the industry has ever produced, is the version most likely to make us forget the difference entirely — because for the first time, the tool doesn't just answer. It sounds like it understands, and it sounds that way whether or not anyone bothered to check.

I am not a technologist by training, in fact I have a B.A. in geography and an M.A. in Planning, so this isn't a book about how large language models work. I'm a practitioner who happened to be in the room for a lot of this history: at Metaphor Computer Systems in the 1980s, at IBM during the transition into predictive analytics, at Teradata and Think Big Analytics through the data warehousing and big data eras, at AWS watching the cloud's partner ecosystem sell objectivity at a scale no vendor had ever attempted before. I've spent the year since building a consulting practice built around a methodology I call Technology Cartography, which exists for one reason: to show a client, visually, where their technology sprawl came from — and it almost never comes from the technology. It comes from a question nobody stopped to ask before the purchase order went through.

Every organization adopting generative AI right now is, whether it realizes it or not, re-running an argument this industry has had with itself every decade since the 70's. This book exists to give you the vantage point to see the pattern while you're still inside it, rather than after, the way most of us saw it the first five times.

What follows is a hybrid: part memoir, part argument. The first seven chapters trace my career chronologically, from that mainframe terminal through Metaphor, IBM, Teradata, AWS, and into my present work, and each one pairs a specific, lived moment with a through-line to today's AI version of the same pattern. Chapter Seven is the hinge,

where the historical evidence finally gets a name. The final three chapters turn from history to the live question in front of every leader right now: why generative AI's fluency makes it the most persuasive version of the old promise yet, what human intuition still does that no model has managed to replicate, and a practical framework for knowing, decision by decision, where to lean on the machine and where to protect the judgment that was never the machine's to make in the first place.

I still think about that terminal sometimes — the slow scroll of dollar signs, the fourteen-hundred-dollar bill, the total, honest silence of a machine that had no opinion about anything beyond the number it had been asked to produce. Every tool I've used since has gotten faster, cheaper, and immeasurably more convincing. None of them, including the one you're probably using this week, has yet managed to tell me which question was worth fourteen hundred dollars to ask. That's still my job. It's still yours too, whether your organization has told you so or not, and this book is about learning to notice the moment you're tempted to forget it.

Chapter One
The Name Was Right

In the fall of 1979 I was a graduate student with a part-time job in the real estate department of Collins Foods International, the parent company at the time of Sizzler Restaurants, and my assignment was to build a model that would tell the company where to put its next restaurant. Nobody called it that, of course — nobody called it a model, or an algorithm, or anything with that much dignity. What I had was a stack of variables somebody in real estate believed mattered — site access, traffic counts, the demographics of the surrounding neighborhood, a handful of others I'd talked my way into adding — and a terminal connected, by phone line, to a time-sharing system called NCSS, a Dun & Bradstreet service that rented computing power by the minute to companies too small or too cautious to buy their own mainframe.

NCSS ran a database called Nomad, amongst many other applications, which is where I loaded everything I had for the sites under consideration across the state of Utah. I remember the loading more than I remember the thinking — hours of keying in numbers, checking them, keying in more. And I remember the night I finally had enough data in the system to let the model run. I typed the command, sat back, and watched dollar signs begin to fill the screen. Not one or two — a column of them, scrolling for what felt like half an hour, the terminal's way of telling me that somewhere on the other end of that phone line, a machine I couldn't see was working very hard on my behalf, and billing my company for every second of it.

The model finished. It gave an answer. Everyone in the real estate department was satisfied, and I went back to the rest of my coursework and forgot about it.

About a week later my boss called me into his office. He had a printout from NCSS in his hand, and on it was a number: my one model run had cost the company fourteen hundred dollars. There was a unit of measurement on the bill called an ARU — an Application Resource Unit — which was NCSS's own private currency for how hard you'd made their machine work, and my ARU count, he told me, was off the charts. He didn't fire me. The model had done its job and done it well enough that nobody wanted to undo the finding. But he did ask me, gently, to go back through my code and see if there wasn't a way to get the same answer for less money.

I mention this not because it's a dramatic story — it isn't — but because it was my first real encounter with the two things that would define the next forty years of my career, long before I had any language for either one. The first was the machine itself: patient, powerful within its narrow lane, and completely indifferent to what any of its numbers meant. The second was the constraint wrapped around it — cost, capacity, the sheer physical expense of asking a computer to think about something — which in 1979 was so severe that the constraint was, for all practical purposes, the whole story. Nobody worried yet about whether the machine could be mistaken for a decision-maker. It was far too busy, and far too expensive, to be mistaken for anything yet major decisions within the department were made on the results. It supported decisions that had huge financial repercussions yet when I talked to one of the real estate reps when he was at the home office, he applauded my work but said "my choices for sites was simple -- find the nearest

McDonalds and locate within a short drive of it". Site model be damned. Decisions were gut level.

Seven years later, in the fall of 1986, I sat down at a desk at Metaphor Computer Systems that still had a coffee ring on it from the person who'd sat there before me. That person was Ralph Kimball — the same Ralph Kimball who would go on to more or less invent the star schema and reorganize how an entire industry thinks about structuring data for analysis. I didn't know any of that yet. Nobody did, including him, probably. He was just the person who'd left, opening a slot that I filled, at a small company making some of the industry's first real bets in a field that didn't yet have a settled name for itself. It's a strange thing, looking back across four decades, to realize how contingent it all was — how many of the people who'd go on to be called founders of an industry were, in the moment, simply the ones standing in a particular room, guessing well.

Metaphor itself was built on a specific conviction, one I'd come to understand more fully as the years went on and I watched it collide with an opposing conviction held just as strongly by other people, in other rooms, at other companies. David Liddle and Donald Massaro, Metaphor's founders, believed the right place for data was directly in the hands of the person making the decision — a desktop, literally, with the numbers on it, no analyst or IT department standing between the question and the answer. You built an application that did your analysis graphically, just as you would if you had all the components at your beck and call but this time, it was made to operate as you would think. It was a good instinct, and an early one; the industry's other founding bet, the one usually associated with Bill Inmon, would pull in the opposite direction, toward one clean, centrally modeled

version of the truth. That argument — the desktop versus the cathedral — is really the subject of the next chapter. What matters here is only that in 1986, both bets were still young enough to feel like common sense rather than a fight.

Metaphor in those days was a small company in the way that only a company inventing something genuinely new can be small — a handful of engineers and analysts who believed, with more conviction than evidence, that the future of business computing belonged on an ordinary person's desk rather than locked inside a data-processing department three floors away. Nobody used the phrase 'user experience' yet, but that was, in effect, what we were arguing for: that a category manager shouldn't need an intermediary to ask a question of his/her own store's numbers. It's easy, four decades later, to underestimate how radical that idea was. In 1986, most of corporate America still routed its questions about its own data through a request form and a queue.

The specific work that fall put me inside Vons Grocery's finance department, where the system Metaphor had installed had been running for months as little more than a weekly financial report generator — what everyone in the building called the green sheet, delivered every Monday morning like a newspaper nobody expected to talk back. The green sheet was static by design: category sales by store, this week against last week, and nothing else. There was no way to drill down into a specific SKU or brand, no way to change the time period, no way to swap in a different set of stores. If a category manager wanted to look past what the green sheet showed him, his only option was to pull the underlying numbers by hand and rebuild the analysis himself, in a spreadsheet, one row at a time. The green sheet was accurate. It was on time. And it used perhaps a tenth of what the system

underneath it was capable of. Nobody in finance had much reason to notice the gap, because the green sheet report did exactly what finance had asked it to do.

Underneath the green sheet, though, sat a genuine capability, built the same way everything was built in that era: one category at a time, because that was what the economics of the day allowed. Sales by store, this week against last week against the same week a year ago. A five-week rolling average. Storage cost roughly eighty thousand dollars a gigabyte in 1986 — not a typo, not adjusted for anything, simply the number a company like Metaphor had to build and sell to its customers. Five weeks of data could live on a spinning disk, where it could be queried in something close to real time. Everything older than that lived on tape, in a vault, and had to be physically mounted and read before it would answer anything at all.

I can still hear the sound the tape drive made when a request came in — a slow mechanical inhale, the click of heads finding a track, then a long hiss while the reels turned, forward and back, hunting for whatever weeks' worth of numbers somebody had asked for. It was not a fast process, and it was not meant to feel like conversation. It felt like exactly what it was: a machine going to considerable physical trouble to fetch a fact and a human had to mount it to work.

What the finance department had never done, in all those months, was hand that capability to anyone who managed a category of product. So, I went looking for someone who would. I found him in soup — a category manager who handled, among a few hundred other things, Vons's soup category — and offered him something nobody had offered him before: a one-time spot analysis, product movement by SKU, five weeks of data, across

ten stores in the Southern California region. Nothing about the offer was permanent or official. It was a demonstration, built to show what the system sitting two floors away in finance could do if somebody asked it a category question instead of a financial one.

There was no great revelation buried in the numbers themselves, and I want to be honest about that rather than dress the story up for the sake of a better chapter. No SKU turned out to be secretly outperforming. No pattern nobody had ever suspected leapt off the page. The revelation, such as it was, had nothing to do with what the data said and everything to do with what he could suddenly do with it. Before that afternoon, if he wanted to look past the green sheet's fixed view of his category, his only path was to pull the raw numbers and rebuild the comparison by hand in a spreadsheet — a slow, manual process he could only afford to do occasionally, for whatever question happened to feel most urgent that week. What I put in front of him instead was five weeks of SKU-level movement across ten stores that he could interrogate directly, on demand, without rebuilding anything from scratch. He could ask a defining question on a Tuesday afternoon — which SKUs were driving the category's movement, whether a slow mover was slow everywhere or only in two stores, what a different slice of stores would show — and get an answer the same day instead of the same month. That shift, from occasional manual reconstruction to something closer to a running conversation with his own data, was what made managing the category active instead of reactive. It wasn't that the paper green sheet had been lying to him. It was that the green sheet had only ever let him confirm what had already happened, never to go looking for what might explain it.

That distinction — between what a system can do and what an organization has actually decided to let it do — turned out to matter as much as anything about the underlying technology, and it's a distinction that resurfaces again and again across the chapters ahead, in nearly every era this book covers. Metaphor hadn't undersold Vons. Vons had, without quite meaning to, undersold Metaphor to itself, by installing a genuinely capable decision-support tool and then only ever asking it finance's questions. It took someone walking two floors down and asking a different kind of question before the tool did the thing it had been built for.

I've turned this over many times in the years since, from the vantage point of roughly a dozen more technology cycles than I had in 1986. And here is what strikes me most, looking back: the system we built for Vons was honestly named, even in the parts of the building where nobody was using it as designed. Nobody called it an oracle. Nobody called it an answer engine. The category of technology my industry was inventing at the time called itself, plainly, Decision Support Systems. Support — not replacement. Whoever settled on that name, deliberately or by accident, drew the line in exactly the right place. The system's job stopped at the edge of the number. Whether a human ever picked up that number and did something with it — asked it a real question, ran a real what-if — was never the system's decision to make. It was always somebody else's, and at Vons that fall, for one afternoon, it was a soup category manager.

That line held in 1986 for a reason worth naming precisely, because it's the reason the rest of this book exists. A system that needed thirty seconds of tape-hiss to hand over a week of SKU movement had no fluency to speak of, no confidence in its voice, no capacity

to sound like it understood something it plainly did not. It could not have talked anyone into believing it grasped the why even if, somehow, it had wanted to. It simply hadn't been built with a mouth — which meant the only way it ever became a decision support system, in practice and not just in name, was if a person decided to support a decision with it.

It's worth pausing on that name for a moment, because it did real work. 'Decision Support' wasn't marketing copy in 1986 — the phrase had a precise, almost technical meaning inside the industry, distinguishing a category of tools built to inform a human decision-maker from the older model of computing, where a request went in, a batch job ran overnight, and an answer came back with no expectation that a person would need to interrogate it further. Decision support systems assumed, by design, that a person would sit down with the number and do something with it that the system itself could not do. The category's name wasn't chosen to sound impressive. It was chosen because it was accurate, and because in 1986, nobody in the room had any incentive to pretend it was anything more.

The name was right in 1986, and the industry has spent the forty years since trying, in one rebrand after another, to talk its way out of what that name meant. Continuously even today.

Every generation of tools that followed offered a faster version of the same tape mount. Cleaner interfaces. Prettier charts. Eventually, tools that could hold something resembling a conversation. But underneath every improvement, the machine never once learned to tell you why a number moved. It only ever got better — remarkably, relentlessly

better — at telling you that it moved, and telling you fast enough, and eventually fluently enough, that it becomes tempting to mistake the telling for the deciding.

Today, a category manager doesn't need someone to walk two floors down and offer a one-time demonstration. They can open a chat window and ask their own what-if question in plain English, and get back, in seconds, an answer written in full, confident sentences — a tone of voice no green-screen terminal in 1986 could have managed even in parody. That part of the gap really has closed. What hasn't closed is the other gap, the one that mattered more at Vons than the technology ever did: the gap between a tool that's capable of supporting a real decision and an organization that's decided to use it that way. Most companies today have access to something far more powerful than what sat two floors above that soup category manager's desk, and a great many of them are still, in effect, only reading the green sheet — pointing extraordinary capability at routine questions because nobody's walked down two floors and asked it something harder. The tools have changed almost beyond recognition. The job of noticing the gap, and deciding to close it, is, in every way that matters, still a human being's job in the year this book is published and it's more important than ever.