Briefing

Outside the hierarchy

An autonomous team, chartered outside the corporate hierarchy, with one mission and a date: the fastest known way to organize new product development.

FTTM best practices.

This briefing describes the autonomous program: what it is, why it is fast, how the Navy operating model works, the role architecture that makes it real, where it has run before, its known risks, and what it asks of the CEO to put in place.

Prepared by

lateralworks
Neal Mitchell

Date

September 2026
Executive briefing

Series

FTTM best practices
lateralworks.com

Table of contents

Outside the hierarchy

Core thesis. A program moves at the speed of its slowest decision path. Move it outside the hierarchy, give one leader the resources and the written authority to act, and hold the mission constant. Everything else in this briefing is mechanics.

For the CEO

Executive summary

The hardest development programs are no longer single products. Companies now commit to ecosystems and platforms: a dozen concurrent projects across hardware, firmware, software, and often an offshore ODM, all converging on one outcome with a date on it. The structure carrying that load is usually the one built for the running business: released products, sustaining, and a handful of projects coordinated through the functions. For a dated mission, that structure is slow, and it does not work.

Measured against the standard research on development organizations, most established companies sit between a functional hierarchy and a lightweight team model [1]. Work lives in the functions. Coordinators report progress without the authority to change it. It is not clear who runs each project, and the program roles that exist are administrative, without overall control. Every decision of consequence travels up a function, across the organization, and back down. That path length, multiplied across a dozen projects, is the program's real schedule risk.

The answer is to move the program outside the corporate hierarchy as an autonomous organization: a small Steering Arm above it, one Program Director running it with every project resource reporting in, Project Triads running each project, and cross-functional core teams staffed on loan from the functions. We call it the Navy model. A sailor reports to the captain of the ship for the duration of the deployment. When the ship returns to home port, the sailor returns to the base commander. Everyone on the program deploys the same way, and everyone comes home when their project is done.

This is the fastest known way to organize new product development. It is the fourth and final form on Wheelwright's spectrum at Harvard [1], and it is the form lateralworks has run for more than thirty years, most recently on a core display technology inside a major consumer AR glasses program, and before that on the Philips Velo, whose engineering leader, Tony Fadell, carried the same practices to Apple and built the iPod [2]. The structure controls the definition of the product, its development, and its manufacturing. It does not sell it. The running business stays with the hierarchy.

The structure asks four things of the CEO: approve it for the mission and the mission only, charter the Program Director in writing with explicit decision rights, confirm the Steering Arm and its cadence, and hold the boundary once it is drawn. The price is before-the-fact functional control over the program's daily work. The return is speed, and speed is the difference between a product that ships on its date and one that arrives after the market has moved.

> **Bottom line.** Keep the org chart for the business you have. Build a ship for the business you are trying to create. Put the team on the ship, name the captain, and let the mission run.

01

Problem statement

The structure is the constraint

A company commits to an ecosystem, and an ecosystem is a harder thing to build than a product. A dozen projects must converge: hardware, firmware, a connectivity layer, a software platform. Each has its own technical risk, its own schedule, and its own dependencies on the others. The commitment has a date, and the date is public. The structure most companies bring to that commitment is too slow to hit it, and it is not working: ownership is unclear, decisions queue in the hierarchy, and the projects drift. This briefing describes the structure that will.

Steven Wheelwright's research at Harvard describes four forms of organizing new product development, from the functional hierarchy through lightweight and heavyweight teams to the fully autonomous team [1, 3]. The forms differ in one essential variable: who controls the work before it happens. In the first two forms the functions control it, and project accountability is weak or absent. In the last two, the team controls it, and accountability is complete. Speed rises left to right across the spectrum, and so does ownership.

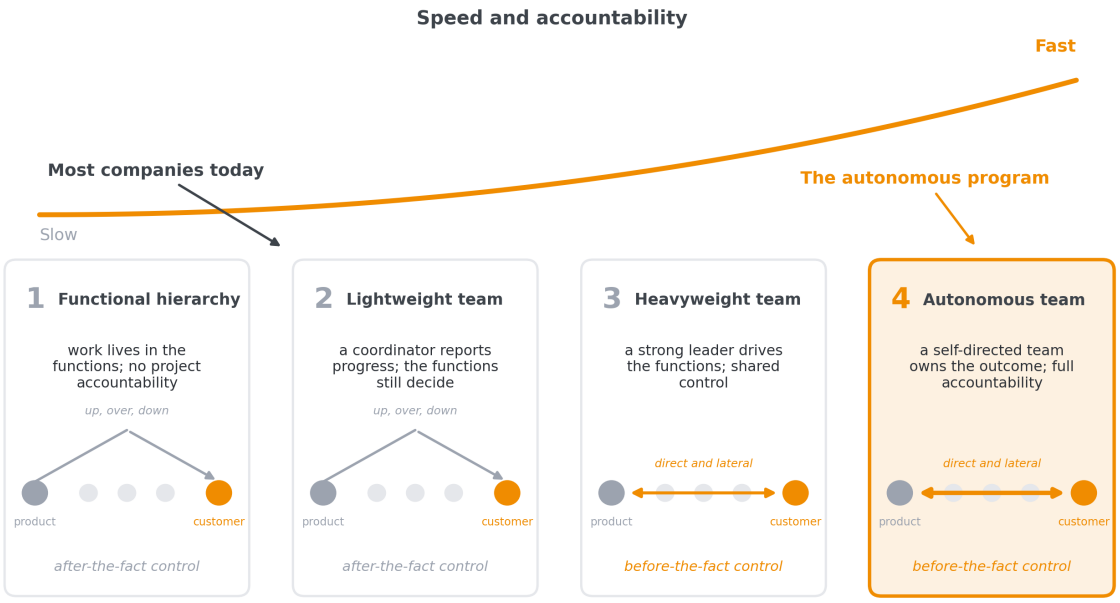

Figure 1. Wheelwright's four forms of development organization [1, 3]. Most established companies operate between forms 1 and 2. The autonomous program is form 4.

Where most companies sit

Most established product companies operate between the first and second forms, leaning to the second. Coordinators exist and they help, but the work still lives in the functions, and the functions still decide. Three symptoms follow directly from that position.

1. It is not clear who is running each project. Ask who owns the outcome of any one project, end to end, and the answer is a committee.
2. The program roles are administrative. They coordinate and report without overall control or clear authority. Responsibility without authority is the defining trait of the lightweight model, and it is why the lightweight model is slow [1].
3. Communication runs vertically. Decisions climb a function, cross the organization at the level where authority lives, and descend again. Each hop adds days. Across a dozen concurrent projects the hops compound into the queue that is quietly setting the schedule.

The people are capable and heavily loaded, and the structure decides how fast capable people are allowed to move. Structures like this were designed to protect a running business, which they do well. They were never designed to deliver a dozen converging projects on a date.

How companies get here

Nobody designs this position on purpose. The structure was built for the running business, projects were added one at a time, and each got a coordinator because a dedicated team felt expensive. The drift is invisible until the company commits to a program too big for it: many projects, one outcome, one date. Then the structure itself becomes the constraint, and no amount of effort inside it changes the arithmetic of the decision path.

02

The proposal

The autonomous program

The autonomous structure moves the program outside the corporate hierarchy for the duration of the mission. It is a complete organization in miniature: governance, leadership, project management, and staffing, all dedicated to one outcome [4]. Figure 2 shows the whole of it.

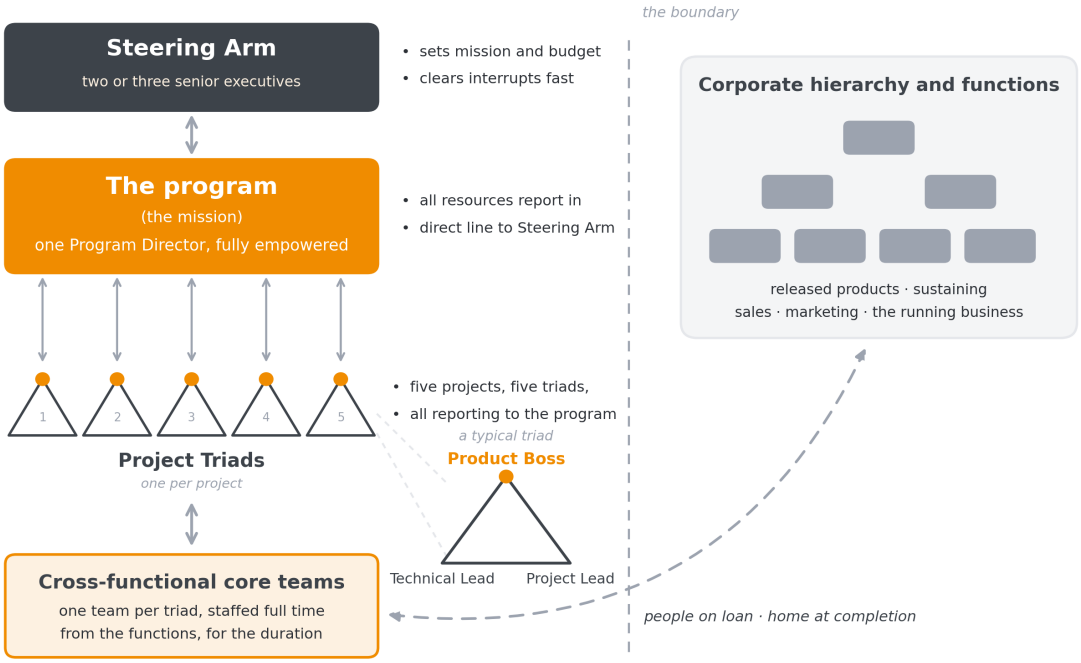

Figure 2. The autonomous program. The program spine on the left sits outside the hierarchy on the right. Team members are on loan and return to their functional home when their project completes.

The four layers

The Steering Arm is two or three senior executives, typically the CEO and the leaders whose organizations feed the program. It sets the mission and the budget envelope, reviews the program at the program level, clears obstacles the team cannot clear itself, and stays out of the daily work. It is the program's board, and like a good board it governs rather than manages.

The program is one Program Director with a direct line to the Steering Arm. Every project resource reports to the Program Director for the duration. One person is responsible for delivering the mission, and because the resources and the authority both sit there, the responsibility is real rather than nominal.

The Project Triads run each project in the program. Each triad pairs a Project Lead, who owns the plan, with a Technical Lead, who owns the engineering, under a Product Boss, who owns what the product is [6]. The triads report to the Program Director.

The cross-functional core teams do the work [5]. Members are assigned from the functions to a project for its duration and report to their triad. When the project completes, they migrate back to the functional hierarchy, carrying what they learned with them.

What the structure controls, and what it does not

The autonomous structure controls the definition of the product, its development, and its manufacturing, through launch. It does not sell it. Sales, marketing, released products, and sustaining stay with the corporate hierarchy, which keeps the running business healthy while the program builds the next one. The boundary is deliberate. The hierarchy is good at operating; the program is built for creating. Asking either to do the other's job is how both get slow [7].

03

The evidence

Why autonomy is speed

The claim that autonomous teams are the fastest form rests on three mechanisms that anyone who has run a development program will recognize. Each one is visible in the research and in thirty years of lateralworks engagement records [4].

Lateral communication beats vertical communication

Inside a hierarchy, information moves vertically: up for permission, across for coordination, down for direction. Inside an autonomous team, the engineer and the buyer and the firmware lead sit on the same team and talk directly. A question that takes a week to route through two functions takes an hour across a table. Clark and Wheelwright documented exactly this effect in heavyweight and autonomous teams: better communication, stronger identification with the project, and cross-functional problem solving as the default rather than the exception [1].

Before-the-fact control beats after-the-fact control

A functional organization controls projects after the fact. It reviews what happened, then corrects. A team that owns its outcome controls before the fact: it makes the decision, on the spot, with the people who have the context. Control systems layered on top of a program do not generate speed; they generate reporting [8]. Ownership generates speed, because the person who owns the outcome has no one to wait for.

Freedom to act is measurable, and it compounds

lateralworks scores empowerment on a five-level freedom scale, adapted from Oncken [9]. At Level 1 a person acts and reports routinely. At Level 5 they wait until told. Every level above 1 inserts a wait into the critical path, and waits on the critical path are the schedule [10]. The point of the autonomous structure is to move the program's everyday decisions to Levels 1 and 2 by design, rather than asking individuals to seize freedom their structure does not grant. The key is not what freedom a team will take. It is what leadership will give [10].

The compounding matters more than any single decision. A multi-project program makes thousands of decisions a year. Shorten the average decision by days and the delivery date moves by months. That is the return on fast decision-making, and it is the cheapest schedule acceleration available to any company [11].

What the hierarchy is for

None of this makes the hierarchy the villain. The functional organization is the host: it provisions the program with people, money, equipment, and services, and a good host is the difference between a fast team and a stranded one [9, 12]. The host is measured on control and staffed for the running business, and that is why the program must sit outside it. Put the mission inside the host and the mission waits in the host's queue. Put it outside, with the host provisioning it, and both do what they are built for.

04

Operating model
The Navy model

The simplest way to understand the operating model is the one the military has used for centuries. A sailor on a deployed ship reports to the captain of the ship for the duration of the mission. Home port, the base hierarchy, and the base commander still exist, and the sailor returns to them when the mission ends. Nobody on a deployed ship radios the base to ask permission to trim a sail.

An autonomous program runs the same way. Every member of a core team is on loan from their function. For the duration of their project they report to their triad, the triads report to the Program Director, and the Program Director reports to the Steering Arm. Their functional home does not disappear. Their career path, their discipline, and their long-term development still live in the function, and when their project completes they migrate back, better than they left. The loan is explicit: named people, named projects, named durations.

What changes on Monday morning

For a core team member, the change is one reporting line and one priority. Their project is their job, their triad sets their week, and no functional manager can redirect their time without going through the program. For a functional manager, the change is the mirror image: the loaned people are gone for the duration, visibly, and the running business is planned around their absence rather than quietly competing for their hours. The half-assignments and priority collisions that consume programs staffed by fractions of people are the thing this model exists to end [5].

A mission, with a date

There is a cultural claim buried in the mechanics, and it is worth stating plainly. Inside a functional structure, projects are jobs. The autonomous structure turns them into a deployment: a ship, a captain, a crew, and one mission with a date on it. People work differently on a mission than they do on a job, and thirty years of watching fast teams says that difference is worth as much as any mechanism in this briefing.

05

People
Role architecture

An autonomous structure lives or dies on role clarity, so this section is specific about the three roles companies wrestle with most: the Program Director, the executive who hosts the program, and the Steering Arm itself.

The Program Director, chartered in writing

The Program Director is the person the company is trusting to deliver the mission quickly: a direct line to fast decision-making, full empowerment, and all project resources reporting in, so the authority to act fast is real. That sentence only becomes true if the authority is written down. Verbal empowerment evaporates at the first disagreement. The charter should state the mission, the date, the budget envelope, the named resources, and the decision rights in Table 1, and the Steering Arm should sign it.

Decision class	Typical today	Target	Held by
Staffing within the approved plan	Level 3: recommend, then wait	Level 1: act, report routinely	Program Director
Design and scope trade-offs within a release	Level 3 to 4	Level 1 to 2	Triad, with the Product Boss
Schedule commitments and pull-ins	Level 3	Level 1	Program Director
Vendor and ODM decisions within the envelope	Level 4: ask what to do	Level 2: act, advise at once	Program Director
Spending within the approved budget	Level 3	Level 1	Program Director
Mission, release scope, budget envelope	Dispersed	Set once, changed rarely	Steering Arm

Table 1. Decision rights for an autonomous program, on the lateralworks freedom scale [9]. Lower level means more freedom to act.

The executive who hosts the program

The structure also resolves the question of where the senior development executive sits relative to the program, and it does so by strengthening the role rather than diminishing it. That executive keeps the functional hierarchy: the development process, sustaining engineering, and the released-product portfolio, plus a seat on the Steering Arm, where product judgment guides the program at the level where guidance belongs. The separation is clean: the Program Director runs the program; the host executive governs it and runs the function that hosts it. Wheelwright's work on the senior manager's role in development makes the same split: executives sponsor, coach, and set bounds; teams execute [13].

The Steering Arm’s discipline

The Steering Arm has three jobs: set the mission and hold it constant, provision the program and arbitrate staffing disputes with the functions, and clear interrupts fast when the team escalates one [12]. It reviews the program monthly, on live data, in a working session rather than a slide show [15]. It has one discipline that outweighs the three jobs: once the boundary is drawn, do not reach across it. The fastest way to break this model is to keep the new org chart and route the decisions the old way anyway.

Staffing the triads

The model concentrates enormous trust in a small number of people, which makes selection the highest-leverage staffing decision of the program. Triad leads should be chosen for demonstrated Level 1 behavior: people who act, close their own loops, and report what they did, rather than people who escalate well [14]. Most companies already have these people. The structure finally gives them somewhere to run.

The mission

One mission, one date

This is not a job. It is a mission.

Outside the hierarchy

lateralworks briefing, September 2026

06

Track record
Precedent

The autonomous model is well traveled. It is the organizing form behind some of the fastest product programs on record, and lateralworks has been building them since 1988 [18].

Philips, the Velo. lateralworks ran FTTM practices on the Philips Velo handheld program in the 1990s. The engineering executive who led that project, Tony Fadell, carried the same practices to Apple, where he created the iPod and led the iPhone and iPad programs [2]. The lineage matters because it shows the model traveling: the practices that moved a PDA through Philips are the practices that moved the iPod through Apple.

A major AR glasses program. Most recently we applied the same structure on a core display technology inside a consumer AR glasses program budgeted in the hundreds of millions of dollars, one of the defining technologies in the product. Autonomous teams reporting outside the corporate hierarchy, on loan from their functions, mission-dated. The details are the client's, but the structure is the one in Figure 2.

Two hundred programs of pattern. Across semiconductors, consumer electronics, instruments, and software, the pattern holds: when the team owns the outcome and sits outside the host's queue, it moves; when it is coordinated through the functions, it waits [18]. The failures we have seen with this model were boundary failures, where the host reached back in, rather than team failures. That is why Section 07 exists.

07 Eyes open
Risks and safeguards

Wheelwright is candid that the autonomous form has known costs, and so are we [1, 3]. Naming them, and building the safeguard into the charter from day one, is what separates a structure that works from a structure that gets quietly reversed in six months.

The product drifts from the market

An empowered team can build the wrong thing faster. The safeguard is already in the structure: the Product Boss owns the definition inside each triad [6], and the Steering Arm reviews at release gates, where the host executive's product judgment and the voice of the customer get their formal say.

Functional standards and reuse decay

Autonomous teams can wander from company standards and reinvent what the functions already have. The Technical Lead in each triad holds the functional thread, and the loan model itself is the long-term answer: people rotate back, and the knowledge returns with them.

Reintegration at the end

The known weakness of the autonomous form is coming home [1]. The Navy model answers it by design: return to home port is written into every loan from day one, with the function committed to the landing. Nobody should join the ship wondering whether they still have a base.

The functions backfill with intentions

A loan honored with a name on paper and a person still doing their old job is not a loan. Loans are named people, full time, for named durations, and the Steering Arm arbitrates when a function cannot release someone. Performance reviews for deployed people take their input from the program, at the team level, on the outcome, so the incentive system points the same direction as the structure [16].

The running business starves

Pulling strong people onto the program leaves holes. The split in Figure 2 is the safeguard: sustaining, released products, sales, and marketing stay with the hierarchy, planned deliberately around the loans rather than absorbing them as surprise.

The company sells around the team

The structure does not sell the product, so the commercial handoff must be designed rather than assumed. Sales and marketing join the release gates as the launch approaches, and the handoff at launch is a chartered event with an owner, the same as any milestone on the critical path.

08

Decisions

What this asks of the CEO

The structure is a CEO decision. It cannot be adopted from the middle of the organization, because its whole point is to change where authority sits. Four decisions put it in place [17].

1. Approve the structure, for the mission only. This is a program design for one mission with one date. It is not a company reorganization, and it should not be sold internally as one. The hierarchy keeps the business; the program gets the mission.
2. Charter the Program Director in writing. Mission, date, budget envelope, named resources, and the decision rights in Table 1, signed by the Steering Arm. The charter is the ship's commission, and it is what the Program Director points to the first time someone routes a decision the old way.
3. Confirm the Steering Arm and its cadence. Two or three senior executives, meeting monthly on live program data, with an explicit fast lane for interrupts between meetings [15].
4. Name the loans. Every core team seat filled with a named person for a named duration, functions committed, triads staffed with Level 1 people [14]. Staffing is where the model becomes real or becomes theater.

The first thirty days

- Week 1. Steering Arm confirmed. Charter drafted and signed. The structure announced with the mission, in one voice.
- Week 2. Loans named. Triad assignments settled across every project. Functional backfill planned.
- Week 3. Triads stand up. Baseline planning begins against the mission date, project by project.
- Week 4. Program baseline reviewed by the Steering Arm. The weekly cadence starts, and the ship sails.

A closing note

Everything in this briefing can be tuned once the executive team has digested it: the cadence, the decision rights, the staffing sequence. The two things that cannot be tuned away are the boundary and the authority, because they are the structure. Hold those two, and the company will have done the single most powerful thing it can do for a program it needs on time: gotten out of its way, and given it a captain.

> **The decision in one sentence.** Put the team on a ship, outside the hierarchy, with a captain in command, the Steering Arm above it, and one mission with a date on it.

Sources

References

[1] Clark, K. B., and Wheelwright, S. C. “Organizing and Leading Heavyweight Development Teams.” California Management Review, 34(3), 1992, pp. 9-28. cmr.berkeley.edu

[2] lateralworks. “The methodology that built the iPod: Philips Velo.” lateralworks results. lateralworks.com/results#philips

[3] Wheelwright, S. C., and Clark, K. B. *Revolutionizing Product Development*. Free Press, 1992.

[4] lateralworks. “Fast autonomous teams.” Ideas library. lateralworks.com/ideas/fast-autonomous-teams

[5] lateralworks. “Integrated core team.” Ideas library. lateralworks.com/ideas/integrated-core-team

[6] lateralworks. “The Product Boss owns the product; the Program Manager owns the plan.” Ideas library. lateralworks.com/ideas/who-owns-the-product-who-owns-the-plan

[7] lateralworks. “What is the host?” Ideas library. lateralworks.com/ideas/what-is-the-host

[8] lateralworks. “Control systems don’t generate speed.” Ideas library. lateralworks.com/ideas/control-systems-dont-generate-speed

[9] lateralworks. “Freedom to act.” Ideas library. lateralworks.com/ideas/freedom-to-act

[10] lateralworks. “What leadership will give.” Ideas library. lateralworks.com/ideas/what-leadership-will-give

[11] lateralworks. “What is the R.O.I. of fast decision-making?” Ideas library. lateralworks.com/ideas/what-is-the-roi-of-fast-decision-making

[12] lateralworks. “The host interrupt.” Ideas library. lateralworks.com/ideas/the-host-interrupt

[13] Wheelwright, S. C. *Leading Product Development: The Senior Manager’s Guide to Creating and Shaping the Enterprise*. Free Press, 1994.

[14] lateralworks. “The Level 1 hire.” Ideas library. lateralworks.com/ideas/the-level-1-hire

[15] lateralworks. “The monthly project deep-dive.” Ideas library. lateralworks.com/ideas/the-monthly-project-deep-dive

[16] lateralworks. “Reward performance. Don’t incentivize it.” Ideas library. lateralworks.com/ideas/reward-performance-dont-incentivize-it

[17] lateralworks. “The FTTM execution system.” Ideas library. lateralworks.com/ideas/the-fttm-execution-system

[18] lateralworks. Methodology: 36 years of FTTM research, 200 plus programs since 1988. lateralworks.com/methodology