

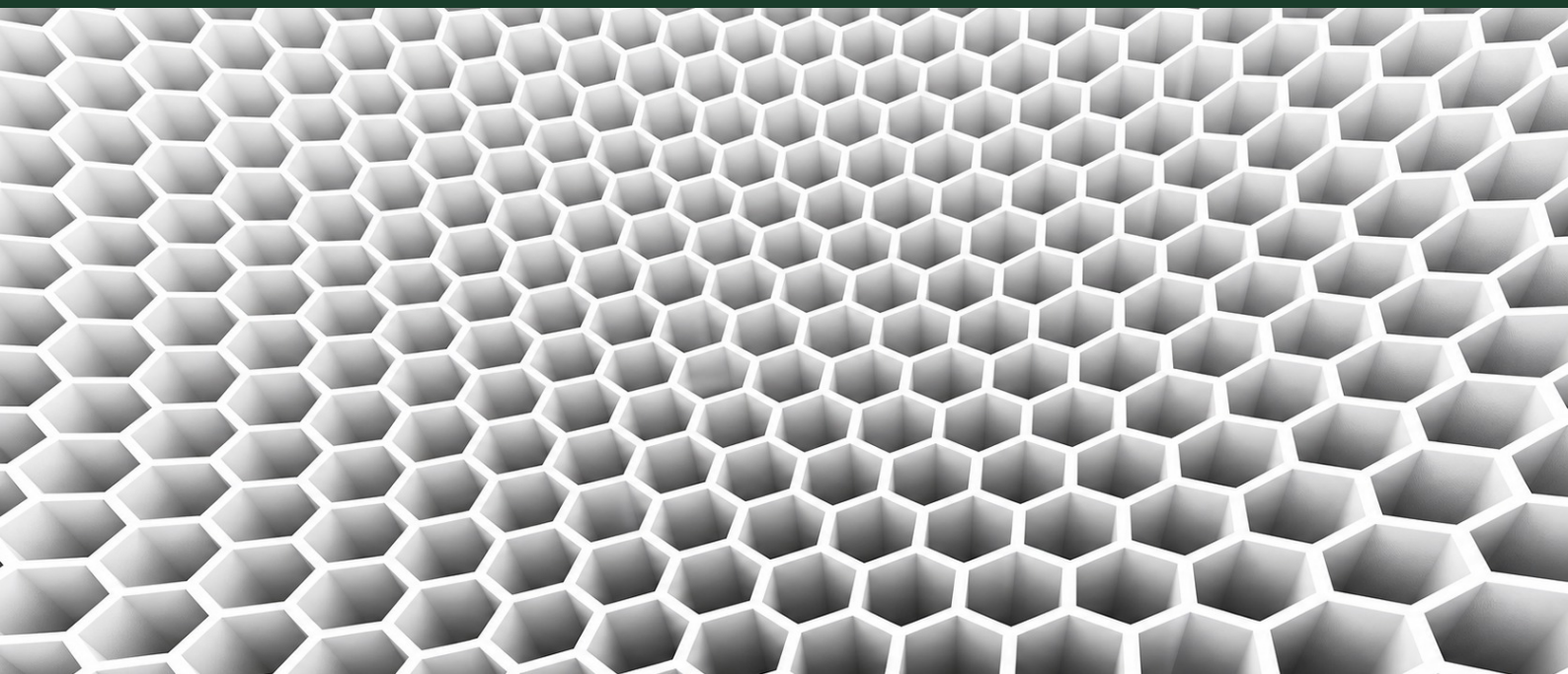
BEYOND RENTED CAPABILITY

The Augmented Delivery Organisation — An Operating Model for the Function That Delivers Everything Else

The Case for Owning — Not Renting — the Organisation's Ability to Land Change

Giovanni Leonardi

May 2026



Contents

- 1. Executive Summary** 3
- 2. The Function We Keep Building and Dissolving** 4
- 3. Why Delivery Capability Becomes an Asset Class** 4
- 4. An Operating Model in Five Layers** 5
 - 4.1 The Capability Core 6
 - 4.2 The Machine Layer 6
 - 4.3 The Formation Studio 7
 - 4.4 The Memory Institution 8
 - 4.5 The Governance Spine 9
- 5. What This Costs — and What It Replaces** 9
- 6. Getting There from Here** 10
- 7. The Objection Worth Taking Seriously** 11
- 8. The Function-Design Choice** 12

The Augmented Delivery Organisation — An Operating Model for the Function That Delivers Everything Else

Programme · No. 308

May 2026

“The organisation that treats them as a cost centre — assembled per programme, funded from project budgets, dissolved after — is liquidating an asset each time a programme closes and repurchasing it, at a premium, when the next one starts.”

1. Executive Summary

The delivery function — the organisational capability that turns strategy into operating reality — is the enterprise's last unredesigned function. Every other support capability has been through its own target operating model exercise: finance has its shared services and robotic process automation, HR its people analytics and employee experience platforms, technology its platform engineering and DevOps cultures. Delivery, by contrast, remains largely where it was a decade ago: assembled ad hoc from contractors and internal secondments at the start of each programme, dissolved when the programme closes, its knowledge walking out the door in the rucksacks of the people who held it.

This paper argues that model is no longer viable. As autonomous agents absorb the execution mechanics of delivery — the scheduling, the status aggregation, the dependency tracking, the risk-register hygiene — the function's value migrates decisively to what remains: the ability to make integration decisions under uncertainty, to hold the envelope between business intent and technical reality, and to land change into organisations that resist it. These are not individual talents; they are institutional capabilities that require deliberate investment, permanent staffing, and organisational memory. They are, in short, an asset class.

This white paper presents a target operating model for the delivery function built around five layers: the Capability Core (the senior, pattern-literate humans who make the decisions agents cannot); the Machine Layer (the delivery agents, their fleet operations, evals, and prompt estate, run as the function's own production system); the Formation Studio (the rebuilt professional ladder — simulation, shadowing, and the drill calendar that develops the next generation when entry-level tasks have been automated); the Memory Institution (the decision records, post-mortems, and pattern library that give the function its institutional continuity); and the Governance Spine (the integrated method maintained as the function's standard product). Sizing, roles, a funding model, and a

maturity path follow. The reader should be able to take this to a budget conversation.

2. The Function We Keep Building and Dissolving

The scene is recognisable to anyone who has spent time inside a large organisation's change portfolio. A new transformation programme is approved. The programme director is appointed — sometimes internal, more often a contractor brought in for the engagement. A delivery team is assembled: programme managers from a preferred supplier, a PMO team recruited through a staffing agency, business analysts borrowed from the functions being changed, technical leads seconded from IT. The team ramps up over twelve to sixteen weeks. They spend the first month learning the organisation's ways of working, its governance rituals, its reporting templates. They build the plan. They begin to deliver.

Eighteen months later, the programme closes. The contractors leave. The secondees return to their functions. The programme director moves on to the next engagement. The lessons-learned document — if one is written — is filed in a SharePoint library that will not be opened again. The next programme starts six months later, and the cycle repeats.

The waste in this model is not primarily financial, although the financial cost is substantial. A mid-sized transformation programme in financial services or utilities typically spends between fifteen and twenty-five per cent of its first-year budget on ramp-up: recruitment, onboarding, context-building, relationship formation. That investment is written off in full when the programme dissolves. But the deeper waste is epistemic. Every programme generates hard-won knowledge about how this specific organisation absorbs change — which governance forums actually make decisions, which stakeholders exercise informal vetoes, where the data integration points fracture under load, what the real dependencies are behind the ones drawn on the plan. That knowledge has no home. It lives in people, and the people leave.

We have tolerated this model because the alternative appeared to be a standing army of delivery professionals with insufficient work between programmes — an overhead the business could not justify. That calculus, however, rests on an assumption that is now breaking: that the delivery function's value is distributed roughly evenly across a spectrum from mechanical execution to strategic judgement. If that assumption holds, you need a large team of mixed seniority, and a large standing team is expensive. The assumption no longer holds.

3. Why Delivery Capability Becomes an Asset Class

The agent revolution in delivery — and it is already under way, if unevenly — is not automating the hard parts. It is automating the easy parts. The scheduling, the status aggregation, the dependency recalculation, the risk-register updates, the meeting preparation, the minutes, the action tracking, the resource reconciliation, the earned-value arithmetic: these are the tasks that consumed the bulk of a programme office's time and that gave employment to the junior and mid-level delivery roles which formed the function's broad base.

What agents cannot do — and what the current generation of large language models gives no credible path towards — is the integration work that sits at the top of the delivery pyramid. The decision about whether to cut scope or slip the date when both options carry political consequences. The reading of a room that tells an experienced programme director that the technology team's confidence is performed rather than felt. The judgement call about when a dependency on another programme's deliverable is genuinely on track and when the reported green status is an artefact of a team that has stopped testing. The negotiation between a business sponsor who needs the benefits to land in this financial year and an architecture team that knows the shortcut will create three years of technical debt.

These are not tasks that can be decomposed into instructions. They require pattern recognition built over years, contextual judgement about this organisation's specific dynamics, and the institutional authority to make a call and be accountable for it. They are, in the language of capability planning, non-fungible and slow to build.

The structural consequence is that the delivery function's shape inverts. The broad base of execution roles shrinks as agents absorb the mechanical work. The narrow apex of senior judgement roles becomes, proportionally and absolutely, more important. And that apex cannot be rented by the programme and returned when the programme ends, because the judgement it exercises is organisation-specific. A senior delivery professional who has spent three years inside an energy company's change portfolio, who knows its governance culture, its integration architecture, its stakeholder landscape, carries knowledge that a contractor arriving next month cannot replicate — no matter how strong their CV.

This is what it means to say that delivery capability is becoming an asset class. An asset, in this sense, is something that depreciates if unattended, appreciates if invested in, and generates returns that compound over time. The delivery function's institutional knowledge, its pattern library, its professional formation pipeline, and its governance standards all have these properties. The organisation that treats them as a cost centre — assembled per programme, funded from project budgets, dissolved after — is liquidating an asset each time a programme closes and repurchasing it, at a premium, when the next one starts.

Delivery capability is becoming an asset class: it depreciates if unattended, appreciates if invested in, and generates returns that compound over time. The organisation that assembles it per programme and dissolves it after is liquidating an asset and repurchasing it at a premium.

4. An Operating Model in Five Layers

The operating model that follows is designed around a single organising principle: the delivery function should be a permanent, funded, professionally managed organisational unit — no different in status from Finance, HR, or Technology — with its own headcount, its own professional standards, its own career ladder, and its own production systems. Each of the five layers that follows addresses a distinct dimension of what the function

must do; none works without the others. The capability-as-asset-class argument is the spine that connects them: each layer exists because some dimension of delivery capability will depreciate without it.

4.1 The Capability Core

The Capability Core is the function's human establishment: the people who make the decisions that agents cannot make, who hold the integrations that span technical and organisational boundaries, and who carry the pattern recognition that turns experience into judgement.

The core is deliberately small and deliberately senior. In a large organisation running a change portfolio of, say, fifteen to twenty concurrent programmes, the Capability Core might number forty to sixty people — roughly a third of what the same portfolio would have required under a fully manual delivery model. But the profile is different. These are not project managers who track progress; they are delivery professionals who make decisions about delivery under uncertainty. Three roles anchor the core:

- Integrators are the programme directors and senior programme managers who hold the envelope between business intent and technical execution. They are accountable for outcomes, not plans. Their distinguishing capability is the ability to read a situation that is not described accurately by its own reporting and to act before the formal escalation reaches the steering group. The integration skill — the subject of much of the profession's recent reckoning with its own purpose — finds its organisational home here.
- Envelope Owners hold the integration boundaries: between programmes, between the change portfolio and business-as-usual operations, between the organisation and its delivery partners. In a portfolio of any complexity, the most common failure mode is not that individual programmes fail internally but that the spaces between programmes are unmanaged. Envelope Owners exist to manage those spaces. The role is distinct from portfolio management as traditionally conceived; it is operational rather than supervisory, concerned with the live state of the boundary rather than a periodic portfolio review.
- Professional Sceptics are the assurance and challenge function: the people whose job is to test the delivery narrative, to probe the green statuses, to ask the question the programme team has not asked itself. They are not auditors arriving after the fact; they are embedded professionals who participate in the delivery rhythm but whose accountability runs to the truth of the situation, not to the programme's success narrative. The question of how assurance operates when reporting is largely automated — when an agent produces a status that is syntactically flawless but substantively misleading — makes the Professional Sceptic's role more important, not less.

These roles require experience that takes eight to twelve years to develop. They cannot be recruited quickly, and they cannot be replaced by agents. They are the function's scarcest resource, and the operating model exists, in large part, to make the best use of them.

4.2 The Machine Layer

If the Capability Core is the function's human establishment, the Machine Layer is its technical one. This is where the delivery agents live — the autonomous and semi-autonomous systems that handle the execution mechanics the function previously staffed with junior and mid-level roles.

The critical design choice is to treat the Machine Layer as a production system, not as a set of tools that individual delivery professionals happen to use. The difference matters. Tools are adopted by individuals, configured to taste, and abandoned when the individual moves on. A production system is centrally operated, systematically maintained, and institutionally accountable. The Machine Layer is a production system:

- **Fleet operations:** the agents are managed as a fleet, with centralised deployment, monitoring, and incident response. When a scheduling agent produces an implausible critical path, there is someone whose job it is to investigate, diagnose, and fix — not a programme manager who shrugs and overrides. The fleet operations discipline borrows from site reliability engineering, adapted for delivery agents rather than microservices.
- **Evals:** the agents' outputs are systematically evaluated against known-good baselines. A risk-assessment agent that consistently underweights integration risk is a production defect, not a quirk. The eval framework is the function's quality system for its machine layer, and it requires the same rigour that a regulated industry applies to any other system whose outputs drive decisions.
- **Prompt estate:** the prompts, templates, and context windows that drive the agents are managed as a maintained asset — version-controlled, peer-reviewed, tested against regression suites. This is the function's equivalent of a codebase, and it requires the same discipline. Prompt drift — the gradual, unmanaged divergence of prompts across a portfolio as individual teams make local adjustments — is the Machine Layer's equivalent of configuration drift, and is managed the same way.

The Machine Layer needs its own small team: two to four people in a mid-sized function, with skills that bridge delivery domain knowledge and AI operations. These are not traditional IT roles, and they are not traditional delivery roles; they are a new professional hybrid whose emergence the function must actively cultivate.

4.3 The Formation Studio

The Formation Studio addresses what is perhaps the most uncomfortable consequence of the agent revolution for the delivery profession: the disappearance of the entry-level work that historically formed the next generation of senior professionals.

The formation question is not new to the profession, but it becomes acute when the tasks that once constituted the apprenticeship disappear. A decade ago, the path to becoming a senior programme director ran through years of progressively responsible roles: project coordinator, project manager, senior project manager, programme manager, programme director. At each stage, the professional learned by doing — by tracking actions, managing risks, running workstreams, and gradually developing the pattern recognition and judgement that define senior capability. The early stages of that ladder are precisely the work that agents now absorb.

The Formation Studio is the function's answer: a deliberate, structured programme of professional development that replaces the apprenticeship-through-doing model with

something more intentional:

- **Simulation:** complex delivery scenarios, built from the function's own post-mortem library, that place developing professionals in decision-making situations they would previously have encountered only after years of operational exposure. Not classroom exercises but high-fidelity reconstructions that demand real-time judgement — a governance escalation with incomplete information, a supplier negotiation where the numbers do not add up, a benefits review where the original business case has quietly become fiction.
- **Shadowing:** structured attachments to senior Capability Core members, with explicit observation protocols and reflection cycles. The developing professional observes the integrator's work across a full programme cycle, records the decision rationale at each significant intervention, and debriefs after each one. The aim is not to transmit a procedure but to develop the perceptual skill — the ability to notice what is not being said — that separates a competent manager from a senior delivery professional.
- **The Drill Calendar:** a regular cadence of exercises — tabletop scenarios, red-team challenges, live rehearsals of governance escalations — that maintain and develop professional readiness across the whole Capability Core, not only the people in formation. The drill calendar is borrowed from professions that have long understood this principle — aviation, medicine, the military — and adapted for the delivery context.

The Formation Studio requires a small dedicated team — typically two to three people with deep delivery experience and facilitation skill — and a modest budget for scenario development. Its output is the function's future senior talent, and without it the Capability Core ages and thins with no pipeline to replenish it.

4.4 The Memory Institution

Every delivery function generates knowledge. Almost none retain it. The argument for institutional memory in delivery has been made many times and in many forms; what changes now is that the practical means to maintain it have improved, while the cost of not maintaining it has increased — because the casual, informal knowledge transfer that occurred when large delivery teams worked together for months is lost when those teams shrink and their mechanical work is handled by agents.

The Memory Institution maintains three assets:

- **Decision records:** not minutes but structured records of why significant delivery decisions were made — the options considered, the constraints in play, the judgement applied, and the outcome observed. These are searchable, categorised by decision type, and referenced routinely when analogous situations arise. A new Integrator taking on a programme can read the decision record of the last programme that touched the same systems and understand not just what was decided but why — and what the team wished it had done differently.
- **Post-mortems:** formal, structured reviews of completed programmes — not the sanitised lessons-learned documents that currently pass for organisational learning, but candid assessments of what worked, what failed, and why, written for an internal professional audience. The post-mortem practice is the Memory Institution's most

culturally demanding product, because honesty requires organisational safety. This is why the institution needs authority, not merely headcount: the knowledge lead must have the standing to require a post-mortem and the independence to write one that a programme director would prefer not to see.

- The Pattern Library: a curated collection of delivery patterns — recurring problem-situation-response sequences that the function has observed across its portfolio. Patterns such as the integration dependency that is nominally managed but actually unowned, or the benefits case that was approved on assumptions the programme team knows are stale but nobody has the incentive to revisit. These are the function's professional knowledge base. They feed the Formation Studio's simulations, inform the Professional Sceptics' challenge protocols, and over time become the tacit curriculum through which the function's institutional judgement reproduces itself.

The Memory Institution requires dedicated headcount: a knowledge lead and one to two curators, with the authority to require post-mortems and the independence to write them honestly. Without headcount and authority, it becomes a document repository that nobody maintains — which is what every organisation already has.

4.5 The Governance Spine

The final layer is the Governance Spine: the integrated delivery method that serves as the function's standard product. This is not a methodology manual filed in a process library and ceremonially reviewed once a year. It is the function's maintained, living decision-making engine — the set of governance patterns, gate criteria, escalation protocols, and reporting standards that the function applies across its portfolio.

The question of whether governance is a reporting mechanism or a decision-making engine — a question the profession has been wrestling with for years — takes on a new shape when the reporting is automated and the decisions are not. In an agent-augmented environment, the Governance Spine serves a dual purpose: it defines the function's professional standards and it provides the interface specification between the human and machine layers. The agents operate within the boundaries the method defines; the method's decision points mark the places where human judgement is required. This is not a theoretical distinction — it is an operational one. An agent that generates a risk assessment hands it to a governance gate where a Professional Sceptic reviews it. The gate's criteria, the review protocol, and the escalation path are all products of the Governance Spine.

The Spine's distinguishing characteristic is that it is maintained as a product. It has an owner, a backlog, a release cycle. When the function observes that a particular governance pattern is consistently failing — that a gate criterion is being routinely waived, for instance, or that a reporting format is generating noise rather than signal — the pattern is investigated, redesigned, tested, and re-released. The method evolves deliberately, rather than drifting through accumulated exceptions that nobody has the authority or the incentive to codify.

5. What This Costs — and What It Replaces

A budget conversation requires numbers. The numbers that follow are composites, drawn from observation of delivery functions across financial services, energy, and public-sector organisations, and they are intended as a sizing framework rather than a quotation.

Component	Headcount	Annual cost range (£)	What it replaces
Capability Core	40–60	£4m–£7.5m	100–150 mixed-seniority contractors at £8m–£15m with ramp-up waste
Machine Layer ops	2–4	£200k–£500k	Tooling fragmentation, manual reporting, unmanaged agent sprawl
Formation Studio	2–3	£250k–£400k	External recruitment at premium rates to replace departing seniors
Memory Institution	2–3	£200k–£350k	Repeated discovery of lessons already learned and lost
Governance Spine team	2–3	£200k–£400k	Methodology shelf-ware, inconsistent governance, exception drift
Total	48–73	£4.85m–£9.15m	A model costing 1.5–2× as much with no institutional memory

The comparison is not between the new model and zero. It is between the new model and the current one — which is not free, and which is not cheap. The current model funds delivery through project budgets, which means every programme carries its own delivery overhead. When those overheads are aggregated across a portfolio — the contractor day rates, the recruitment fees, the twelve-to-sixteen-week ramp-ups, the knowledge that is purchased once and then discarded — the total is almost always higher than a standing function would cost.

The funding model itself needs to change. Delivery capability funded from project budgets is delivery capability that appears as a cost to each project sponsor — and project sponsors, rationally, minimise cost. The operating model proposed here requires central funding from the enterprise change budget, treated as infrastructure investment rather than project overhead. This is the same funding model that organisations already use for enterprise architecture, for the technology platform, and for the finance function itself: the cost is borne centrally because the benefit — institutional capability that serves the whole portfolio — cannot be allocated to a single programme.

The most persuasive line in the budget case is not the cost comparison, although the comparison favours the new model. It is the risk argument. An organisation that liquidates its delivery capability between programmes is an organisation that has no institutional capacity to course-correct a failing transformation — because the people who would recognise the failure pattern and know what to do about it are, by design, not there.

6. Getting There from Here

No organisation can stand up the full operating model in a single step, and no sensible sponsor would try. The maturity path has four stages, and each stage delivers measurable value before the next one begins.

Stage 1 — Anchor the Core. Identify the fifteen to twenty most experienced delivery professionals already in the organisation — staff and long-tenure contractors — and consolidate them into a single organisational unit with a single leader. Fund it centrally. Stop redeploying these people through project budgets. This is the nucleus of the Capability Core, and it is the single most consequential move: it creates an institutional home for delivery judgement that did not previously exist. Timeline: three to six months to establish.

Stage 2 — Stand Up the Machine Layer. Establish fleet operations for the delivery agents already in use — and by the first half of 2026, most large portfolios are using some, even if informally. Appoint an owner. Begin building the eval framework and formalising the prompt estate. The goal at this stage is not sophistication but ownership: someone is accountable for the agents' performance, their outputs are evaluated, and their prompts are version-controlled. Timeline: six to nine months from Stage 1.

Stage 3 — Build the Memory. Commission the first round of honest post-mortems from recently completed programmes. Begin building the pattern library and establishing the decision-record practice. Appoint the knowledge lead. This stage is often the most culturally difficult, because honest post-mortems require organisational safety, and organisational safety requires visible senior sponsorship. The cultural work cannot be delegated. Timeline: six to twelve months from Stage 2.

Stage 4 — Complete the Model. Stand up the Formation Studio and formalise the Governance Spine as a maintained product with an owner and a release cycle. By this stage the function has a track record, a cost baseline, and a demonstrable value proposition — which makes the case for the full model considerably easier than it would have been at Stage 1. Timeline: six to twelve months from Stage 3.

An organisation that begins in late 2026 could have the full model operational by 2029, with measurable benefits from Stage 1 onward. The maturity path is designed so that each stage strengthens the case for the next — and so that an organisation that stops at Stage 2, for whatever reason, still has something substantially better than what it started with.

The maturity path begins with a single move: consolidate the organisation's most experienced delivery professionals into one centrally funded unit. Everything else builds from that anchor.

7. The Objection Worth Taking Seriously

The strongest objection to this model is not that it costs too much — the cost comparison generally favours it — but that it creates a bureaucratic overhead that slows the programmes it is meant to serve. A standing delivery function, the argument goes, becomes an empire: it imposes process, it demands reporting, it creates governance that

exists for its own perpetuation rather than for the portfolio's benefit. Organisations have seen this before. The PMO that becomes a post office. The methodology team that produces frameworks nobody uses. The centre of excellence that is neither.

This objection deserves a serious answer, because it is grounded in real and repeated experience. Standing functions do tend towards bureaucratic self-perpetuation, and a delivery function is no exception. The history of centralised programme offices is littered with examples of exactly this failure mode.

The answer is structural, not aspirational. The operating model contains its own feedback mechanisms — not guarantees against drift, because no organisational design can guarantee that, but structures that make drift observable and correctable:

- The Capability Core's professionals are deployed into programmes, not stationed in a central office reviewing reports. Their value is visible daily to the programme teams they serve, which creates a natural accountability that a distant centre of excellence lacks. A programme director who is not adding value to their programmes will hear about it, directly and quickly.
- The Governance Spine is maintained as a product with users, not a policy with subjects. If a governance pattern is not serving the portfolio — if a gate is being routinely gamed, if a reporting requirement is generating effort without insight — the Spine's product owner is accountable for changing it. The post-mortem practice provides the evidence base for that change, and the Professional Sceptics provide the independent voice to surface it.
- The Machine Layer's eval framework creates an objective performance baseline. The function can demonstrate, with data, that its agents are producing more accurate schedules, earlier risk signals, and fewer false escalations than the fragmented tooling they replaced. This is a harder standard than most centralised functions are held to, and it is the right one.
- The function's funding model — central, not per-project — means the function must justify its total cost to the enterprise, not merely its marginal cost to each programme. This is the accountability mechanism that a project-funded model lacks: there is a single number, attached to a single leader, reviewed in a single forum.

The honest answer is that this model can still fail. An incompetent function head, an unsupportive executive sponsor, an organisational culture that punishes candour — any of these can defeat the structural safeguards. But the model at least creates the possibility of a delivery function that improves over time, learns from its own experience, and adapts its methods to what actually works. The current model — assemble, deliver, dissolve, forget — does not even create that possibility.

8. The Function-Design Choice

The question this paper addresses is not whether to use agents in delivery — that question has been settled by practice across most large portfolios. Nor is it whether the delivery function will change — it is already changing, as the mechanical execution work migrates to machines and the remaining human work concentrates at higher levels of judgement. The question is whether the organisation will design a function around this

new reality or allow the function to emerge by accident, as individual programmes adopt agents in their own ways, the contractor market adjusts its rate cards, and the institutional knowledge continues to walk out the door between engagements.

The case for deliberate design is the case for treating delivery capability as what it has become: an asset the enterprise must own, staff, and maintain — not a cost centre it rents programme by programme. The operating model presented here — five layers, forty-eight to seventy-three people in a mid-sized enterprise, centrally funded from the change budget — is one way to do it. The maturity path makes it achievable in stages. The funding comparison makes it defensible in a budget conversation. The structural feedback mechanisms make it answerable to the portfolio it serves.

The delivery function has spent a generation redesigning everyone else's organisation. The argument of this paper is that it is time — past time — to redesign its own.

*“The delivery function has spent a generation
redesigning everyone else's organisation. It is time to
redesign its own.”*
