

BEYOND ROLE SUBSTITUTION

Replacing the Project Manager with a Scrum Master Did Not Remove Management — It Removed Accountability

What agile delivery gained inside the team and lost across the programme boundary



Giovanni Leonardi

August 2012

Replacing the Project Manager with a Scrum Master Did Not Remove Management — It Removed Accountability

Programme · No. 171

Originally written August 2012 | Re-edited and re-rendered for the WhereBeyond Edition, 2025

“Methods do not abolish accountability; they redistribute it.”

The sprint was green and the release was late

The team had completed every sprint it had forecast for three months. The burndown charts were healthy, impediments were discussed daily and the retrospective actions were visible on the wall.

The release was eleven weeks late.

No single sprint had failed. The delay sat between the teams: a supplier interface had not been contractually agreed, the shared test environment was available only twice a month, an operations manager had not accepted the support model, and the business case still assumed benefits from a process that the business owner had quietly decided not to retire.

The Scrum Master had done the job well. The mistake was expecting that job to replace project management.

Across many organisations adopting agile delivery, the project manager has been removed with a confidence that is not matched by a redesign of accountability. The title disappears, the ceremonies arrive, and the unresolved management work is left to migrate into committees, sponsors' diaries and informal negotiation.

That is not agility. It is an organisational role vacancy disguised as methodological progress.

Two roles were treated as one because both organised work

The confusion is understandable. Traditional project managers have often become schedulers, meeting convenors and controllers of task lists. When an effective Scrum team begins to organise its own work, expose progress daily and remove internal impediments, much of that administration becomes unnecessary.

The Scrum Master protects the framework, helps the team improve and removes obstacles to its flow. These are valuable responsibilities. They are also deliberately centred on the team's ability to deliver.

Project management, at its best, addresses a wider system:

- the continuing justification for spending;
- dependencies between teams, suppliers and business functions;

- commitments made through contracts and governance;
- readiness of operations, users and support;
- risks that cannot be solved inside the team;
- the conversion of delivered capability into an owned business benefit.

Replacing one role with the other assumes that team flow and programme accountability are the same problem. They are not.

The Scrum Master asks, “What prevents this team from completing valuable work?” The programme still needs somebody to ask, “What prevents the organisation from turning that work into the promised outcome?”

What was lost did not vanish

Management work has an inconvenient quality: removing the role does not remove the obligation.

A composite £6 million change programme used three Scrum teams and two external suppliers. The project manager position was removed at mobilisation because the programme wanted to avoid command-and-control behaviour. A lead Scrum Master coordinated the ceremonies, while the product owners maintained separate backlogs.

For four months, delivery appeared strong. The teams completed 83 per cent of committed stories. Yet three obligations accumulated outside the backlogs:

- a data conversion supplier required a scope decision before fixing its price;
- operations needed six weeks to recruit and train service-desk staff;
- the finance function had not agreed how the expected £1.4 million annual saving would be measured.

Each matter belonged to everyone in principle and nobody in practice. The supplier decision waited through two steering meetings. Operational readiness was treated as a release activity rather than planned work. Benefits remained in the business case but had no owner in the delivery cadence.

When these issues finally converged, the software was substantially complete and the organisation was not ready to use it. Eleven weeks of delay added £370,000 in team and supplier cost. The programme had increased the flow of software while leaving the flow of decisions untouched.

A self-organising team can own how work is done; it cannot appoint the executive who must own why the organisation is spending the money.

The missing responsibilities did not disappear. They reappeared as escalation, delay and sponsor intervention.

The strongest argument for removing the project manager

There is a serious case against retaining the traditional role. Too many project managers translate uncertainty into detailed plans, assign work that specialists understand better, and stand between the team and the customer. Their reports create an appearance of control while decisions wait. Their success measures favour conformance to scope and date over usefulness.

From this perspective, replacing the project manager is not careless. It is necessary surgery. The product owner provides direction; the team manages delivery; the Scrum Master protects the system. Adding a project manager can restore the very hierarchy that agile working seeks to remove.

That argument is correct about bad project management. It is wrong about the work that remains.

The solution is not to reinstall a task controller above the team. It is to separate delivery facilitation from programme accountability, then assign both explicitly.

Scrum Master focus	Programme accountability focus
Team effectiveness and improvement	Continuing business justification
Impediments within or near the team	Dependencies across organisational boundaries
Integrity of the delivery framework	Integrity of commitments and governance
Sustainable sprint flow	Readiness for release and adoption
Team transparency	Executive decisions and benefit ownership

One person may sometimes carry elements of both, particularly in a small initiative. But the responsibilities should not be merged by accident, and the Scrum Master should not be judged for failing to exercise authority the role was never given.

The programme boundary is where the method thins

Scrum provides strong answers inside a team boundary. Large organisations create most of their difficulty outside it.

Budgets are approved annually. Suppliers work to contracts. Operations protects service stability. Architecture decisions span several products. Benefits depend on process, training and management action. Senior committees retain authority for risk and expenditure.

None of these facts invalidates agile delivery. They do mean that the team cannot resolve every impediment through self-organisation.

The pattern becomes especially visible when several teams share a release. Each backlog may be well ordered, yet the programme has no integrated view of:

- which cross-team dependency will constrain the release;
- which business decision has a last responsible date;
- which non-software activity must finish before value can be realised;
- which risk exceeds the authority of any product owner;
- which benefit has lost its operational sponsor.

Without that view, local transparency can coexist with programme blindness.

Accountability must become lighter, not absent

The language of project management carries baggage, and some organisations may reasonably choose another title. The title matters less than the mandate.

The needed role should not allocate daily tasks, own the team's estimates or turn sprint commitments into fixed contractual promises. It should create the conditions in which team-level agility can survive contact with organisational reality.

That means doing four things with discipline.

The original case is not a historical permission slip. Costs, benefits and assumptions should be tested as evidence emerges. If the expected process saving no longer has an owner, the programme must confront that before more software is built.

A programme plan should show the decisions and dependencies that sit outside the teams, with owners and useful dates. It should not duplicate the backlog at a higher level.

Training, data, support, controls, supplier mobilisation and business change are part of delivery even when they do not fit naturally into a development team's backlog. Someone must see the whole outcome.

An impediment is only removable if the person receiving it has the authority to act. The programme role connects team evidence to sponsors, commercial owners and operational leaders who can make the necessary choice.

This is management stripped of task control and returned to accountability.

The question organisations avoided

The debate has often been framed as project manager versus Scrum Master, old versus new, control versus self-organisation. That binary is emotionally satisfying and operationally weak.

A good Scrum Master should replace much of what a poor project manager used to do. The team should no longer need somebody to distribute tasks, police updates or manufacture progress reports.

But the organisation still needs a named answer to a harder question: who is accountable when every team is performing and the programme is still not capable of delivering the outcome?

If the answer is “the sponsor,” then the sponsor needs the time, evidence and programme discipline to perform that role. If it is the product owner, that person needs authority over suppliers, operations, funding and benefits, not merely the backlog. If it is a programme manager, the mandate must protect team autonomy while owning cross-boundary decisions.

What matters is that the answer exists before the delay does.

The lesson from the delivery side is not that Scrum Masters are inadequate or that project managers must be restored unchanged. It is that methods do not abolish accountability;

they redistribute it.

When organisations removed the project manager without redesigning that distribution, they did not become less managed. They became managed later, through escalation, recovery plans and executive intervention.

The role transition succeeds only when the team gains autonomy and the programme retains an owner for the obligations that autonomy cannot reach.