

BEYOND THE ITERATION

Iterative Delivery and the Benefits Problem

A Practitioner's Exploration of Why Iterative Delivery Has Not Solved the Benefits Problem



Giovanni Leonardi

October 2006

Iterative Delivery and the Benefits Problem

Programme · No. 69

Originally written October 2006 | Re-edited and re-rendered for the WhereBeyond Edition, 2025

“Agile has given us a faster way to build things, but it has not given us a better way to know whether the things we build are worth building.”

1. The Promise That Felt Different

When iterative and incremental approaches to delivery began gaining traction in the late nineteen-nineties, they carried an implicit promise about benefits that was genuinely exciting. The argument ran like this: if we deliver in short cycles, releasing working capability every few weeks rather than waiting months or years for a single monolithic deployment, then benefits begin to flow earlier. Users get value sooner. The organisation can observe what is actually working and adjust course. The traditional problem of the business case — that benefits are promised at the start and assessed, if at all, only at the end — is dissolved by the rhythm of continuous delivery.

This was a compelling narrative, and it contained a real insight. The waterfall model, with its long feedback loops, genuinely did make benefits realisation harder by delaying the point at which anyone could observe whether the thing being built was the thing that was needed. Shorter cycles, tighter feedback, earlier releases — these were real improvements to delivery, and they had real implications for how organisations could think about value.

But something has gone wrong in the translation. Across the programmes I have observed adopting iterative methods, a pattern has emerged that is both consistent and troubling: the delivery cadence has changed, but the benefits discipline has not improved. In many cases, it has got worse.

2. Velocity Is Not Value

The first and most fundamental confusion is between delivery velocity and benefits realisation. Agile methods are exceptionally good at measuring throughput — stories completed, features shipped, releases deployed. The rituals of iteration planning, daily stand-ups, and retrospectives create a powerful sense of momentum. Teams can demonstrate, sprint by sprint, that they are building things. The question they cannot answer with equal confidence is whether the things they are building are producing the outcomes the organisation invested in.

This is not a trivial distinction. A programme that delivers forty features in twenty iterations has demonstrated delivery capability. It has not demonstrated that customer satisfaction has improved, that operational costs have fallen, or that the organisation is better positioned in its market. Those are benefits, and they live in a different domain from delivery metrics.

Agile has given us a faster way to build things, but it has not given us a better way to know whether the things we build are worth building.

The danger is that the visible productivity of agile delivery creates an illusion of value. Stakeholders see working software at the end of every sprint. They attend demonstrations. They see the burndown chart trending in the right direction. All of this feels like progress, and it is — but it is progress measured in outputs, not outcomes. The conflation of the two is one of the most persistent traps in contemporary programme management.

3. The Backlog as Benefits Case

In traditional programme delivery, the business case is the document that articulates what benefits the programme expects to deliver and how. Whatever its limitations — and they are many — it creates at least a formal link between investment and expected return. There is a moment, however ritualistic, at which someone writes down what the organisation expects to get for its money.

In agile delivery, the backlog has quietly assumed this role, and it is not equipped for it. The backlog is a prioritised list of features, stories, and tasks. It is an excellent tool for managing what needs to be built next. It is a poor substitute for a benefits case, because it describes capability to be delivered without articulating the organisational outcomes that capability is expected to produce.

The product owner, in most agile frameworks, is responsible for prioritising the backlog based on value. But value, in practice, tends to mean value to the user or value to the next release, not value in the sense of strategic benefits realisation. The product owner is optimising locally — which is exactly what the role is designed to do — but no one is tracking whether the accumulation of locally optimised decisions is producing the globally intended benefits.

I have seen programmes run dozens of sprints, delivering working software every fortnight, with no mechanism at all for assessing whether the strategic benefits in the original investment case are being realised. The product owner is prioritising. The team is delivering. The sponsors are seeing demonstrations. Everyone is busy, and no one is asking the question that matters.

4. The Retrospective That Never Looks Far Enough Back

Agile retrospectives are one of the methodology's genuine contributions to professional practice. The discipline of pausing regularly to ask what went well, what went badly, and what should change is valuable, and most traditional programme approaches lack an equivalent. But retrospectives, as practised, almost always focus on the delivery process — how the team worked, what impediments arose, whether the estimation was accurate. They rarely, if ever, address benefits.

This is structurally inevitable. The retrospective happens at the end of an iteration, and it looks back over that iteration. The time horizon is two to four weeks. Benefits, by their nature, materialise over months or years. A retrospective that asked whether the programme is on track to deliver its intended benefits would require a different kind of

conversation — one that connects the work of the last few weeks to the strategic outcomes the organisation is pursuing, and one that draws on data from outside the delivery team's immediate domain.

Such conversations do happen, but they happen outside the agile cadence, in steering committees and programme boards that operate on a different rhythm. The problem is that these governance forums often lack the granular understanding of what has actually been delivered to make meaningful assessments of benefits progress, while the delivery teams that have that understanding lack the strategic perspective to connect it to benefits. The two conversations happen in parallel, and the gap between them is where benefits accountability disappears.

5. Why the Early-Value Argument Misleads

The strongest claim made for agile delivery in the context of benefits is that it enables earlier value realisation. By releasing functionality incrementally, organisations begin to capture benefits before the programme is complete. This is true in principle, and genuinely valuable where it applies. But it applies less often and less cleanly than its advocates suggest.

For the early-value argument to hold, three conditions must be met. First, the functionality delivered in early iterations must be independently usable — it must create value in its own right, not merely contribute to a larger capability that only becomes valuable when complete. Second, the organisation must be capable of absorbing and exploiting the new functionality quickly enough to realise benefits within the iteration cycle. Third, someone must actually measure whether the expected value has materialised.

In practice, all three conditions are frequently unmet.

- Many programmes deliver capability in slices that are technically complete but operationally dependent on later slices. A customer-facing portal that lacks integration with the back-office system is working software but delivers no benefit.
- Organisational adoption is rarely as fast as delivery. New functionality requires training, process change, behaviour change, and sometimes structural reorganisation. These happen on timescales measured in months, not sprints.
- Measurement of early benefits is almost never performed. Teams move on to the next sprint, and the question of whether the last release actually delivered value is lost in the momentum of ongoing delivery.

The result is that the early-value promise becomes an article of faith rather than an observed reality. Everyone believes that iterative delivery enables earlier benefits, but no one checks.

6. The Governance Vacuum

Traditional programme governance, for all its bureaucratic weight, at least creates formal checkpoints at which benefits are reviewed. Stage gates, investment committee reviews, benefits realisation reviews — these mechanisms force a periodic conversation about whether the programme is delivering what it promised. They are often perfunctory, but they exist.

Agile delivery, as currently practised in most organisations, has not created equivalent mechanisms. The assumption seems to be that continuous delivery makes periodic benefits review unnecessary — that the ongoing visibility of working software provides sufficient assurance. This is a dangerous assumption, because visibility of outputs is not the same as visibility of outcomes.

What is needed is not a return to heavyweight governance, but a deliberate integration of benefits tracking into the agile cadence. This might take several forms:

- A quarterly benefits retrospective, distinct from the sprint retrospective, that reviews progress against the original benefits case using actual outcome data rather than delivery metrics.
- Benefits indicators embedded in the definition of done — not for individual stories, where they would be meaninglessly granular, but for releases or programme increments.
- Product owners equipped with and accountable for benefits data, not just backlog prioritisation.
- Sponsors who are willing to engage with benefits evidence at the cadence the programme demands, rather than waiting for the quarterly steering committee.

None of this is methodologically radical. It is simply the application of the same rigour that agile brings to delivery, extended to the domain that actually matters to the investing organisation.

7. The Deeper Problem

But there is a deeper issue here, and it goes beyond methodology. The pattern I am describing — agile delivery without benefits discipline — persists because it serves multiple interests.

Delivery teams prefer it because benefits measurement introduces accountability they would rather avoid. Demonstrating that you shipped forty features is comfortable. Demonstrating that those features improved customer retention by five per cent is uncomfortable, because it might reveal that they did not.

The agile movement has optimised brilliantly for the question of how to build things well. It has barely engaged with the question of whether the right things are being built.

Sponsors prefer it because benefits realisation exposes the quality of the original investment decision. If the programme was funded on the basis of a benefits case that was aspirational rather than analytical, a rigorous benefits review will reveal that — and the sponsor who championed the investment would rather not have that conversation.

Product owners prefer it because their role, as defined by most agile frameworks, is focused on maximising the value of the product, not on tracking organisational benefits. These are related but not identical objectives, and conflating them gives the product owner an excuse not to engage with the harder question.

The result is a collective conspiracy of silence. Everyone involved in agile delivery has an incentive to conflate velocity with value, to mistake outputs for outcomes, and to assume that if the team is shipping, the organisation must be benefiting. The methodology itself — with its focus on working software, customer collaboration, and responding to change —

provides the intellectual cover for this conflation.

8. What Honest Practice Looks Like

I am not arguing against agile delivery. The improvements it has brought to how organisations build and deploy capability are real and substantial. I am arguing that the profession needs to be honest about what agile has and has not solved.

It has solved — or at least significantly improved — the problem of delivering working capability in a timely, responsive, and transparent manner. It has not solved the problem of knowing whether that capability produces the benefits the organisation invested in. And by creating an illusion of continuous value through continuous delivery, it may have made the second problem harder to see.

Honest practice would look like this: programmes that adopt agile delivery would simultaneously adopt an explicit benefits framework that operates alongside the delivery cadence. This framework would define the intended benefits clearly and measurably before delivery begins. It would establish baselines against which benefit realisation can be assessed. It would create regular, data-driven checkpoints — integrated with but distinct from the delivery retrospective — at which benefits progress is reviewed. And it would assign clear accountability for benefits realisation to someone with the authority and perspective to act on what the data shows.

This is not a return to waterfall. It is a recognition that agile delivery and benefits management are complementary disciplines, not substitutes for each other. The fact that we are delivering faster does not excuse us from the obligation to demonstrate that what we are delivering is worth the investment. If anything, the speed of agile delivery makes that obligation more urgent, because the cost of building the wrong thing accumulates faster when you are building it more efficiently.

The question the profession needs to ask itself is uncomfortable but necessary: has agile delivery given us continuous value, or has it given us a more sophisticated way of avoiding the question of whether value has been delivered at all?