



BEYOND TWO SPEEDS

Bimodal IT Was the Wrong Answer to the Right Question

Why two-speed delivery is hardening legacy and trapping cloud ambition at the edge

Bimodal IT Was the Wrong Answer to the Right Question

Portfolio · No. 71

Originally written March 2017 | Re-edited and re-rendered for the WhereBeyond Edition, 2025

“Two speeds do not create one enterprise moving faster; they create two enterprises learning to blame each other.”

The Compromise That Became an Operating Model

The portfolio review has acquired a familiar pair of slides. The first celebrates an eight-week release from the new digital team: clean screens, rapid testing, enthusiastic users. The second explains why the service cannot yet go live: a security approval is pending, the required customer data sits behind a quarterly release window, and operations has not accepted the support model.

Both slides are accurate. Together, they reveal the flaw in the increasingly fashionable idea of bimodal IT.

Bimodal IT answered a real question. Large organisations cannot treat an ageing transaction system, a customer-facing experiment and a regulatory change as though they carry the same uncertainty and risk. Some work benefits from short iterations and discovery; some demands deep assurance and controlled change. But we have mistaken different work for separate worlds. The result is not two speeds of one enterprise. It is a fast team whose progress ends wherever the slow team's boundary begins.

Two speeds do not create one enterprise moving faster; they create two enterprises learning to blame each other.

Two Definitions of Done

The mechanism is straightforward. Mode 2 is usually given the visible work: new channels, prototypes, analytics and early cloud services. Mode 1 retains the systems of record, service management, supplier contracts, security controls and most production accountability. The first group is rewarded for release frequency. The second is rewarded for stability and avoidance of failure.

Each behaves rationally. The innovation team declares a feature complete when it passes its own tests. The operational team regards it as incomplete until support, recovery, access control, data integrity and change scheduling are settled. What one

calls bureaucracy, the other calls unfinished engineering.

Consider a composite example typical of current portfolio reviews. A retail lender builds an online quotation service in eight weeks. The calculation logic is sound and user testing is positive. Yet live use depends on four items outside the team's remit:

- a nightly extract from the account system, available only through the incumbent supplier;
- a firewall change requiring the monthly infrastructure board;
- security accreditation based on documents the delivery team did not know it needed;
- a support rota and recovery procedure that no budget currently funds.

The work then spends twenty weeks crossing queues. Seventy-one per cent of the elapsed time from start to service lies outside the team celebrated for moving quickly. Hiring more developers or tightening the sprint cadence cannot alter that figure. The constraint is the operating model, not the development method.

Cadence should vary with the work. Accountability for the whole service should not.

Cloud First, Operating Model Last

Cloud-first mandates make this weakness more expensive, not less. Moving virtual machines out of a data centre can shorten capacity lead times. It does not, by itself, change how services are owned, how costs are allocated, how security decisions are made, how suppliers are governed or how applications are released.

When those mechanisms remain intact, the organisation changes the location of computing while preserving the economics and delays of the old estate. Provisioning may take minutes, but approval still takes weeks. Capacity may be elastic, but annual project funding still encourages teams to overstate demand and then abandon responsibility at handover. A new environment can therefore reproduce an old queue with a more modern invoice.

The distinction that matters is not fast technology versus slow technology. It is work organised around temporary projects versus accountability organised around enduring services.

Question	Two-speed answer	One operating-model answer
Where does innovation live?	In a separate fast team	In the end-to-end service
How is risk controlled?	Slow work passes gates; fast work seeks exceptions	Assurance is designed into each delivery path
How is legacy treated?	A Mode 1 constraint to be managed later	A portfolio dependency to be funded now

Question	Two-speed answer	One operating-model answer
What proves progress?	Prototype pace and sprint output	Time to a stable, supportable live service

The Strongest Case for Two Speeds

The serious defence of bimodal IT is not foolish. A complex transaction platform cannot be changed with the same tolerance for failure as an experimental web service. Forcing every team into one method would be theatre, and asking every legacy supplier to behave like a small product team would ignore contractual and technical reality. Different risk profiles require different controls, skills and release rhythms.

But that argument supports multiple delivery paths, not two organisational castes.

The error is to make the boundary permanent. Once labelled Mode 1, an asset attracts maintenance funding, long queues and defensive leadership; once labelled Mode 2, a team attracts attention, discretion and ambitious people. Legacy remains legacy because the model expects it to. Innovation remains experimental because production accountability lives elsewhere. The classification begins as a pragmatic description and ends as a self-fulfilling allocation of talent and capital.

A better response accepts variation while refusing separation. A low-risk customer experiment and a high-risk ledger change may travel through different controls, but they should share service ownership, architectural direction, portfolio priorities and measures of elapsed value. The people responsible for security, operations and supplier management must shape the delivery path early, not appear as gates at its end.

The Portfolio Decision We Are Avoiding

Bimodal IT is attractive because it postpones a difficult executive choice. Leaders can protect the existing estate, announce a digital capability and claim both stability and speed without resolving the dependencies between them. The unresolved cost reappears later as integration delay, duplicated data, fragile handovers and cloud expenditure that delivers little operational change.

Portfolio governance should expose that cost at the point of decision.

1. Fund the dependency with the proposition. If a new service requires an interface, identity change or supplier amendment, include it in the same investment decision rather than leaving it in a legacy backlog.
2. Measure end-to-end elapsed time. Start the clock when work is authorised and stop it only when the service is stable in production. Team velocity is useful locally; it is a poor measure of enterprise flow.
3. Assign one accountable service owner. That person must hold the trade-offs across change, run, risk and cost, even where several teams and suppliers perform the work.

4. Make the control functions part of delivery design. Security, architecture and operations should define proportionate paths before work starts, with heavier evidence only where consequence demands it.

This is less comfortable than drawing a line between old and new. It makes visible which systems constrain growth, which controls add assurance, which merely add waiting, and which senior decisions have been quietly delegated to queues.

The right question was never whether all technology should move at one speed. It was how an organisation can learn quickly without losing control of the services on which it depends. Bimodal IT named the tension, but institutionalised it. In 2017, as cloud-first ambitions move from experiments into portfolio commitments, that compromise has reached its limit. The next advantage will not come from a faster second mode. It will come from making the whole system capable of change.