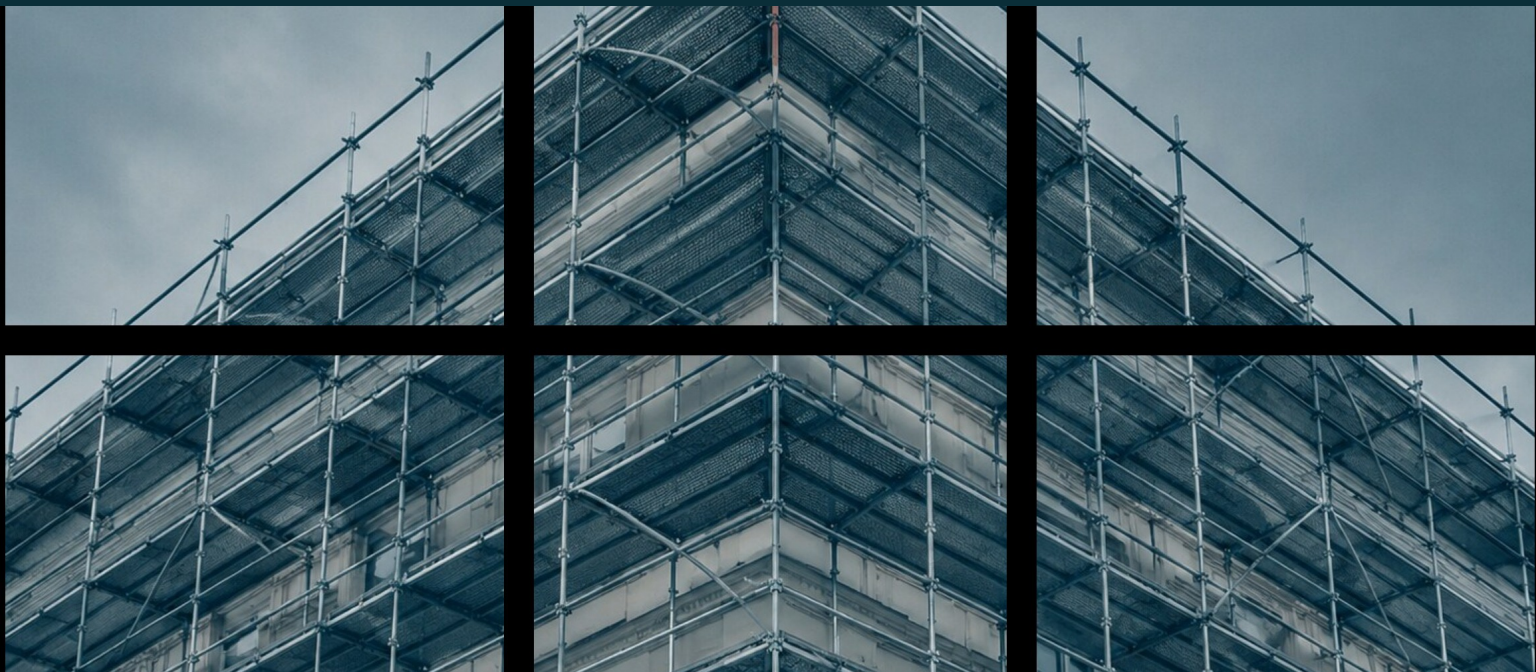


BEYOND THE INTEGRATOR

When the Integrator Owns the Strategy: The Hidden Cost of Vendor-Led Transformation

A Practitioner's Perspective on Systems Integrator Dependency in Large-Scale Change



Giovanni Leonardi

May 2001

Contents

1. Executive Summary	3
2. How Strategy Migrated to the Vendor	3
3. The Anatomy of the Dependency	4
4. The Evidence: Where Vendor-Led Programmes Break Down	5
5. What Was Tried, and What Actually Worked	6
6. A More Effective Model: Own the Strategy, Buy the Capacity	7
7. Recommendation	8

When the Integrator Owns the Strategy: The Hidden Cost of Vendor-Led Transformation

Transformation · No. 31

Originally written May 2001 | Re-edited and re-rendered for the WhereBeyond Edition, 2025

“The organisation that cannot articulate its own transformation without reaching for the vendor's slides has already lost the argument it did not know it was having.”

1. Executive Summary

The programmes commissioned in the eighteen months since the market turned share an uncomfortable characteristic. The organisation pays for the transformation, lives with its outcomes, and answers to its board for the result — but no longer owns the strategy that shapes it. That strategy has migrated, quietly and largely by default, to the systems integrator.

This paper examines how that migration happened, why it persists, and what it costs. The conditions were specific to this moment. A Year 2000 remediation effort left organisations dependent on external hands for their most critical systems. A collapse in market confidence stripped internal ambition and internal headcount in the same quarter. An enterprise software wave — the packaged suites now sitting at the centre of every large operating model — arrived too large and too fast for most in-house teams to absorb. Into that gap stepped the global integrators, offering not merely delivery capacity but the thinking itself.

The evidence assembled here, drawn from the observable pattern across large package-implementation, e-business and shared-services programmes, points to a consistent failure mode. When the integrator owns the strategy, the client loses three things in sequence: the standing to challenge the design, the knowledge to operate the result, and the practical option to leave. Each loss makes the next harder to reverse.

The paper sets out a defensible alternative — a model in which the organisation retains strategic authorship and a genuine intelligent-client capability while buying delivery capacity on contestable terms. The recommendation is not to stop using integrators. Few organisations could deliver a programme of this scale without them, and pretending otherwise helps no one. The recommendation is to stop outsourcing the one thing that cannot later be bought back.

The question is not whether to engage a systems integrator. It is whether, eighteen months into a programme, anyone inside the organisation still understands it well enough to change its direction.

2. How Strategy Migrated to the Vendor

The dependency did not arrive through a single decision. It accumulated through a sequence of reasonable ones, each defensible on its own terms, and it is that reasonableness which makes the pattern so difficult to see from inside.

The first condition was the remediation programme that consumed the closing years of the last decade. Preparing the estate for the century date change was not a strategic exercise; it was a vast, deadline-driven scramble that pulled in external hands at every level. Organisations emerged from it having proven, to themselves and to their boards, that large technical undertakings could be handed to a partner and delivered against a fixed date. The lesson learned was seductive and only half true: that the hard problems of the estate were problems best given away.

The second condition was the market itself. The correction that began last spring did not merely deflate valuations; it changed the internal posture of the organisation almost overnight. The appetite for ambitious, self-authored change evaporated. Discretionary hiring stopped. The internal architects and programme leaders who might have owned a transformation were, in many organisations, the same people whose roles were now under review. Ambition and capacity were withdrawn together, and the vacuum was immediate.

The third condition was the software. The packaged enterprise suites that now anchor finance, supply chain and human resources are not tools an organisation configures over a weekend. They embed a model of how the business should work, and implementing one is as much an act of organisational design as of engineering. Very few internal teams had lived through such an implementation more than once. The integrators had lived through dozens.

- The remediation years normalised handing critical work to external partners against fixed dates.
- The market correction withdrew internal ambition and internal capacity in the same quarter.
- The packaged-software wave demanded design experience almost no client possessed and every integrator did.

When these three conditions met, the integrator was not merely the natural choice to build the transformation. It was the only party in the room that appeared to know what the transformation should be. And so the client began, without ever deciding to, to buy the strategy along with the build.

3. The Anatomy of the Dependency

It is worth being precise about what "owning the strategy" means in practice, because the phrase can sound abstract. It is not abstract. It shows up in artefacts, in meetings, and in who can answer which questions.

Strategy ownership reveals itself first in authorship. Ask who wrote the target operating model, the business case, the benefits map, the design principles that govern the programme. In the pattern under examination, the honest answer is the integrator — often working from a method and a set of reference designs the client has never seen in full and could not reproduce. The client reviewed and approved. But reviewing a document you could not have written, and could not now rewrite, is not the same as owning it.

It reveals itself second in the ability to ask the awkward question. A programme owned internally has people who can look at a design decision and say, with authority, that it is wrong for this organisation. A programme where strategy has migrated outward loses that

voice. The awkward question still gets asked — but it is asked by the same party that would have to unpick its own work to answer it honestly, and that is a structurally weak position from which to expect candour.

It reveals itself third, and most damagingly, in the operating knowledge left behind at the end. A transformation is not finished when the system goes live; it is finished when the organisation can run, adapt and extend what it has been given. Where the strategy was authored externally, the design rationale often leaves with the team that authored it. The client is left operating a machine whose reasoning it never held.

“Reviewing a document you could not have written, and could not now rewrite, is not the same as owning it.”

These three — authorship, challenge, and operating knowledge — are not independent. They fail together and in order. Lose authorship and you weaken challenge; lose challenge and you stop accumulating the understanding that would let you operate independently later. By the time the deficit becomes visible, usually at or after go-live, the cheapest moment to have corrected it is long past.

4. The Evidence: Where Vendor-Led Programmes Break Down

The failure of a vendor-led programme rarely looks like a failure of delivery. This is the part that most confuses boards. The system is often delivered. The milestones are often met. The integrator, judged on the contract it actually signed, frequently performs. The breakdown occurs in the space between what was delivered and what the organisation needed — a space the contract never described because the strategy that would have described it was never the client's to write.

Three patterns recur.

The first is the well-built answer to the wrong question. Because the design embeds the integrator's reference model rather than the organisation's actual distinctiveness, the programme optimises for a generic version of the business. The parts of the operating model that were genuine sources of advantage — the unusual way the organisation served a particular customer, the process that competitors could not easily copy — are quietly standardised away, because standardisation is what the method rewards and what the client lacked the authority to contest.

The second is the benefits case that no one can now defend. The business case was authored to secure funding, often with the integrator's assistance, and it made promises about cost and value that were plausible at the time. But because the client never fully owned the logic beneath those numbers, no one inside the organisation is positioned to track them, challenge them, or explain to the board why they have not materialised. The benefits case becomes an artefact of approval rather than an instrument of management.

The third is the lock-in that reveals itself only at the exit. The point of maximum dependency is not the middle of the programme; it is the moment the organisation first tries to change direction, change partner, or bring the work back inside. That is when it discovers that the design knowledge, the configuration decisions, and the undocumented reasoning all sit with a party whose commercial interest is served by that knowledge

remaining external. The switching cost was there all along. It simply was not visible until someone reached for the door.

What the client believes it is buying	What the pattern actually delivers
Delivery capacity for a strategy it owns	Delivery of a strategy the vendor owns
An objective expert view on the design	A design shaped by the vendor's reference method and commercial interest
Knowledge transfer at the programme's end	Knowledge departure with the team that held it
Optionality to change course later	A switching cost that only becomes visible at the exit

None of this requires bad faith on the integrator's part, and it is important to say so plainly. A capable partner, delivering competently against the mandate it was given, will still produce these outcomes if the client has surrendered the strategy. The failure is structural, not moral. It is a failure of ownership, and it is the client's to prevent.

5. What Was Tried, and What Actually Worked

Organisations have not been blind to this. Several responses have been attempted over the past two years, with markedly different results, and the contrast between them is instructive.

The most common response has been contractual. If the risk is dependency, the reasoning goes, then tighten the contract — fix the price, specify the deliverables, write in knowledge-transfer clauses and penalty regimes. This helps at the margin, and a well-structured contract is certainly better than a loose one. But it treats a strategy problem as a procurement problem. A knowledge-transfer clause cannot transfer knowledge the client's own people were never staffed to receive. A fixed price does not confer authorship. The contractual response manages the symptom and leaves the cause intact.

A second response has been to appoint a second firm to watch the first — an assurance or client-side advisory role, independent of the integrator. This is better, because it reintroduces the awkward question from a party with no stake in the answer. But it has a subtle weakness: it can become another form of outsourced thinking. If the organisation cannot itself judge whether the assurance firm is right, it has simply added a layer, not reclaimed the strategy. Independent assurance works only when the client is capable enough to act on what it hears.

The response that has actually worked, in the cases where the pattern was avoided, was neither contractual nor supervisory. It was structural. Those organisations kept a small number of genuinely capable people — an intelligent-client function, in the language now emerging — who owned the strategy, the design principles and the benefits logic outright, and who used the integrator for capacity rather than for direction. This function did not need to be large. It needed to be senior, permanent, and unambiguously accountable for the thinking. Its members could write the target operating model themselves, even if they chose to have it built by others. They could ask the awkward question and understand the answer. And crucially, they stayed after go-live, holding the operating knowledge that

would otherwise have walked out of the door.

1. The contractual response manages the symptom; it cannot manufacture ownership.
2. The assurance response helps only if the client can itself judge the assurance.
3. The intelligent-client response works because it keeps authorship, challenge and operating knowledge inside the organisation.

The difference in outcome was not marginal. Programmes run on the third model absorbed integrator capacity without absorbing integrator dependency. They emerged able to operate, adapt and extend what they had built. The others emerged owning a system and renting the understanding of it.

6. A More Effective Model: Own the Strategy, Buy the Capacity

The alternative this paper recommends can be stated as a single principle and then unpacked: buy delivery capacity, never strategic authorship. Everything else follows from holding that line.

The first requirement is an intelligent-client function that exists before the programme is scoped, not after it has gone wrong. This is the part organisations most often get backwards. The capability to own a transformation cannot be assembled once the transformation is underway, because by then the design decisions that mattered most have already been taken by someone else. The function must be in place early enough to author the strategy itself.

The second requirement is that this function holds four artefacts as its own, regardless of who physically drafts them: the target operating model, the design principles, the benefits logic, and the data and integration architecture that ties the estate together. These are the load-bearing decisions. They can be developed with the integrator's help, but if the client cannot rewrite them without the integrator, the client does not own them.

The third requirement is that the commercial relationship is kept genuinely contestable. Dependency thrives where the client could not, in practice, change partner. Preserving optionality – through modular scope, documented design decisions held by the client, and a deliberate refusal to let any single partner become the sole holder of critical knowledge – is what keeps the integrator honest and the client free.

Decision	Owned by the client	Delivered with the integrator
Target operating model	Yes – authored and defensible internally	Drafting support, reference patterns
Design principles	Yes – the client's, not the method's	Challenge and refinement
Benefits logic	Yes – tracked and defended internally	Modelling support
Build and configuration	Governed by the client	Yes – this is what capacity buys
Operating knowledge	Retained internally by design	Transferred continuously, not at the end

The fourth and final requirement is temporal: knowledge transfer must be continuous, not an event scheduled for the programme's close. Knowledge scheduled for handover at the

end is knowledge that leaves at the end. The intelligent-client function should be embedded in the design decisions as they are made, accumulating understanding in real time, so that there is nothing left to "transfer" because the client held it all along.

7. Recommendation

The organisations reading this fall, broadly, into two groups, and the recommendation differs for each.

For those about to commission a major transformation, the instruction is to build the intelligent-client function first and scope the integrator's role second. Decide, explicitly and in writing, which decisions will be authored internally and which capacity will be bought. Treat the target operating model, the design principles, the benefits logic and the integration architecture as non-negotiable client property. Then engage a partner — a capable one, on a contestable footing — to deliver against a strategy the organisation owns outright.

For those already inside a programme where the strategy has migrated outward, the instruction is harder but more urgent. The dependency will not reveal its full cost until the exit, and the exit is the worst moment to discover it. The remedy is to reclaim authorship now, while the programme is still live: staff the intelligent-client function even at this late stage, require the design rationale to be documented and held by the client, and begin the continuous transfer of operating knowledge before the team that holds it disperses. It is more expensive than getting it right at the outset. It is far cheaper than discovering, at go-live, that the organisation owns a system it cannot run.

The systems integrator is not the problem. The integrators are, in the main, capable and necessary, and the scale of change now underway could not proceed without them. The problem is a client posture that mistook buying capacity for buying certainty and gave away the one asset it could not repurchase. The organisation that cannot articulate its own transformation without reaching for the vendor's slides has already lost the argument it did not know it was having. Reclaiming that argument — owning the strategy while buying the capacity — is the single most consequential decision a board will make about the programmes it is funding this year.