Case study

One program, many companies

Running a multi-site advanced-node development program as a single, integrated team

Engagement series.

How lateralworks acted as the technical program manager that pulled a research institute, a foundry partner, equipment suppliers, and a corporate sponsor into one empowered program — and told the truth when the program turned hard.

Prepared by
lateralworks
FTTM engagement

Sector
Advanced-node semiconductors
Consumer product device

Online
lateralworks.com
Case study series

Table of contents

One program, many companies

Core thesis. When a program depends on many companies, the work is not coordination — it is integration: building one team, one schedule, and one set of decisions out of organizations that arrive as strangers. The integrator's highest-value output is an early, honest read of the two things that actually bind the schedule: the maturity of what you inherit and the capability of the partners you depend on.

Overview

Abstract

A large, Silicon Valley–headquartered technology company set out to build a new advanced-node device technology. It was one building block inside a billion-dollar development program that spanned dozens of major technology inventions. In plain terms, the program had to take a device that worked only in a laboratory and make it run on a factory line the company itself controlled — fast enough to feed decisions on a flagship consumer product, and on a path to volume manufacturing for years of product generations after that.

The work was spread across many sites on three continents, it ran straight through the COVID-19 lockdowns, and it depended on companies that had never operated as one team: a European research institute, a foundry partner, equipment suppliers, a separate driver-chip program, and the sponsor's own functions.

lateralworks did not advise from the sidelines. We were brought in to run the program — to act as the technical program manager that pulled every company into a single integrated effort. Over three years we set up the program, built a co-development environment with the foundry partner, retrofitted a fab in the middle of lockdown, installed and qualified over one hundred process tools, qualified the line, and transferred the process out of the research institute.

Two hard programs were fused into one: a facilities and fab build, and an advanced process-development program that needed major technical breakthroughs on a short clock. The fab and the tools were the easier part. Getting them installed on time, and bringing the line up to world-class operating standards, was only the foundation. The hard part was making the technology perform to its targets. The client scoped this as a two-year project. We assessed it at five. It took five, exactly as we said — and without the planning and orchestration that held it together, our estimate is that it would have taken eight.

This is also an honest account. The program met limits no schedule could erase: a process that left research far less mature than anyone admitted, a foundry partner that could not change at the pace the program demanded, and the human cost of carrying that load. We name those problems plainly, because the value lateralworks delivered was not only the build — it was the early, candid diagnosis that gave the sponsor the truth in time to act on it.

The engagement at a glance

Role Technical program manager — we ran the program, not just advised it	**Schedule call** Client scoped two years; we assessed five; it took five. Without the program structure, an estimated eight
Footprint Multiple sites across three continents, integrated into one program	**Build (the foundation)** Brownfield fab retrofit through lockdown; over one hundred tools installed and qualified; line brought to world-class standards
The hard part An advanced process-development program needing major breakthroughs on a short clock, matured from a research result on a controlled line	**What we told them** An early, written diagnosis of the binding constraints, in time to act

01

Section one

The challenge

The sponsor had a research result and a deadline. The device worked in a European research institute, on a prototype line, at the maturity you would expect from a proof of concept: an unfinished flow, incomplete recipes, and tool requirements that were still moving. The business needed that same technology running on a development line it controlled, fast enough to feed product decisions and on a path to manufacturing well beyond the first product.

Section one

A research result, a deadline, and many sites

The gap between a research result and a controlled development line is where most advanced-technology programs lose their schedule. The sponsor faced that gap in its hardest form. The work touched facilities, tools, and process development at an advanced node all at once, and it was distributed across many sites: a Silicon Valley headquarters, a European research site, a European foundry site, an Asian equipment-supplier base, and the teams building the driver chip and the product itself.

Two hard programs, fused into one

This was never one project. It was a facilities and fab build fused with an advanced process-development program, and the two ran on very different kinds of difficulty. Standing up the fab and the tools — specifying them, ordering them, installing and qualifying them, and bringing the line up to world-class operating standards — was the easier part. Hard to schedule, capital-intensive, unforgiving on lead times, but ultimately a known kind of problem. It was the foundation, not the summit.

The summit was the technology. The program needed an immature process to make major leaps in performance, on a short clock, to hit specified product targets — the kind of work where the breakthrough either comes or it does not, and no amount of project management manufactures it. That asymmetry defined the program. Many projects were combined into one, and the binding risk lived in the hardest of them. Careful planning, orchestration, and execution management could not invent the breakthroughs, but they could make sure everything else was ready the moment a breakthrough landed — and that nothing else was the reason the program slipped.

An organization built for silos, not for this

The organization that had to deliver this was not built for it. Each building block reported vertically, up its own chain and back down. There was no single owner of the integrated outcome, no process architect tying the technology blocks together, and no integrated schedule that could show how a delay in one place moved the finish line everywhere else. Teams were not dedicated, not co-located, and not empowered to decide.

Silos produce a predictable result. Decisions climb the hierarchy and crawl back down. Integration happens late. The system-level problems surface near the end, exactly when there is no time left to fix them. Each silo optimizes its own corner — requirements, performance, cost, size — with no view of the impact on the whole.

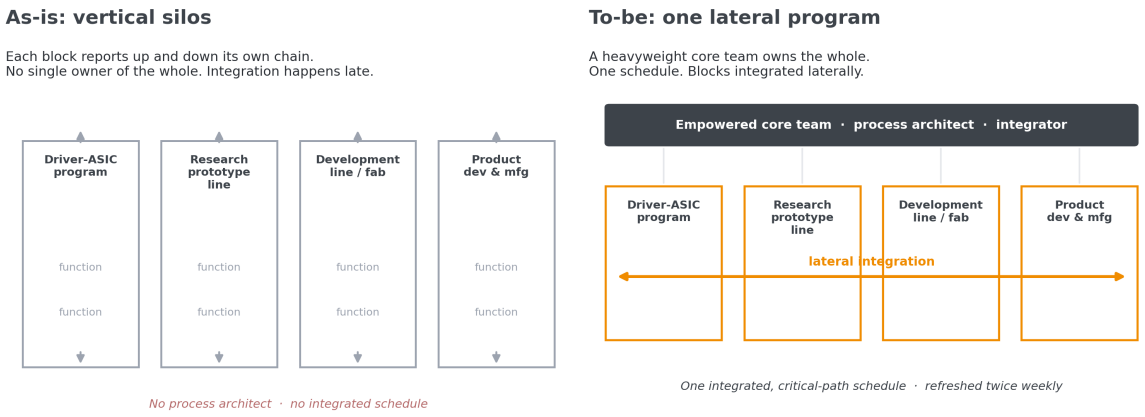


Figure 1. The starting organization reported vertically with little lateral integration. lateralworks added the connective structure: an empowered core team and the architect and integrator roles that tie the building blocks into one program.

And then the schedule got harder

The program had to retrofit an existing fab — a brownfield site, not a clean sheet — lay it out for development and later production, run the hook-up design and bid packages, and install and qualify over one hundred tools. It had to do all of this through the COVID-19 lockdowns, with equipment suppliers, travel, and labor constrained at the same time.

02

Section two

Run it, do not advise it

The sponsor did not ask for recommendations. It asked lateralworks to take the program and make it come together on time.

Section two

Own the schedule, own the bad news

That distinction shaped everything. A consultant who advises hands over a deck and waits. A technical program manager who runs the program owns the schedule, drives the decisions, and stands in front of the sponsor when the news is bad. lateralworks took the second role. We set up the core team, ran the integrated schedule, drove the critical path, and managed the interfaces between every company in the program.

A heavyweight team, fully empowered

We built that core team as a heavyweight team in the precise sense the term carries in the product-development literature: a single team, led by a manager with real authority over the program, staffed by people dedicated to it full time and accountable for the whole outcome rather than their own function. The team was chartered to drive the program schedule, integrate every function, and make the trade-offs needed to hit the product targets on the revised timeline. It reported by exception, not by permission. A steering committee existed to break cross-functional ties, not to run the work.

One team out of many companies

The aim was simple to state and hard to do. Pull the research institute, the foundry partner, the facilities and industrial-engineering partner, the equipment suppliers, the driver-chip program, and the sponsor’s own functions into one program, with one schedule and one set of priorities. Close enough that, in a meeting, you could not tell which company a person came from.

03

Section three

The lateral program architecture

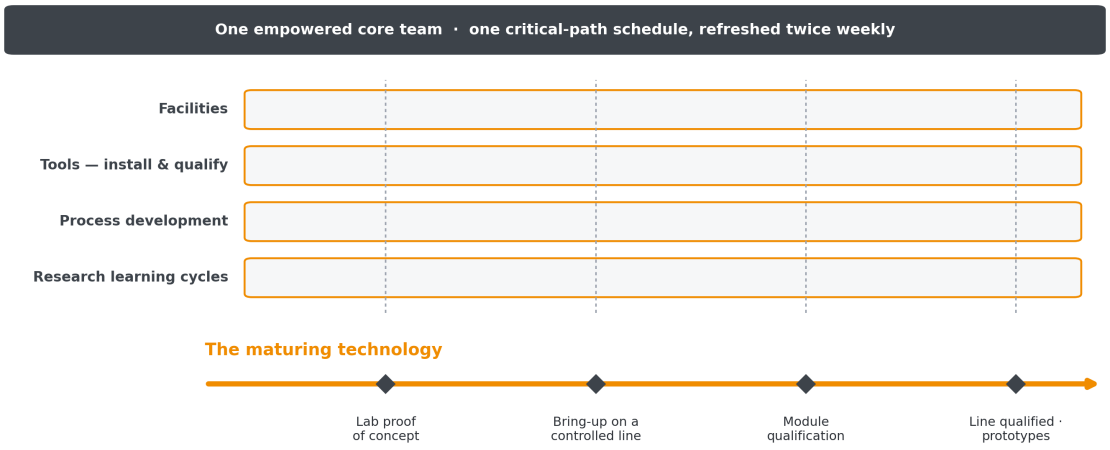
A multi-site program does not fail for lack of effort. It fails at the seams — the interfaces between organizations, where work is handed off and ownership goes vague. Managing those seams was the job.

Section three

One schedule, the hidden path, and lateral glue

From functional silos to a flow built around the technology

The starting organization was functional. Facilities ran its project, tools ran theirs, process development ran a third, and the research learning cycles ran a fourth — each on its own clock, reporting up its own chain. We rebuilt it as a single lateral process flow, organized around the one thing that actually mattered: the maturing technology. Facilities, tool install and qualification, process development, and the research learning cycles stopped being four projects and became four streams of one flow, synchronized to the state of the device as it moved from a lab result toward a qualified line. The integrated schedule kept them in step, so that each stream delivered exactly when the technology was ready to use it — no sooner, building idle capacity, and no later, stalling the program.



Functionally organized projects became four streams of one lateral flow, synchronized to the state of the device as it matured.

Figure 2. We re-cast four functionally separate projects — facilities, tools, process development, and research learning cycles — as four streams of one lateral flow, synchronized by the integrated schedule to the maturing technology.

One integrated schedule, driven by the critical path

We built a single critical-path schedule for the whole program and made it the source of truth. Every tool, every transfer step, every process and product qualification lived in one model. A working tool list — names, suppliers, costs, dependencies, lead times — fed the schedule and took its dates back from it, so the plan and the tracking never drifted apart. We refreshed the schedule twice a week. That cadence is not administrative habit; it is how a fast program keeps the plan honest against reality and catches a slip while there is still time to act on it.

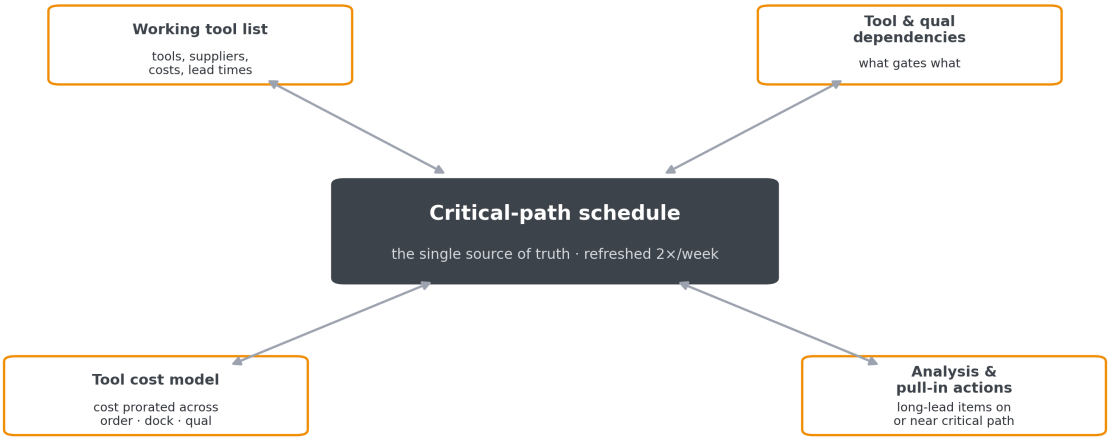


Figure 3. The critical-path schedule was the driver. The working tool list, dependency map, cost model, and pull-in analysis all fed from it and back into it, refreshed twice weekly.

Managing the critical path you cannot see

The obvious critical path is rarely the one that derails a program. It is the second or third — the one hiding in plain sight — that does the damage. Here, the hidden path ran through facilities: until the fab was retrofitted and tools could go in, nothing downstream could start. We sequenced the program around that constraint, ordered long-lead tools at risk where the exposure was small, and treated schedule buffer as a managed resource to be spent deliberately rather than slack to be quietly consumed.

Figure 4. The acceleration logic is a chain: securing the site lets you order tools, which lets you qualify them, transfer the process, and learn faster. Facilities was the hidden constraint that gated all of it.

Lateral integration of the building blocks

Vertical reporting was already in place. What the program lacked was lateral integration — the connective tissue between technology building blocks, and between technology and product. We added the roles that supply it. An integrator looked across the whole and tied it together. A process architect made the system-level trade-offs so that no single block was optimized at the expense of the program. A chief engineer owned the final technical calls. Those three roles turn a stack of silos into a lateral system.

Decisions, not status

Speed in a program like this comes from decision quality, not from meeting frequency. We pushed decision authority down to the core team and kept the sponsor’s role to provisioning, fast tie-breaking, and exception handling. When outside groups tried to pull the team into endless justification, we insulated the team on purpose, routing external communication through the program lead so the people doing the work could keep doing it.

The integration test

One team, many logos

"In a meeting you could not tell which company a person came from.

lateralworks engagement

Advanced-node development program

04

Section four

Execution under lockdown

The plan met the world in early 2020.

Section four

Building a line while the world shut down

A co-development environment with the foundry

We set up a co-development environment with the sponsor's primary foundry partner — shared tooling, shared process ownership, and a joint qualification path — so the device could be brought up and matured on a line the sponsor could actually steer, rather than handed over the wall. This was a deliberate departure from a copy-exact transfer. The process arriving from research was not finished; treating it as a clean transfer-and-ramp would have locked in an immature flow. We structured the work as a process bring-up with continued development, which let the line keep maturing the process past where research had left it.

Retrofitting a fab in the middle of a lockdown

We retrofitted the brownfield fab while the pandemic shut down travel and strained every supply chain. We laid out the line for development and for later production, ran the hook-up design and bid packages, and installed and qualified over one hundred process tools. When equipment suppliers and procurement were constrained, the program sent a team to finalize critical equipment agreements in person and pulled long-lead orders forward where the schedule justified the risk. Tool by tool, the line came up.

Transferring the process and qualifying the line

We transferred the process out of the research institute one super-module at a time — the large process blocks for substrate preparation, wafer bonding, planarization, device patterning, and back-end integration. For each, we installed and qualified the sponsor's tools, moved the recipes onto them, and proved the module on a full-flow demonstrator before starting the next. The strategy was to take the highest-risk modules first, so the program could fail early, learn fast, and pull problems forward instead of meeting them at the end.

05

Section five

The honest diagnosis

A program manager who only reports good news is worthless at the moment the news turns. By the spring of 2022 it had turned, and lateralworks said so — directly, in writing, to the sponsor's leadership.

Section five

Three facts that no schedule could erase

Three facts had collided. First, the transferred process was far less mature than the program had been led to believe. Bringing it to a baseline was a twelve-to-eighteen-month effort, not a six-month one; the target had been wrong from the start, and the expectations stacked on top of it were higher than the stage of development could support. Second, the foundry partner could not change at the pace the program demanded. A relationship that began as transactional had hardened, and the sponsor held less control over the outcome than its investment implied. Third, the people carrying the load were running out of road — worn down by the daily fight, short on support, and at real risk of leaving at the worst possible moment.

Put it all on the table

We did not soften it. We recommended radical transparency with the sponsor's leadership, a realistic re-baselining of process maturity, decisive use of the contractual levers the sponsor held over the foundry, and an urgent, funded effort to hold the core team together. None of this was comfortable. All of it was true, and the sponsor heard it early enough to act — which is the entire point of putting a program manager, not a cheerleader, in the seat.

Why the model surfaced it early

This is where the lateralworks model earns its keep. The same integrated schedule and lateral structure that drove the build also surfaced the problems early and made them legible to leadership. A siloed program would have buried these failures inside functional reporting until the end. An integrated one names them while there is still time.

06

Section six

What the model delivered

Measured against scope, the program moved an extraordinary amount of work. Measured against the original product targets, it ran into limits that were structural, not managerial.

Section six

What integration bought, and what it could not

lateralworks set up and ran the program through its critical first three years. We stood up a co-development environment with the foundry partner. We retrofitted a brownfield fab through the COVID-19 lockdowns, installing and qualifying over one hundred process tools and qualifying the line. We transferred an unfinished, advanced-node process out of a research institute and matured it on a controlled development line. We built and ran a single integrated schedule across many sites and three continents, and we turned a set of vertically siloed organizations into one laterally integrated program.

We called the schedule

The clearest measure of the work is the schedule itself. The client scoped the program at two years. On our assessment of what an immature process, a brownfield fab, and a multi-company structure actually required, we put it at five. It took five — to the year. The planning and orchestration did not make the program faster than physics allowed; it made the program take the time it genuinely needed and no more. Our estimate is that without that structure, the same program would have run to eight years, lost to false starts, idle capacity, and problems found too late.

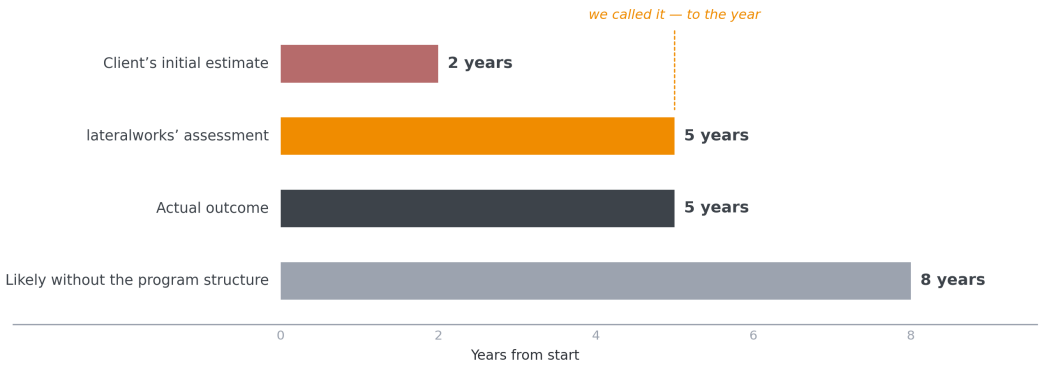


Figure 5. The client scoped two years; we assessed five; the program took five. Our estimate of the same program without the integrated structure: roughly eight.

Against the original product targets, the program met limits that were structural, not managerial: a process that needed far more development than the schedule assumed, and a foundry partner that could not move fast enough. lateralworks' contribution was twofold — the integration that made the build possible at all, and the candor that gave the sponsor an accurate picture in time to change course. Both are the job. Neither is optional.

The lesson generalizes

On a program that depends on many companies, the binding constraints are usually the maturity of what you inherit and the capability of the partners you depend on. Neither is fixed by working harder. Both are exposed early by an integrated schedule, a lateral structure, and a program manager willing to say what is true. That early, honest read is the integrator's highest-value output — worth more than any status report,

because it is the one thing a siloed program cannot produce.

07

Section seven

Best practices applied

The program ran on the lateralworks FTTM body of practice — best practices observed across hundreds of fast technology programs since 1988. The practices that shaped this engagement, and where they come from:

Section seven

The practices behind the program

A heavyweight, fully empowered core team. A single dedicated team with real authority and end-to-end accountability, reporting by exception. This is the heavyweight-team model documented by Clark and Wheelwright [1, 2], and the lateralworks standard for programs of this scale.

Lateral integration of the building blocks. Adding the integrator, process architect, and chief-engineer roles that connect technology blocks to each other and to the product. This lateral structure sits at the center of the FTTM system [9], and is consistent with Allen's findings on lateral communication in technical organizations [3].

One integrated, critical-path-driven schedule. A single source of truth, refreshed twice weekly, with a working tool list feeding it — lateralworks' refresh-planning practice in action [9].

Managing the critical path you cannot see. Sequencing the program around the hidden facilities path rather than the obvious one. A core FTTM principle: the critical path is rarely where you think it is [9].

Buffer as a managed resource. Treating schedule buffer as something to be spent deliberately, not consumed by default — the discipline Reinertsen formalizes in the economics of product-development flow [4].

Speed from decision quality, with authority pushed down. Empowered decisions at the core team, exception reporting upward, and the team insulated from interrupts. This is the FTTM decision practice, reinforced by Oncken and Wass on where the responsibility — the "monkey" — belongs [5].

Fail faster to learn faster. Taking the highest-risk modules first to pull problems forward. The FTTM acceleration logic, which trades directly on Reinertsen's economics of delay [4] and the break-even-time thinking of House and Price [6].

The right-product, right-team, right-time environment. Continuously defining the product, selecting and empowering the team, and planning the schedule as one connected system rather than three separate problems [9].

> **How to read this.** Every practice above is a lateralworks observation first, grounded in the wider product-development literature second. The engagement did not apply a theory; it applied a body of field practice that the literature happens to corroborate.

08

Section eight

The four common problems, solved

lateralworks' field research names four conditions as the hardest in technology development, and the most reliable way to sink a multi-party program. This program had all four at once.

Section eight

Four hard conditions, met at once

Most programs struggle with one of these conditions. This one carried all four simultaneously: work spread across many sites, many companies on the team, several corporate stakeholders with their own agendas, and the hardest combination of all — a technology that still had to make breakthroughs while a product was being built around it. Each condition has a known failure mode, and each has a documented lateralworks solution. We implemented all of them. The figure below maps the four problems to what we actually did on this program.

Figure 6. The four common problems of multi-party technology programs, mapped to the lateralworks solutions applied on this engagement.

The fourth condition deserves a closing word, because it was the one that ultimately bound the program. Running technology innovation and product development at the same time is the trap that no amount of effort escapes. The discipline is to isolate the breakthrough work and dedicate an insulated team to it, manage it by learning milestones rather than product-performance milestones, and keep everything else known and ready. We did that. What it cannot do is manufacture a breakthrough on demand — which is exactly why the honest read of process maturity, delivered early, mattered as much as the build.

Sources

References

[1] Clark, K. B., and Wheelwright, S. C. “Organizing and Leading ‘Heavyweight’ Development Teams.” California Management Review, vol. 34, no. 3, Spring 1992, pp. 9–28.

[2] Wheelwright, S. C., and Clark, K. B. Revolutionizing Product Development: Quantum Leaps in Speed, Efficiency, and Quality. Free Press, 1992.

[3] Allen, T. J. Managing the Flow of Technology: Technology Transfer and the Dissemination of Technological Information Within the R&D; Organization. MIT Press, 1977.

[4] Reinertsen, D. G. The Principles of Product Development Flow: Second Generation Lean Product Development. Celeritas Publishing, 2009.

[5] Oncken, W., and Wass, D. L. “Management Time: Who’s Got the Monkey?” Harvard Business Review, November–December 1974 (reprinted November–December 1999).

[6] House, C. H., and Price, R. L. “The Return Map: Tracking Product Teams.” Harvard Business Review, January–February 1991, pp. 92–101.

[7] Clark, K. B., and Fujimoto, T. Product Development Performance: Strategy, Organization, and Management in the World Auto Industry. Harvard Business School Press, 1991.

[8] Brooks, F. P. The Mythical Man-Month: Essays on Software Engineering. Anniversary edition, Addison-Wesley, 1995.

[9] lateralworks. “The FTTM Framework.” lateralworks.com/methodology, accessed 2026.

[10] lateralworks. “Internal engagement assessment.” Advanced-node development program, 2019–2022. Confidential; anonymized for publication.