FORRESTER®

Mainframe Application Development Maturity Model

How Mainframe And Hybrid Developers Can Evolve Their Understanding Of Organizational Maturity In An AI-Enabled World

A FORRESTER CONSULTING THOUGHT LEADERSHIP PAPER COMMISSIONED BY BROADCOM, MAY 2026

Table Of Contents

Project Team:

Madeline Harrell,

Senior Market Impact Consultant

Contributing Research:

Forrester's technology architecture and delivery research group

ABOUT FORRESTER CONSULTING

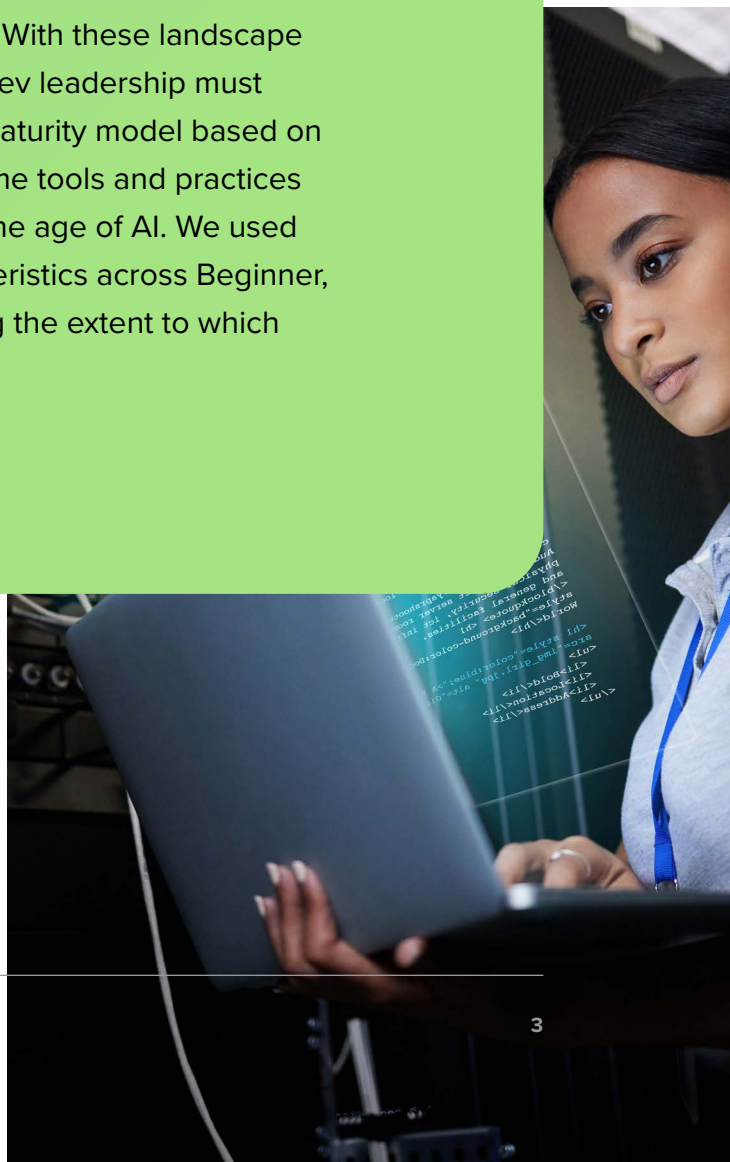
Forrester provides independent and objective research-based consulting to help leaders deliver key outcomes. Fueled by our customer-obsessed research, Forrester's seasoned consultants partner with leaders to execute their specific priorities using a unique engagement model that ensures lasting impact. For more information, visit forrester.com/consulting.

Executive Summary

Application development (app dev) leaders increasingly recognize that AI is reshaping how software is built and how success is measured.

Mainframe practice leaders, in particular, are adapting by modernizing without destabilizing. Rather than replacing legacy systems, they are evolving tooling, refactoring or encapsulating code, and integrating AI-assisted development, testing, and operations into existing environments. Metrics are shifting toward time to insight, AI integration readiness, developer productivity, and secure exposure of trusted core data to intelligent services. Modernization is no longer just about cost or technical debt — it is about making app dev AI-ready while preserving legacy strengths and unlocking new value.

In our survey of 390 global enterprise developer decision-makers across industries, we found that app dev leadership teams are prioritizing legacy tooling and code modernization, while enabling their workforce in a time of hybrid work and emerging technology adoption. With these landscape changes increasing complexity for the workforce, dev leadership must rethink their strategies for success. We created a maturity model based on respondents' adoption and use of modern mainframe tools and practices as they seek to further enable their developers in the age of AI. We used this model to identify and analyze different characteristics across Beginner, Intermediate, and Advanced maturities, recognizing the extent to which they had already modernized.



Key Findings

Leaders operationalize advanced capabilities. High-maturity organizations are far more likely to use automated code quality tools, code coverage, synthetic test data, and pipeline-integrated testing. Fewer than one-third of lower-maturity peers do so consistently, leaving risks undiscovered until late stages.

Developer experience drives performance. Organizations prioritizing modern mainframe development — IDE parity, CLI/API workflows, and streamlined onboarding — deliver faster and retain talent better. Those with manual or siloed experiences face slower onboarding and growing skills risks.

Automation depth differentiates maturity. While many have modern tools, leaders embed them into daily workflows. Laggards rely on manual processes for nonfunctional testing, rollback, and approvals, increasing lead times and developer friction.

AI impact depends on fundamentals. High-maturity organizations apply AI to code understanding, quality enforcement, and modernization, reinforcing existing practices. Laggards rely on mandates or pilots without guardrails, metrics, or integration to drive sustained performance.

Effectiveness And Maturity Gaps Become Visible

Organizations have invested heavily in modern DevOps tools but usage is inconsistent, with advanced capabilities like AI assistance underused and tool sprawl fragmenting workflows. Operationally, teams rely on a mix of automated and manual processes, leading to slow, inconsistent release cycles and limited performance visibility. Culturally, developers seek autonomy, modern tools, and less toil, while leadership prioritizes predictability and control, creating ongoing tension. Inconsistent training investment also leaves some teams unprepared to adopt emerging technologies effectively. The reassessment of enablement strategies is forcing a candid look at existing delivery models, governance structures, and skill sets, as teams confront the reality that today's "doing things right" may not translate into tomorrow's competitive advantage.

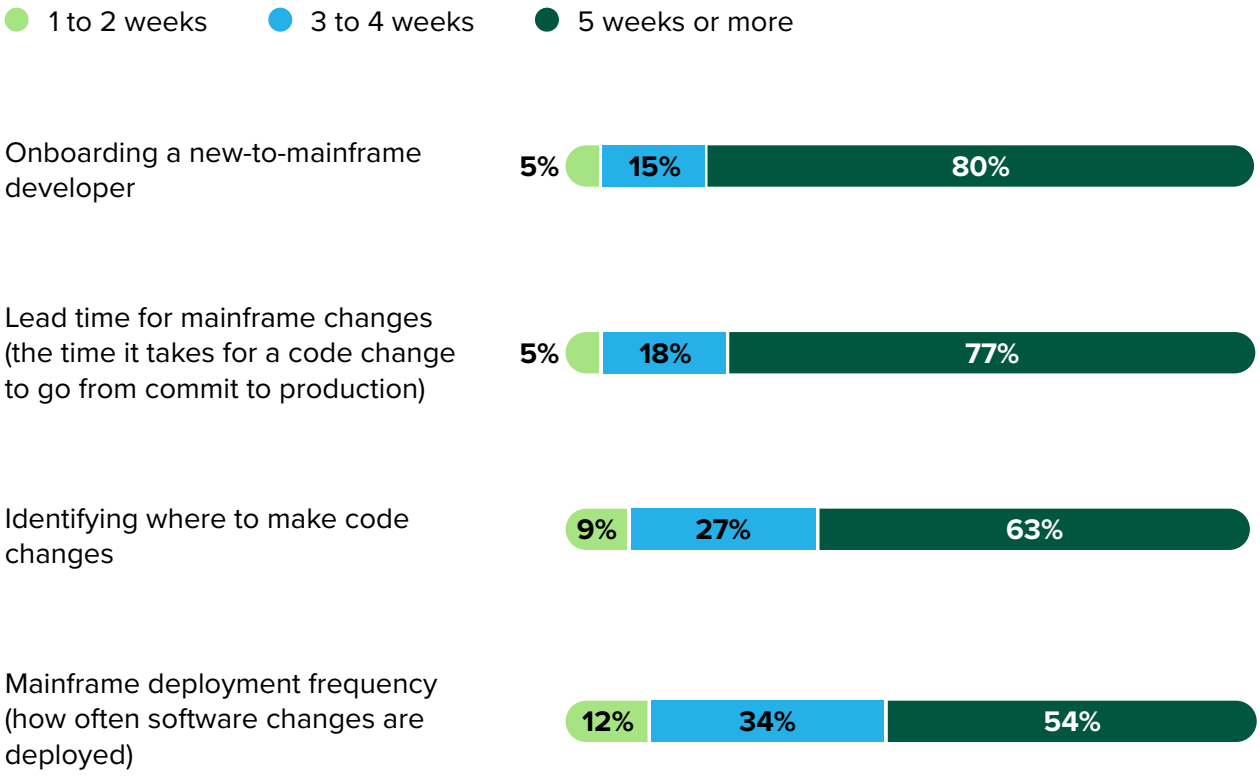
In surveying 390 app dev practice leaders at global enterprises, we found the following:

- **Effectiveness.** Respondents are generally satisfied with the quality of their teams' output; however, 44% are dissatisfied with the length of software delivery cycles and the time required to introduce new features or operability improvements. Nearly two-thirds of respondents indicate that deploying software changes into production takes five weeks or longer, highlighting a persistent gap between quality outcomes and delivery speed.

  Historically, organizations measured mainframe development success by stability, predictability, cost containment, and throughput. While these metrics remain important, the data shows they no longer suffice on their own. In the AI era, leading organizations are questioning whether legacy KPIs adequately reflect value in an environment where adaptability, data accessibility, experimentation speed, and intelligent automation are increasingly critical. Organizations that lag continue to prioritize uptime over responsiveness, resulting in longer lead times and slower improvement cycles (see Figure 1).

FIGURE 1

Time To Complete App Dev Tasks

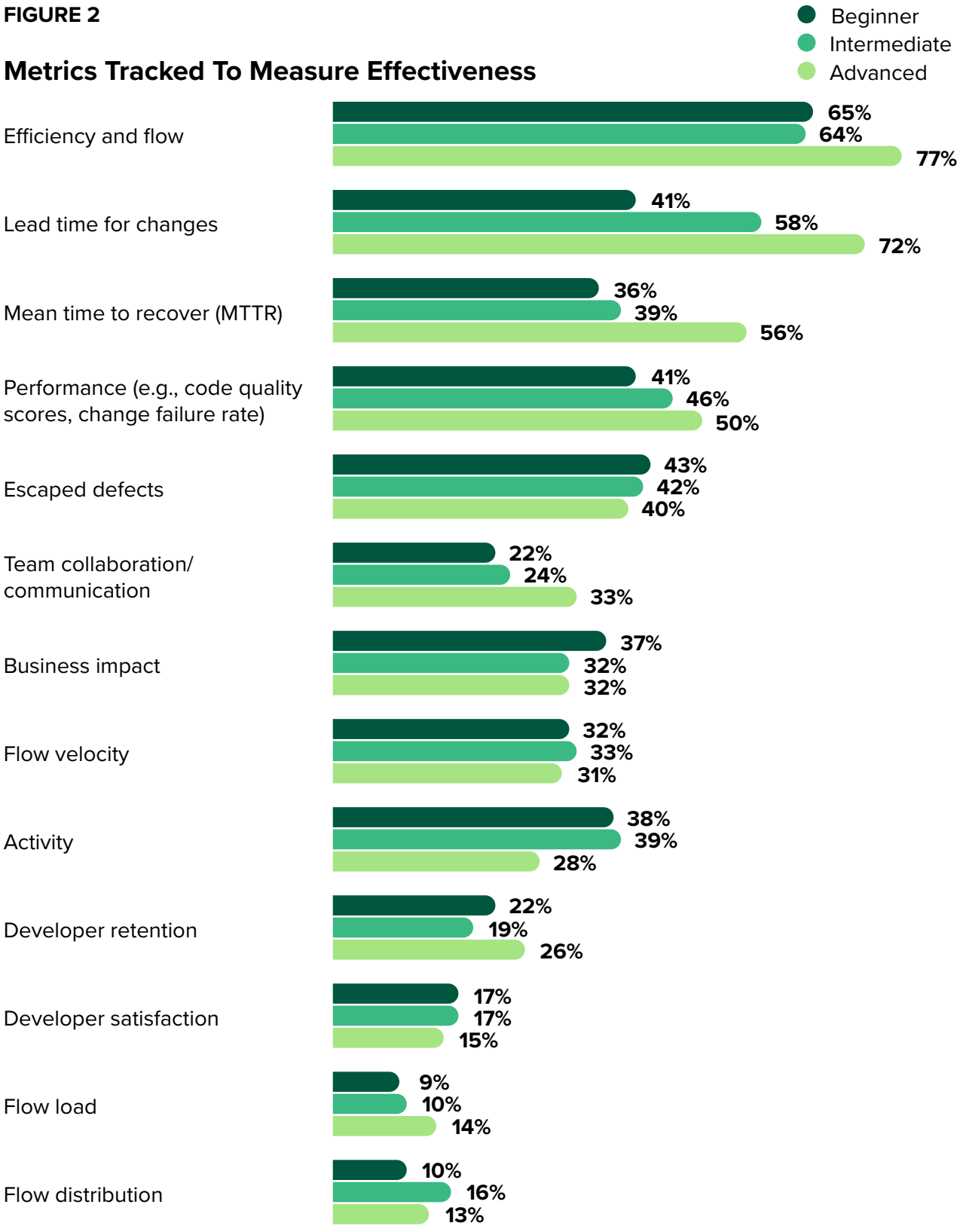


Base: 390 global enterprise developer decision-makers
Source: Forrester's Q1 2026 Contemporary Z/os DevX Survey [E-64876]

- **Metrics strategy.** More than 80% of respondents report using Flow, SPACE, DORA, or DX Core 4 metrics to measure development effectiveness. Among these, they most commonly track metrics for efficiency and flow, lead time to changes, and performance indicators (e.g., code quality, failure rate). Respondents less frequently emphasize developer satisfaction and retention metrics, despite long onboarding times and ongoing skills challenges. This pattern suggests that while respondents have widely adopted measurement frameworks, application varies. Higher-maturity organizations use metrics to guide investment and change, while their lower-maturity peers track metrics primarily for visibility rather than action (see Figure 2).

FIGURE 2

Metrics Tracked To Measure Effectiveness



Base: 390 global enterprise developer decision-makers
Source: Forrester's Q1 2026 Contemporary Z/os DevX Survey [E-64876]

- **Developer experience related to the tools that are provided and used for development.** Most organizations provide core development tooling, including build pipeline automation, code quality and security tools, test automation, and deployment pipelines. Self-service infrastructure, database schema tooling, and APM or observability capabilities are less common.

  Tool usage highlights sharper maturity differences. Mainframe developers most frequently use modern version control systems, IDEs such as Eclipse and VS Code, merge and pull requests for code review, and CLI/API-driven tooling like Zowe. In contrast, 33% or fewer report regular use of code coverage tools, systematic load or performance testing, testing harnesses with synthetic data, or AI-assisted development tools. Organizations with higher maturity are more consistently adopting these advanced capabilities, reducing manual effort and improving feedback loops (see Figure 3).

- **DevOps, testing, and AI usage.** Hybrid and cloud-based environments, containerized development, and automated dev environment provisioning remain below majority adoption.

  Testing practices also remain largely manual for respondents. Manual unit testing is most common, followed by

FIGURE 3

Most-Used Developer Tools And Practices By Maturity Leaders

(Showing responses from “Advanced”)

56%	Merge/pull requests to signal code reviews
55%	VS Code or variant
51%	Eclipse
50%	CLIs/APIs
36%	Code coverage tools
35%	Systematic load/ performance testing
33%	Dev sandbox aligned with production
28%	Harnesses
24%	Code explanation/ understanding tools
24%	AI tools
21%	Synthetic test data
21%	Container technologies
18%	Record/replay tools

Base: 390 global enterprise developer decision-makers
Source: Forrester's Q1 2026 Contemporary Z/os DevX Survey [E-64876]

automated unit testing in the same source language with mocking and stubbing. AI-assisted testing is still limited. Respondents note that their organizations primarily govern AI use overall via higher-level corporate and regulatory mandates, with less emphasis on team-level policies, monitoring, or metrics. While adoption is strongest for use cases such as code assistance, application understanding, and document generation, more advanced approaches — including agentic workflows and vibe coding — remain aspirational and are most closely associated with more mature development organizations.

Traditional Practices Struggle To Meet Modern Expectations

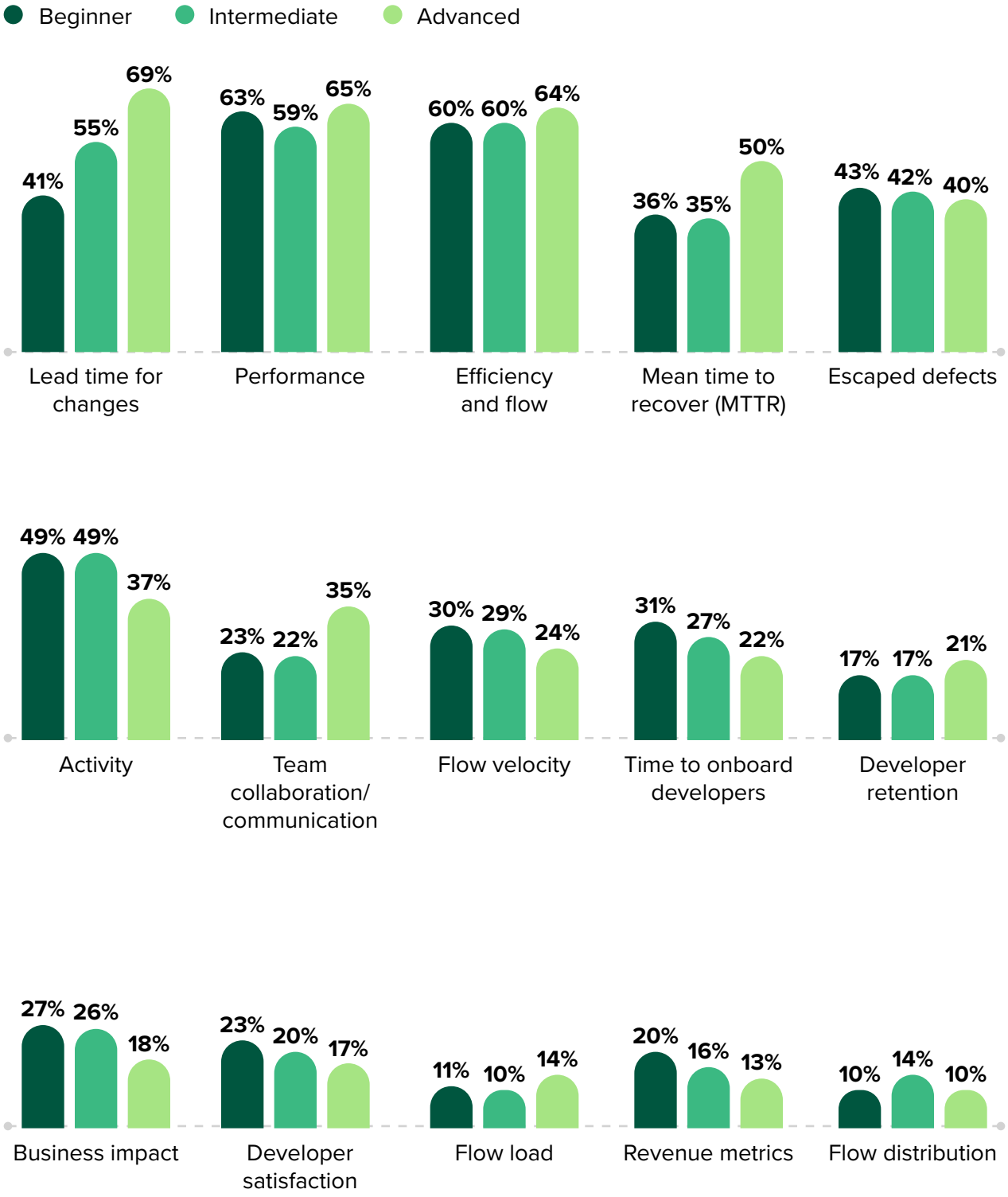
Respondents report that their developers struggle most with integration into modern pipelines, legacy code modernization, persistent expertise gaps, and limited tooling or automation. While many teams have deep domain experience, it is no longer sufficient to keep pace with increasing environment complexity, expanding toolchains, and higher expectations for speed and quality. Code testing and modernization efforts in particular lag behind platform and tooling changes, signaling a growing mismatch between skills and delivery demands.

As success metrics evolve alongside AI adoption and hybrid delivery models, development organizations face increasing pressure to identify which challenges most constrain their effectiveness. More mature organizations focus on using automation and tooling to offset skills gaps, while their less mature peers continue to rely on individual expertise and manual processes — amplifying risk as change volume increases.

- **Time and visibility remain persistent constraints on effectiveness.** Onboarding timelines highlight a structural challenge: Onboarding to platforms excluding the mainframe takes the shortest amount of time, while onboarding developers new to the mainframe — and delivering changes on the mainframe — takes the longest. While many organizations still struggle to meet key effectiveness metrics, AI adoption shows its strongest impact where priorities are highest, improving performance, efficiency and flow, and lead time for changes. Maturity leaders are better positioned to translate these gains into sustained improvement — in metrics like lead time for changes — while other, less mature respondents see benefits remain uneven (see Figure 4).

FIGURE 4

AI's Influence On Key Mainframe DevEx And DevOps Performance Indicators



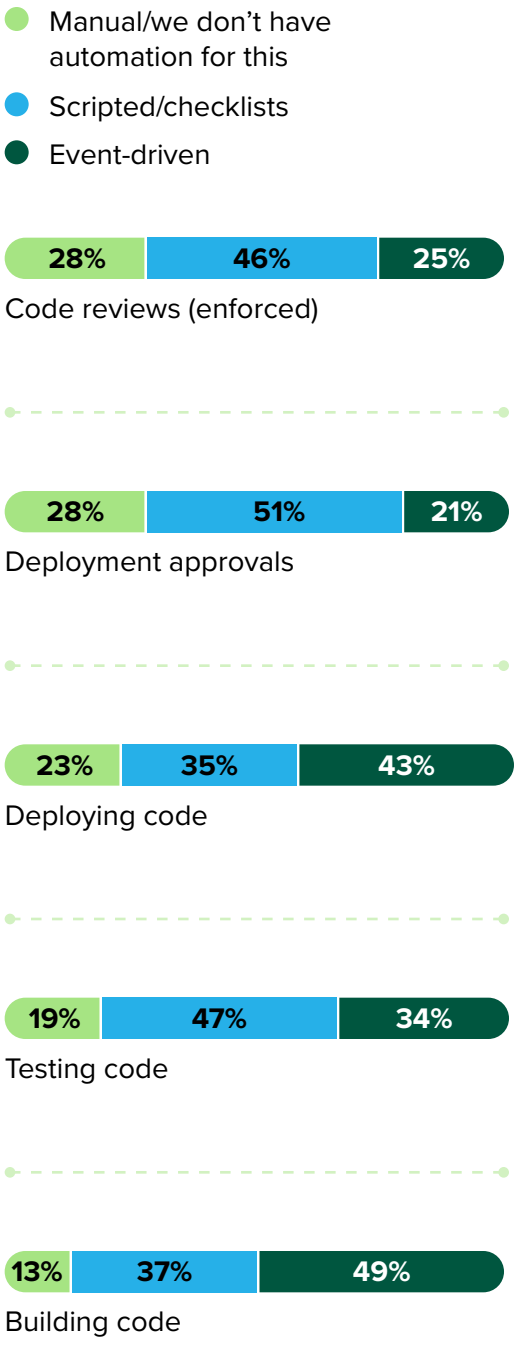
Base: 390 global enterprise developer decision-makers
Source: Forrester's Q1 2026 Contemporary Z/os DevX Survey [E-64876]

- **Developers expect parallel experiences, but they are not consistently delivered.** Sixty percent of respondents agree that the developer experience on Z is — or should be — similar to that of other platforms. This expectation reflects growing pressure to provide consistent tooling, workflows, and feedback loops across environments. Organizations that fail to deliver this parallel experience increase onboarding friction and strain developer satisfaction, while more mature organizations reduce platform-specific barriers and improve talent mobility.

- **DevOps and automation script scales, but manual gates persist.** On average, nearly one-third of daily mainframe development tasks can be completed using scripts or CLI/API-driven workflows, indicating progress toward automation. However, many core pipeline functions remain manual. Functional and nonfunctional testing and rollback are still largely manual, while build, deploy, and unit testing processes vary between standalone mainframe and integrated pipelines. These manual control points disproportionately slow lower-maturity organizations, while maturity leaders continue to reduce human dependency across the delivery lifecycle (see Figure 5).

FIGURE 5

Mainframe Development Automation Levels



Base: 390 global enterprise developer decision-makers
Source: Forrester's Q1 2026 Contemporary Z/os DevX Survey [E-64876]

Development Organizations Can Thrive Amid Moving Goal Posts

Respondents clearly identified the barriers limiting efficiency, effectiveness, and developer satisfaction, but it is their near-term investment decisions that will determine whether they mature or stall. As delivery expectations shift alongside AI adoption and hybrid architectures, organizations face growing pressure to decide which constraints to address first. Leading organizations distinguish themselves by aligning tools, processes, and skills investments to their most acute bottlenecks, while lower-maturity organizations spread effort broadly, slowing progress. With finite resources and increasing modernization complexity, targeted expertise and technical support are critical for moving from incremental improvement to sustained automation.

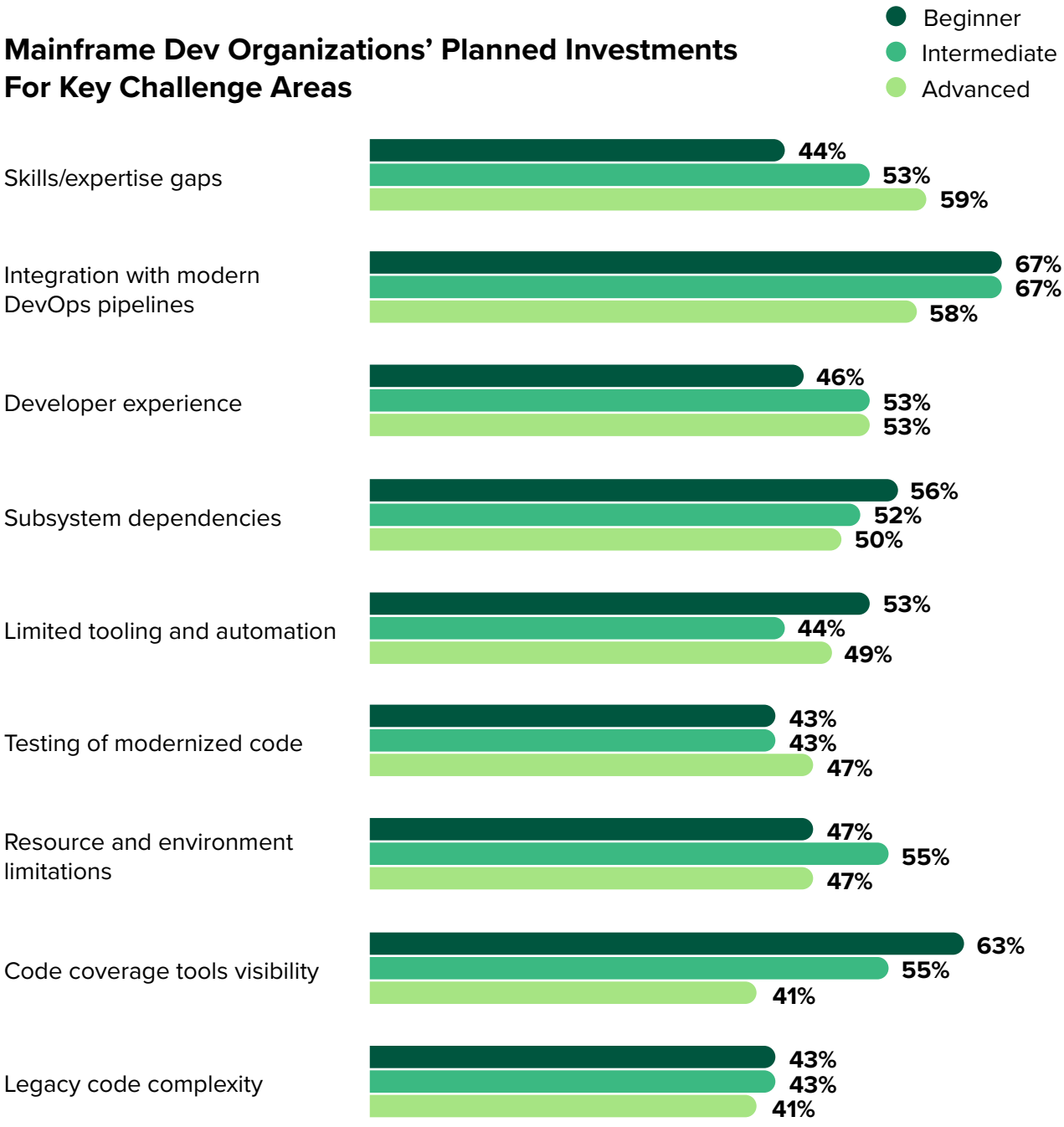
- **Near-term investments can reinforce or reset maturity trajectories.** Respondents' planned investments over the next two years closely mirror current challenge areas. More than half expect to invest in integrating pipelines, improving visibility into code coverage tools, closing skills gaps, managing subsystem dependencies, improving the developer experience, and addressing resource limitations. These areas form the foundation for reducing cycle time and improving reliability. Investments in deeper automation, modernized code testing, and legacy complexity simplification also remain important, but tend to be prioritized by higher-maturity organizations that have already stabilized foundational capabilities. Organizations earlier in their journey continue to focus on establishing baseline enablement before pursuing more advanced optimization (see Figure 6).

- **AI amplifies gains where automation and standards are in place.** While AI adoption has not followed a linear progression, respondents expect AI-enabled tools to meaningfully address key problem areas, including skills gaps, limited code understanding, low productivity, complex or manual testing, and legacy code conversion. These improvements directly influence core effectiveness metrics at the enterprise level.

Organizations with stronger automation and governance foundations are better positioned to translate AI adoption into measurable impact, while their less mature peers often experience fragmented or uneven benefits.

FIGURE 6

Mainframe Dev Organizations' Planned Investments For Key Challenge Areas

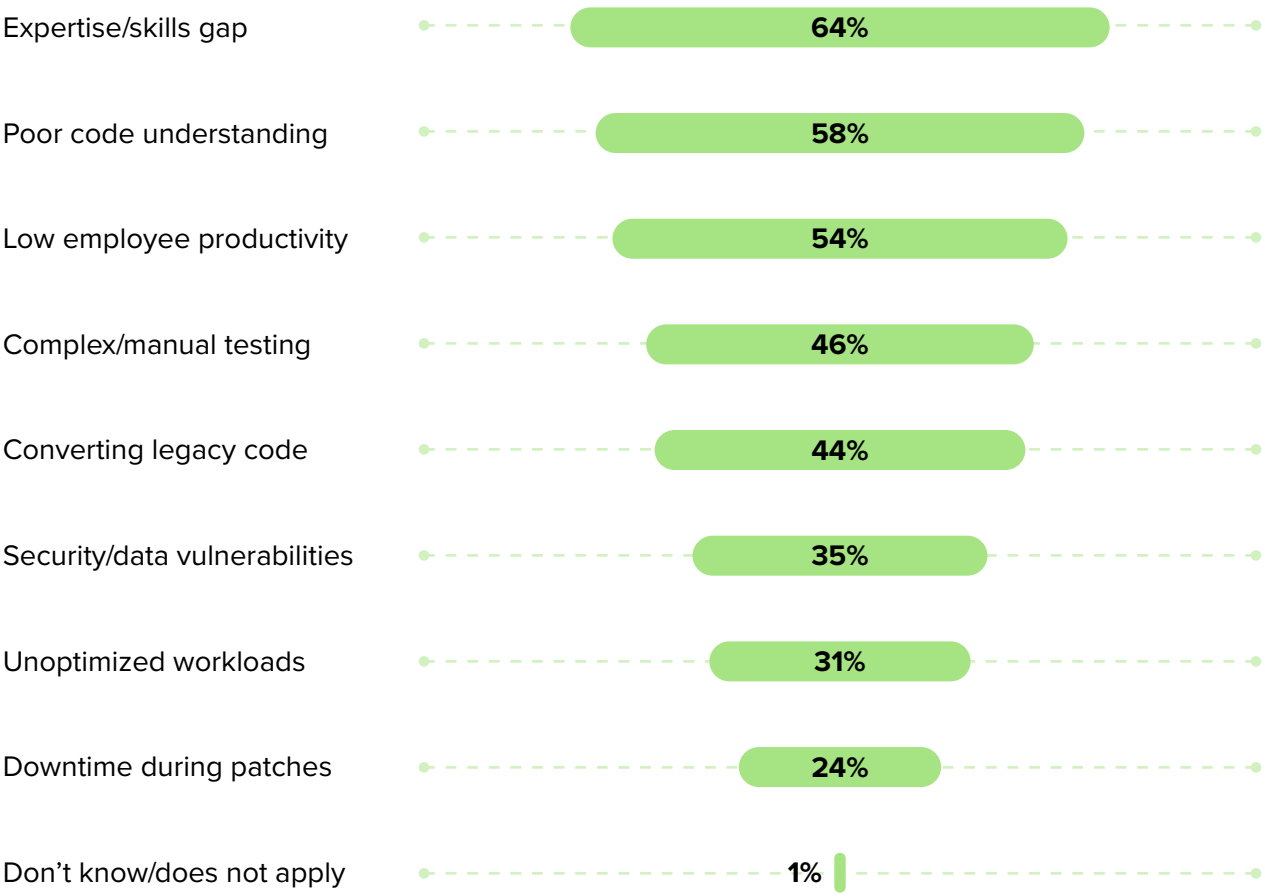


Base: 390 global enterprise developer decision-makers
Source: Forrester's Q1 2026 Contemporary Z/os DevX Survey [E-64876]

Together, these patterns suggest that solutions are less about adopting new tools in isolation and more about sequencing investments to reinforce maturity. Organizations that align automation, AI, and developer enablement to their most pressing constraints are better equipped to adapt to shifting delivery expectations and sustain improvement as the goal posts continue to move. As this alignment becomes essential to how work gets done, platform engineering emerges as the operating model that integrates these capabilities into a coherent, scalable system, positioning it for new investment and modernization (see Figure 7).

FIGURE 7

Problems Expected To Be Mitigated By AI Tools



Base: 390 global enterprise developer decision-makers
Source: Forrester's Q1 2026 Contemporary Z/os DevX Survey [E-64876]

Key Recommendations

Forrester's in-depth survey of 390 app dev decision-makers yielded several important recommendations:

Augment the core pipeline with modern tools. New developers entering the mainframe space expect to use toolchains like those used for cloud. Give them the tools they need to increase their productivity: contemporary IDEs like VS Code, flexible build and deployment pipelines, and hybrid source control.

Automate functional and nonfunctional testing. Many mainframe teams spend too much time testing functional and nonfunctional elements. This can mean it takes too long for important signals to reach the developers, resulting in last-minute panic fixes to components that developers thought they had already completed. Avoid this by building functional and nonfunctional tests into the build pipeline, giving developers immediate feedback when code is too slow or doesn't behave correctly.

Adopt AI with guardrails. Make use of AI, particularly to understand and help modernize existing code bases. However, avoid the temptation to rely on AI-generated code, especially in highly regulated environments. Make it clear that developers own the code they ship, regardless of any AI assistance they use to create it. Adopt strong, non-AI guardrails in the build, and deploy pipelines to discover and prevent hallucinated code from entering production.

Act on developer satisfaction metrics. Many teams measure some developer productivity metrics, but few include measures of developer satisfaction. Your developers know where their pain points are. By addressing them, you can improve retention, reduce the need for expensive onboarding, and simultaneously improve developer productivity.

Appendix A: Methodology

In this study, Forrester conducted an online survey of 390 global app dev decision-makers to evaluate the maturity of their app dev organizations. Questions provided to the participants asked for details relating to the capabilities, tools, and strategies they use to enable and understand developer success. Respondents were offered a small incentive as a thank-you for time spent on the survey. The study began in December 2025 and was completed in January 2026.

Appendix B: Demographics/Data

GEOGRAPHY	
United States	**28%**
United Kingdom	**16%**
Canada	**12%**
Germany	**12%**
France	**12%**
India	**11%**
Brazil	**10%**

NUMBER OF EMPLOYEES	
20,000 or more	**11%**
15,000 to 19,999	**22%**
10,000 to 14,999	**30%**
5,000 to 9,999	**37%**

DECISION INFLUENCE	
Hybrid development strategy	**100%**
Mainframe development strategy	**100%**
Development strategy, overall	**100%**

DEMOGRAPHICS HEADLINE	
Technology (SI/services)	**17%**
Financial services and/or banking	**16%**
Retail	**9%**
Consumer product goods and/or manufacturing	**8%**
Healthcare	**8%**
Manufacturing and materials	**8%**
Energy, utilities, waste management	**7%**
Insurance	**5%**
Transportation and logistics	**5%**
Automotive manufacturing/ aftermarket	**4%**
Education and/or nonprofit	**3%**
Government	**3%**
Telecommunication services	**3%**
Travel and hospitality	**2%**

DEPARTMENT	
IT	**100%**

Note: Percentages may not total 100 due to rounding.

Appendix C: Supplemental Material

RELATED FORRESTER RESEARCH

Knowledge Management And The Developer Experience, Forrester Research, Inc., June 20, 2025.

The State Of Mainframe, Global, 2025, Forrester Research, Inc., May 23, 2025.

Tackle The Overwhelming Challenge Of Mainframe Modernization, Forrester Research, Inc., February 29, 2024.


FORRESTER®