FORRESTER®

Mainframe Application Developer Study

A FORRESTER CONSULTING THOUGHT LEADERSHIP PAPER COMMISSIONED BY BROADCOM, AUGUST 2024

Table Of Contents

Project Team:

Madeline Harrell,
Market Impact Consultant

Jemimah Charles,
Associate Market Impact Consultant

Contributing Research:

Forrester's Technology Architecture & Delivery research group

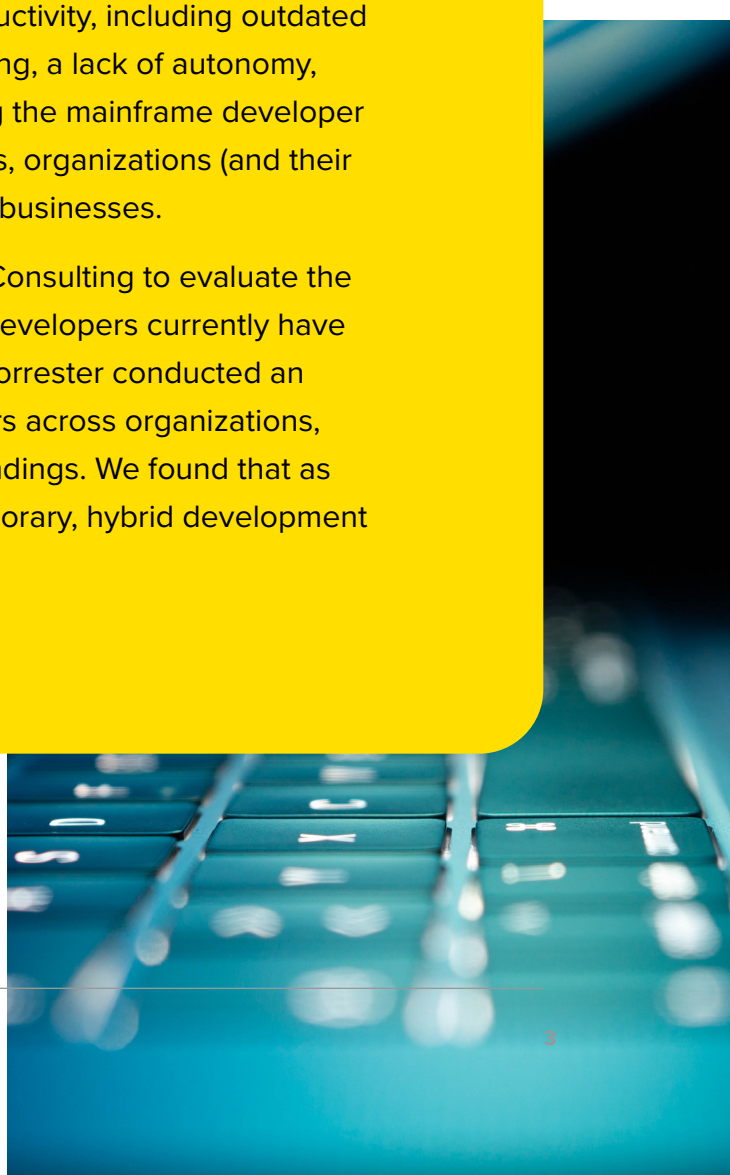
Executive Summary

To ensure the long-term health and vitality of mainframe applications, it's critical to understand those responsible for them. While the voice of the global developer community is widely reported, the voice of mainframe developers is often drowned out. This study highlights their perspectives, challenges, and the opportunities they perceive.

As organizations continue to integrate the mainframe into multiplatform architectures, including hybrid cloud, the role of the mainframe application developer is evolving in tandem. Mainframe developers want practices and tools that enable them to perform at the highest levels for their organizations. But organizations are at different maturity levels in offering new tools and practices to their mainframe developers. Because of this, their strategic roadmaps should match their maturity level.

Though most mainframe developers are happy and productive, they face considerable barriers to increased speed and productivity, including outdated tools and frameworks, interruptions/context switching, a lack of autonomy, and a lack of agile/DevOps practices. By prioritizing the mainframe developer experience (DevX) and automating manual activities, organizations (and their developers) can deliver even greater value to their businesses.

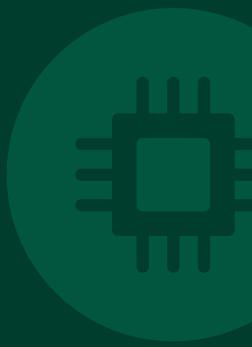
In June 2024, Broadcom commissioned Forrester Consulting to evaluate the capabilities and processes mainframe application developers currently have or desire to improve their organizations' maturity. Forrester conducted an online survey with 838 global mainframe developers across organizations, generations, and geographies, and analyzed the findings. We found that as developer needs grow and evolve toward contemporary, hybrid development practices, so do their tools and frameworks.



Key Findings

Traditional productivity barriers now have straightforward solutions. The biggest barriers, which include outdated tools and frameworks, a lack of autonomy, and a lack of agile/DevOps practices, can be traced to a conservative approach to change. While cloud developers were "failing fast" with new, shift-left DevOps techniques, mainframe teams were protecting mission-critical applications with time-tested methods and tools. However, many practices that were once cutting-edge, such as DevOps, are now well-vetted and more easily adopted by mainframe teams. A well-designed developer experience is paramount to increasing time spent coding.

There is real hunger for hybrid development tools and practices. As career mainframe developers retire, replacements typically have experience with contemporary platforms and tools, creating a groundswell of demand for common enterprise standards (e.g., popular IDEs [integrated development environments], DevOps practices and toolchains). Fifty-six percent of surveyed developers have professional experience with other platforms, reflecting an important tipping point.



To accelerate development, look first to automation and, in the long term, AI. Developers in this study demonstrated a high level of satisfaction in their roles but recognize the need to improve their time to market. When asked about the biggest opportunities to speed mainframe development, automation stood out as the most common theme. There is also an acute awareness of AI potential.

Traditional Productivity Barriers Now Have Straightforward Solutions

Mainframe applications are typically mission-critical transaction processing applications that entire industries — from financial services to healthcare to government — rely on. These applications enable core lines of business by ensuring transactional accuracy and regulatory compliance. The stakes are high and, as a result, changes to time-tested development practices and tools are subject to a high level of distrust. However, the art and science of software engineering have advanced significantly over the past decade with the introduction of DevOps, major innovations with IDEs, version control tools, and more.

An examination of the top reported barriers to productivity yields solutions that are now well understood at the enterprise level. The top reported barriers are:

- Outdated tools and frameworks (35%).
- Interruptions/context switching (34%).
- A lack of agile/DevOps practices (32%).
- A lack of autonomy (dependencies) (32%).

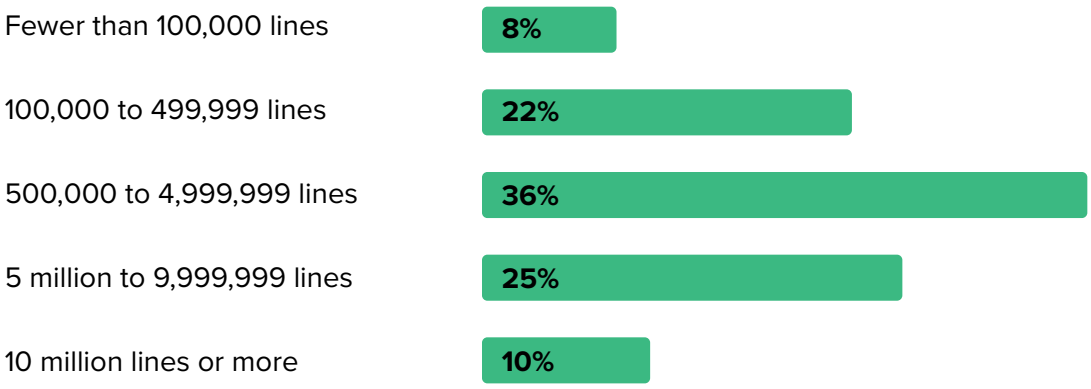
Modern practices like DevX workflow design and DevOps automation with continuous integration and continuous delivery (CI/CD) pipelines and test automation are now widely accepted and deployed within the enterprise. While these practices are starting to expand into platform engineering at the enterprise level, mainframe teams should evaluate the well-established ones that address the top barriers. Regarding development environments, for example, 31% of surveyed respondents use the out-of-favor Eclipse IDE while the green screens of ISPF remain prevalent. Widely used IDEs that facilitate code navigation like Visual Studio Code are readily available for mainframe use with extensions for mainframe languages, subsystems, etc.

This need is particularly acute with mainframe development as applications are often large and complex. Twenty-five percent of respondents reported their applications exceed 5 million lines of code, and 10% reported 10 million

or more lines, so having an editor that facilitates code navigation offers significant impact (see Figure 1). Applications this large present unique coding, testing, and deployment challenges that modern tools can help address.

FIGURE 1

Average Size Of Applications



Base: 838 global mainframe application developers at enterprise companies
Source: A commissioned study conducted by Forrester Consulting on behalf of Broadcom, June 2024

The traditional organizational alignment among many mainframe teams reflects a desire for specialized roles: developers write the code, and testers test the code while systems programmers control resources and database administrators control tables.

Contemporary development, however, reflects the shift-left mindset where developers are empowered with self-service to access the resources and tools they need to code, test, and deploy efficiently. By adopting this empowerment mindset, mainframe teams can reduce the disruptions and context switching that dependencies create.

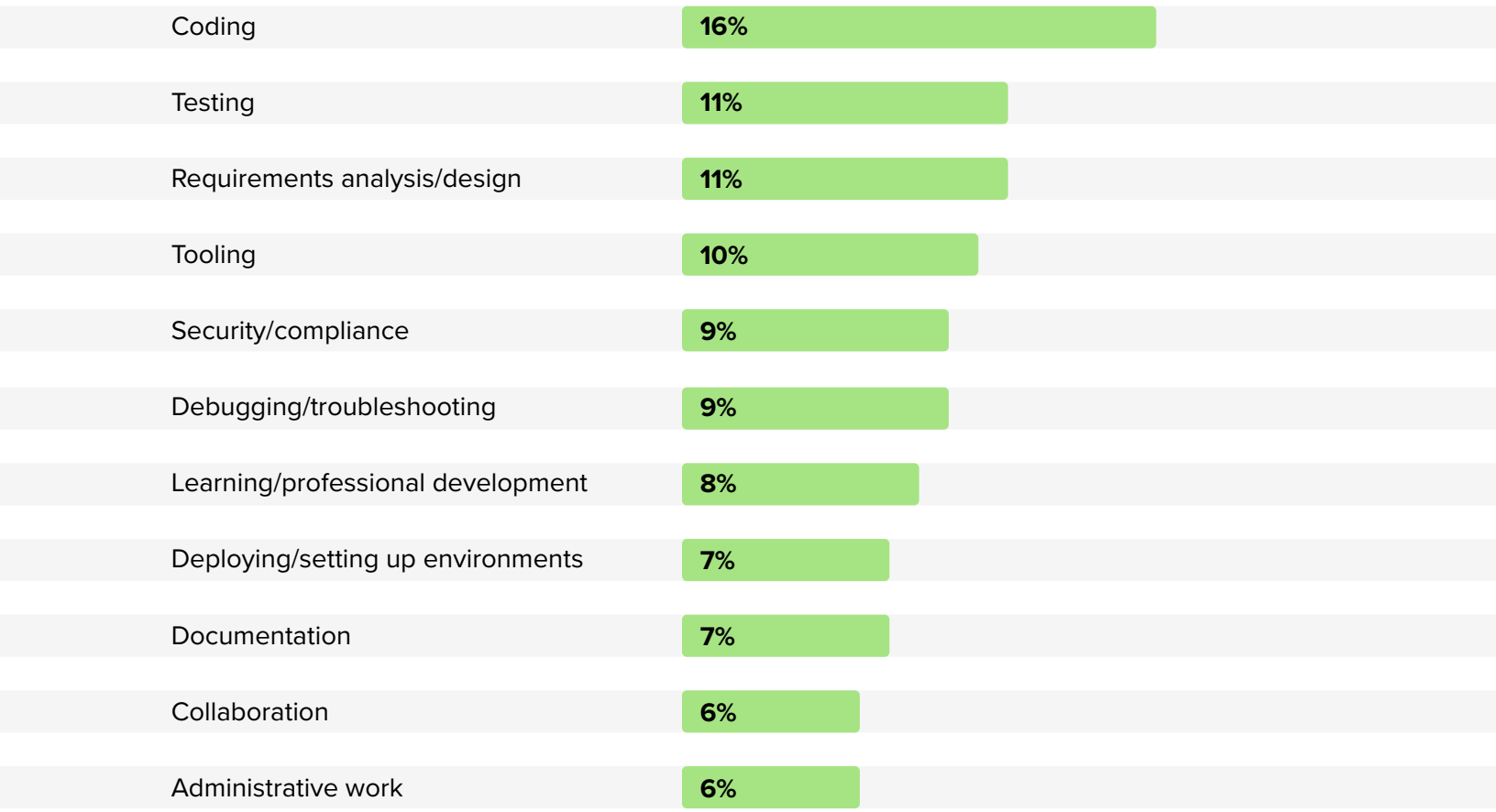
Coding effort was a mixed bag for the surveyed developers. Of all the activities involved in a typical day, they spent more time coding than on any other. However, it was only 16% of their time, which means they're spending 84% on noncoding activities. Empowering these developers with shift-left

autonomy will free them to spend more time coding and in the “flow state” — those windows of maximum concentration and engineering productivity — and less time waiting for others and toggling between coding and noncoding tasks (see Figure 2).

Following the recommendations in this report will help increase their time spent coding.

FIGURE 2

Average Mainframe App Developer Time Spent On Daily Tasks



Base: 838 global mainframe application developers at enterprise companies
Source: A commissioned study conducted by Forrester Consulting on behalf of Broadcom, June 2024

There's A Real Hunger For Hybrid Development Tools And Practices

Fifty-six percent of surveyed developers have professional experience on other platforms and, when education and nonwork coding are factored in, the number is likely much higher. These developers already know and love hybrid development tools and practices so making them available is an easy win.

A developer's primary toolset is their editor/IDE and version control tool. Visual Studio Code is consistently rated the most popular and widely used IDE, and 70% of mainframe developers say the adoption of VS Code will result in a major productivity increase. Similarly, with version control, 74% say Git adoption will result in a major productivity increase. Git adoption has the added benefit of opening up the world of DevOps toolchains, which are typically integrated with enterprise Git servers.

Legacy development tools and practices create several challenges for mainframe leaders, not least of which is staffing. A limited talent pool means new recruits often require extensive training, which impacts onboarding timelines. Opening mainframe development to hybrid development tools and practices offers a wide range of benefits, including:

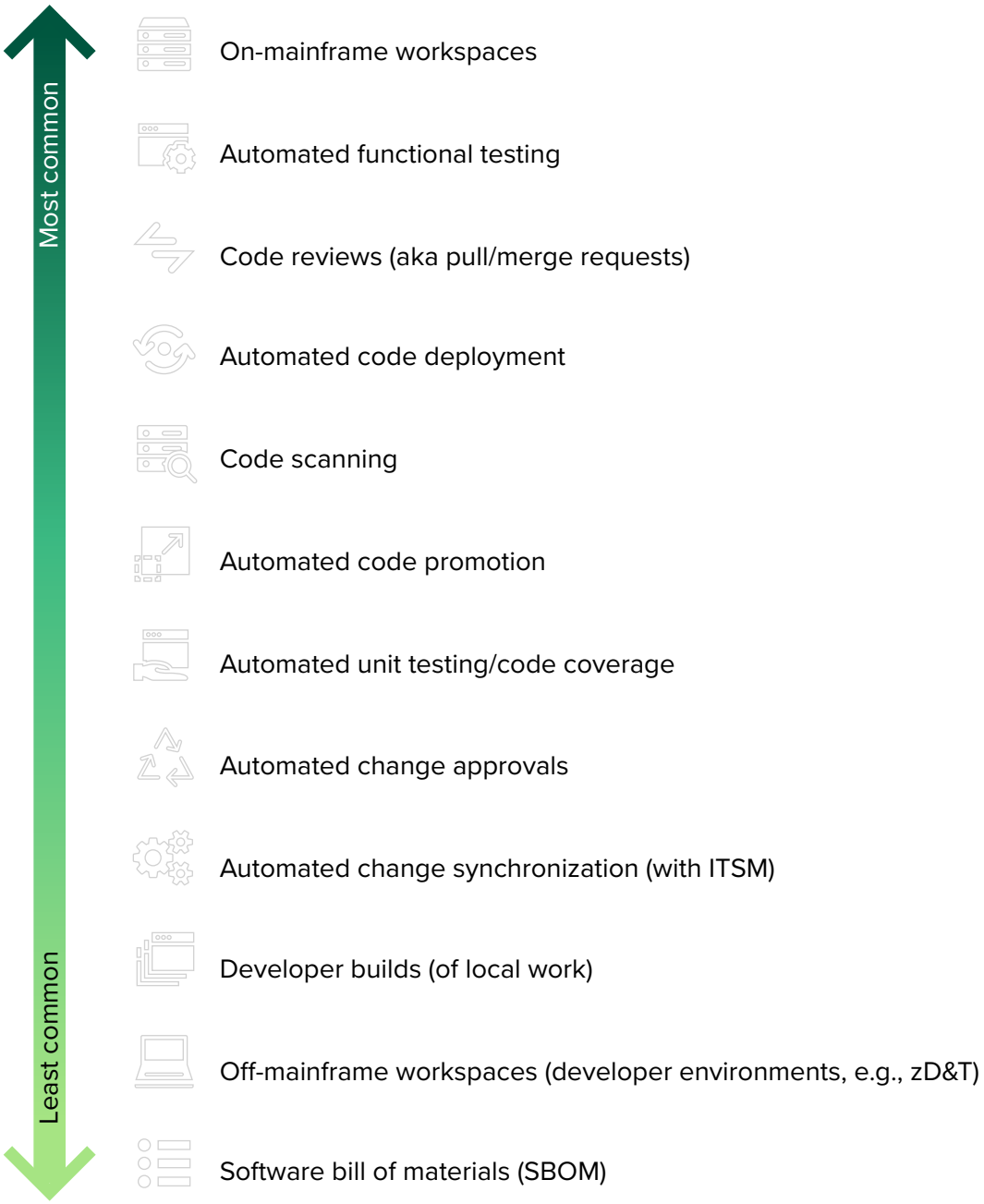
- A larger talent pool, easier internal transfers, and reduced onboarding time.
- Improved communication and collaboration at all levels (e.g., intra-team, cross-team, cross-functional, between mainframe and nonmainframe developers).
- A virtuous loop of more productive developers becoming happier developers, increased retention, easier recruiting, etc.

In summary, using the same tools and practices as other development teams helps create a common vocabulary and fabric across the enterprise (see Figure 3).

FIGURE 3

Regularly Used AppDev Practices

(Showing "Yes, I use this regularly.")



Base: 838 global mainframe application developers at enterprise companies
Source: A commissioned study conducted by Forrester Consulting on behalf of Broadcom, June 2024

Look First To Automation Then To AI To Accelerate Development

Mainframe developers are generally happy in their roles as illustrated by a 93% satisfaction rate (with 68% being **extremely** satisfied). Eighty-nine percent view their teams as productive. Yet the need to accelerate the delivery cycle is clear. While CI/CD has become commonplace in the cloud world, only 16% of mainframe developers reported typical code changes more frequently than monthly. Twenty-six percent reported typical code changes of six months or more. So, we asked developers "What would be the single biggest improvement to accelerate mainframe development?"

Automation emerged as the top theme — mentioned over 130 times — which may seem counterintuitive as almost half reported their build, test, and deploy processes were mostly automated already. In the speed context, however, automation was cited as reducing manual workloads and eliminating human error. Repetitive tasks were called out specifically.

Automation can now be safely orchestrated off-platform with tools like the open-source Zowe Command Line Interface (CLI). Innovations like Zowe open the door for developers to take advantage of contemporary automation tools and practices and access mainframe resources using a familiar CLI to integrate tools with the mainframe. Mainframe developers are also increasingly likely to have experience with the most common automation languages: JavaScript and Python.

In terms of functional areas, testing was most frequently mentioned in terms of speeding development.

In a separate question about a range of modernization options, test automation was rated **extremely** important. Like most areas of technology, AI is having an impact on software development and mainframe developers are already experimenting.

Generative AI initiatives designed to help developers with COBOL, PL/I, and Assembler applications understand that their code (aka code explain) and edit/write code (aka code assist) are being developed and tested via collaboration between leading software vendors and their innovation partners,

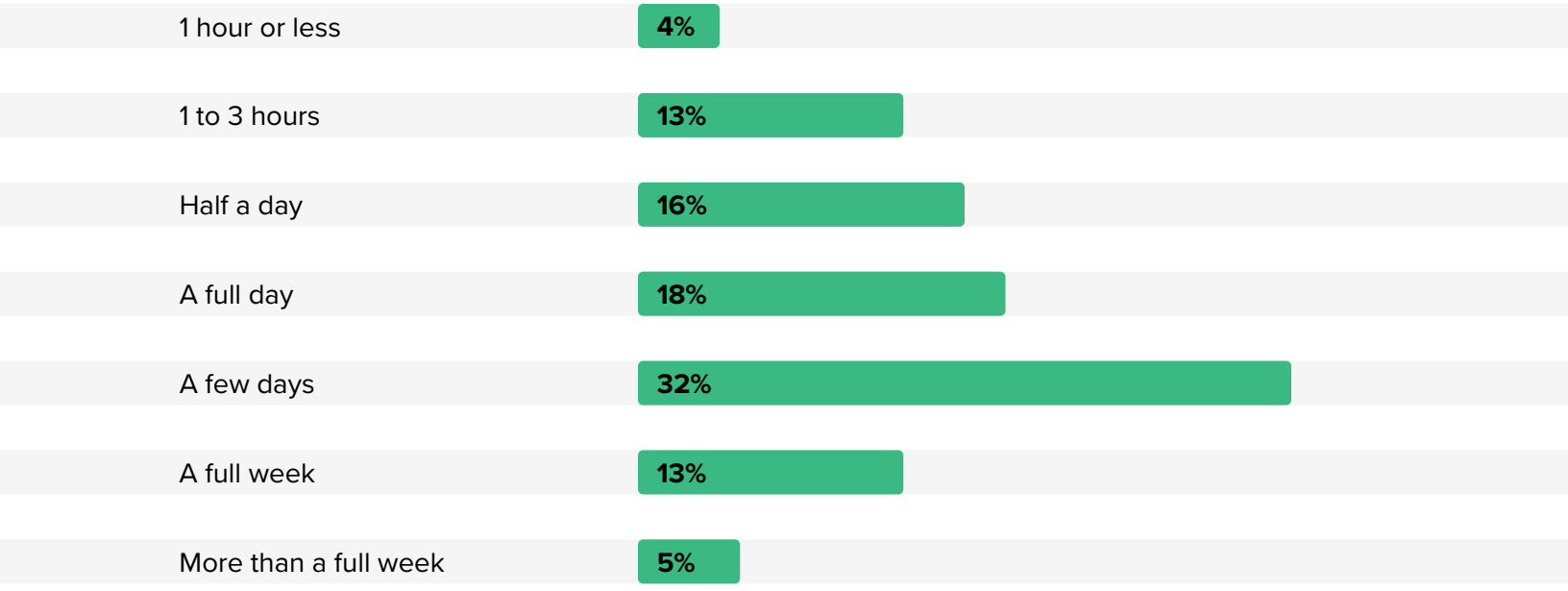
open-source projects, and other entrepreneurs/innovators. With such expansive and complex applications, helping mainframe developers understand where to make changes with assistive intelligence could provide the best short-term ROI.

This need is particularly acute as retiring staff are taking their application expertise (e.g., code structure, logic, and flow, data structures, and industry knowledge) with them. Combine this with this survey finding that half of respondents need three days or more to understand where to make code changes, and the opportunity is clear (see Figure 4).

As issues like security and code confidentiality are addressed, the potential impact of AI-driven chatbots for code explanation and code assistance in mainframe development could be enormous.

FIGURE 4

Average Time It Takes To Understand Where To Make Code Changes



Base: 838 global mainframe application developers at enterprise companies
Note: Total does not equal 100 due to rounding.
Source: A commissioned study conducted by Forrester Consulting on behalf of Broadcom, June 2024

Key Recommendations

Forrester's in-depth survey of mainframe application developers yielded several important recommendations:

Prioritize the mainframe developer experience (DevX).

To maximize developers' time in the flow state, mainframe leaders should put the end-to-end developer experience front and center. Take a holistic view with special consideration for those aspects of mainframe development that are unique to address critical productivity barriers like context switching and dependencies.

Adopt hybrid development tools and practices (i.e., existing enterprise standards).

Enable developers to use the same tools their contemporaries use like VS Code, Git, and DevOps workflows and toolchains. However, tools like IDEs can be part of a developer's identity so don't force hybrid standards on them. Once they observe the level of self-service autonomy and collaboration, even entrenched users will embrace these tools and practices, too.

Automate while staying connected to key AI initiatives.

Encourage teams to adopt modern automation frameworks and automate repetitive manual activities highlighted in your DevX research. Expand on existing Rexx-based automations and prioritize opportunities to automate manual testing activities. Generative AI (genAI) initiatives like COBOL code explanation and assistance are fast-moving, so stay involved as they will likely have a profound impact on mainframe development over time.

Appendix A: Methodology

In this study, Forrester conducted an online survey of 838 global z/OS mainframe application developers to evaluate the practices and capabilities utilized by mainframe application developer organizations. Survey participants included global mainframe developers across developer organizations, generations, and geographies. This study analyzed the findings as related to their tools, frameworks, and roadmaps. Respondents were offered a small incentive as a thank you for time spent on the survey. The study began May 2024 and was completed in June 2024.

Appendix B: Demographics

YEARS OF MAINFRAME EXPERIENCE	
0 to 2	**9%**
3 to 5	**20%**
6 to 10	**26%**
11 to 16	**23%**
16 to 25	**14%**
26+	**8%**

COUNTRY	
India	**31%**
United States	**25%**
United Kingdom	**12%**
Germany	**10%**
France	**10%**
Brazil	**8%**
Canada	**5%**

PRIMARY ROLE	
z/OS mainframe application developer	**100%**

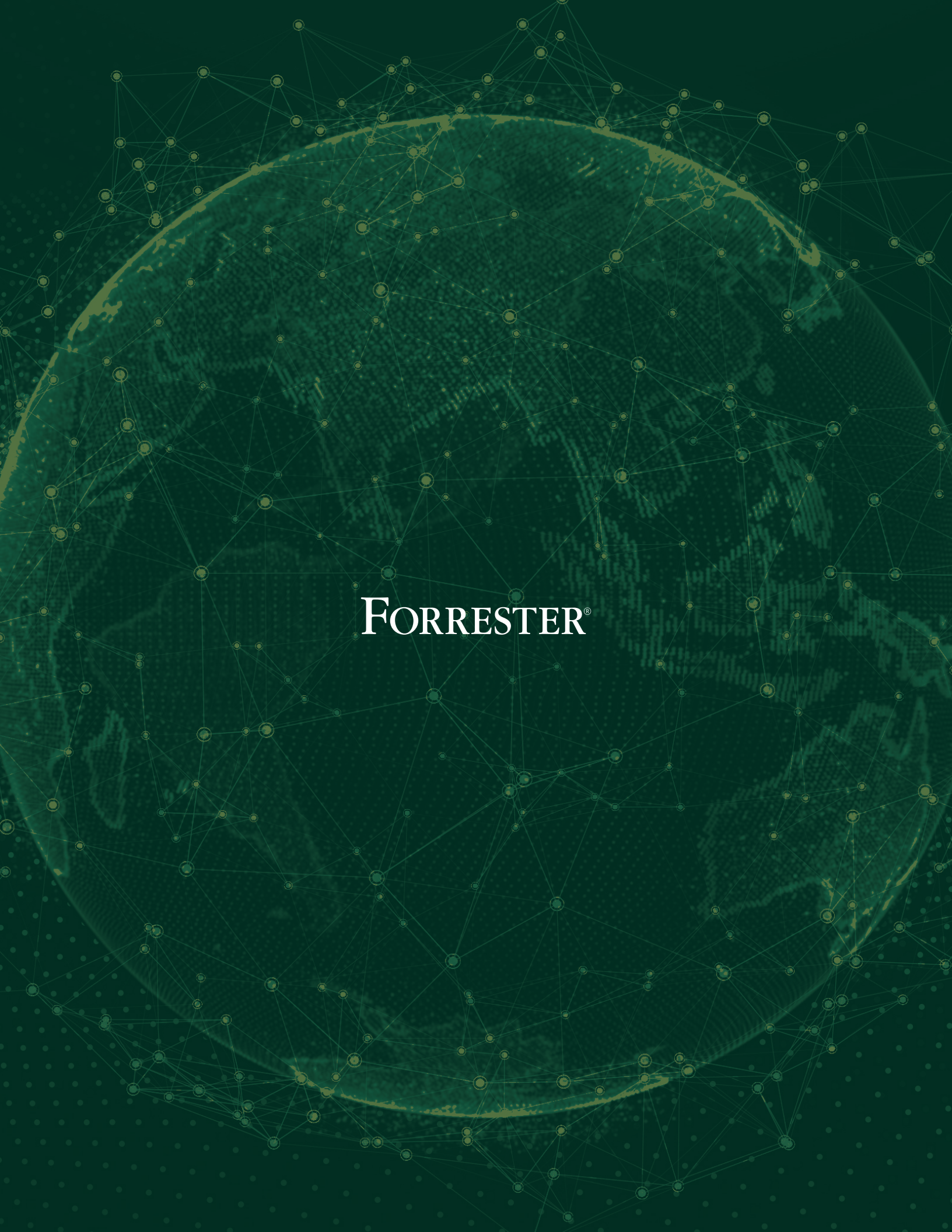
PROFESSIONAL PLATFORM CODING EXPERIENCE	
Cloud	**45%**
Mobile	**27%**
Distributed	**28%**
Mainframe	**100%**

COMPANY SIZE	
2,000 to 4,999 employees	**12%**
5,000 to 19,999 employees	**23%**
20,000 to 29,999 employees	**26%**
30,000 to 39,999 employees	**20%**
40,000 to 49,999 employees	**13%**
50,000 or more employees	**7%**

INDUSTRY (TOP 7)	
Financial services and/or banking	**12%**
Retail	**10%**
Telecommunications services	**10%**
Technology — systems integration/ services	**9%**
Insurance	**8%**
Healthcare	**7%**
Manufacturing and materials	**7%**

PRIMARY CODING LANGUAGE	
Java	**28%**
Python	**23%**
COBOL	**22%**
C++	**10%**
Assembler	**6%**
PL/I	**5%**
C	**4%**
Rexx	**1%**

Note: Percentages may not total 100 due to rounding.


FORRESTER®