



BEYOND THE END DATE

The Programme That Refuses to Close

On the discipline of the temporary endeavour, and the digital products that never end — a view from the delivery side

The Programme That Refuses to Close

Programme · No. 173

Originally written April 2013 | Re-edited and re-rendered for the WhereBeyond Edition, 2025

*"We keep patching it with the vocabulary of closure,
asking living things when they intend to die."*

1. The question the committee keeps asking

At some point in the life of every digital programme I have worked near, the same question arrives across the steering committee table: when does this close? It is a fair question. The programme was chartered to close. It has a business case that assumed closure, a plan that ends on a date, and a governance model whose final act is a closure report and the orderly disbanding of the team. The awkwardness in the room is that the thing the programme has built — a customer-facing online channel, a mobile application, a platform that other teams have started to depend on — has no intention of closing. It went live a year ago and it has been changing every fortnight since.

I want to argue something that the standard texts on programme and project management are oddly silent about: that the discipline we have inherited is built, at its very foundations, on the assumption of the temporary, and that the shift now underway — from projects that deliver and end to products that live and continue — quietly breaks those foundations. We have not seen the break clearly because we keep patching it with the vocabulary of closure, asking living things when they intend to die.

2. The temporary organisation

It is worth remembering how load-bearing the end date is. A project, in the discipline as we were taught it, is a temporary organisation assembled to deliver a defined output and then dissolved. A programme is a temporary coordination of such work, assembled to deliver a strategic benefit and then closed. The word temporary is not incidental decoration; it is the assumption on which everything else rests. The business case assumes a one-off cost purchasing a one-off benefit. The funding is released against milestones on the way to a finish. Benefits realisation assumes that, at the end, the new capability is handed across to business-as-usual and the delivery team goes away. Success is defined as closure achieved on time and to budget. Remove the finish line and every one of these mechanisms is left reaching for something that is no longer there.

3. What changed

What changed is that software stopped being delivered and started being operated continuously. A channel that releases every fortnight is never in a finished state; it is only ever in its current state. The application in the customer's pocket updates itself and is expected to keep improving. The idea of a minimum viable product — by now firmly in the vocabulary — makes the point almost too well: the first version is explicitly not the finished thing but the first of many, launched precisely so that it can keep changing in contact with real users. You do not finish an online channel while competitors improve theirs every month. And so the end date on the plan became a polite fiction — a date everyone privately knows will be followed by more work, funded somehow, carried out by a team that will not, in fact, be disbanded.

4. Where the discipline breaks

The breakage is not vague; it shows up at four specific joints.

- The business case. It was written to justify a one-off investment against a one-off return. A living product has a standing run-rate cost and a benefit that must be continually re-earned. The case, signed once, is never opened again — and so nobody notices when the economics drift.
- The funding model. Money is released against milestones, not against a standing capability. To keep a living product alive, organisations invent new phases and releases and dress each one as a fresh project with its own closure — closures that then never happen, because the product simply carries on.
- Benefits realisation. The model assumes a handover to a stable operational home. But there is no settled business-as-usual for a product that is still being built. The handover, when you look closely, is to the same team that just built it.
- Closure as success. The programme manager is trained to regard closure as the job well done. For a product, closure is not success at all — it is the moment the thing stopped evolving, which is to say the moment it began to fall behind.

Consider a composite that will be recognisable. A programme is chartered to deliver the new online channel, with a £4.2m business case and a go-live date, its benefits handed on completion to the business. Eighteen months after go-live it is still spending roughly £180,000 a month, still has no closure report, and its benefits case has not been opened since the day it was signed. Governance keeps asking when it will close. The honest answer is never — but the funding model has no word for never, and so the team keeps manufacturing phases to give the money a shape the system will accept.

5. The obvious objection

The fair reply to all this is that I am making heavy weather of nothing. Is a product not simply a very long programme? Keep the governance, extend the dates, roll one phase into the next — a rolling programme — and the machinery copes. There is, on this view, nothing new under the sun.

I think this mistakes a longer rope for a different animal. Extending an end date is not the same as removing the idea of one. A rolling programme still funds by phase, still measures by milestone, still holds closure as its goal and treats the run state as an afterthought to be handed to someone else. And the assumption that there is a someone else — a stable operational home waiting to receive the finished thing — is precisely what a living product removes. You cannot roll your way out of a category error. However far you extend it, the temporary organisation is still built to end; the product is built to continue. The mismatch is in the foundations, not in the duration.

“A rolling programme is still a programme that believes in endings. The product does not.”

6. What the posture has to become

What this asks for is not a new methodology but a change of stance. Fund the capability rather than the project: a standing team with a known run-rate, held to account for outcomes over time rather than for hitting a closure date. Govern the flow of value rather than the completion of milestones. Stop treating the handover to business-as-usual as the finish, and accept that for a living product, build and run are the same team drawing on the same budget line. And retire, for this class of work, the equation of closure with success. The question is no longer did it finish on time and to budget but is it still earning its keep, and are we still improving it faster than the alternative would.

The hardest habit to give up is the closure report. For a living product, the day you can finally write it is the day the thing stopped mattering.

I am not arguing that programme management is obsolete. For genuinely temporary endeavours — a new billing system, a data-centre migration, a regulatory change with a statutory deadline — it remains exactly the right instrument, and the end date is real. I am arguing that we have been applying the grammar of the temporary to a growing class of work that is permanent, and then calling the resulting discomfort a delivery problem when it is a conceptual one. The textbooks leave this out because they were written for a world in which software was delivered and then, for the most part, left alone. That world is going. The steering committee that keeps asking when the channel will close is not being unreasonable; it is speaking the only language it was given, about a thing that language was never designed to describe.