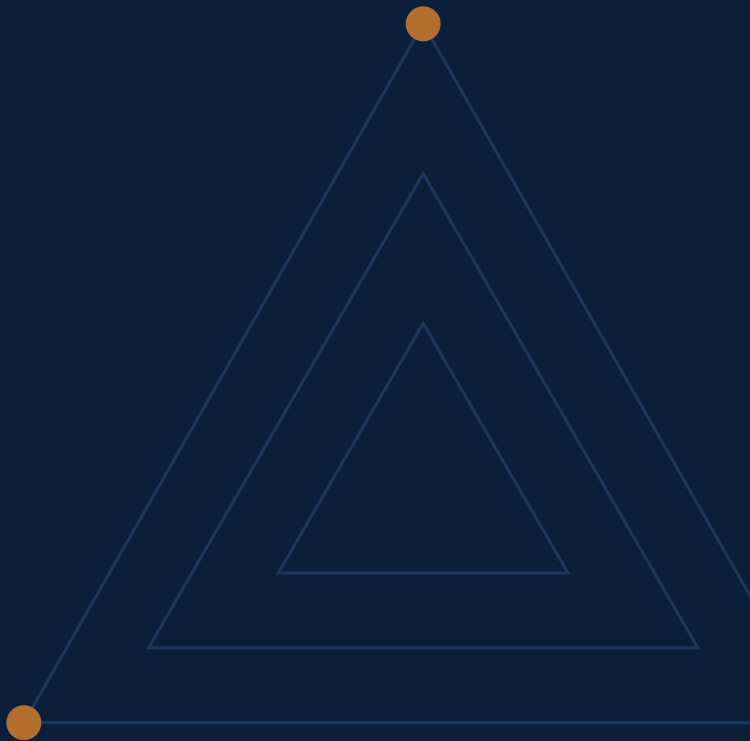


ITSM · AUTOMATION · AI GOVERNANCE · PEOPLE

Automated, Not Absolved

Who's Accountable When AI Runs IT — Real Stories and a Governance Playbook for Service Leaders



Ahmed Nabil

FREE SAMPLE · CHAPTER 1 · ITSMCOMPASS.COM/BOOK/



Contents

01	The Work Behind the Ticket	1
	Analysis	2
	Forty-five jobs and a busier phone line	3
	What we are really automating	5
	Notes	10

CHAPTER

01

The Work Behind the Ticket

COLD OPEN

THE CALL CAME IN angry, and it came in already tired.

I was a technical support engineer at a large telecom operator in Cairo, one of the people answering a hundred or more contacts a day: mobile data that wouldn't connect, home broadband that crawled, business lines, handsets with settings nobody could find. Most calls had a shape you learned to recognise in the first ten seconds.

This one had a shape too. It was the shape of someone who had done this before.

The customer's connection kept dropping. Not all the time, which would almost have been easier, but often enough to ruin a working day and rarely enough that it was usually behaving when someone looked. They had called before. More than once. Each time, someone had taken them through the troubleshooting steps, the connection had come back, and the ticket had been closed as resolved.

And each time, a few days later, it dropped again.

So by the time the call reached me, the anger wasn't really about the fault. It was about every previous person who had said "that should fix it".

CUSTOMER: Every time, someone tells me it's fixed. Every time.

The script had a place for this. Apologise, verify the account, run the checklist. I didn't start there. I let them talk. All of it: the previous calls, the dropped connections, the feeling of being handed from one polite voice to the next. It takes longer than you'd think, and it is the most important part of the call. People can't hear a solution while they're still waiting to be heard.

Then I told them something the script doesn't say: that I would personally see this through, and that I wouldn't close the call until we knew why it kept happening, not just that it was working right now.

The checklist had been run before, several times. It had passed several times. That was the clue. So I went beyond it, into the checks the standard flow skipped, looking for the reason an intermittent fault stays intermittent. We found it. We fixed it. I stayed on the line until the connection had held long enough for both of us to believe it.

At the end, the customer thanked me. By name.

Later that week, I wrote down what I'd done: the extra checks, in order, and why the standard flow missed them. It became one of the first articles in the knowledge base I built for the team.

...

The ticket said resolved. It had said resolved every time.

It had never once recorded the work.

Analysis

If you look at that call as a ticket, it is unremarkable: one contact, fault identified, fixed, closed. First-contact resolution, if you're generous, because this time it didn't come back. A good customer satisfaction score. A few extra minutes of handle time.

Now look at what actually happened, layer by layer.

Listening. An angry person was allowed to finish. Nothing was resolved in those first minutes, and everything depended on them.

History. The real problem was not the fault. It was the pattern of previous “resolved” tickets. Seeing that pattern meant reading the past, not just the present.

Ownership. Somebody said “I will see this through”. That sentence changed the call, because it changed who was responsible for the outcome.

Judgement. The checklist had passed repeatedly, and a passing checklist was the evidence that the checklist was the wrong tool. Knowing when to leave the script is a skill, and it takes years to learn.

Persistence. Staying on the line until the fix had proved itself, rather than until the ticket could be closed.

Teaching. Turning one call into an article, so the next person didn’t have to rediscover it. Across the team, that habit eventually cut our average resolution time by more than a third.

Six layers of work, and the ticket captured roughly one of them.

This is the gap at the centre of this book. When organisations decide what to automate, they look at the ticket, because the ticket is what they can see. Volumes, categories, handle times, resolution codes. It all looks wonderfully automatable. But the value of a good service desk rarely lives in the fields of the ticket. It lives in the work behind it.

Automate the ticket, and you may keep the fields while losing the work.

Forty-five jobs and a busier phone line

In July 2025, the Commonwealth Bank of Australia, the country’s largest bank, cut 45 customer service roles after introducing an AI voice bot on one of its inbound enquiry lines. The business case was

straightforward: the bot would handle the routine calls, and fewer people would be needed.

Call volumes went up.

Managers offered overtime. Team leaders were pulled onto the phones to cope. The Finance Sector Union took the case to Australia's Fair Work Commission, and in August 2025 the bank conceded that the roles had not in fact been redundant. It apologised to the staff and offered them their jobs back, redeployment or an exit payment.

It would be easy to read this as a story about a bad bot. I don't think it is. It is a story about a business case built on the ticket rather than the work.

When a routine call is handled badly, the customer doesn't disappear. They call back. They call back angrier, and they ask for a human. The systems thinker John Seddon called this *failure demand*: demand caused by a failure to do something, or do it right, the first time. My customer with the dropping connection was failure demand, several times over, long before any AI was involved. Automate carelessly and you don't remove that demand. You manufacture more of it.

When I asked myself what a voice bot would have done with that call, the answer came quickly. The customer would have asked for a human, and been angrier by the time they reached one. The bot might even have run the same checklist that had already passed several times, and closed another ticket as resolved.

THE NAME ON THE REPORT

Who decided: the bank's leadership, on a business case that counted calls rather than the work inside them. **Who paid first:** 45 people told a machine had made them unnecessary, and later told it hadn't. **Who absorbed the gap:** the team leaders and remaining staff working overtime to answer the calls the bot generated. **Who carried the accountability in the end:** the bank, publicly, after a union and a tribunal made it.

WHAT THE DASHBOARD SAID / WHAT PEOPLE FELT

The dashboard said: bot deployed; contacts contained; ticket closed as resolved; forty-five roles removed from the cost base. **What people felt:** a customer telling the same story for the fourth time; a team leader back on the phones at the end of a long shift; forty-five people wondering what they had done wrong.

What we are really automating

None of this is an argument against automation. I have spent my career automating things, and the knowledge base that came out of that angry call was itself a form of it: capturing a human judgement so it could be repeated without the human.

The question is not *whether* to automate. It is *what* exactly we are automating, and whether we have looked closely enough to know.

From scripts to agents

Service management has been automating for decades, and each wave has quietly moved the line between the ticket and the work.

- **Scripts and runbooks** automated steps a human had already decided to take.

- **Workflow and robotic process automation** automated the routing of work between people and systems.
- **AIOps** began to automate noticing: correlating alerts, spotting anomalies, suggesting probable causes.
- **Generative AI and agents** now automate talking and doing: answering the customer, writing the summary, and increasingly taking the action.

Each step up that ladder takes on a little more of the work behind the ticket. That is exactly why each step needs a closer look at what is being handed over.

The six things AI actually does

The latest version of ITIL offers a helpful lens here. Rather than treating “AI” as one thing, it describes six kinds of capability, which it calls the six Cs. Think of them as six different jobs, each with its own risks:

Capability	What it does	In that call
Creation	Produces new content	Drafting the knowledge article afterwards
Curation	Keeps knowledge accurate and current	Flagging that the old troubleshooting article kept “passing” and failing
Clarification	Summarises, rephrases, translates	Summarising the customer’s previous tickets in seconds
Cognition	Finds patterns and anomalies	Spotting that “resolved” plus “called back” is a pattern
Communication	Talks to people in natural language	Answering the call itself
Coordination	Takes actions across systems	Running the deeper line checks and applying the fix

Laid out like this, something interesting happens. AI could have helped with almost every layer of that call. Summarising the history

in seconds would have been a gift. Spotting the pattern of repeat contacts would have been better than relying on one engineer's instinct. Drafting the article would have saved an evening.

But look at what is missing from the table. *Listening until someone is ready to hear you. Saying "I will see this through" and meaning it. Deciding that a passing checklist is the problem.* Those are not capabilities you install. They are what the capabilities are supposed to serve.

ITIL itself is careful on this point. The six Cs describe functions, not levels of maturity, and ITIL pairs them with an emphasis on the competence of the people who design, manage and oversee AI. The model tells you what the machine can do. It does not tell you who owns the outcome.

The hidden workload

There is one more thing the ticket-level view misses: the work that automation creates.

In SolarWinds' 2026 State of ITSM survey of 844 IT professionals, most respondents said their AI investments had met their expectations for return. Yet more than half said their workload had gone up, and 44% said they spent more time maintaining AI than using what it produced. Teams measured on activity, such as tickets closed, were far more likely to report a heavier workload than teams measured on outcomes.

That is the Commonwealth Bank story at a smaller scale. The work doesn't vanish. It moves: into maintenance, into exceptions, into repeat contacts, and into whoever is left to catch what the automation drops.

AUTOMATED, NOT ABSOLVED

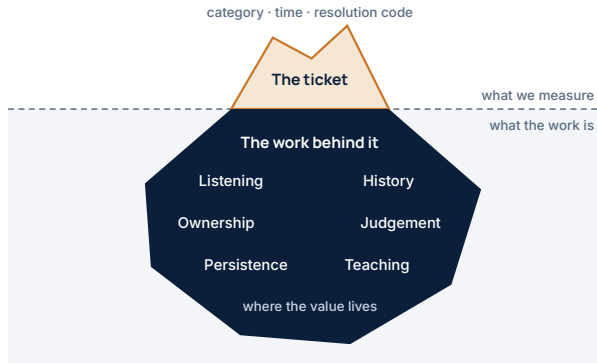


Figure 1.1 — The ticket is the tip. Most of the value of service work sits below the fields we measure, which is exactly the part an automation business case tends not to count.

KEY IDEA

A ticket records the problem, rarely the work. Before you automate a kind of contact, map the work behind it — listening, history, ownership, judgement, persistence, teaching — and decide which parts AI should take, which it should support, and which must stay human. Automate from the ticket alone, and you keep the fields while losing the work.

CHECKLIST · BEFORE YOU AUTOMATE A CONTACT TYPE

- ☐ We have listened to real contacts of this type, not just read the category reports
- ☐ We know how many of these contacts are failure demand (repeat or “chaser” contacts)
- ☐ We have mapped the work behind the ticket, not only its fields
- ☐ Each part of that work is marked: AI does it, AI supports a human, or human only
- ☐ Customers can reach a human easily, and we know how often they ask to
- ☐ We will measure outcomes (problem gone, customer stays) as well as activity
- ☐ We will measure repeat contacts for at least three months after go-live
- ☐ No roles are removed until the new model has proved itself on real demand
- ☐ Someone owns the knowledge the AI relies on, and keeps it current
- ☐ Affected staff were consulted, and their knowledge was captured before any change

MONDAY MORNING CARD

Three actions. Listen to five recorded or live contacts from your highest-volume category and list the work that never reaches the ticket. Pull the repeat-contact rate for that category. Find one knowledge article your best engineer carries in their head, and get it written down. **One question for your next team meeting:** “If a bot took our most common call tomorrow, what would it miss?”

Workbook: exercise 1.1, the AI capability inventory map.

Notes

- Commonwealth Bank of Australia voice-bot job cuts and reversal: ABC News, 21 August 2025, <https://www.abc.net.au/news/2025-08-21/cba-backtracks-on-ai-job-cuts-as-chatbot-lifts-call-volumes/105679492>
- “Failure demand”: John Seddon, *Freedom from Command and Control* (Vanguard, 2003) and later work.
- The ITIL AI capability model (the “six Cs”): ITIL® (Version 5) learner materials, PeopleCert. ITIL® is a registered trademark of PeopleCert. Descriptions paraphrased.
- SolarWinds, 2026 State of ITSM, <https://www.solarwinds.com/blog/2026-state-of-itsm-what-800-it-professionals-say-about-ai-automation-and-the-year-ahead>
- The author’s own story: the customer is not identifiable, and the dialogue is reconstructed from memory.