

Solving the Enterprise AI Last-Mile Problem with Forward-Deployed Engineers

Enterprise AI often stalls between a successful prototype and meaningful adoption. Forward-Deployed Engineers can close that last-mile gap by combining technical depth, business context, and hands-on delivery.

04 OCT 2026 · 22 MIN READ



Darren Sim

Senior technology & product leader

All content published here is shared in a personal capacity. Nothing on this site should be interpreted as an official statement, endorsement, or communication from my employer.

Enterprise AI rarely fails because the underlying AI system is not powerful enough. More often, it fails in the final stretch—the difficult journey from a promising prototype to a dependable solution that people trust, adopt, and use in their daily work.

This is the **enterprise AI last-mile problem**.

Organizations can buy capable AI systems, build impressive demonstrations, and announce ambitious AI strategies. Yet real value only appears when AI is connected to business processes, enterprise data, security controls, existing platforms, and the realities of how work gets done. That final integration requires more than a conventional handoff between a product team and the business. It requires people who can work across both worlds.

This is where **Forward-Deployed Engineers** can play a critical role.

What is a Forward-Deployed Engineer?

A Forward-Deployed Engineer is a hands-on technical owner who works directly with users and business teams to turn high-value problems into trusted, production-ready capabilities.

The role combines four disciplines that enterprises have traditionally separated: **management consulting, solution architecture, AI-enabled software engineering, and product and change leadership.**

- **Management consulting**, to understand the operating context, frame ambiguous problems, identify where value exists, and define measurable outcomes
- **Solution architecture**, to design the complete solution across AI systems, data, applications, security, governance, reliability, and enterprise platforms
- **AI-enabled software engineering**, to rapidly prototype, evaluate, build, deploy, and operate production-quality solutions
- **Product and change leadership**, to build alongside users, navigate organizational, security, risk, and governance constraints, drive adoption, learn from real usage, and turn recurring needs into reusable capabilities

THE FORWARD-DEPLOYED ENGINEER

From technical capability to business impact



Discover → **Build** → **Deploy** → **Adopt**

Turn field learning into a stronger product or platform.

They stay close to the problem from discovery through production adoption, navigating enterprise constraints and moving continuously between workflows, architecture, code, and user feedback. They also turn field learning into reusable capabilities that improve the broader product or platform.

The defining characteristic is not simply proximity to the user. It is **end-to-end ownership of the gap between technical capability and business impact**.

Why enterprise AI has a last-mile problem

A GenAI capability can perform well in isolation and still fail inside an enterprise. Production environments introduce complexity that a proof-of-concept (POC) rarely captures.

An MIT NANDA study found that **only about 5%** of integrated enterprise GenAI pilots delivered **measurable financial value**, while the remaining **95%** showed **no measurable P&L impact**. The report attributed this divide less to the quality of the AI system or regulation than to the **implementation approach**—including **brittle workflows**, **limited contextual learning**, and **poor operational integration**. (MIT NANDA, 2025)

A major reason is the mutual context gap between business and technology teams.

- **Business teams** understand the problem, the workflow, and the desired outcome, but may not know how AI can address them.
- **Technology teams** understand AI, architecture, and engineering, but may not fully understand the operating context, exceptions, and outcomes that matter to the business.

Traditional delivery approaches attempt to bridge this divide through requirements documents, project handoffs, and layers of coordination. That can **work for stable and well-understood systems**. Enterprise AI, however, is highly iterative: the problem and the solution often need to be discovered together as users interact with working software.

That gap becomes visible in production. A solution may need to retrieve information from fragmented systems, respect fine-grained access controls, produce auditable outputs, handle exceptions, meet latency expectations, and fit naturally into an existing workflow. It must also **earn the confidence of users** who may be accountable for decisions influenced by its output.

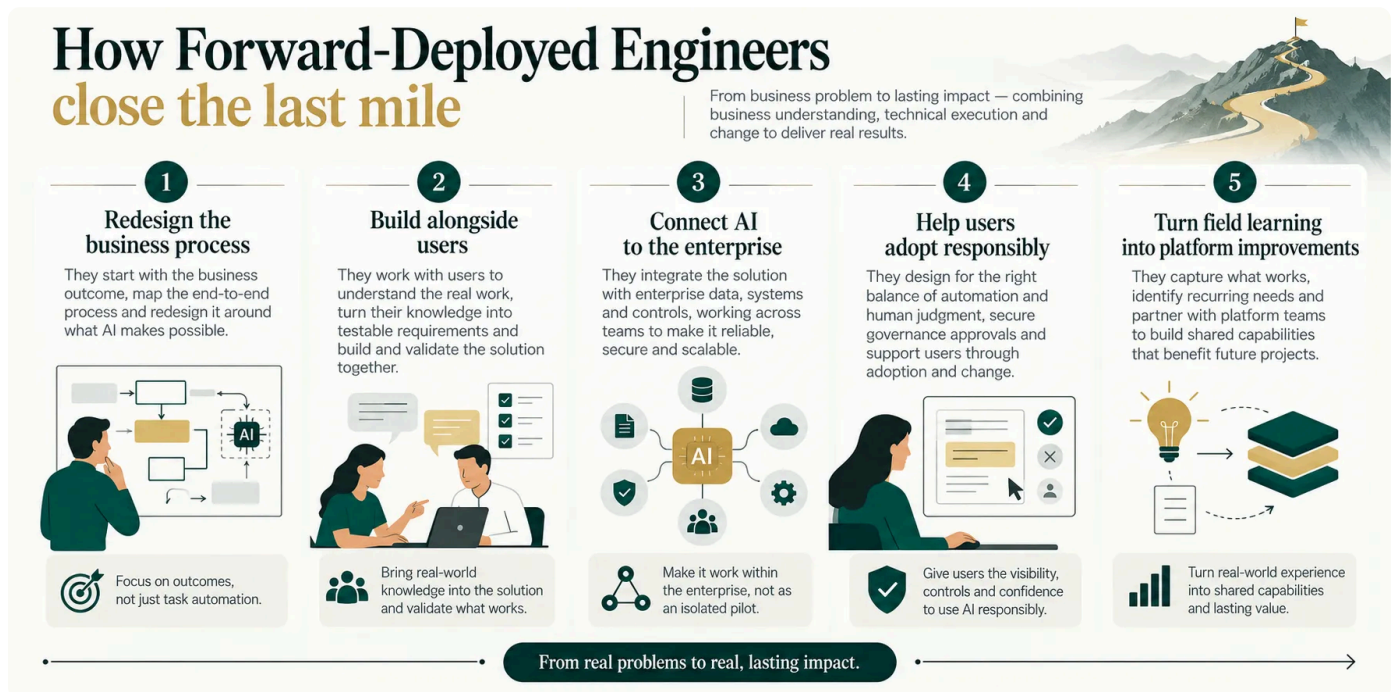
This mutual context gap is compounded by four others:

1. **The integration gap:** The prototype works independently but is disconnected from enterprise systems and data.
2. **The trust gap:** The output appears useful, but users cannot easily verify, explain, or rely on it.
3. **The adoption gap:** The capability exists, but it does not fit how people actually work.

4. **The ownership gap:** Multiple teams contribute, but no one owns the outcome from end to end.

Forward-Deployed Engineers shorten that learning loop.

How Forward-Deployed Engineers close the last mile



1. They redesign the business process, not just deploy the AI system

The greatest value from enterprise AI **may not come from automating individual tasks within an existing process**. It **comes from redesigning the process around capabilities that were not previously available**. McKinsey's 2025 *State of AI* survey found that workflow redesign had the strongest relationship with reported EBIT impact among 25 organizational attributes studied. Harvard Business Review similarly argues that companies often struggle to generate enterprise-wide returns because they **automate isolated tasks rather than redesign the workflows that create business value**. (McKinsey & Company, 2025; Shunko & Netessine, 2026)

This is where management-consulting skills become essential. Before selecting a technical solution, the Forward-Deployed Engineer must **structure an ambiguous problem, map the end-to-end process**, identify root causes, quantify the opportunity, and challenge assumptions

about how the work must be performed. Without this discipline, teams risk using AI to automate visible symptoms rather than redesigning the process that produces them.

The starting question is therefore not, “Where can we deploy an AI system?” It is, **“Which business outcome are we trying to improve, and how should the underlying process work now that AI is available?”**

Forward-Deployed Engineers **examine the process end to end** rather than inserting AI into an existing task. They **identify which activities can be eliminated, automated, augmented, or resequenced**; where human judgment and accountability must remain; and which controls are required. By prototyping alongside users, they progressively redesign both the technology and the way work gets done—a test-and-learn approach consistent with guidance from MIT Sloan and IBM Consulting. (Stackpole, 2024; Limbert, 2026).

2. They build alongside users

A well-defined business process gives us a clear starting point for AI implementation. However, knowing how the work should happen does not tell us how reliably AI can support it. While we can specify the intended outcome, workflow and controls upfront, then test the AI system against representative inputs, exceptions and agreed **acceptance criteria**, working alongside users helps us establish where AI adds value, where **human review** remains necessary and how the solution fits into daily work.

That proximity also helps us understand the work beyond the documentation. In large enterprises, experienced employees may spend years developing checks, out-of-system efficiencies and ways of handling exceptions. These practices become **muscle memory**, yet may never appear in official business processes or standard operating procedures. By working alongside users, we bring this **tacit knowledge** into view and assess which practices the AI-enabled solution should preserve, change or replace.



Invoice Approval Example

Consider an invoice-approval process with documented steps. A finance specialist may recognize when an apparently routine discrepancy warrants closer scrutiny, even if the procedure does not capture that judgment. Working with engineers, finance specialists assess whether an AI-generated explanation correctly identifies the discrepancy and provides enough evidence for the next decision; their contribution goes beyond interface feedback to translating established policies and process requirements into **concrete tests**. They identify the evidence a recommendation needs, the omissions that matter and the exceptions that require escalation. The aim is to make **existing standards testable** in the AI-enabled solution, rather than rediscover the process.

That assessment helps define the boundary between assistance and action: a useful recommendation is not necessarily safe to automate, so teams need to validate AI's **autonomy**, when it needs **human approval** and how users correct outputs or take over when it fails. Anthropic's agent-building guidance emphasizes clear success criteria, feedback loops and meaningful human oversight, reinforcing this collaboration. ([Anthropic, 2024](#))

Addressing these decisions in **short cycles**, Forward-Deployed Engineers turn user observations into evaluation cases, design changes and functioning software, testing routine cases and consequential exceptions against agreed acceptance criteria and existing practice. Cycles test whether the solution is reliable and useful enough to expand scope or justify scaling investment.

3. They connect AI to the enterprise

An AI system does not become an **enterprise capability** simply because it produces a useful answer. It needs to work with the organization's **data, applications** and **controls**, and support the business process from beginning to end. This may involve data pipelines, retrieval mechanisms, APIs, identity controls, evaluation frameworks, monitoring, human-review steps and user interfaces.

In practice, these components rarely sit within one team's control. Information may be fragmented across systems, data quality may vary, and legacy applications may impose integration constraints. Forward-Deployed Engineers work with business owners, application teams, platform engineers, software architects, security and governance specialists to resolve these dependencies. Their role is to connect the pieces without bypassing the controls or ownership that make the enterprise function.

That responsibility extends beyond integration. The solution needs clear behavior when information is unavailable, an integration fails or a human must intervene. It also needs an agreed support approach and a long-term owner. The FDE brings these requirements together so that the result works within the enterprise, rather than remaining an isolated demonstration.

4. They Help Users Adopt AI Responsibly

Connecting AI to enterprise systems establishes the technical foundation; helping people use it responsibly and comfortably is the next part of the work. Alongside the integration and support arrangements, users need to:

- Understand what the system is authorized to do
- Inspect the evidence behind its recommendations
- Challenge or correct those recommendations

Forward-deployed engineers help the business **secure approvals** from relevant enterprise Gen-AI Governance councils, required to put the solution into use. They work with users, security, risk and governance teams from the outset to understand approval requirements and agree on a design, controls and supporting evidence that meet them. Rather than treating governance as a final hurdle, they resolve concerns with the responsible reviewers as the solution takes shape, then coordinate the submission and address any remaining findings through to an approval decision. Their role is to build a solution that earns approval through **collaboration** and **demonstrated compliance**, not to bypass requirements or pressure reviewers. Approval authority remains with the designated reviewers, and accountability remains with the business owner.



Invoice Approval Example

In the invoice-approval example, these requirements shape how the solution works. If the AI system flags a discrepancy and recommends withholding payment, the user should be able to inspect the invoice, purchase order and relevant policy before deciding what to do. Where approval is required, the system should request it and record the recommendation, supporting evidence and resulting decision. If the evidence is insufficient or the case falls outside its operating boundaries, it should refer the case for human review rather than proceed automatically.

These safeguards give users a way to reject a recommendation, correct an error or take over when the system cannot complete a task reliably. However, providing those options does not tell people when or how to use them. Alongside business owners and change-management teams, forward-deployed engineers help users understand AI's capabilities and limitations, practice the revised process and recognize how their responsibilities change.

Through **change management**, the team helps turn these controls into everyday behavior. Training, hands-on practice and clear guidance prepare users to review, approve, correct or escalate as appropriate, while ongoing feedback allows the team to address concerns and refine the solution. The aim is to help people adopt AI within clear limits, retaining the judgment and accountability their work requires.

5. They Turn Field Learning into Platform Improvements

Working close to business problems, forward-deployed engineers identify requirements that are unique to an engagement and needs that recur across products or projects. With platform architects, platform teams and principal engineers, they agree which should remain solution-specific and which belong in the shared platform. Architects and principal engineers help define the build boundaries and review shared designs; they do not necessarily form a separate build team.

FDE teams then build the business-specific implementation using supported platform capabilities rather than recreating them. That implementation remains with the product or project it serves. Where a recurring need warrants a shared capability, the relevant platform or engineering team should build it and own its design, integration and ongoing maintenance. An FDE engagement may provide an initial implementation, evaluation evidence or practical lessons to inform that work.

This division creates a feedback loop between business delivery and platform development. It helps the platform evolve around demonstrated needs and gives subsequent teams a stronger foundation, without producing an endless series of standalone systems or turning the FDE function into a separate platform-development organization.

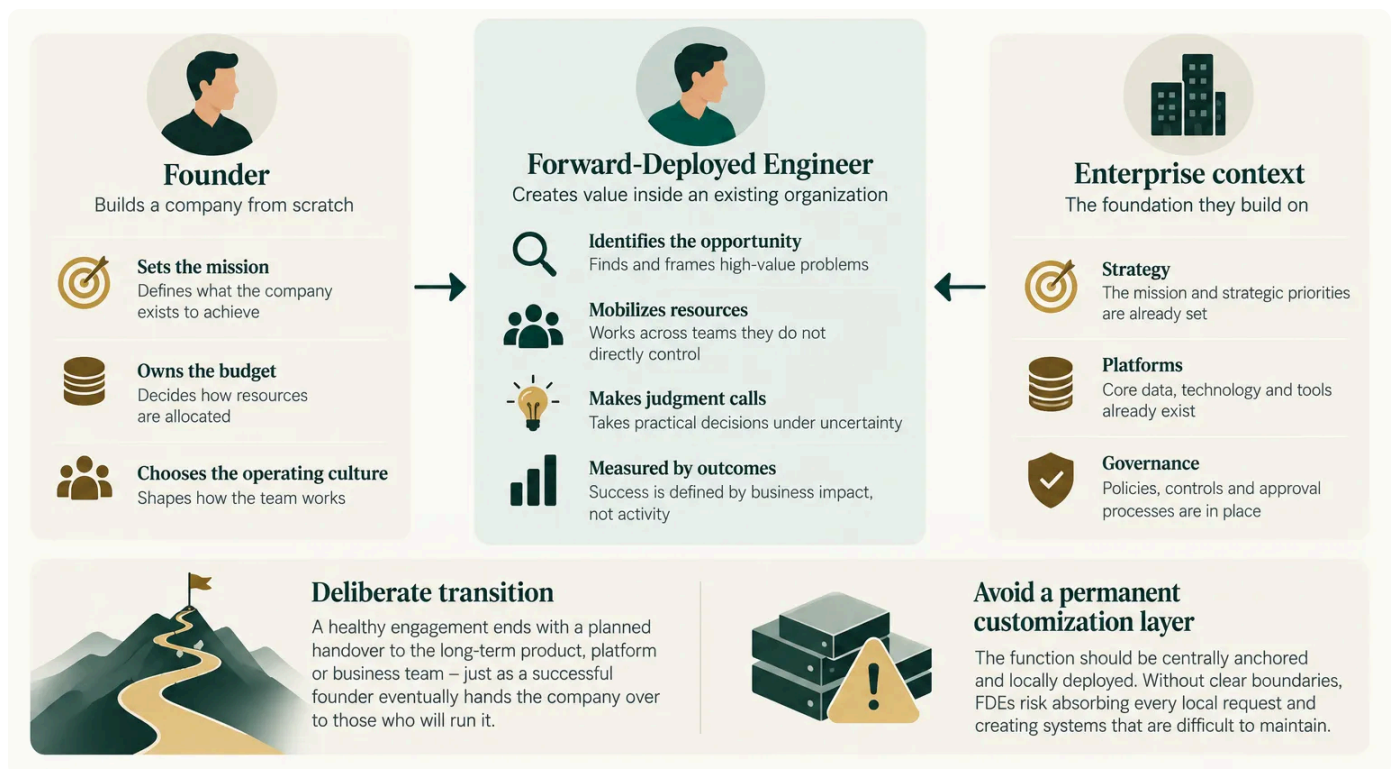
How forward-deployed engineering should be organized

Forward deployment is not simply a role design; it is an operating approach and mindset.

Auseful way to think about it is a **founder operating inside someone else's company**.

A Forward-Deployed Engineer identifies the opportunity, mobilizes resources they do not directly control, makes judgment calls under uncertainty, and is measured by outcomes rather than activity. Unlike a founder, however, they do not set the mission, own the budget, or choose the operating culture—they create value inside an organization whose strategy, platforms, and governance already exist.

The role borrows the founder's **bias toward action** and **end-to-end accountability**, while accepting that the enterprise keeps ultimate ownership of the outcome. And just as a successful founder eventually hands the company over to those who will run it long term, a healthy forward-deployed engagement ends with a **deliberate transition** to the long-term product, platform, or business team.



There is a risk that Forward-Deployed Engineers become a **permanent customization layer**, absorbing every local request and creating **systems that are difficult to maintain**. The function should instead be **centrally anchored and locally deployed**, drawing on the engagement model that Google has refined for its Site Reliability Engineers (SREs), which we discuss in a later section.

When forward-deployed engineering is not the right fit

Forward-deployed engineering **should not become the default delivery approach for every AI initiative**. It is most valuable when a technically complex or highly configurable capability must be translated into an outcome for business users who cannot implement it independently. If that condition does not exist, a conventional product, platform, engineering, or consulting team may be more appropriate.

An FDE engagement may not be the right fit when:

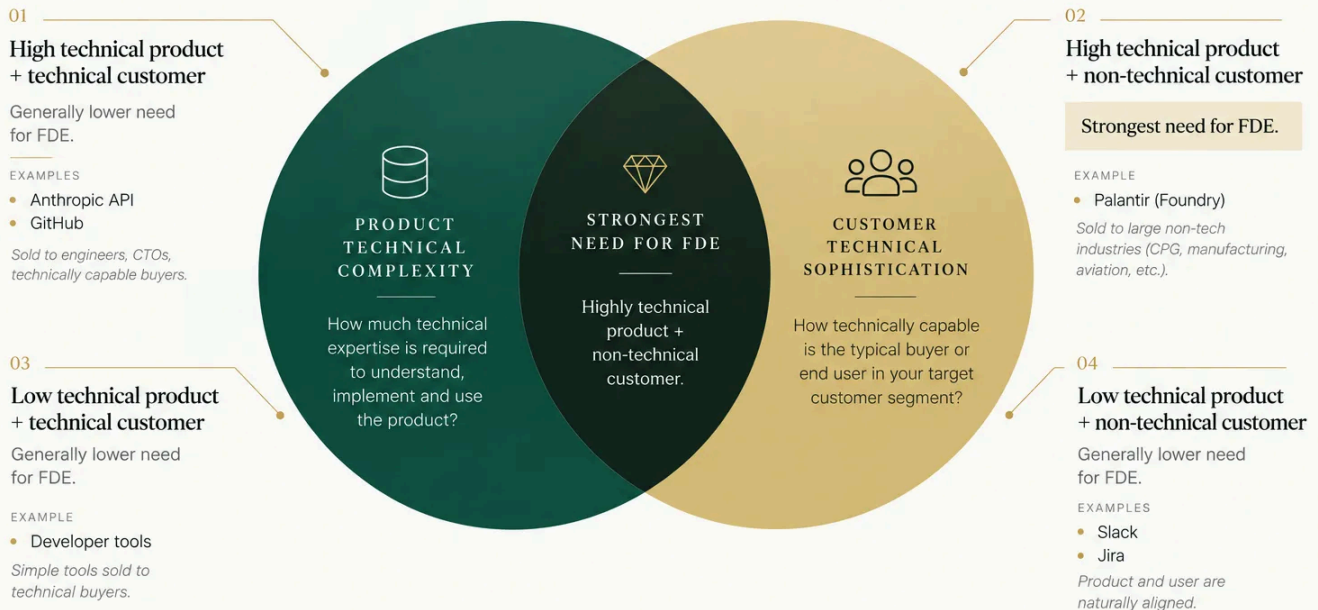
- A standard product already meets the need.** If the problem can be addressed through configuration, training, or normal product adoption, embedded engineering adds unnecessary cost.

- **The receiving team has sufficient technical capability.** A technically mature product or engineering team may be able to adopt the platform directly with documentation and architecture support.
- **The problem and solution are already well understood.** Stable requirements may be better served through normal product delivery rather than an exploratory, embedded engagement.
- **There is no shared platform or reusable foundation.** If every engagement must be built independently, the function risks becoming a custom development shop with unsustainable maintenance costs.
- **The expected value does not justify dedicated engineering capacity.** Embedded deployment should be reserved for outcomes significant enough to warrant scarce multidisciplinary talent.
- **There is no accountable business owner.** FDEs cannot compensate for unclear ownership, unavailable users, or a lack of commitment to process and organizational change.
- **The need is permanent operational support.** Once the solution and operating process have stabilized, long-term ownership should pass to a product, platform, or business team.

The objective is not to deploy an FDE wherever AI is involved. It is to use forward deployment selectively where proximity, experimentation, technical depth, and process redesign materially increase the probability of achieving a valuable outcome.

When do you need a Forward Deployed Engineering function?

FDE is most valuable when there's a gap between a technical product and a non-technical customer.



A shared platform is a prerequisite for scale

Forward-deployed engineering scales only when teams build primarily from shared platform capabilities rather than creating every solution independently. The platform should provide reusable technical primitives, integration patterns, evaluation methods, security controls, and operational foundations that FDEs can assemble and extend for a particular business context.

Without that foundation, each engagement produces another standalone application, codebase, or integration that must be maintained separately. The FDE function gradually becomes a custom development shop, and the cost of supporting bespoke solutions grows faster than the value it creates.

A shared platform does not eliminate customization. It defines the boundary between what should remain specific to a business process and what should become a reusable enterprise capability. Customer- or domain-specific requirements may remain local, while repeated patterns should progressively move into the shared platform.



Example from practice: Palantir's Dev/Delta split

Palantir separates the two sides of this boundary between its "Devs" (platform teams) and "Deltas" (its forward-deployed engineers):

- **Platform teams (Devs):** own platform components end to end and build capabilities that are "deployed to many different customers for many different use cases."
- **Forward-deployed engineers (Deltas):** work with a single customer to deploy and customize those existing platforms against critical business problems, without rebuilding the platform itself—"we are actually deploying existing software products to achieve the customer's outcomes."

Palantir summarizes the division as: "You can think of a Dev's focus as 'one capability, many customers,' while a Delta's focus is 'one customer, many capabilities.'" ([Palantir, 2019](#))

Centrally anchored, locally deployed

A central Forward-Deployed Engineering (FDE) home team can own talent development, delivery methods, quality standards, reference architectures, reusable components, engagement intake, and relationships with platform, security, and governance teams. FDEs or small multidisciplinary pods can then embed with business and product teams for defined periods, while remaining connected to their professional home.

This resembles Google's Site Reliability Engineering (SRE) engagement approach, which ranges from consultation and architecture guidance to part-time and full-time assignments. For full-time engagements, Google prefers embedding an SRE within the product-development team and recommends shared goals, delivery milestones, regular feedback, and explicit disengagement planning. ([Google, n.d.](#))

The same pattern is visible among leading AI companies. OpenAI and Anthropic describe FDEs as embedding with strategic customers, owning work from discovery through production, and returning field learning to central product and engineering teams. Palantir similarly positions FDEs around customer outcomes while encouraging reusable capabilities to flow back into the core product. (OpenAI, n.d.; Anthropic, n.d.; Palantir, 2019)

A tiered engagement approach

Not every business need requires a fully embedded engineer. A forward-deployed function can offer several levels of engagement:

1. **Advisory:** Opportunity assessment, office hours, and architecture guidance
2. **Discovery:** A short engagement to define the outcome and redesign the business process
3. **Embedded deployment:** A dedicated FDE or multidisciplinary pod that prototypes, evaluates, and builds alongside users
4. **Production adoption:** Integration, governance, operational readiness, and change leadership
5. **Transition:** Sustainable ownership passes to the long-term product, platform, or business team
6. **Platform feedback:** Reusable components and field learning return to the broader enterprise platform

IBM Consulting describes a similar Forward Deployed Unit: a small, outcome-oriented team combining FDEs, architects, and domain specialists to drive continuous transformation rather than provide conventional advisory support. (Limbert, 2026)

A deliberate feedback loop

A healthy operating approach defines responsibilities across the enterprise:

- Forward-Deployed Engineers discover patterns through real-world delivery.
- Product teams convert repeated needs into scalable capabilities.
- Platform teams provide secure, reliable foundations.

- Business teams own outcomes and operational adoption.
- Governance teams define guardrails that enable responsible speed.

The boundary should evolve as the solution matures. Early in the journey, Forward-Deployed Engineers may do significant exploration and custom integration. Over time, stable components should move into supported products and platforms, leaving the forward-deployed team to focus on the next unresolved problem.

What makes the role effective?

Technical skill is necessary, but it is not sufficient. What distinguishes strong Forward-Deployed Engineers is everything around it: curiosity, empathy, pragmatism, and the ability to communicate with both executives and engineers. They are as comfortable in a process discussion or an architecture decision as they are in a codebase or a user-feedback session, and they have the judgment to know when to build quickly, when to pause for risk review, and when a local request should become a reusable enterprise feature.

Reduced to its essentials, the role comes down to five characteristics:

1. **Diagnose — Find the real business problem.** Listen deeply and translate symptoms into measurable outcomes.
2. **Prioritize — Focus on the highest-value intervention.** Balance customer impact, feasibility, speed and scope.
3. **Build — Turn ambiguity into working software.** Own architecture and implementation, not merely recommendations.
4. **Own — Stay accountable through production adoption.** Success is the customer outcome, not project completion.
5. **Compound — Turn field learning into product leverage.** Feed recurring patterns back into the core platform so each deployment makes the next one easier.

Of the five, the last is particularly important at the organizational level. Without it, you can build a very effective professional-services organization; with it, you create a genuine **Forward-Deployed Engineering system**. The flywheel is self-reinforcing: **customer problem** → **FDE**

discovery → rapid deployment → measurable outcome → field insight → core product improvement → faster next deployment.

What it comes down to is measurement. Most importantly, they must be measured by outcomes — not activity. Useful measures might include adoption, time saved, cycle-time reduction, quality improvements, risk reduction and the number of reusable capabilities contributed back to the broader platform.

A new role for a new delivery challenge

Enterprise AI has made experimentation easier than ever. It has not made operational transformation easy.

The hardest work still happens in the last mile: understanding the workflow, connecting fragmented systems, designing for trust, navigating enterprise constraints, and helping people change how they work.

Forward-Deployed Engineers are valuable because they bring engineering to the point where technology meets reality. They do not merely deliver AI to the enterprise. They help the enterprise turn AI into a working, trusted, and repeatable capability.

For organizations struggling to move from AI prototypes to measurable value, the answer may not be another AI system or another demonstration. It may be a new kind of engineer—one who is prepared to own the last mile.

Looking Ahead: Building an FDE Practice

What does it take to put forward-deployed engineering into practice?

In the articles that follow, we will explore how to establish the function, choose the **right engagements** and define how FDEs work with business, product and platform teams. We will also hear from **practicing FDEs** about what has worked, what has proved difficult and the **lessons** they would share with teams starting out.

References

- Anthropic. (2024, December 19). *Building effective agents*. <https://www.anthropic.com/engineering/building-effective-agents>
- MIT NANDA. (2025). *The GenAI divide: State of AI in business 2025*. https://mlq.ai/media/quarterly_decks/v0.1_State_of_AI_in_Business_2025_Report.pdf
- McKinsey & Company. (2025). *The state of AI: How organizations are rewiring to capture value*. https://www.mckinsey.com/~media/mckinsey/business%20functions/quantumblack/our%20insights/the%20state%20of%20ai/2025/the-state-of-ai-how-organizations-are-rewiring-to-capture-value_final.pdf?shouldindex=false
- Shunko, M., & Netessine, S. (2026, September 14). *Stop automating old processes. Design new ones instead*. Harvard Business Review. <https://hbr.org/2026/09/stop-automating-old-processes-design-new-ones-instead>
- Stackpole, B. (2024, November 25). *How to redesign work for the age of AI*. MIT Sloan School of Management. <https://mitsloan.mit.edu/ideas-made-to-matter/how-to-redesign-work-age-ai>
- Limbert, N. (2026, August 18). *How forward deployed engineering is redefining client transformation*. IBM. <https://www.ibm.com/think/perspectives/how-forward-deployed-engineering-is-redefining-client-transformation>
- Google. (n.d.). *SRE engagement model*. <https://sre.google/workbook/engagement-model>
- OpenAI. (n.d.). *Forward deployed engineer*. [https://openai.com/careers/forward-deployed-engineer-\(fde\)-sf-san-francisco/](https://openai.com/careers/forward-deployed-engineer-(fde)-sf-san-francisco/)
- Anthropic. (n.d.). *Forward deployed engineer*. <https://job-boards.greenhouse.io/anthropic/jobs/5302966008>
- Palantir. (2019, April 8). *Dev versus Delta: Demystifying engineering roles at Palantir*. <https://blog.palantir.com/dev-versus-delta-demystifying-engineering-roles-at-palantir-ad44c2a6e87>



— THE AUTHOR

Darren Sim

Darren is a senior technology and product leader based in Singapore. He writes about the decisions, systems, and people behind meaningful transformation across Asia-Pacific.

[More about Darren](#) ↗

[Connect on LinkedIn](#) ↗