avansys lab
www.avansys-solutions.com

"WE KEEP MISSING DEADLINES" EXECUTIVE SUMMARY

The full paper in one page · for the reader who needs the key points before reading the analysis · 11 pages of substance distilled to four moves

The Bottom Line: Three consecutive missed milestones. Five reasonable execution-layer fixes tried, all in the same direction, all failed the same way. The deadlines are not moving because the team is executing poorly. They are moving because "done" was never defined precisely enough to hold, and every fix so far made the team work harder against a target that keeps quietly relocating.

The Diagnostic

The Dominant Frame

"How do we get the team to hit its deadlines?"

Sounds right. Feels like the PM's job. Pre-commits every answer to the execution layer — tracking, reporting, effort — before asking what is actually being tracked.

The Three Category Errors

1. **A definition problem treated as a scheduling problem.** Deadlines move because scope expands. Scope expands because "done" is not defined. More check-ins make the expansion visible — they do not stop it.
2. **A requirements problem treated as an execution problem.** Every "actually, we also need…" conversation is a requirements failure, not an execution failure. The team is executing well against an incomplete definition.
3. **A stakeholder problem treated as a team problem.** The stakeholder's picture of what they want becomes clearer as they see work in progress. Normal human behaviour — in the wrong role at the wrong time.

Five Fixes Already Tried

- **Daily standups** — added visibility, no change to the trajectory
- **A new project management tool** — retracked the same moving scope, faster
- **Smaller task breakdown** — distributed the same undefined scope into smaller undefined pieces
- **Buffer time added to estimates** — consumed, then exceeded, by the next scope clarification
- **Leadership escalation for more resources** — added capacity against a target that was still moving
- **Result: three consecutive milestones missed** despite all five fixes

The Solution

The Refined Reframe

REPLACEMENT FRAME

"How do we define what each milestone looks like when it's done — specifically enough that everyone agrees before work begins, and scope cannot expand once it starts?"

The Five-Phase Sequence

1. **Phase 1 — Run the Definition Session.** Two hours with the stakeholder before the next milestone starts. One written, testable definition of done, signed by both parties. No new tools.
2. **Phase 2 — Install the Change Decision Gate.** Any request that changes the definition after sign-off triggers a formal impact assessment. The stakeholder moves the date or defers the change.
3. **Phase 3 — Rename the Structural Problem With Leadership.** The conversation shifts from "the team missed the deadline" to "the milestone lacked a mechanism to hold scope stable."
4. **Phase 4 — Repeat the Gate at Each Remaining Milestone.** The same session and decision gate apply to every milestone still ahead, testing whether the pattern holds once the fix moves layers.
5. **Phase 5 — Make the Gate Standard Practice.** The definition session becomes a standing step in the team's process, not a one-time fix for this project.

The Choice

Both paths cost something. A sixth execution-layer fix is easy to propose and has already failed five times in five forms. The definition session costs two hours of stakeholder time before work is allowed to start — time nobody currently budgets, because nobody has had to ask for it explicitly before.

FINAL VERDICT

Sarah was solving the right problem stated the wrong way. Three consecutive misses across five different execution-layer fixes is not a signal to try a sixth. It is empirical confirmation that the fix belongs one layer up. **Read Sections 1–4 for the substantiation.**

"WE KEEP MISSING DEADLINES" THE FULL REFRAMEX© CASE STUDY

Applying the five-pattern method to a mid-level PM's third consecutive missed milestone — what five reasonable fixes could not reach, and what one definition session does instead

Central Argument: *A mid-level PM has tried five different execution-layer fixes across three missed milestones: more visibility, a new tool, smaller tasks, more buffer, more people. None of it worked, because the deadlines are not moving due to an execution problem. They are moving because "done" was never defined precisely enough to hold, and every fix so far made the team work harder against a target that keeps quietly relocating. This paper applies the ReFrameX© method to name that dynamic precisely, and to design the definition-layer gate that would have prevented all three misses.*

Scope and Disclosure

This paper is a methodological exercise applied to an ordinary, painful project situation. The ReFrameX© method is content-neutral; it surfaces category errors and layer-mistakes in problem framing regardless of whether the underlying domain is a six-month software integration or a national labor market. This case is used here because it exhibits, with unusual clarity, a symptom — missed deadlines — that is real and a diagnosis — the team isn't executing well — that is wrong.

The author holds no position on Sarah's individual performance. The argument is narrower: the frame within which she and her leadership have been responding measures a downstream symptom, and a realistic alternative can be built from the same method, in under thirty minutes, without new tools.

What Happened — Factual Baseline

Sarah is a mid-level PM managing a cross-functional team of eight on a six-month software integration project. Four months in, three consecutive milestones have been missed. She has already tried daily standups, a new project management tool, breaking work into smaller tasks, adding buffer time to estimates, and escalating to leadership for more resources. None of it has worked.

She brings the problem to ReFrameX© as: "How do we get the team to hit its deadlines?" That question sounds right. It is not the right question. And that is exactly what the method is designed to find.

METHODOLOGICAL HONESTY

Section 4 proposes a definition session that adds two hours of stakeholder time before a milestone is allowed to start. The author acknowledges this adds friction to an intake process that currently feels informal and fast, and that some stakeholders will experience the session as a delay rather than a safeguard. The point is not that the addition is costless. The point is that the two hours are already being spent — repeatedly, informally, mid-milestone — in the form of "actually, we also need" conversations that this session would front-load and settle once.

Reader's Guide

This paper is in four sections.

1. **Section 1 — The Reframing Analysis.** Applies the five-pattern diagnostic to the scheduling frame and surfaces three category errors, an accountability gap, and a naming trap.
2. **Section 2 — A Structured Alternative.** Builds the five-layer architecture, the bounded roles, the authority gates, and the load-bearing vocabulary of a definition-first response using the full scaffold.
3. **Section 3 — Five Fixes, One Pattern.** Goes through each of Sarah's five prior interventions and shows why each failed for the same underlying reason, confirming the category errors named in Section 1.
4. **Section 4 — Three Fixes, No New Tools.** Sequences the definition session, the change decision gate, and the leadership reframe into a five-phase rollout, and compares the resulting response to the execution-layer response side by side.

How the Method Applies Here

The ReFrameX© method's five patterns (Reframe-First, Frame-the-System, Place-the-Parts, Authority-Where-Accountability-Lives, Naming-as-Anchoring) operate at any scale — a national policy or a single team. This case is particularly useful for practitioners because the evidence that the frame is wrong is entirely self-contained: five different fixes, tried in good faith, all failing the same way, is itself the diagnostic signal.

For practitioners who have not previously applied the method to project delivery, this case is instructive precisely because the same diagnostic that catches DOGE's misframing, Boeing's misframing, Brexit's misframing, the Iran strikes' misframing, the Fixed-Price Order's misframing, and the labor market's displacement misframing catches Sarah's scheduling misframing here — and points toward the same kind of solution: naming the layer where the problem actually originates before choosing the fix.

THE DOMINANT FRAME AND ITS CATEGORY ERRORS

What the scheduling frame says · what it pre-commits to · what it cannot see

The Dominant Frame

"How do we get the team to hit its deadlines?"

This is the frame in the standups, the new tool, the smaller tasks, the added buffer, and the escalation for more resources. It is intuitive. It sounds like exactly the question a PM should be asking. It pre-commits every answer to the *execution layer* — how the work is tracked, how tasks are broken up, how progress is reported. Within that frame, all five fixes Sarah tried are coherent interventions: increase visibility, increase capacity, increase precision of tracking, increase tolerance for slippage.

The frame is not wrong in some absolute sense. The team's execution genuinely matters, and reasonable execution hygiene is not a waste of effort. The question the method forces is whether the frame captures the actual problem, or only the most visible place to look for it.

The Category Errors

Error 1 — A definition problem treated as a scheduling problem

The deadlines are moving because the scope keeps expanding. And the scope keeps expanding because "done" was never defined precisely enough to hold. Each time a milestone approaches, the stakeholder sees the work in progress and clarifies what they actually wanted. The work expands. The deadline moves. The team absorbs it and tries harder next time. This is not a performance problem. It is a definition problem. And no execution fix addresses a definition problem.

Error 2 — A requirements problem treated as an execution problem

Every "actually, we also need…" conversation is a requirements failure, not an execution failure. The team is executing well against an incomplete definition. Adding standups, tools, or smaller tasks makes execution against that incomplete definition more visible and better organized. None of it makes the definition complete.

Error 3 — A stakeholder problem treated as a team problem

The stakeholder's picture of what they want becomes clearer as they see work in progress. This is normal human behaviour — people often cannot fully specify what they want until they see a version of it. The error is not that this happens. The error is treating it as the team's execution failure rather than as a normal human behaviour occurring in the wrong role at the wrong time, with no mechanism designed to catch it before it re-enters the schedule as scope.

DIAGNOSTIC SUMMARY

Three stacked category errors: definition mistaken for scheduling, requirements mistaken for execution, stakeholder behaviour mistaken for team performance. Each is a recognized pattern in the method's library. Each is correctable. None is corrected by executing harder against the same undefined target.

THE DIAGNOSTIC SIGNAL

When multiple reasonable fixes all fail in the same way, the fix is almost always being applied to the wrong layer. The problem is not what it looks like. That is the pattern ReFrameX© is built to catch.

AUTHORITY, NAMING, AND WHAT THE FRAME FORECLOSED

Pattern 4 and Pattern 5 applied · the parts of the problem the frame structurally cannot engage

Authority-Accountability Gaps

The scheduling frame creates a clear accountability structure. Sarah is accountable for hitting the milestone date. The team is accountable for executing its tasks. Leadership is accountable for supplying resources when asked. Everyone has a role.

What the frame does not create accountability for is **the stability of the definition itself**. The stakeholder who clarifies scope mid-milestone faces no accountability for that clarification arriving late. Leadership, evaluating the team on deadline adherence, has no mechanism for learning that the deadline was never achievable against a moving definition. Sarah is accountable for a date that was never fully hers to control, because the thing that kept moving — the definition — was never assigned an owner or a lock-in point.

This is the precise pattern Pattern 4 names as a brittleness signature: when the actor accountable for the outcome (Sarah, for the date) has no authority over the variable actually driving the outcome (the stakeholder, over the definition), the system cannot close its own feedback loop. The diagnostic prediction: more fixes get applied to the accountable party's layer, the actual driver remains untouched, and the pattern repeats on the next milestone — which is precisely what happened three times in a row.

Naming-as-Anchoring — The Trap

"We keep missing deadlines"

The phrase names the team as the subject and the date as the object, anchoring the fix to team behaviour. It hides that the actual variable in motion is the definition of done, not the team's pace. Naming the symptom as the problem makes the actual cause — an unstable definition — invisible inside the sentence used to describe it.

"The team needs to move faster"

The implied fix inside the dominant frame's own vocabulary. It assumes the target is fixed and the team's speed toward it is the variable. When the target itself is moving, speed is irrelevant to whether it is reached — a faster team executing against an unstable definition simply reaches the wrong, larger scope sooner.

What the Frame Cannot Engage

1. **Is "done" defined specifically enough that a third party could test it?** The frame cannot ask this because it treats the deadline, not the definition, as the object needing management.
2. **What is this milestone actually costing in re-work when the definition shifts mid-flight?** Cannot ask this because accountability for the date sits with Sarah, not with the definition's stability.
3. **Who decides whether a scope change moves the date or gets deferred?** Cannot ask this because scope changes currently enter execution silently, with no decision point at all.
4. **Was this a performance miss or a structural miss?** Cannot ask this because the frame has no vocabulary for a miss that originates above the execution layer.

THE METHOD'S VERDICT — SECTION 1

The team's situation is intelligible within the frame "get the team to hit its deadlines." That frame contains three stacked category errors, an accountability gap where nobody owns the stability of the definition, and a naming trap that keeps the actual driver — a moving target, not a slow team — out of the conversation. The instruments built from this frame are not wrong. They are aimed one layer below where the misses actually originate. Section 2 attempts the rebuild.

THE REFRAME, THE ARCHITECTURE, THE LAYERS

Building what the dominant frame foreclosed · what would have to be true · why five layers, not one

The Replacement Frame

"How do we define what each milestone looks like when it's done — specifically enough that everyone agrees before work begins, and scope cannot expand once it starts?"

This is the reframe the method produces. It contains four explicit moves, each of which reverses a specific category error from Section 1.

Replacement move	Reverses
"What each milestone looks like when it's done"	Definition mistaken for scheduling (Error 1)
"Everyone agrees before work begins"	Requirements mistaken for execution (Error 2)
"Scope cannot expand once it starts"	Stakeholder behaviour mistaken for team performance (Error 3)
"How do we define" (not "how do we get the team")	The authority-accountability gap (Pattern 4)

The original question pre-commits every answer to the execution layer. The replacement question points directly at the definition layer — which is where the problem actually lives. Same team, same project, same stakeholder. Different question, completely different set of actions.

Before and After the Reframe

Before ReFrameX©	After ReFrameX©
"How do we get the team to hit deadlines?"	"How do we define done before committing to a date?"
Add more check-ins and reporting	Run a definition session before each milestone
Add buffer to absorb uncertainty	Refuse to commit a date against an unstable definition

The System Structure

The method (Pattern 2) requires that any reframe be followed by a designed system structure — named layers, each with a bounded role, each with explicit interfaces between them. The reframed problem decomposes into five layers.

1. **Definition (foundational layer).** What "done" means, specific enough that a third party could test it. The layer where the crisis actually originates. Not addressed by any of Sarah's five fixes.
2. **Requirements Architecture (architecture-to-component bridge).** How requirements are gathered, clarified, and locked before execution begins. Not addressed.
3. **Execution (component layer — where all five fixes operated).** Tracking, task breakdown, reporting, tools. Primary focus of every fix tried so far.
4. **Change Governance (interface layer).** How scope changes after sign-off are processed — decision versus silent absorption. Not addressed.
5. **Organizational Reporting (governance layer).** What leadership hears, and how a miss gets attributed. Currently defaults to a performance narrative.

WHY FIVE LAYERS, NOT ONE

All five of Sarah's fixes operate at Layer 3. The problem originates at Layer 1. Daily standups, a new tool, and smaller tasks are not wrong; they are complete instruments for a layer the problem never lived at. A frame that produces only the execution layer cannot reach the definition layer where the scope is actually expanding. The method's diagnostic for "why did five fixes all fail the same way?" is not "the team needs to try harder" — it is "every fix so far addressed one layer out of five."

BOUNDED ROLES, GATES, AND THE HARD TRADE-OFFS

Placing the parts · locating authority · naming the operationally inconvenient honestly

Component Placement (Pattern 3)

Actor	Bounded to	Currently doing instead
Project Manager	Running the definition session; surfacing scope changes as decisions	Absorbing scope changes and trying to execute faster
Stakeholder	Committing to a specific, testable definition of done before work begins	Clarifying requirements informally, as they see work in progress
Team	Executing against a stable, well-defined scope	Executing against a moving target and absorbing the difference
Leadership/Sponsor	Holding the definition gate accountable; evaluating structural stability	Evaluating the team on deadline adherence alone

The component-placement work above is what does not happen when the frame is "get the team to hit deadlines." Under that frame, every actor's job is to help the team go faster. Under the reframed structure, each actor is bounded to a different piece of a definition architecture nobody currently owns in full.

Authority Gates (Pattern 4)

1. **Definition Sign-Off Gate.** No milestone execution begins until a written, testable definition of done is signed by the PM and stakeholder. Does not exist by default.
2. **Change Decision Gate.** Any request that changes the definition after sign-off triggers a formal timeline and resource impact assessment before it enters execution. Does not exist by default.
3. **Escalation Language Gate.** When a milestone is at risk, the structural cause is named explicitly to leadership, not folded into a general "the team is behind" report. Rarely happens by default.
4. **Retrospective Attribution Gate.** Post-milestone reviews identify whether a miss originated at the definition or execution layer before any performance consequence is assigned. Does not exist by default.

The Hard Trade-Offs — Named Honestly

The method requires that the costs of the reframed solution be named explicitly. Otherwise the comparison to the execution-layer response is rigged. Three trade-offs are operationally inconvenient and immediately real.

TRADE-OFF 1 — THE DEFINITION SESSION COSTS TIME NOBODY CURRENTLY BUDGETS

Two hours before "real work" starts will feel like a delay to a stakeholder under their own time pressure, even though skipping that conversation is exactly what generates every overrun that follows it.

TRADE-OFF 2 — THE CHANGE DECISION GATE SLOWS DOWN "QUICK ASKS"

A stakeholder's small, reasonable-sounding addition can no longer enter silently. It must be named, costed, and decided, which will feel like new bureaucracy to a stakeholder used to informal, mid-flight clarifications.

TRADE-OFF 3 — NAMING THE STRUCTURAL CAUSE MEANS GIVING UP THE EASIER NARRATIVE

"We tried harder and still missed it" protects the team from scrutiny of the process. "The definition was never stable" is more accurate and invites scrutiny of the stakeholder relationship and intake process — a less comfortable conversation for a PM to initiate with leadership, even though it is the honest one.

VOCABULARY, UPTAKE, AND THE METHOD'S VERDICT

Naming the architecture · who would have to carry it · why this is easy to name and harder to finish

Load-Bearing Vocabulary (Pattern 5)

Term	Anchored to
Definition of Done	Agile and Scrum literature's Definition of Done concept, extended here to a formal, testable, signed artifact
Change decision, not absorption	PMI and PMBOK change control and integrated change management practice
Requirements volatility	Software and systems engineering literature identifying unstable requirements as a leading driver of schedule overruns
Structural miss, not performance miss	Root-cause and systems-thinking distinction between process failure and individual performance failure

Each term displaces a piece of the scheduling frame's vocabulary with one that creates a distinction the original language could not. "Definition" instead of "deadline." "Decision" instead of "absorption." "Structural" instead of "underperforming."

Uptake — Who Would Have to Carry This

1. **Inside the project** — Sarah needs to hold the definition session before work begins on each remaining milestone, which requires stakeholder time she has not previously had to ask for this explicitly.
2. **With the stakeholder** — accepting that a signed, testable definition is a precondition for a date commitment, not a formality to get through quickly.
3. **With leadership** — accepting the reframed escalation language, which shifts scrutiny partly onto the intake process rather than resting entirely on team velocity.
4. **Across the team** — trusting that a stable, signed definition will actually be held, so the change decision gate is not experienced as one more process layered on top of the same old absorption dynamic.

The Method's Final Verdict — Section 2

> **FINDING**
>
> Applying the full ReFrameX© diagnostic to this project yields a precise finding: the response so far is a Layer 3 instrument — execution — applied five different ways to a problem that originates at Layer 1. This incompleteness is not a reflection of Sarah's competence. It reflects treating a definition problem with instruments built for an execution problem, which is the natural first move for any PM trained to manage schedules rather than requirements architecture.

The reframed solution outlined above is not free. It requires the stakeholder to accept slower, more disclosed intake, the team to trust a gate it has not seen hold before, and Sarah to have a harder conversation with leadership than "we're behind, we need more resources." These are real costs. The scheduling frame's defenders are correct that a sixth execution fix is simpler to propose than a new stakeholder process.

The reframed solution's defenders are correct on a different point: the scheduling frame's cost is also real, and already paid five times over in fixes that did not hold. The choice is not between an easy fix and a harder one. It is between a sixth fix aimed at the wrong layer and a first fix aimed at the right one.

THE EXECUTION-LAYER FIXES, TESTED AND FAILED

Every fix Sarah tried, and why each one confirms the category errors named in Section 1

Factual Baseline — What Was Already Tried

Daily standups. A new project management tool. Smaller task breakdown. Buffer time added to estimates. Leadership escalation for more resources. Three milestones. All five fixes tried, in sequence, over four months. Still missed.

Each fix was a reasonable response to a visible symptom. None addressed the definition layer where the scope was actually expanding. The five failures, taken together, are not five separate problems. They are the same category error, tested five different ways.

Fix 1 — Daily standups (Confirmed)

Standups increase visibility into execution. They made the scope expansion more visible — the team could see, sooner, that the target had moved — without providing any mechanism to stop the next "actually, we also need." Visibility without a gate produces anxiety, not correction.

Fix 2 — A new project management tool (Confirmed)

A tool tracks tasks against whatever scope currently exists. When the scope moves, the tool faithfully re-tracks the new scope. It cannot enforce a definition that was never written down as testable and signed, because tracking software has no opinion about whether the target it is tracking should have moved.

Fix 3 — Smaller task breakdown (Confirmed)

Decomposition makes execution more legible. It distributes the same undefined scope into smaller undefined pieces. Each piece can still expand the same way the whole did, because the instability was never in the size of the tasks. It was in what each task was ultimately supposed to produce.

Fix 4 — Buffer time added to estimates (Confirmed)

Buffer absorbs uncertainty in execution speed. It does not absorb uncertainty in what is being built. An unstable definition consumes any buffer, then keeps expanding past it, because the buffer was sized against the wrong kind of variance.

Fix 5 — Leadership escalation for more resources (Confirmed)

More people can execute a defined scope faster. They cannot execute an undefined scope faster, because the missing constraint was never capacity. Additional resources against a moving target typically just accelerate the appearance of "almost done," which the next stakeholder clarification then reopens.

> **WHAT THREE MISSED MILESTONES NOW MAKE LEGIBLE**
>
> ReFrameX© identifies windows when category errors become legible to actors who had previously been protected from seeing them. Three consecutive misses using five different execution-layer fixes is such a window. The team's defenders cannot claim the problem is effort or process discipline when every reasonable process fix has already been tried. The definition frame's defenders cannot claim the fix is costless: it requires two hours of stakeholder time before the team is comfortable calling itself "started," and it requires the PM to name a structural gap to leadership rather than promise to try harder.

BEFORE AND AFTER, AND THE RESPONSE VERDICT

The execution response vs. the definition response, side by side — and the honest verdict on what five fixes could reach

Before ReFrameX© vs. After ReFrameX©

Before ReFrameX©	After ReFrameX©
"How do we get the team to hit deadlines?"	"How do we define done before committing to a date?"
Add more check-ins and reporting	Run a definition session before each milestone
Add buffer to absorb uncertainty	Refuse to commit a date against an unstable definition
Treat scope changes as part of the work	Treat scope changes as formal decisions with a cost
Escalate for more resources	Escalate the structural gap at the definition layer

THE VERDICT, IN SARAH'S WORDS

Sarah was solving the right problem stated the wrong way. The replacement question — "how do we define done before we commit?" — produces three specific, actionable changes. No new tools. No new resources. No culture overhaul. A definition session before each milestone, a formal scope change process during execution, and a different escalation language with leadership. That is it.

The Response Verdict

Every fix Sarah tried is a genuinely competent execution-layer instrument. None of them are costless mistakes, and none of them are wrong to have inside a well-run project. They are complete instruments for Layer 3, aimed at a problem that originates at Layer 1.

Daily standups (visibility) genuinely surface problems earlier. They cannot fix a problem visibility alone cannot solve. Real, but incomplete.

The PM tool (tracking) genuinely organizes work. It tracks against whatever scope exists, including a moving one. Accurate, not corrective.

Task breakdown (decomposition) genuinely clarifies execution. It cannot clarify a definition nobody wrote down. Structural, not definitional.

Buffer time (schedule padding) genuinely absorbs execution variance. It cannot absorb scope variance, a different variable entirely. Misapplied, not wrong.

Resource escalation (capacity) genuinely increases throughput. Throughput against an undefined target does not converge. Amplifying, not resolving.

THE PHASED SEQUENCE AND THE SIDE-BY-SIDE

What changes the structure, not the symptoms — sequenced, and compared instrument for instrument

The Five-Phase Implementation Sequence

The three fixes unfold in five sequenced phases. Each phase has named participants, named deliverables, and a defined gate to the next phase.

1. **Phase 1 — Run the Definition Session (before the next milestone starts).** Two hours with the stakeholder. One output: a written definition of done that both Sarah and the stakeholder sign. Specific enough that a third party could test it — not "the feature is built" but "the feature processes X transactions per minute, passes QA, and the client signs off on the test results." Gate to Phase 2: definition signed.
2. **Phase 2 — Install the Change Decision Gate (during the milestone).** Any request that changes the definition of done after sign-off triggers a formal assessment of timeline and resource impact. The stakeholder chooses: accept the change and move the date, or keep the date and defer the change. Gate to Phase 3: first change request processed through the gate.
3. **Phase 3 — Rename the Structural Problem With Leadership.** "The milestone definition was not stable enough to hold a commitment. We have implemented a definition gate for future milestones." Gate to Phase 4: leadership accepts the structural framing.
4. **Phase 4 — Repeat the Gate at Each Remaining Milestone.** The same definition session and change decision gate apply to every milestone still ahead, testing whether the pattern holds once the intervention moves to the right layer. Gate to Phase 5: at least one milestone lands on its committed date.
5. **Phase 5 — Make the Gate Standard Practice.** The definition session and change decision gate become a standing step in the team's process for future projects, not a one-time fix for this one.

Side by Side — Execution Response vs. Definition Response

Execution response	Definition response
Daily standups	Definition session before the milestone starts
New project management tool	Written, testable, signed definition of done
Smaller task breakdown	Same decomposition, applied to a stable, signed scope
Buffer time added to estimates	Change decision gate: extend the date or defer the change, decided explicitly
Escalation for more resources	Escalation naming the structural gap, not a capacity request
"The team needs to move faster"	"The milestone lacked a mechanism to hold scope stable"
No formal definition-of-done artifact	Signed, testable definition-of-done document per milestone

THE HONEST COMPARISON AND FINAL VERDICT

What changes on Monday morning · the honest comparison · the method's final word

The Honest Comparison

The execution-layer response has one lever, tried five different ways: help the team move faster, see more, absorb more, get more hands. Three consecutive misses show what happens when the lever is pulled five times against a target that is still moving: each fix produces a plausible-sounding improvement in how the team works and no change in whether the milestone lands, because the milestone was never going to land against a definition that keeps being clarified mid-flight.

The definition-layer response is inexpensive in the currency that matters most here: no new tools, no new headcount, no culture initiative. It is not costless in the currency of time and candor: it costs two hours of stakeholder attention before work is allowed to start, and it costs the PM a harder conversation with leadership that names a structural gap instead of promising more effort. The execution frame's defenders are correct that daily standups and a new tool are easier to introduce than a signed definition-of-done session. The definition frame's defenders are correct that a sixth execution-layer fix aimed at the wrong layer produces a sixth miss, not a smaller one.

FINAL VERDICT

Sarah was solving the right problem stated the wrong way. Three consecutive misses across five different execution-layer fixes is not a signal to try a sixth. It is the empirical confirmation that the fix belongs one layer up, at the definition the milestone was built on. A two-hour definition session before the next milestone, a formal change decision gate during it, and a different escalation sentence with leadership are the entire intervention. That is the sequence with a chance of actually holding a date.

FOR THE PRACTITIONER

This case is a canonical example of why the method matters at the scale of a single team, not just a national policy. The same diagnostic that catches DOGE, Boeing, Brexit, the Iran strikes, the Fixed-Price Order, and the labor market's displacement frame catches Sarah's scheduling frame here. The fix that is fastest to try is rarely the fix that reaches the layer where the problem originates, and the method's job is to make that gap visible after the third miss, not the thirtieth.

What Changes on Monday Morning

For a Project Manager

Execution frame: run another standup, try a new tracking tool, escalate for more people. **Definition frame:** block two hours with the stakeholder before the next milestone starts, and leave that session with a written, testable definition both parties sign. Nothing else changes until that exists.

For a Stakeholder

Execution frame: clarify what you actually want once you see the work in progress, informally, whenever it occurs to you. **Definition frame:** commit to a specific, testable definition of done before work begins, and treat any later change as a decision with a cost, not a quick ask.

For a Leadership Sponsor

Execution frame: ask the team to try harder and report on progress more often. **Definition frame:** ask whether the milestone's definition was stable enough to hold a date before asking whether the team executed well against it.

— Avansys Lab, August 2026. This document is a complete ReFrameX© case study: the five-pattern analysis, a five-layer system structure, an empirical validation using Sarah's own five prior fixes, and a forward-looking three-intervention design sequenced into five phases. Methodological, not partisan. Sarah is a composite of real project management situations. Companion case studies: The Labor Market Crisis, The Iran Conflict 2025–2026, and Trump's Fixed-Price Order.