Whitepaper

Right product, right time

How an honest schedule reveals whether you are building the right thing, in time to do something about it

FTTM Best Practices Series.

A lateralworks methodology paper on right-time product development. An anonymized AI-datacenter memory program shows how building an honest schedule surfaced a five-year reality hiding behind a two-year wish, and how that one truth forced the requirements and the roadmap back into alignment with the market. Connects the practice to the literature on cost of delay, the planning fallacy, strategic misrepresentation, and time-to-market economics.

Prepared by
lateralworks
FTTM methodology

Date
June 2026
Methodology paper

Online
lateralworks.com
FTTM Best Practices Series

Table of contents

Right product, right time

Core thesis. Every program carries three separate truths: the target the market sets, the schedule the team can actually deliver, and the requirements that define what the product even is. Most organizations collapse all three into a single date. The schedule is the only one of the three that produces a measurable signal every week, so an honest schedule is the first truth to break the silence. Once the team can see the real schedule trending out, it forces a harder question than "are we late." It forces "are we building the right thing at all."

Overview

Abstract

A late product and a wrong product are usually the same failure seen from two angles. A team commits to a market date, discovers far too late that the work takes much longer, and ships something that misses the window. Underneath that story is a quieter one: the requirements the team was chasing were never achievable in the time available, and often were never the requirements the customer actually needed. Both failures trace back to the same root. The organization never built a schedule honest enough to tell it the truth.

This paper makes a single claim. A truthful schedule does more than predict a ship date. It is the first instrument sensitive enough to reveal that the product itself is wrong, because the schedule is the only one of a program's three truths that moves and can be measured every week. Market timing is slow and arguable. Requirements correctness is slow and arguable. The schedule trend is a number that changes at every refresh, and a number that moves is a number a team can argue with. When the trend refuses to bend toward the committed date, it is telling the team that the plan behind the date is fiction. Acting on that signal opens the requirements question that the organization had agreed, without ever saying so, to leave closed.

The argument runs through an anonymized engagement. An engineering team spent a year developing an advanced memory device aimed at AI datacenter workloads. They ran expensive design-of-experiments cycles with their foundry partner, with learning cycles of four to eight months and no plan to connect them. They had a two-year target, set by the expected obsolescence of the technology they meant to replace, and they treated that target as a schedule. Every engineer on the program understood the performance requirements were not reachable in two years. Nobody said so. The FTTM planning process produced the program's first real schedule, which came out near five years. The schedule trend, tracked over a four-month cycle, was moving away from the target, not toward it. That single fact reopened the requirements, the roadmap, and the customer conversation the team had skipped at the start.

The paper builds on three earlier lateralworks practices, bringing pain forward, dropping the happy schedule, and knowing the gap [4], and extends them into the right-time dimension. Section one separates the three truths of a program. Section two explains why the schedule is the truth that breaks the silence first, and why the silence was rational to begin with. Section three works through the case. Section four shows how one schedule truth re-timed an entire product roadmap, and closes with a diagnostic a leader can run on a live program this week.

01

Section one

The three truths of a program

Ask a product organization when its next program will ship and you will get one date. Ask why that date, and the answers blur together. The date is when the market needs the product. The date is when the team can build it. The date is also, somehow, a statement about what the product will be. Three different questions have been answered with one number, and the organization has lost the ability to tell them apart.

Separating those questions is the first move in fast time to market. A program carries three truths, and they are not the same truth wearing different clothes.

Section one

Target, schedule, requirements

The target is what the market needs

A target is a business aspiration. It says the product has to be in the market by a certain date or the opportunity degrades. In short-lifecycle technology markets the target is rarely arbitrary. Product lifecycles are steep, global competition is intense, and entering at the right time often means being first while demand is just beginning to accelerate, which lets a company hold a higher average selling price before the inevitable price erosion sets in [21]. Aggressive targets are healthy. They create urgency, and research on goal-setting supports the idea that ambitious goals outperform comfortable ones [20]. The target is not the problem. The problem starts when the target is handed down as if it were a plan.

The schedule is what the team can do

A schedule is a different artifact entirely. It is the sequence of work, the dependencies, the resources, and the durations that get the product to market, built by the people who will do the work. A target says "we need to be in the market by September." A schedule says "here is the work that gets us there, and based on what we know today, that date is December" [22]. The gap between the two is not a morale problem or a discipline problem. It is a planning problem, and planning problems have solutions. A team that can see the gap can negotiate scope, add resource to a critical path, parallelize sequential work, or rebaseline the target. None of those moves exists if the gap is invisible.

The requirements are what the product has to be

The third truth is the quietest and the most dangerous when it is wrong. Requirements define what the product actually is: the performance, the features, the specification the team is trying to hit. They are supposed to be a prediction of what the customer will need by the time the product arrives. Often they are an assumption made at kickoff and never revisited. Wheelwright and Clark located the highest-leverage point in all of product development at the front end, where the concept and its requirements are set, and argued that the firms that "design it right the first time" hold a structural advantage in the race to market [16]. Getting the requirements wrong is more expensive than getting the schedule wrong, because a late product can still be the right product, while a finished product built to the wrong requirements is waste no matter when it ships.

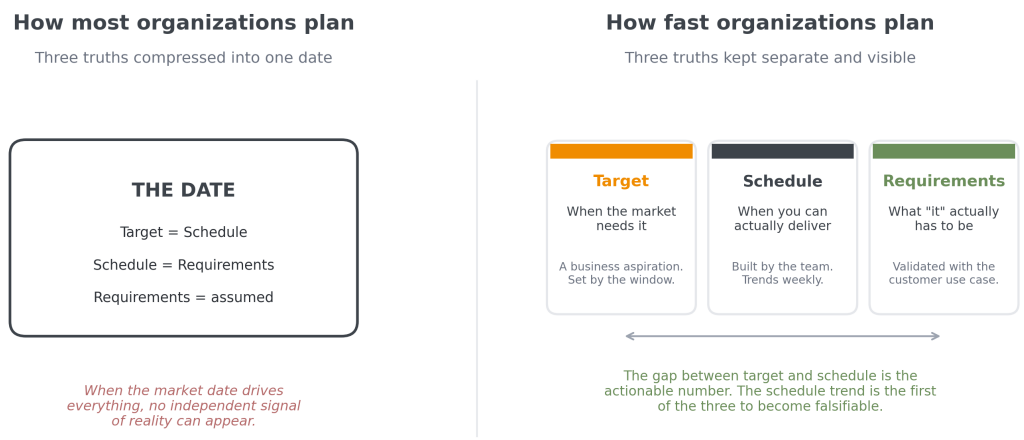

Figure 1. The three truths of a program. Most organizations compress target, schedule, and requirements into a single date, which removes any independent signal of reality. Fast organizations keep them separate. The gap between target and schedule is the actionable number, and the schedule trend is the first of the three to become falsifiable.

Figure 2. The three truths and how they relate. Each corner is one truth; each edge is the question that connects two of them. Target against schedule is the gap, which governs right time. Target against requirements is the fit, which governs right product. Requirements against schedule is feasibility, whether the spec can be built in the time available. Collapsing all three into one date erases every edge at once.

Why collapsing them is the default

Organizations do not collapse the three truths out of carelessness. They collapse them because keeping them apart is uncomfortable. A separate schedule can disagree with the target, and that disagreement has to be voiced by someone. A separate requirements truth can turn out to be wrong, and admitting that means

reopening decisions the organization thought were settled. The path of least resistance is to let the market date stand in for all three. The cost of that convenience is hidden until the program is too far along to change course cheaply. Lovallo and Kahneman described the cognitive half of this pattern as the delusions of success that lead executives to anchor on best-case scenarios and discount base rates even when the data is available [5]. The organizational half is the subject of the next section.

02

Section two

Why the schedule breaks the silence first

If three truths are in play and all three can be wrong, why does the schedule deserve pride of place? Because of a property the other two lack. The schedule produces a fresh, falsifiable number every time the team refreshes the plan. Market timing and requirements correctness change slowly and are argued in words. The schedule changes weekly and is argued in dates. A truth you can measure every week is a truth that eventually wins.

Section two

The falsifiable truth

A number that moves is a number you can argue with

Reinertsen made the case that the single most valuable quantity in product development is the cost of delay, and found that roughly 85 percent of product managers cannot say what a few months of delay would cost their program [8]. His advice was blunt: if you only quantify one thing, quantify the cost of delay [8]. Each refresh records the predicted finish date for each target milestone, and over weeks those predictions form a line. A flat line means the team is holding the milestone; a line drifting up means it is slipping. The intercept of that line with the target date tells the team how long it has to act. The analogy to a stock chart is exact. One day of price says nothing; the trend over months says something about the health of the enterprise. Most project organizations track only the latest update, which means they hold the daily prices and never draw the chart.

Why the silence was rational

It is tempting to read a team that hides a schedule gap as either dishonest or incompetent. Neither is usually true. The silence is a rational response to how organizations treat early bad news. Argyris named the underlying pattern skilled incompetence: capable people becoming expert at saying things that look responsible and produce results that are not [3]. Around schedules, the defensive routine has two rules. The first is that the gap is undiscussable. The second is that the undiscussability is itself undiscussable [3, 14]. A team member who breaks the first rule reads as difficult. One who breaks the second reads as a threat. The cost of breaking either rule lands immediately on the individual. The cost of keeping both rules in place lands months later on the program. Given that asymmetry, staying quiet is the locally sensible move. The military version is sharper. A commander who punishes scouts for reporting bad news will soon be surrounded by scouts who report only good news, and will walk straight into an ambush [22]. Engineering teams learn the same lesson and produce schedules that read as commitments and behave as forecasts. The artifact looks rigorous. The data behind it is wishful.

Optimism and misrepresentation, working together

Two distinct forces keep the happy schedule alive. The first is the planning fallacy. Buehler, Griffin, and Ross showed that people systematically underestimate how long their own tasks will take even when they have direct experience of similar tasks running long [1]. The second is strategic misrepresentation. Flyvbjerg, studying infrastructure megaprojects across decades and geographies, found cost and schedule overruns of remarkably consistent magnitude and attributed them to a mix of optimism bias and the deliberate underestimation of cost and time to get projects approved [2]. He found the deliberate component the larger of the two. In private product development the same behavior appears under gentler names: sandbagging, stretch goals, aggressive baselines. The schedule trend cuts through both because it does not care why the original date was wrong. It only reports where the work is actually going.

This is where the schedule earns its primacy. A team cannot easily measure, week by week, whether its requirements are correct; the customer is not in the room every week. It can measure, every week, whether the plan to deliver those requirements is converging or diverging. When a plan diverges hard enough for long enough, the divergence becomes an argument the requirements cannot ignore, and a five-year trend against a two-year target is that argument made undeniable.

The mechanism

Schedule first

The schedule is the only truth that moves every week. So it is the first one that can break the silence.

lateralworks

Right product, right time

In their own words

The practitioner case

"This was forced because we have a real schedule now that shows how long it takes. When there is a fake schedule they can pretend they can do it. So even the late schedule has value."

The observation that opens this paper came from a program leader watching the pattern resolve in real time. The team had spent a year defending a date everyone privately knew was impossible. What broke the standoff was not a better argument about the requirements. It was a schedule honest enough that the requirements could no longer be defended. Once the real duration was on the table, the conversation the company had been avoiding, about whether this was the right product for the right time, finally happened.

03

Section three

The case: a memory device for AI datacenters

The engagement is anonymized at the company’s request; the device, the segment, and the dynamics are real. An engineering team was developing an advanced memory device for AI datacenter workloads, a market where high-bandwidth memory roadmaps move on roughly eighteen-to-thirty-six-month cadences and a missed generation is a missed supercycle [24]. The team meant to replace a predecessor technology it expected to go obsolete in two years. That expectation set the target, and the target quietly became the plan.

Section three

A year of experiments with no plan

Expensive cycles, no schedule

The team did not start with a plan. They started experimenting. Working with their foundry partner, they ran a series of expensive design-of-experiments runs, each one a learning cycle of four to eight months. The cycles were real work and produced real learning. What they lacked was any structure connecting them to a finish date. With no schedule, there was no way to ask whether the rate of learning was fast enough to reach the performance targets inside the two-year window. The team was busy, the foundry was engaged, and the program looked active. Activity is not the same as convergence.

Everyone knew, no one said

The engineers understood the performance requirements were not reachable in two years. The amount of invention and innovation standing between the current device and the target specification was large, and large invention does not schedule like incremental engineering. Yet the team continued for a full year inside a plan that assumed the invention would simply arrive on time. This is the rational silence of section two in its natural habitat. The two-year date had been set by the predecessor's expected obsolescence, which made it feel like a fact about the world rather than a choice the company had made. Arguing with it meant arguing with the market. So the gap between the achievable and the assumed stayed wrapped in the language of aggressive-but-achievable, and the program kept its head down.

The requirements were never the customer's

The deeper flaw sat underneath the schedule. The performance targets had never been aligned with a real customer requirement for the AI datacenter segment. The team had made a set of assumptions at kickoff about what that segment would need in two years, written them into the specification, and never questioned them or checked them with a customer. The program was therefore exposed on two fronts at once. It was chasing a specification it could not reach in the time available, and it had no evidence the specification was the one the market would actually want when the product arrived. Either failure alone is serious. Together they describe a program heading toward the wrong product at the wrong time, with no instrument on board to detect either condition.

The FTTM schedule, and what it revealed

lateralworks engaged the team in the FTTM planning process to build the program's first real schedule [25]. Built participatively with the engineers doing the work, factoring the invention as learning cycles with honest durations rather than as placeholder tasks, the schedule came out near five years. Then the team tracked it. Over a four-month cycle of weekly refreshes, the predicted time to qualification did not hold and did not pull in. It trended out, refresh after refresh, away from the two-year target and toward the five-year reality. The trend, not any single estimate, was the evidence. A plan moving away from its date under honest tracking is not a plan that needs encouragement. It is a plan revealing its true length.

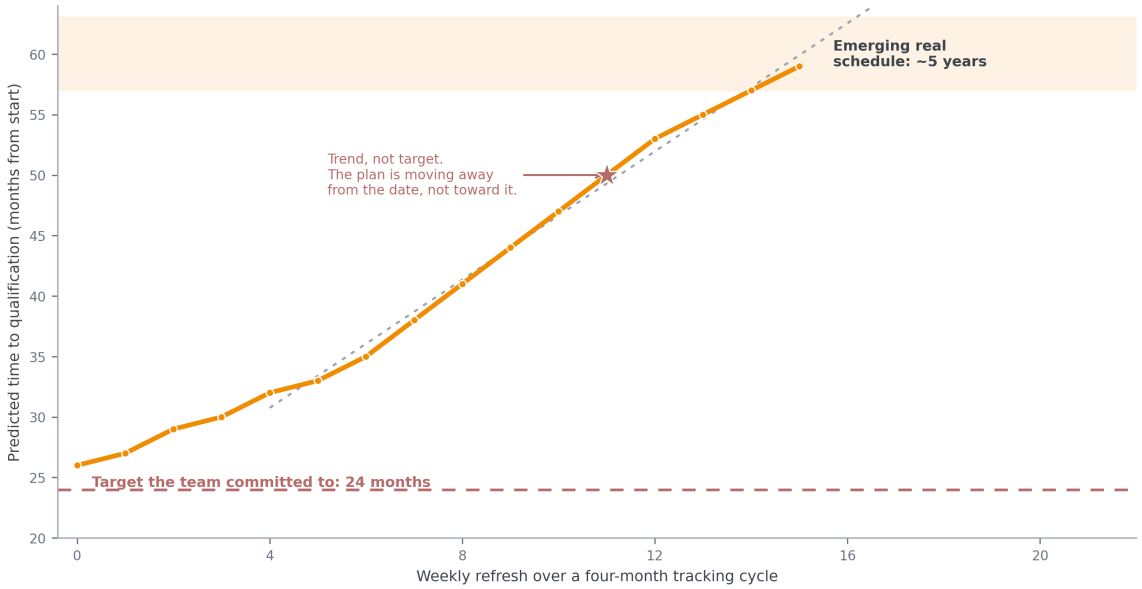

Figure 3. The case milestone, tracked over a four-month cycle. The predicted time to qualification trends away from the committed two-year target rather than toward it. The extrapolated trend intercepts a band near five years. The trend, not any single estimate, is what told the team the two-year plan was fiction.

Confronting the reality

With the real schedule in hand, the team took the reality to the CEO. The conversation that had been impossible for a year became straightforward once the date was no longer arguable. The CEO acknowledged that the requirements could be relaxed, or detuned, so the device could actually be developed inside a sensible window. The schedule drove that realization. Without the schedule truth, the company would never have revisited the requirements, and it would have continued toward a product that was both late and wrong: late because the real work took five years, and wrong because the specification had never been validated against a customer need. The order of operations matters. The schedule did not decide anything. It made the decision possible by replacing a private suspicion with a number the organization could no longer pretend away.

04

Section four

Right product, right time, right roadmap

The schedule opened the requirements question. It did not answer it. Answering it took the work the team should have done at kickoff, now done with the benefit of a year of real learning and an honest sense of how long real progress takes. The result re-timed not one product but the company’s roadmap.

Section four

Detune, validate, re-time

Scaled requirements on a real schedule

The team aligned a scaled-down requirements set with a more realistic three-year schedule. The detuning was not a retreat. It was the first time the specification had been set against an achievable plan rather than against a date borrowed from the predecessor's obsolescence. A three-year device the company could actually build and sell beat a two-year device that existed only on a slide. This is the participatory step that imposed schedules never reach. When the people doing the work own the plan, they hunt for the efficiencies and the scope trades that pull time out of it. When the plan is handed to them, they execute someone else's fiction and stop volunteering improvements [22].

Validating against the customer

The new requirements set was validated with the customer as an accurate prediction of needs in the two-to-three-year window the program now targeted. This is the step the original program skipped entirely, and it converts the exercise from guesswork into evidence. House and Price built their return map around exactly this discipline of estimating and re-estimating against market reality, and reported the McKinsey finding that companies shipping six months late lose on the order of a third of a product's lifecycle profit, against roughly a twentieth of that for a 50 percent development overspend [23]. Time is the expensive variable. A validated specification on a real schedule protects the variable that matters most.

Extending the predecessor, re-timing the roadmap

The reset reached past the single product. The predecessor technology's life had also been mapped to assumption rather than to a validated customer requirement, and once the team looked honestly, it could extend that life by another year. That extension covered the gap the detuned product's longer-but-real schedule created, and it kept revenue flowing from the installed base while the new device matured. The company ended with right-product, right-time success across the roadmap rather than a single rescued program. The same schedule truth that exposed the two-year fiction also exposed the soft assumption under the predecessor's end-of-life, and re-timed both.

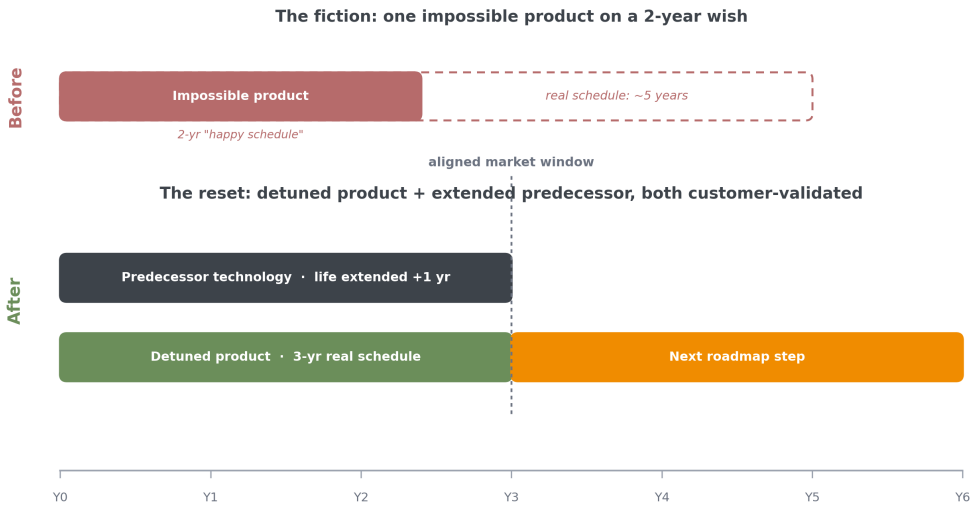

Figure 4. Re-timing the roadmap. Before: one impossible product on a two-year wish, with a five-year reality hidden behind it. After: a detuned product on its real three-year schedule, the predecessor's life extended a year to cover the gap, and both validated against the customer. One schedule truth re-timed the whole roadmap.

What the host had to provide

None of this survives without a host environment that rewards the early truth. The CEO had to be willing to hear that a date he had likely repeated to a board was wrong, and to treat that as useful information rather than as a failure of commitment. The first question on a program sets what the team will produce. A host who asks "how confident are you" gets confidence. A host who asks "what is the gap between your real finish date and the target, and what would it take to close it" gets a gap number [4]. The fast version of this organization separates targets from schedules as a standing habit, refreshes at a cadence that keeps the trend legible, and makes naming a problem cheaper than concealing one. The practices in this paper are the visible artifacts. The host is the cause.

The honest qualifier belongs here, because it is what separates a method from a slogan. A truthful schedule is necessary but not sufficient. It reveals that the product is wrong in time to do something about it. It does not, by itself, decide what the right product is. That still takes customer validation, scope judgment, and a leadership team willing to act on what the schedule surfaced. What the schedule guarantees is that the conversation happens while the cheap options are still on the table, instead of at the milestone review when they are gone.

A

Appendix
The diagnostic

A leader does not need a full engagement to find out whether a live program is in the trap this paper describes. Three questions, asked of any active program, separate a real plan from a target wearing a schedule’s clothes.

Appendix A

Three questions for a live program

One. Is there a real schedule, or a target in disguise?

Ask how the finish date was built. If it traces back to a sequence of work, dependencies, durations, and learning cycles owned by the people doing the work, it is a schedule. If it traces back to a market date, a board commitment, or the obsolescence of something else, it is a target wearing a schedule's clothes, and the program has no instrument for detecting either lateness or wrongness.

Two. Over the last four refreshes, is the trend pulling in or trending out?

Pull the predicted finish date for the top milestones over the last four refreshes. A flat or pulling-in trend means the team is converging on its plan. A trend moving out, refresh after refresh, means the real schedule is longer than the committed one and is still emerging. A program with no refresh history at all is the most exposed of the three, because it has chosen not to look.

Three. Have the requirements been validated, or assumed?

Ask when the requirements were last checked against a real customer use case for the segment the product will actually serve when it ships. If the answer is a validation exercise with a date, the requirements truth is being managed. If the answer is a set of assumptions made at kickoff and never revisited, the program may be optimizing toward the wrong specification regardless of how its schedule trends.

> **Reading the answers.** A target in disguise, a trend moving out, and assumed requirements together describe a program building the wrong product for the wrong time, with no instrument on board to detect either condition. The first move is not a rebaseline. It is a real schedule, tracked weekly for six weeks, so the trend can surface reality on its own. Once the trend is visible, the requirements conversation becomes possible on data rather than on opinion.

Sources

References

[1] Buehler, R., Griffin, D., and Ross, M. "Exploring the planning fallacy: Why people underestimate their task completion times." Journal of Personality and Social Psychology, 67(3), 1994, pp. 366-381.

[2] Flyvbjerg, B. "Curbing optimism bias and strategic misrepresentation in planning: Reference class forecasting in practice." European Planning Studies, 16(1), 2008, pp. 3-21.

[3] Argyris, C. "Skilled incompetence." Harvard Business Review, September-October 1986, pp. 74-79.

[4] lateralworks. "Bring pain forward: happy schedules, the gap, and the discipline of finding reality early." FTTM essentials series, 2026.

[5] Lovallo, D., and Kahneman, D. "Delusions of success: How optimism undermines executives' decisions." Harvard Business Review, July 2003, pp. 56-63.

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

[14] Argyris, C. Overcoming Organizational Defenses: Facilitating Organizational Learning. Prentice Hall, 1990.

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

[20] Locke, E. A., and Latham, G. P. "Building a practically useful theory of goal setting and task motivation: A 35-year odyssey." American Psychologist, 57(9), 2002, pp. 705-717.

[21] lateralworks. "Right-time ensures financial return from new products." Ideas archive, 2019. https://lateralworks.com/ideas/2019/6/21/right-time-ensures-financial-return-from-new-products

[22] lateralworks. "The scheduling debate." Working paper, 2025.

[23] House, C. H., and Price, R. L. "The Return Map: Tracking Product Teams." Harvard Business Review, January-February 1991, pp. 92-101.

[24] JEDEC Solid State Technology Association. "High Bandwidth Memory (HBM4) DRAM, JESD270-4." Standard, 2024; with industry roadmap reporting on HBM4 and HBM4E, 2025-2026.

[25] lateralworks. FTTM Self-Paced Tutorial and Reference Guide, section on participatory scheduling and wigglechart trend analysis, 2019.

[26] lateralworks. Internal assessment database. Engagement data across client programs, 2015-2026.