



BEYOND AGILE THEATRE

Agile Was a Management Critique Before It Was a Method — And Management Won

On the Corporate Capture of a Developer Rebellion

Giovanni Leonardi

April 2015

Agile Was a Management Critique Before It Was a Method — And Management Won

Transformation · No. 192

Originally written April 2015 | Re-edited and re-rendered for the WhereBeyond Edition, 2025

"We have built a management-controlled programme to implement a philosophy that argued against management control."

1. The Training Room and the Org Chart

I sat recently through a two-day agile training session for a financial services organisation that had just committed to an enterprise-wide transformation. The trainer was good — energetic, experienced, fluent in the manifesto's language of empowerment and self-organisation. Forty managers took diligent notes. They practised estimation with playing cards. They built towers out of paper cups to learn about iteration.

On the morning of the second day, during a break, I asked one of the programme directors whether the reorganisation would change the approval process for budget reallocation across the portfolio. She looked at me as though I had asked whether the building would be relocated. "No," she said. "That's governance. This is delivery."

That sentence — that's governance, this is delivery — captures, with an economy the manifesto's authors might have envied, exactly what has happened to agile as it crossed from software teams into the wider enterprise. The organisation had adopted the method. It had neutralised the critique.

2. What the Manifesto Actually Said

It is worth going back to the source document, because what is striking about the Agile Manifesto fourteen years on is not how radical it was, but how modest. Four value statements. Twelve principles. No methodology, no certification programme, no scaling framework. The seventeen signatories at Snowbird in February 2001 were not proposing a new process. They were proposing that the existing processes were built on a mistaken model of how knowledge work actually gets done.

The manifesto's core claim was not about velocity or throughput. It was about people. Individuals and interactions over processes and tools. Customer collaboration over contract negotiation. Responding to change over following a plan. Read without the delivery-methodology lens that has since been bolted onto it, the manifesto is a document about trust: trust that the people closest to the work understand the work

better than the people managing them, and that rigid processes exist not because they produce better outcomes but because they make management feel in control.

This was, in other words, a critique of management — one written by practitioners who had spent years watching talented people constrained by stage-gate processes, waterfall documentation requirements, and the presumption that planning could substitute for learning. It joined a tradition that runs from Deming through Ohno to Senge: the argument that the industrial management model, built for repeatable physical production, damages knowledge work by design.

That the manifesto took root first in software development is no accident. Software was where the gap between the management model and the work was most visible, because software development is irreducibly a learning process — you cannot fully specify what you are building before you build it, and the attempt to do so produces bureaucracy, not clarity.

3. The Crossing

For a decade, agile remained largely within software teams. Scrum spread. Extreme Programming's technical practices gave teams real tools. Kanban offered a gentler on-ramp. The results, where teams were genuinely empowered, were often remarkable — not because the ceremonies were magic, but because telling skilled people they were trusted to organise their own work, and then actually trusting them, produced the predictable effect of skilled people organising their work well.

The crossing happened somewhere around 2010 to 2012. Large organisations — banks, insurers, government departments, telecoms — began to notice that their agile teams consistently outperformed their waterfall programmes, and they drew the reasonable but catastrophic conclusion that agile should be applied everywhere.

The problem was structural. Agile as practised in a small, empowered team is an expression of a set of values. It works because the team has authority commensurate with its accountability: it can change direction, reprioritise, simplify scope, and talk directly to users. The enterprise, by contrast, runs on precisely the mechanisms the manifesto challenged — centralised planning, delegated execution, phase-gated funding, and separation of decision-making from work.

To adopt agile at scale, organisations had two options. They could reorganise themselves around the values the manifesto articulated — distributing authority, collapsing approval chains, funding outcomes rather than projects, and accepting the discomfort that genuine empowerment produces in a management culture built on control. Or they could adopt the ceremonies and leave the power structure untouched.

We know which option won.

4. The Scaling Machine

The market delivered what the market was asked for. By 2013, scaling frameworks had arrived to fill the gap between agile's values and the enterprise's appetite for method. The Scaled Agile Framework — SAFe — is the most prominent, and it is instructive precisely because it is so competently constructed.

SAFe takes the manifesto's vocabulary and maps it onto the existing enterprise structure with remarkable fidelity. Teams do sprints. Product owners prioritise backlogs. There are iterations and increments and releases. The language is agile throughout. But the architecture is not. SAFe introduces programme-level and portfolio-level constructs — Agile Release Trains, Solution Trains, portfolio Kanban — that restore the hierarchy the manifesto dissolved. Decision authority flows downward. Information flows upward. The team "self-organises" within boundaries set by people who do not do the work.

The genius of SAFe — and I use the word advisedly, because what it has accomplished commercially is extraordinary — is that it gives the enterprise agile's appearance while preserving management's control. It is the answer to a question the manifesto never asked: how do we make agile safe for the org chart?

I have watched a programme of several hundred people operate under SAFe, and the most telling feature was not any ceremony or artefact — it was the quarterly planning event. Two hundred people in a room for three days, producing a programme-level plan that would then be decomposed into team-level commitments. The exercise was meticulous. It was also a waterfall planning event conducted in agile vocabulary. The teams did not emerge from that room empowered; they emerged with their next quarter's work prescribed at a level of detail that the manifesto's authors would have recognised instantly as a project plan.

None of this means SAFe is useless. In the organisations I have worked with, it has often provided genuine value — not by delivering agility, but by providing a common cadence, a shared vocabulary, and a mechanism for cross-team coordination that is better than what preceded it. The mistake is not in using the framework. The mistake is in believing that adopting it constitutes an agile transformation.

5. What We Adopted Instead of What We Were Told

The pattern is worth naming precisely, because it recurs beyond agile and beyond technology. An idea emerges from practice. It challenges the prevailing management model. The management model adopts the idea's language, its ceremonies, its visible artefacts — and discards the critique. What remains is a process, shorn of the values that gave it meaning.

Consider what the enterprise typically adopts when it "goes agile":

- The stand-up survives, but as a status report to the Scrum Master rather than a coordination mechanism owned by the team.
- The sprint survives, but with scope fixed by people outside the team who treat the sprint commitment as a delivery contract.

- The backlog survives, but populated from above with requirements that leave the team no room for discovery.
- Velocity survives, but as a productivity metric reported upward — precisely the surveillance use the manifesto's principles warned against.
- The retrospective survives, but its outputs disappear into an actions log that no one with authority ever reads.

Each of these is a ceremony detached from its purpose. The stand-up was meant to make the team's work visible to itself. The sprint was meant to create a protected space for focus. Velocity was meant to help the team calibrate its own capacity. The retrospective was meant to be a mechanism of genuine self-correction.

"We took the rituals and left the religion."

6. The Honest Objection

There is a serious counter-argument, and it deserves better than a caricature. The manifesto was written by and for small, co-located software teams. It said nothing about how to coordinate two hundred people, or how to align dozens of teams with an enterprise strategy, or how to govern a portfolio of programmes each of which depends on shared platforms and services. These are real problems, and the manifesto's silence on them is not a virtue — it is a gap.

The scaling frameworks exist because organisations needed answers the manifesto did not provide. To dismiss SAFe or LeSS as betrayals of agile purity is to ignore the genuine organisational challenge they address: how do you maintain coherence across many teams without reimposing command and control?

This is fair. And yet it concedes too much, because it accepts the premise that the only way to achieve coherence is through hierarchical coordination. The manifesto's implicit argument — that trust, transparency, and direct communication scale further than we believe — has never been seriously tested in most of the organisations now running scaling frameworks. We defaulted to hierarchy not because we tried the alternative and it failed, but because hierarchy is what we know. The scaling frameworks are answers to the question the enterprise was comfortable asking, not the question the manifesto was actually posing.

7. The Temperature of the Room

What troubles me — and I write this as someone who has used agile ideas to rescue genuinely struggling programmes, not as a purist — is the sheer momentum of the adoption. The agile consulting market is now a multi-billion-dollar industry. Certification programmes have produced hundreds of thousands of Scrum Masters and SAFe practitioners. Organisations speak of agile transformations the way they once

spoke of ERP implementations — as multi-year, enterprise-wide programmes with dedicated change teams, governance structures, and reporting dashboards.

The irony is structural. We have built a management-controlled programme to implement a philosophy that argued against management control. We have created a certification industry for a movement that began by challenging the value of prescribed process. We measure the progress of agile transformation with exactly the metrics — milestone tracking, percentage completion, adoption scorecards — that agile was designed to make unnecessary.

And the human cost is real. I have sat with teams who feel the gap between the promise and the reality. They were told they would be empowered. They attend stand-ups where they are interrogated. They were told they would self-organise. They receive their priorities quarterly from a planning event they cannot influence. They were told the retrospective was their mechanism for change. They watch their concerns acknowledged and filed.

The real agile transformation has nothing to do with frameworks. It is a decision about power — who holds it, and whether those who hold it are willing to share it with the people who do the work.

8. What Might Still Be Saved

None of this is inevitable, and I have seen enough exceptions to believe the manifesto's insight is not dead — only dormant. In organisations where senior leaders genuinely redistribute authority rather than merely relabelling it, agile still delivers what it promised. I have worked with programmes where the product owner truly owned the product, where teams had genuine authority over their technical decisions, where the retrospective produced changes that were visible the following week. These programmes share a common feature: someone with organisational power chose to give some of it away.

The manifesto was, at bottom, an argument for that redistribution. Everything else — the sprints, the stand-ups, the backlogs — was scaffolding. We are now fourteen years past Snowbird, and the scaffolding has become the building. The question practitioners face is whether the original insight — that knowledge work is done best by trusted, empowered people working in small groups with direct access to the problem — can survive its own success.

The manifesto won the argument. It lost the implementation.

What remains is the oldest challenge in the management of complex work: whether organisations can tolerate the discomfort of distributing power, or whether they will always, in the end, recapture it — wearing whatever language the moment provides.