



BEYOND THE ROADMAP

Legacy Modernisation and the Programme That Never Ends — Why Enterprise Technology Renewal Defeats Conventional Delivery

A Practitioner's Perspective on Technical Debt at Enterprise Scale

Giovanni Leonardi

May 2014

Legacy Modernisation and the Programme That Never Ends — Why Enterprise Technology Renewal Defeats Conventional Delivery

Programme · No. 179

Originally written May 2014 | Re-edited and re-rendered for the WhereBeyond Edition, 2025

“Legacy modernisation does not fail because the technology is difficult; it fails because the organisation treats a permanent condition as a temporary project.”

1. The Programme That Restarts

Every large organisation has one. The legacy modernisation programme. It has a different name in each organisation — sometimes a brand, sometimes a code name, sometimes simply the name of the system it is meant to replace — but the shape is always recognisable. A substantial investment case, approved after years of advocacy. A programme team assembled with considerable care. An ambitious scope covering the replacement of core platforms that have been in service for a decade or more. A timeline measured in years. And then, somewhere between the second and third year, the quiet acknowledgement that the programme is not going to deliver what was promised, in the time that was promised, for the cost that was approved.

What happens next varies in its specifics but not in its pattern. The programme is rescoped. The timeline is extended. The business case is revisited. In some cases, the programme is formally closed and a new programme is initiated with a different name, a different leadership team, and a substantially similar scope. The cycle restarts. In my experience, it is not unusual to encounter organisations on their third or fourth attempt at modernising the same core platform.

This is not a story about incompetent delivery. Many of these programmes are led by capable, experienced programme managers and supported by skilled technical teams. The pattern persists across organisations with very different levels of delivery maturity, different technology stacks, and different governance frameworks. It persists, I would argue, because the problem is not in the execution. It is in the conception.

2. Why Legacy Systems Are Not What They Appear to Be

The standard framing of legacy modernisation treats it as a technology replacement problem. The existing system is old. It is expensive to maintain. It cannot support new business requirements. It must be replaced with something modern. The programme is therefore a large technology delivery initiative — a matter of specifying the new system, building or configuring it, migrating the data, and switching over.

This framing is not wrong in its individual claims. The old system usually is expensive. It usually cannot support new requirements well. It usually does need to be replaced. But the framing misunderstands what a legacy system actually is, and this misunderstanding is the source of almost everything that goes wrong.

A legacy system is not simply a piece of technology. It is the encoded operating model of the organisation. Over the years and decades of its operation, the system has accumulated not just data and functionality but the organisation's business rules, exception handling, regulatory interpretations, process workarounds, and institutional knowledge. Much of this is undocumented. Some of it is unknown — understood only by the system's behaviour, not by any person or document. The system does not merely support the business process; in many cases, the system is the business process, and the humans around it have organised their work to accommodate its logic rather than the other way around.

Replacing such a system is therefore not a technology project. It is an organisational archaeology project, followed by an operating model redesign project, followed by a technology delivery project, followed by an organisational change project. Programme structures that account for only the third of these four phases are structurally incapable of delivering the outcome.

3. The Discovery Problem

The first phase — the archaeology — is where programmes most consistently underestimate the work. The assumption is that the requirements for the new system can be gathered through conventional means: workshops with business users, analysis of existing documentation, review of the system's configuration. What this assumption misses is the scale of what is undocumented.

In my experience, large legacy systems contain three layers of logic:

- The documented layer — the business rules and processes that are written down somewhere, whether in requirements documents, process maps, or user manuals. This layer is typically well understood and relatively straightforward to replicate.
- The configured layer — the rules and behaviours that have been implemented through system configuration, custom code, or parameter settings over the years. These are theoretically discoverable through technical analysis, but the volume is often staggering. A core banking platform with fifteen years of customisation may contain tens of thousands of configured rules, many of which interact in ways that no single person understands.
- The tacit layer — the rules and behaviours that exist only in the practices of the people who use the system. The workarounds that operators have developed to compensate for system limitations. The exception handling that has never been

formalised. The knowledge that a particular field must be populated in a particular way for a downstream process to work correctly, even though nothing in the system enforces this. This layer is invisible to conventional requirements gathering and emerges only when the new system is tested against real operational scenarios.

Programmes typically budget for the documented layer, make some allowance for the configured layer, and are blindsided by the tacit layer. The tacit layer is where the schedule breaks.

The most dangerous assumption in legacy modernisation is that the organisation knows how its own systems work. In practice, the system has become its own specification — the only complete record of how the organisation actually operates.

4. The Scope Problem

The second structural challenge is scope, and it manifests in a way that is peculiar to legacy modernisation. In most programme types, scope can be managed through conventional prioritisation — the programme delivers the most valuable features first and defers the rest. Legacy modernisation resists this approach because the system being replaced is already in production. Every function it performs must be replicated in the new system before the old system can be retired, regardless of whether that function is strategically important or not.

This creates what might be called the completeness trap. The programme cannot deliver partial replacement, because a partially replaced legacy system is not replaced at all — it is merely supplemented. The organisation now operates two systems instead of one, with the associated cost and complexity. The business case for modernisation depends on retiring the old system entirely, and retirement requires complete functional coverage.

The result is that scope in legacy modernisation is not a variable to be managed. It is, to a first approximation, fixed by the functionality of the existing system. The programme can influence how the functions are implemented in the new system, but it cannot easily choose which functions to implement. Every function the old system performs is, by definition, a function the organisation needs — because the organisation is performing it.

This is why the conventional programme management approach of managing scope, time, and cost as trade-offs works so poorly in legacy modernisation. Scope cannot be meaningfully reduced without abandoning the objective. Time extends accordingly. And cost follows time.

5. The Organisational Change Problem

Even when the technology delivery succeeds — when the new system is built, configured, tested, and ready for deployment — the programme faces a challenge that is often underestimated to the point of invisibility: the organisation must change how it works.

Legacy systems shape organisational behaviour over time. Teams are structured around the system's architecture. Processes are designed to match the system's workflow. Skills are developed to operate the system's interfaces. Performance is measured against the system's outputs. When the system changes, all of this must change too — and the organisation's resistance to this change is frequently the factor that stalls or kills the programme in its final phase.

The pattern is familiar. The new system is technically ready. Parallel running begins. Users discover that the new system works differently from the old one — not because it is inferior, but because it is different. Tasks that took three clicks now take five, or take two but in a different sequence. Reports that were generated automatically now require a different process. Exception handling that was managed through familiar workarounds now requires new approaches. The organisation experiences the change not as modernisation but as disruption, and the resistance is proportionate.

Programmes that treat the technology cutover as the finish line consistently fail at this point. The technology is ready; the organisation is not. The result is either a forced deployment that damages operational performance and user confidence, or a prolonged parallel running period that doubles operating costs and erodes the business case.

6. What the Pattern Reveals

The persistence of this pattern across organisations, sectors, and technology generations suggests that the problem is not with individual programmes but with the model of programme delivery being applied to the challenge.

Conventional programme management assumes that a programme has a defined scope, a beginning, a middle, and an end. Legacy modernisation does not fit this model. The scope is defined by the existing system, not by business requirements. The beginning is typically the second or third attempt. The middle is prolonged by discovery of undocumented complexity. And the end — the retirement of the old system — is contingent on organisational readiness that the programme cannot directly control.

“Legacy modernisation does not fail because the technology is difficult; it fails because the organisation treats a permanent condition as a temporary project.”

The more honest framing — the one that the most effective organisations are beginning to adopt — is that legacy modernisation is not a programme at all, in the conventional sense. It is a capability — a permanent organisational function that manages the continuous evolution of core platforms. It requires stable funding, not investment cases. It requires standing teams, not mobilisation and demobilisation cycles. It requires governance that measures progress in operational capability, not in project milestones.

7. Towards a Different Approach

The organisations that are managing legacy modernisation most effectively — and effectiveness here means sustained progress without the restart cycle — tend to share several characteristics.

They have abandoned the big bang replacement model in favour of incremental modernisation. Rather than attempting to replace an entire platform in a single programme, they decompose the system into domains and modernise domain by domain, allowing the old and new systems to coexist through well-defined integration points. This reduces the completeness trap: each domain can be modernised, tested, and deployed independently, and the old system is retired incrementally rather than all at once.

They invest heavily in the discovery phase. Rather than treating requirements gathering as a three-month activity at the front of the programme, they treat it as a continuous process that runs in parallel with delivery. Technical teams analyse the legacy system's behaviour systematically, documenting the configured and tacit layers before attempting to replicate them. This is slow and unglamorous work, and it does not produce impressive milestone reports. But it prevents the late-stage discovery that derails programmes.

They plan for organisational change from the outset. The programme includes dedicated capability for process redesign, training, and operational readiness, resourced at a level commensurate with the scale of change. Cutover is planned not as a technology event but as an organisational transition, with the time, support, and tolerance for disruption that implies.

And they measure differently. Progress is not measured by the number of features built or the percentage of the project plan completed. It is measured by the reduction in operational risk, the number of legacy functions successfully retired, and the improvement in the organisation's ability to change. These are slower, less dramatic metrics. They are also more honest.

8. The Uncomfortable Truth

The uncomfortable truth about legacy modernisation is that it is never finished. Technology ages continuously. The system being deployed today will be tomorrow's legacy. The organisation that treats modernisation as a programme to be completed

and closed will find itself, in five or ten years, commissioning another programme to replace the system it has just built.

The alternative is to accept that modernisation is a continuous process — a permanent feature of how the organisation manages its technology estate — and to build the structures, capabilities, and governance mechanisms that support sustained evolution rather than periodic revolution. This is less exciting than a transformation programme. It produces fewer opportunities for dramatic announcements. But it is, in my observation, the only approach that actually works.

The programme that never ends is not a failure of delivery. It is a signal that the problem has been misunderstood. Legacy modernisation is not a destination to be reached. It is a discipline to be sustained.