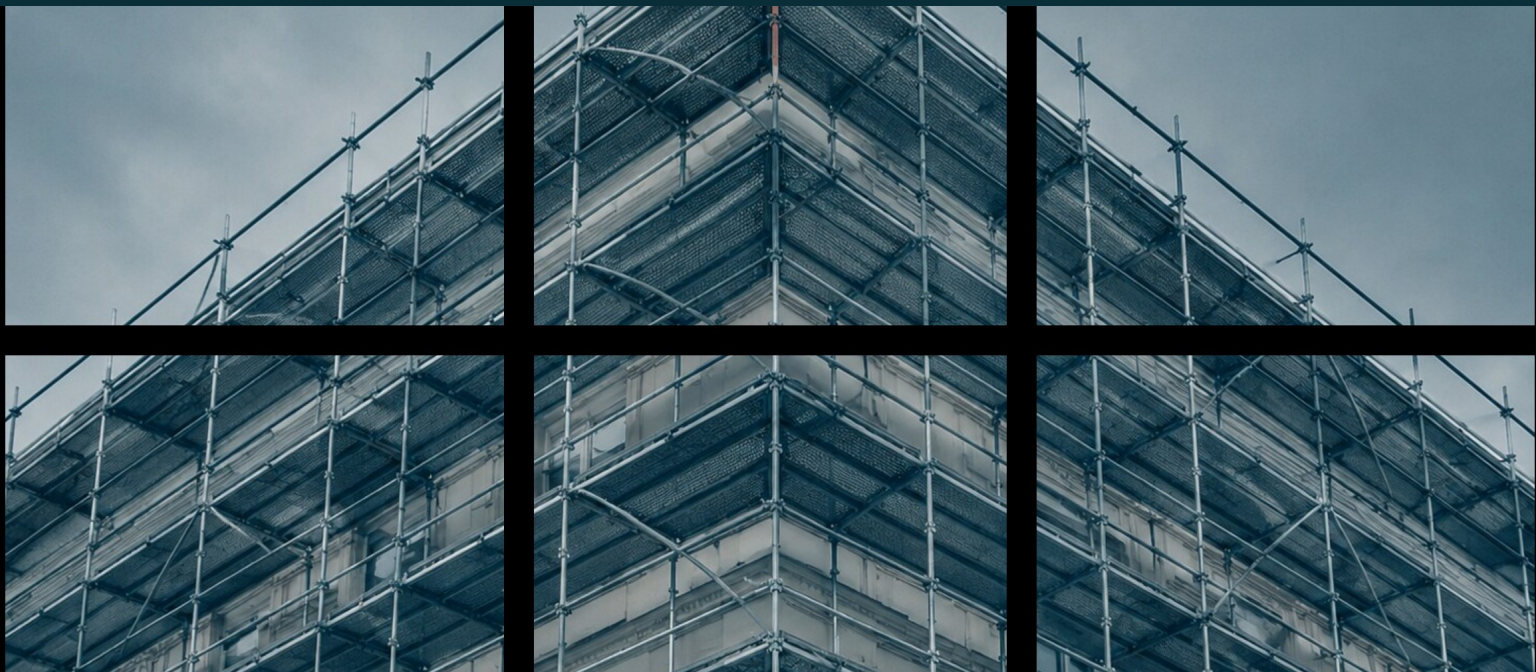


BEYOND LABOUR ARBITRAGE

The Transferability Illusion — Why the Back Office Did Not Offshore the Way the Code Did

How a genuine victory in application outsourcing became the flawed blueprint for everything that sat behind it



Giovanni Leonardi

May 2004

The Transferability Illusion — Why the Back Office Did Not Offshore the Way the Code Did

Transformation · No. 53

Originally written May 2004 | Re-edited and re-rendered for the WhereBeyond Edition, 2025

“Sending the process away did not send the accountability away. It never does.”

1. The Month-End That Moved Eight Thousand Kilometres

The first sign was not in the numbers. It was in a corridor. A financial controller who had, for fifteen years, resolved a coding query by walking twenty paces to the ledger clerk's desk now raised a service request, attached the exception, and waited. The answer came back the following afternoon — correct, courteous, and thirty-six hours too late for a conversation that used to take ninety seconds. Nothing had failed. The close still balanced. And yet something the organisation had never priced, and never named, had quietly left the building.

That something is the subject of this essay. Over the past few years, organisations have learned to send work abroad, and they have learned it well. What began as a scramble to find enough hands for Y2K remediation and application maintenance hardened into a capable, industrialised model: specify the work, transfer the knowledge, run it at a fraction of the domestic cost. The saving was real, the quality often improved, and the confidence it produced was justified. The trouble is what that confidence was then asked to do. Emboldened by success with systems, organisations turned to processes — finance and accounting, human-resource administration, procurement, the whole apparatus of the back office — and applied the same logic, the same business case, and the same expectations. This is the moment worth examining: not the decision to offshore, but the assumption that a process would offshore the way a system did.

It will not, and the reason is not operational. It is that a business process carries something an application does not.

2. Why the Code Went First

There was nothing accidental about IT leading the way. Software work has properties that make it unusually portable. Its inputs and outputs are explicit — a specification in, a tested build out. Its quality can be inspected at the boundary without watching the work being done. It had, by the early part of this decade, a maturity apparatus around it — capability assessments, defect metrics, the discipline of the appraised process — that let a buyer trust a supplier eight time zones away on the strength of an audit rather than a handshake. And the demand shock of Y2K had forced a generation of organisations to prove, under deadline, that remote delivery could work at all.

So the code went first because the code was, in a specific sense, separable. You could lift application maintenance out of the organisation and the organisation would still know what it was doing; the work already sat at arm's length from the business's own

judgement.

The success was then misread. What had travelled well was not "work that could be sent away" in general. It was a particular kind of work — codifiable, inspectable, loosely coupled to the daily exercise of business judgement. Reading the win as a general proof of portability, rather than as evidence about one category of task, is the original error from which most of the disappointment that followed descends.

The lesson of IT outsourcing was never "processes can be moved." It was "codifiable, inspectable, loosely coupled work can be moved." Everything since has turned on the difference.

3. What a Process Carries That a System Does Not

Sit with a mature business process for a week and you find it is only partly the thing written in the procedure manual. The documented flow — receive the invoice, match it to the order, post it, pay it — is real, and it is perhaps four-fifths of the volume. But the value, and nearly all of the risk, lives in the other fifth: the exceptions, the judgement calls, the quiet knowledge of which supplier always invoices in the wrong currency, which internal manager will escalate if a payment slips, and which apparent discrepancy is a fraud signal rather than a typing error. None of that is in the manual. It lives in the people, and in the hundred informal relationships that let them resolve in a corridor what the system cannot resolve on a screen.

Three things, in particular, resist the boundary that offshoring draws:

- Tacit knowledge. The part of the work that has never been written down, because those doing it never needed to write it down. It transfers, if at all, slowly and by apprenticeship — not in a six-week knowledge-transfer window with a departing incumbent who has already been told their role is ending.
- Exception handling. The transactional core is genuinely portable; the exceptions are where the process meets the specific, messy reality of this organisation, and they demand a context the manual never held.
- Entanglement with decision. An application sits beside the business's judgement. A finance or HR process is woven into it. Move the process and you have not relocated a task; you have put a border, a time zone, and a service-level agreement between the organisation and part of its own thinking.

This is the heart of it. IT was separable because it stood at arm's length from judgement. The back office is not separable in the same way, because in many places the back office is the judgement, wearing the costume of a routine.

4. The Costs That Never Reached the Business Case

The business cases of this period are strikingly consistent, and strikingly incomplete. A typical one takes a loaded domestic cost, applies a wage differential of sixty per cent or more, and presents the gap as the saving. On that arithmetic a back-office transition promises to take, say, forty-five per cent out of the cost base.

Watch the same transition eighteen months on and the realised figure is very often in the high teens. The difference is not failure, and it is not fraud; it is the set of costs the model never carried.

1. The retained organisation. The work does not leave cleanly. A shadow team stays behind to manage the relationship, own the exceptions, and hold the accountability that cannot be exported — and it is frequently larger, and more senior, than anyone forecast.
2. Coordination. Every query that once crossed a desk now crosses a boundary. The thirty-six-hour loop from the opening of this essay is not an anecdote; multiplied across thousands of exceptions a month, it is a standing tax on the organisation's own speed.
3. Knowledge erosion. Once the last incumbent has gone, the organisation loses not merely the doing of the process but the understanding of it. It can no longer easily improve what it no longer comprehends — and, increasingly, it cannot answer its own auditors.

That last cost has lately grown teeth. With the new demands on internal control over financial reporting now bearing down on listed companies, the organisations that dispersed their finance processes are discovering that "we transferred it to the provider" is not an answer a controller can give about a control they must personally attest to. Sending the process away did not send the accountability away. It never does.

*"A wage differential is a fact about a labour market. It is not,
on its own, a fact about your costs."*

5. The Honest Objection: Give It Time and Discipline

The strongest case against everything above is not naïve, and it deserves to be met at full strength. It runs like this. These are teething problems, not structural ones. The pioneers who struggled did so because they transferred messy, undocumented processes in a hurry. Do it properly — reengineer and document the process before you move it, put real capability discipline around the transition, hold the provider to genuine service levels, and give the relationship three years to reach steady state — and the tacit knowledge gets codified, the exceptions get systematised, and the saving arrives after all. On this view the problem is immaturity, and immaturity is cured by time and rigour.

There is real truth in it, and the discipline it prescribes is worth having. Processes that are documented and rationalised before they move do transfer better than those flung over the wall in a panic. But the objection concedes more than it intends. To codify a process well enough to hand it over is itself a large and skilled act of transformation — often larger than the outsourcing it was meant to enable — and the capability to perform it is precisely the capability the organisation is proposing to send away. Then there is the limit the argument cannot cross: it holds only for the transactional core. The judgement layer is not tacit by accident, to be tidied up with more documentation; it is tacit by nature, because it responds to a specific reality that will not hold still long enough to be written down. And there is a sting in the tail. The very act of freezing a process into a transferable specification can ossify something that ought to keep evolving — you buy stability at the price of adaptability, and discover the price only later. Discipline improves the fraction of

the process that was portable all along. It does not make the rest portable. It cannot.

6. Keeping the Territory While Selling the Map

None of this is an argument against offshoring. The saving is real, the talent is real, and the organisations that refuse the model out of sentiment will be out-competed by those that learn to use it well. The argument is against offshoring by analogy – against treating "the process" as a single object to be picked up and moved because "the systems" were.

What the wiser operators of this period do differently comes down to a few disciplines.

- Separate the transactional from the judgemental, and mean it. Send the high-volume, rules-based, inspectable core. Keep the exception-handling, the relationships, and the calls that carry real risk – not as a grudging residue, but as a deliberate design.
- Treat the retained organisation as the main event, not the leftovers. Its size, its seniority, and its capability decide whether the whole arrangement works. Under-resourcing it to flatter the business case is the most common and most expensive mistake in the field.
- Never export the ability to improve the process. Ownership of how the work evolves must stay inside the organisation. A provider can run your process; a provider cannot be trusted to reinvent it in your interest.
- Price coordination and erosion into the case from the start. A business case that counts only the wage gap is not conservative; it is wrong, and it will be embarrassed by its own actuals inside two years.

There is a deeper pattern beneath these rules, and it is worth stating plainly. A system is a map of the work. A process is the territory – the living thing the map describes. What offshored so well in the first wave was, in truth, a set of maps. The disappointment of the second wave came from treating the territory as though it were merely a larger map.

The organisations that will look wise a few years from now are the ones drawing that distinction on purpose: selling the map, keeping the territory, and never confusing the arbitrage of labour with the transfer of capability. The saving was always real. So was the thing the corridor lost. The whole art is in knowing which is which – and in refusing to let a genuine victory with systems argue you into a careless defeat with everything that sits behind them.