Forward Deployment Is a Leverage Decision, Not a Service Tier

Omid Razavi, Ph.D. | Founder, SuccessLab

AI has exposed the most expensive gap in enterprise software: the distance between what a product can technically do and the value customers see in production. A product can pass the demo, clear procurement, and still fail to change how work gets done.

The breakdown lives in the operating layer: workflow design, data quality, governance, evaluation, user trust, escalation logic, and the daily discipline that turns AI into results. That is the gap forward deployment exists to close.

Start with a definition.

Forward deployment is the deliberate placement of a vendor's most capable technical people inside a customer's live operation: not to deliver a project, but to turn one customer's real-world complexity into production value the customer could not reach on its own, and into product and market knowledge the vendor can reuse elsewhere.

That second half is where the discipline begins.

Forward deployment should be funded only when it creates one of three assets: customer value that would not otherwise reach production, product capability that can be reused, or a market pattern that can scale.

If the work creates none of these, call it services and manage it that way. Saying no to forward deployment protects the talent that lets you say yes to the next customer.

None of this devalues services. A well-run professional services business is a deliberate, margin-positive engine for customer outcomes. The two motions simply answer different questions. Services is paid to deliver an outcome for one customer. Forward deployment is funded to create an asset for the company. Both create value. Confusing them weakens both.

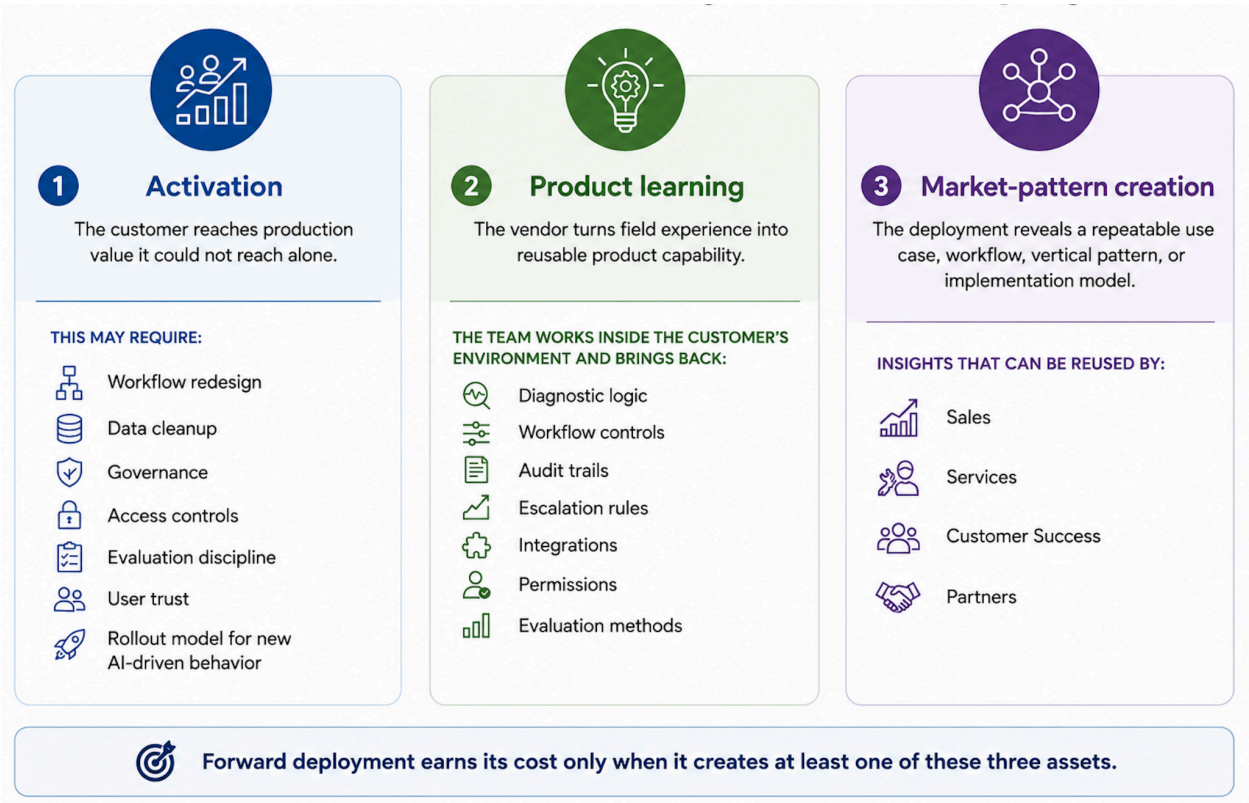
The Three Assets That Justify Forward Deployment

Activation. The customer reaches production value it could not reach alone. This may require workflow redesign, data cleanup, governance, access controls, evaluation discipline, user trust, and a rollout model for new AI-driven behavior.

Product learning. The vendor turns field experience into reusable product capability. The team works inside the customer's environment and brings durable improvements back into the product: diagnostic logic, workflow controls, audit trails, escalation rules, integrations,

permissions, and evaluation methods. The goal is not to leave one-off configurations behind. It is to make the product better.

Market-pattern creation. The deployment reveals a repeatable use case, workflow, vertical pattern, or implementation model that sales, services, customer success, and partners can reuse.



The Three Assets That Justify Forward Deployment

Forward deployment should not be assigned because an account is large, loud, political, or late. It should not absorb the cost of weak documentation, shaky onboarding, immature partners, or sales promises. Those are operating problems. Treating them as forward deployment problems burns scarce talent and hides the signal leaders need to fix the system.

Forward Deployment Is an Executive Allocation Question

The enterprise AI adoption gap is no longer theoretical.

Most companies are not failing because AI tools are unavailable. They are failing because the operating system around them is not ready: workflows, governance, data access, evaluation, and decision rights.

McKinsey's 2025 State of AI research and BCG's AI Radar land on the same conclusion: value comes from rewiring workflows and governance rather than layering AI tools onto existing processes. Most companies invest. Far fewer track hard financial results.

In customer service, the pressure is more immediate. Gartner reported in February 2026 that 91% of service and support leaders feel executive pressure to implement AI this year.

Pressure, however, is not readiness.

The scarce resource is not only engineering capacity. It is judgment about where embedded technical talent can create reusable advantage.

The market is already formalizing the motion. OpenAI launched the OpenAI Deployment Company. Palantir remains the earlier reference point, with Forward Deployed Engineers (FDEs) working closely with customers to integrate data, configure platforms, and translate operational needs into deployed software.

The reason is simple: AI products discover their real requirements in production.

The Forward Deployment Decision Matrix

The decision turns on two questions:

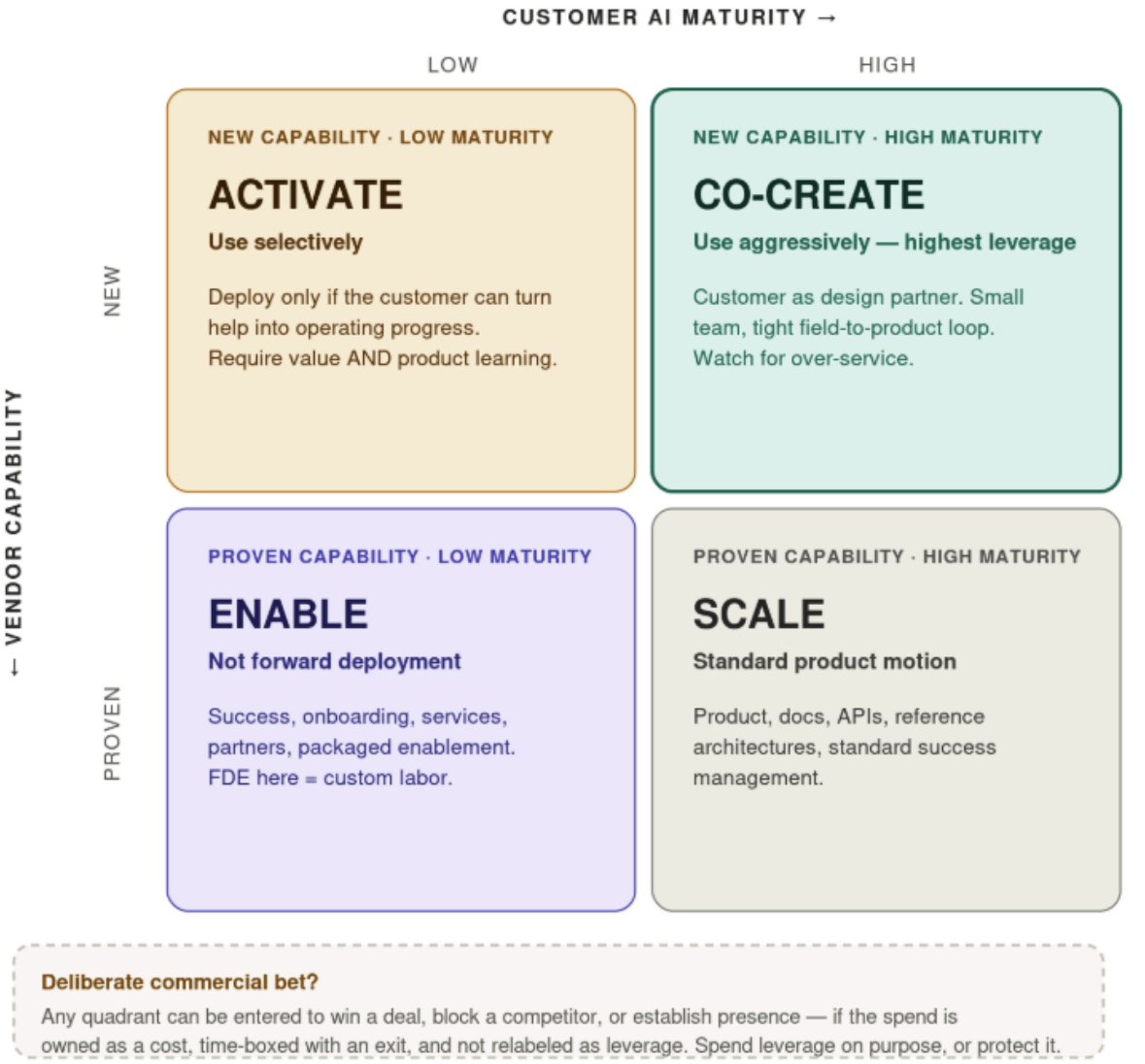
1. How proven is the vendor capability?

Has the vendor delivered this before, or is it still learning the product, workflow, feature set, or implementation pattern?

2. How mature is the customer?

AI maturity is not enthusiasm. It is the ability to provide data access, sponsorship, workflow ownership, governance, evaluation discipline, and decision speed.

Together, those dimensions create four motions: activate, co-create, enable, and scale.



The Forward Deployment Decision Matrix — vendor capability × customer AI maturity.

1. New Capability, Low Customer Maturity: Activate

Use forward deployment sparingly.

Both sides need help, but need alone is not enough. A low-maturity customer can absorb enormous field capacity and still fail without sponsorship, operational ownership, data access, and decision speed.

The question is whether the customer can turn intense field support into operating progress.

Use forward deployment here only when the customer reaches production value and the vendor learns what the product still needs. The bar is customer value plus product learning.

2. New Capability, High Customer Maturity: Co-Create

This is often the highest-return quadrant.

The vendor is building something new, and the customer has the maturity to move fast. A mature customer can pressure-test the use case, expose edge cases, sharpen evaluation criteria, and help convert field learning into reusable product capability.

The caution is over-servicing. A capable customer needs speed and a tight field-to-product loop, not a large permanent team.

3. Proven Capability, Low Customer Maturity: Enable

The capability is proven, and the path is known. The problem is adoption, not product uncertainty.

This customer needs packaged enablement, services, partners, and customer success, not scarce forward deployed talent.

If the motion is repeatable, package it into playbooks, accelerators, reference architectures, and certified delivery.

4. Proven Capability, High Customer Maturity: Scale

The vendor has delivered the capability many times. The customer knows how to adopt.

Documentation, APIs, reference architectures, partners, and the customer's internal center of excellence should carry the work.

If the vendor learns nothing new and the customer can adopt without deep field help, forward deployment spends capacity instead of creating an asset.

Saying yes may feel customer-centric. Saying no may be the more strategic decision.

The Lighthouse Test

A lighthouse account is a highly visible customer whose success can create proof for the market, open a segment, or establish a repeatable pattern.

Lighthouse accounts can justify an exception.

A visible deployment can create proof, define a category, open a segment, or change market perception. That can be worth the investment.

But leaders should separate lighthouse work from vanity deployment.

A vanity deployment is sent because the account is large, urgent, political, or loud. A lighthouse deployment sends the team because cracking the workflow can create a reference, a reusable implementation model, and a path into a segment.

Before making the exception, answer three questions:

1. What pattern are we trying to prove?
2. What segment does it unlock?
3. How will the learning be captured and reused?

Revenue pressure or competitive dynamics may still justify a low-leverage deployment. That can be a sound commercial bet.

But call it what it is.

If the work only unblocks one account, treat it as a commercial cost. Time-box it, define the exit, and do not count it as strategic reuse.

How Forward Deployment Compounds

“Bring learning back to the product” is easy to say. It is much harder to make true.

Forward deployment compounds only when field work changes what the company can repeatedly sell, ship, implement, govern, or support.

Two disciplines make that possible.

Field-to-product loop: Each deployment should answer one question:

What did we build by hand that should become product?

The answer may be a feature, control, integration, evaluation method, governance rule, permission model, implementation pattern, or partner playbook. If there is no answer, ask why the work required forward deployment at all.

Graduation: Co-create becomes scale once the capability is proven, the pattern is documented, and the standard organization can carry it. Define up front when the field team exits and the account moves to customer success, services, partners, or the standard product motion.

Without graduation criteria, forward deployment becomes permanent staffing on problems the company claims to have solved.

Bespoke field work can also create a shadow product layer: account-specific prompts, logic, data flows, controls, permissions, workflows, and escalation rules outside governed architecture.

In AI, undocumented field logic becomes operational variance, customer exposure, and product debt.

The Talent Constraint

Forward deployment depends on rare talent.

The constraint extends beyond technical depth. It requires judgment under customer pressure: the ability to read a live operation, build against imperfect systems, earn trust across functions, and still see the reusable pattern behind the immediate problem.

Think of it as a team archetype. Depending on the work, it may span solution architects, applied AI engineers, field engineers, technical consultants, and product-minded services leaders who can move between workflow detail and executive conversation.

The tension is real. Forward deployed teams must solve the immediate problem without becoming captive to one account. They must earn trust without turning every request into product direction. They must convert field complexity into learning the company can reuse.

Using this talent for weak onboarding, missing documentation, partner gaps, sales promises, or routine escalation work wastes scarce judgment and hides the operating problems leaders should fix elsewhere.

Use this talent where the answer can become product, pattern, or playbook.

Measuring Whether It Compounds

If reuse is the test, it has to be measured.

Track four signals:

Reuse rate: How much field-built work becomes product capability, documentation, implementation pattern, partner playbook, or repeatable service motion.

Time-to-repeatability: Whether each deployment of the same pattern requires fewer field hours than the last. A falling curve shows learning. A flat curve signals custom work.

Graduation rate: How many co-create accounts move into the standard motion within a defined window.

Customer production outcome: The business result the customer could not reach alone, such as resolution quality, self-service rate, transactional NPS, cost-to-serve, or adoption of a new workflow.

Track the economics with the same discipline. Forward deployment becomes risky when it is funded as product discovery but operates like low-margin services. Watch gross margin, utilization, attach, and cost-to-serve.

Funding follows the same logic. If forward deployment exists to produce reusable product capability, fund it from the product or R&D line, and govern it there. Work funded as services gets managed for margin and utilization. Work funded as product gets managed for learning and reuse. The budget line decides which behavior you get.

The warning sign: field capacity grows faster than reusable product capability. At that point, the company is building a services dependency, not a deployment advantage.

Compounding requires better customer outcomes and a more repeatable next deployment.

The Operating Rule

Before assigning forward deployment, ask three questions:

1. Will this unlock production value standard adoption cannot?
2. Will this teach us something product should absorb?
3. Will this create a repeatable segment, vertical, or workflow pattern?

If none are yes, use another motion. If one is yes, treat it as borderline and look for a cheaper way to de-risk it. If two or more are yes, forward deployment is likely the right call.

Use your best field talent where the answer can become product, pattern, or playbook.

What Customer Leaders Should Do Differently

Every customer wants bespoke attention. Few merit forward deployment.

A customer earns it by bringing sponsorship, workflow access, data readiness, governance discipline, and the ability to operationalize what gets built.

Size, dissatisfaction, complexity, and quarter-end urgency are not enough.

The better question is not “How much help does this account need?” It is, “Will this work create reusable learning?”

FDE-Washing

The most common failure mode is relabeling.

FDE-washing happens when custom services, escalation cleanup, or deal-driven deployment is renamed as strategic forward deployment.

The signs are easy to spot: no pattern to prove, no learning mechanism, no product owner, no graduation path, and no exit. A field team stays inside one account for months, fixes the same issues by hand, and little changes in the product, documentation, partner motion, or standard implementation path.

The issue is not the occasional commercial exception. Revenue pressure, competitive dynamics, or lighthouse value can justify one. The issue is confusing account importance with reusable output.

Once non-reusable work is counted as strategic, the system loses discipline. Reuse rate gets overstated. Time-to-repeatability loses meaning. Graduation rate gets ignored.

The company pays twice: it consumes scarce technical talent and avoids fixing the operating problem that created the need.

The Executive Takeaway

Forward deployment is a scarce-capacity decision. It earns its cost only when customer complexity becomes an asset the company can reuse.

Use it when standard adoption will not get the customer to production value, when field learning should become product capability, or when the deployment can prove a repeatable market pattern.

Do not use it to absorb weak onboarding, missing documentation, partner gaps, sales overreach, or routine escalation work. Those belong in customer success, services, enablement, partners, documentation, or product-led scale.

The strongest AI companies will treat forward deployment as a learning system: a way to convert production reality into better product, sharper patterns, and faster repeatability. The weaker ones will turn it into bespoke services and call it strategy.

The rule is this: fund forward deployment only when customer complexity can become reusable advantage.

Otherwise, call it services and manage it that way.

Sources

- McKinsey, *The State of AI 2025: How Organizations Are Rewiring to Capture Value*
- BCG, *Closing the AI Impact Gap (AI Radar 2025)*
- Gartner, *91% of Customer Service Leaders Under Pressure to Implement AI in 2026* (Feb 2026)
- OpenAI, *OpenAI Launches the Deployment Company* (approximately 150 forward deployed engineers and deployment specialists via the Tomoro acquisition; the $4B+ backing is per Reuters and other press reports)
- Palantir, *A Day in the Life of a Palantir Forward Deployed Software Engineer*

About the Author:

Omid Razavi, Ph.D., is the founder of SuccessLab, an advisory and community brand at the intersection of AI transformation and enterprise customer leadership. He publishes CCO Perspectives, organizes the SuccessLab Executive Forums and Roundtables, and advises revenue and customer leaders across the enterprise software industry.