split

2023

FEATURE MANAGEMENT IMPACT REPORT

REPORT CONTENTS

ABOUT THE REPORT

This study was conducted by **Virtual Intelligence Briefing** on behalf of Split Software. While Split guided the research brief, Virtual Intelligence Briefing sourced and processed all survey data, and produced the insights contained within. Split did not attempt to influence any of Virtual Intelligence Briefing's conclusions.

INTRODUCTION

Developing new, successful software features is essential for growth and competitive differentiation. As a result, software product development teams release new features and measure their impact on users and the business overall. They may also experiment with new features to gauge how users are reacting to them.

Despite strong interest in feature management and experimentation, not all organizations have embraced these practices. Whether the issue is inertia or immaturity in feature management and experimentation, it is time to get serious, even if adoption lags behind enthusiasm, so far. This report explores the issues in depth. It is based on a survey of software developers, development managers, QA managers, product managers and other stakeholders in software feature development.

HIGHLIGHTS FROM THE SURVEY

92% of respondents either agree or completely agree that software feature management is critical to developing and releasing successful digital experiences (**Fig. 2**)

87% either agree or completely agree that software feature experimentation is critical to developing and releasing successful digital experiences (**Fig. 3**)

84% consider themselves "somewhat successful," "neither successful nor unsuccessful," or "somewhat unsuccessful" at feature delivery and management (**Fig. 4**)

60% felt that their top area for improvement involved software feature release quality and reliability (**Fig. 5**)

65% are measuring the performance of software features (**Fig. 7**)

61% are measuring the business impact of software features (**Fig. 7**)

51% feel accelerating software delivery and software feature delivery are important priorities (**Fig. 10**)

50% feel easily pinpointing unexpected issues during a feature rollout is an important priority (**Fig. 10**)

50% measure performance for fewer than 25% of their feature releases (**Fig. 12**)

40% said consequences of an issue with a feature release resulted in late nights and long hours rolling back issues; 10% said someone on the team lost their job (**Fig. 19**)

32% said that 50% or more of their software engineering efforts results in technical debt (**Fig. 15**)

27% experiment with features, looking at feature-level data attribution and how it is associated with business metrics like revenue (**Fig. 8**)

20% have implemented a feature management platform; 32% are investigating; 33% plan to implement (**Fig. 22**)

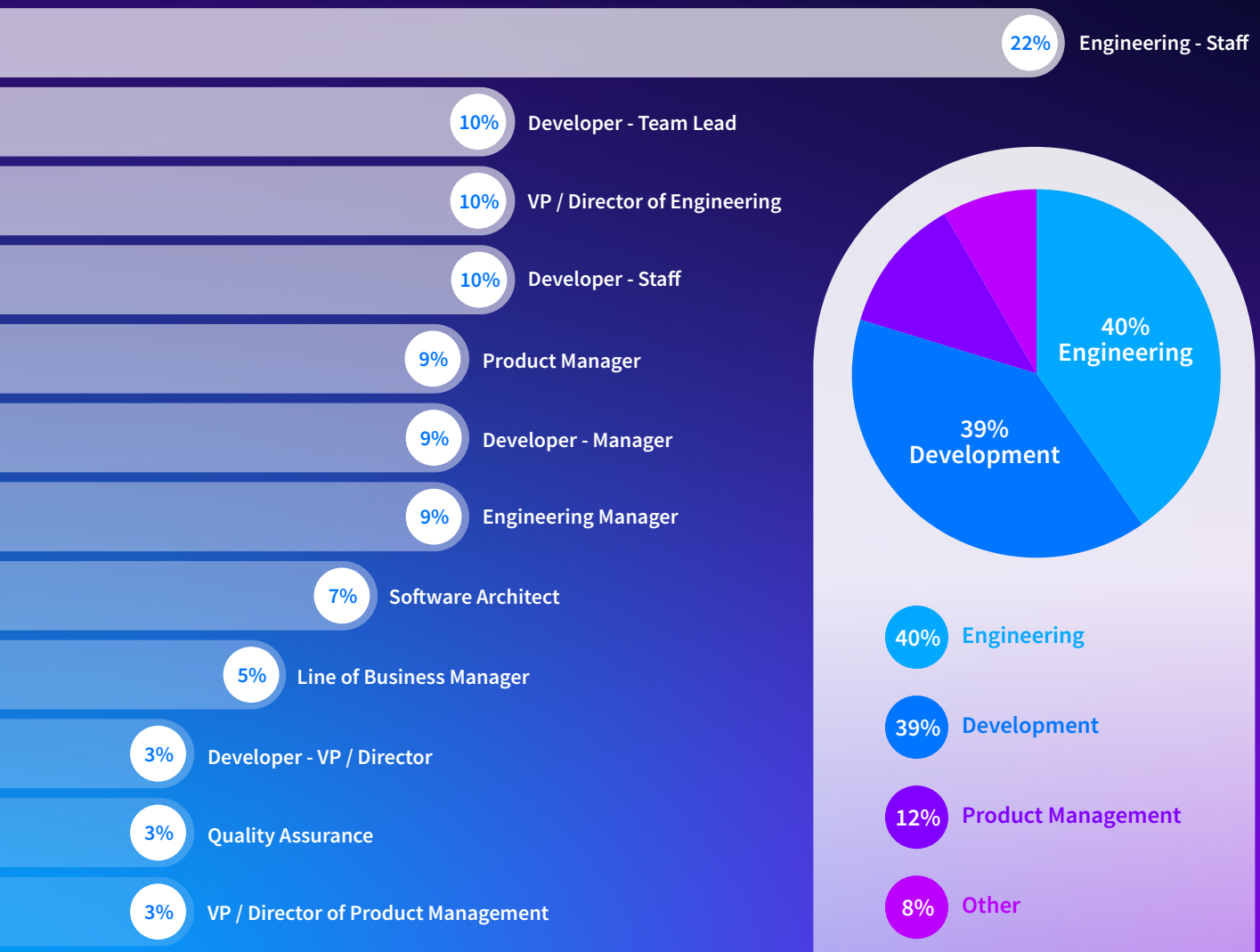
14% have implemented a feature experimentation platform; 39% are investigating; 29% plan to implement (**Fig. 24**)

SURVEY DESCRIPTION AND DEMOGRAPHIC SUMMARY

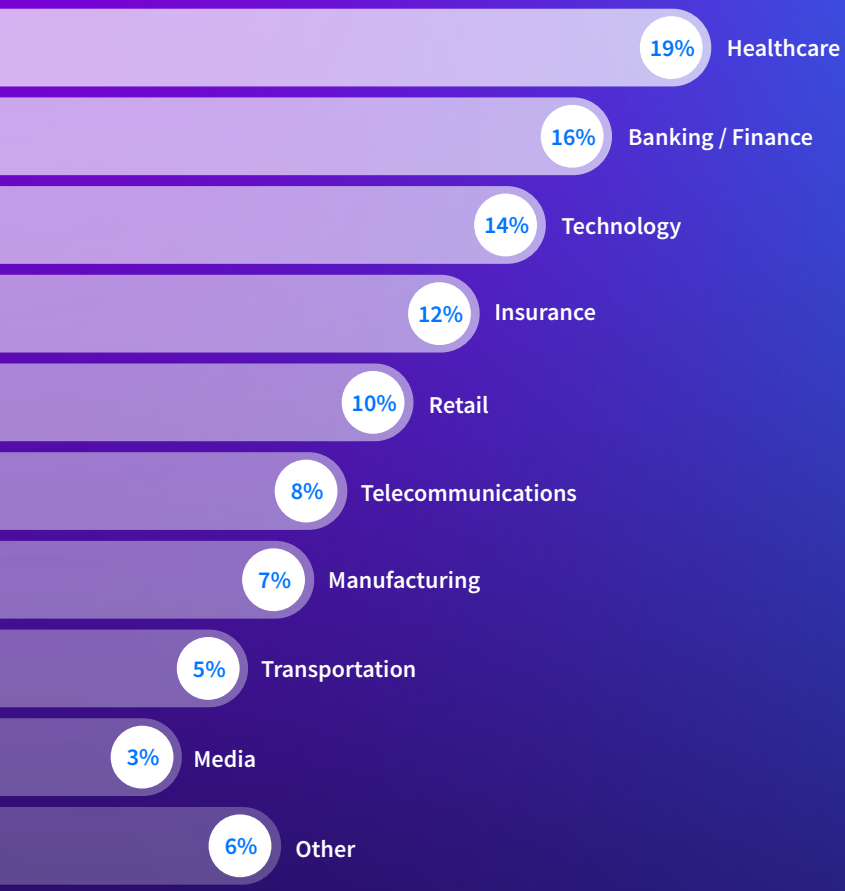
This report is based on a survey of **353 software professionals**, conducted in late 2022. Respondents come from a variety of backgrounds, with 40% serving in engineering roles, 39% in development and 12% in product management. They represent companies in a range of industries, including healthcare (19%), banking and finance (16%) and technology (14%). Thirty-six percent of respondents work at companies with more than 10,000 employees; 26% at companies with between 2,500 and 9,999 employees; 22% with 1,000 to 2,499; and 16% with 500-999.

This diverse group provides a cross section of organizations that are trying to make their way with software feature management and experimentation. The survey asked respondents for their views on software feature management and experimentation, as well as, whether they think their organizations are succeeding with these processes. It also probed their plans for the future.

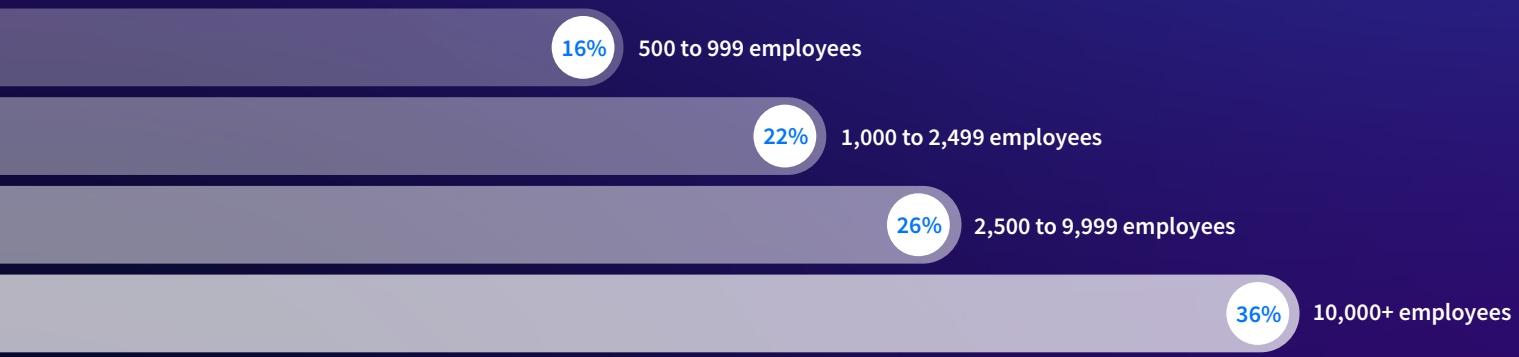
353 respondents broken down by **role / function:**



Respondents broken down by **industry:**

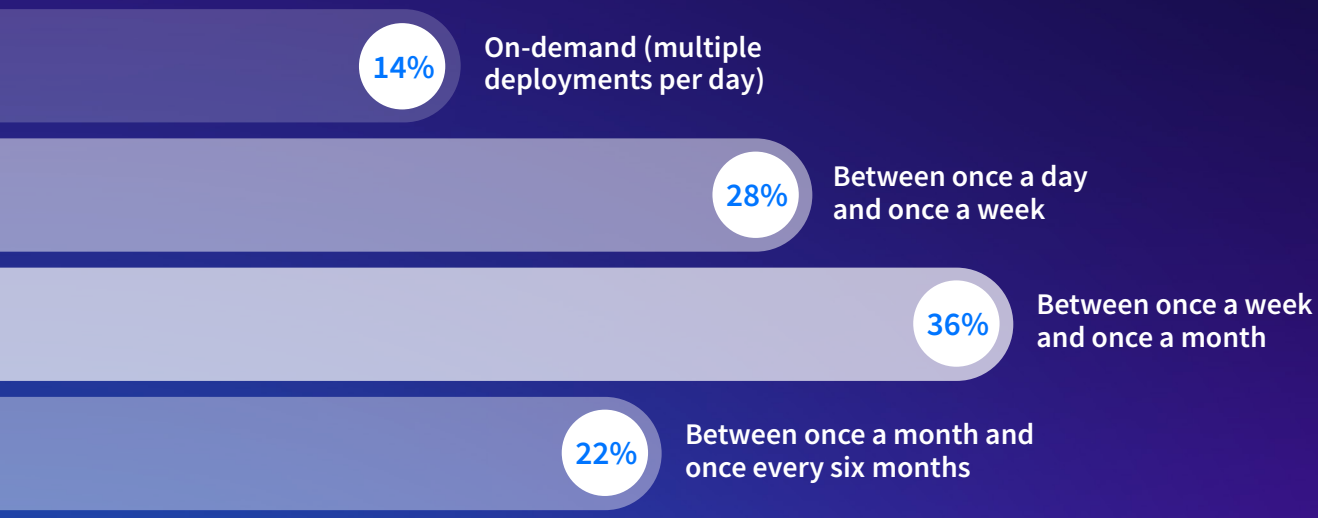


Respondents broken down by **organization size:**

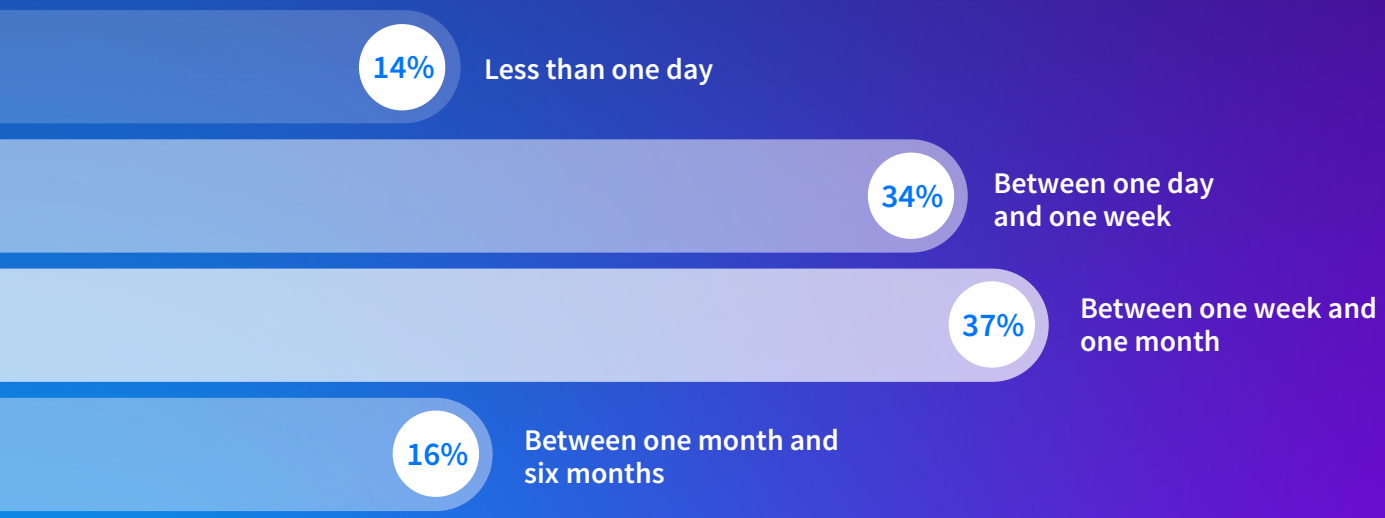


The respondents' software feature deployment frequencies run the gamut from on-demand, meaning multiple deployments per day, to scheduled, meaning between once per month and once every six months. The greatest proportion deploys code to production between once per week and once per month. Very fast and very slow code deployment schedules are less common.

1 **Deployment Frequency:** How often does your organization deploy code to production or release it to end-users?



Lead Time for Changes: How long does it take to go from code committed to code successfully running in production?



The lead time for changes in software varied, as well. Just 14% needed less than a day to go from code committed to code successfully running in production. This group most likely represents organizations with well-implemented DevOps and CI/CD workflows. Thirty-four percent required between one day and one week, while 37% needed between a week and a month. Sixteen percent required between one month and six months.

HIGH-LEVEL TAKEAWAYS

The survey results suggest several high-level takeaways:

- Interestingly, there's a disconnect between perceptions of the importance of feature management and experimentation and self-perceptions of success in software feature delivery and management. Ninety-two percent and 87% of respondents considered feature measurement and experimentation critical for successful digital experiences. However, 84% of respondents do not consider themselves entirely successful at these practices.
- A further misalignment appears between perceptions of the importance of feature measurement and experimentation and plans to invest in these capabilities. While 92% and 87% of respondents, respectively, believe feature measurement and experimentation is important, only about 50% either perform these practices or intend to implement them.
- Forty percent of developers think feature management and experimentation makes an impact on the business, but only 16% of leadership agree. This finding is all the more significant considering the consequences of feature release issues, which include job loss, long hours, loss of respect for engineering, and negative impacts on team morale.
- These misalignments are striking in light of areas where a high percentage of respondents feel there is room for improvement, e.g., technical debt, inability to measure business impact of features, impact of problematic features, etc.
- One conclusion that can be drawn from the tensions between these findings is that feature management and experimentation represent an area of immaturity for software organizations. Stakeholders want these capabilities. They understand their importance. Yet, adoption lags.
- Overall, the findings support the idea that it's time to move toward feature management and experimentation practices and platform adoption. Organizational inertia, which translates into immaturity with feature management and experimentation, leads to problems with feature releases. These problems can be eased, if not solved altogether, by feature management and experimentation platforms.

ABOUT FEATURE MANAGEMENT AND EXPERIMENTATION

What's in a software feature release? While end users may not think in terms of feature releases, their user experience (UX) is definitely affected by feature quality. For the people who develop, manage and sell software, features are everything. A well-designed and executed feature can easily translate into competitive differentiation for a software product, driving adoption and potentially vaulting the software to industry dominance.

Furthermore, feature quality does not just affect software companies. Virtually every business today is dependent on software to operate–and many write their own applications. It is no longer controversial to contend that all companies are now software companies, to some degree. That is the effect of digital transformation.

This is the context for the important work of managing software features. A feature doesn't just appear in applications (one hopes). Rather, a feature comes into existence after a careful analysis of user behavior and product management deliberations. Innovative software makers tend to experiment with features, too. For instance, will adding a button help with user experience (UX) or not? Will getting rid of a drop-down menu make UX better or worse?

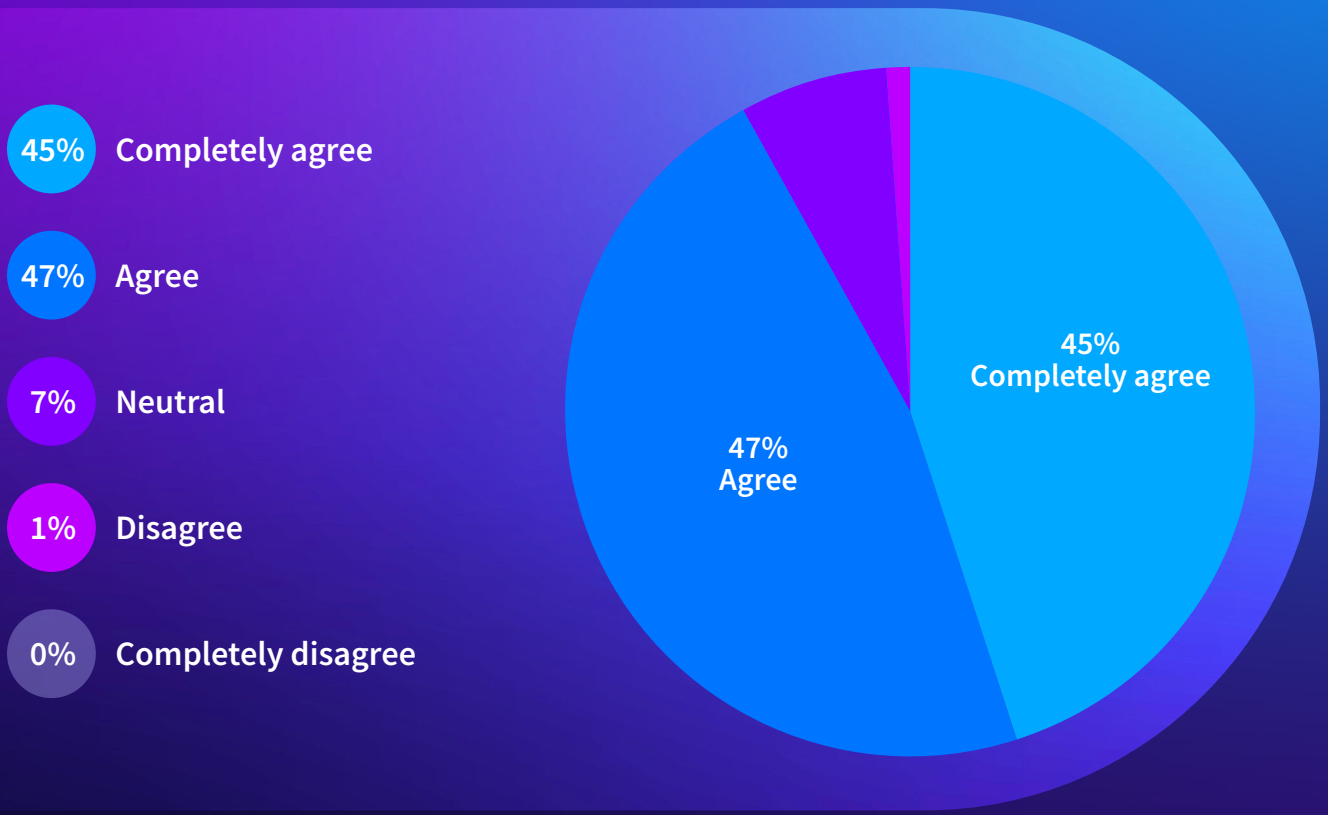
A new class of software development tools has emerged to enable product development teams and other stakeholders in the production of software to track how various features are performing. Some, such as Split, provide "feature flags" that allow developers to turn a feature on for certain users, but not others. Feature flags make it possible to reveal features to a constrained subset of users and assess whether that feature has helped or hindered UX, among other factors.

WHAT RESPONDENTS THINK ABOUT FEATURE MANAGEMENT AND EXPERIMENTATION

The first part of the survey dealt with what respondents think about feature management and experimentation. It covered their perceptions of these processes and their abilities to execute them. The survey then explored what respondents are doing in this context, and what they plan to do.

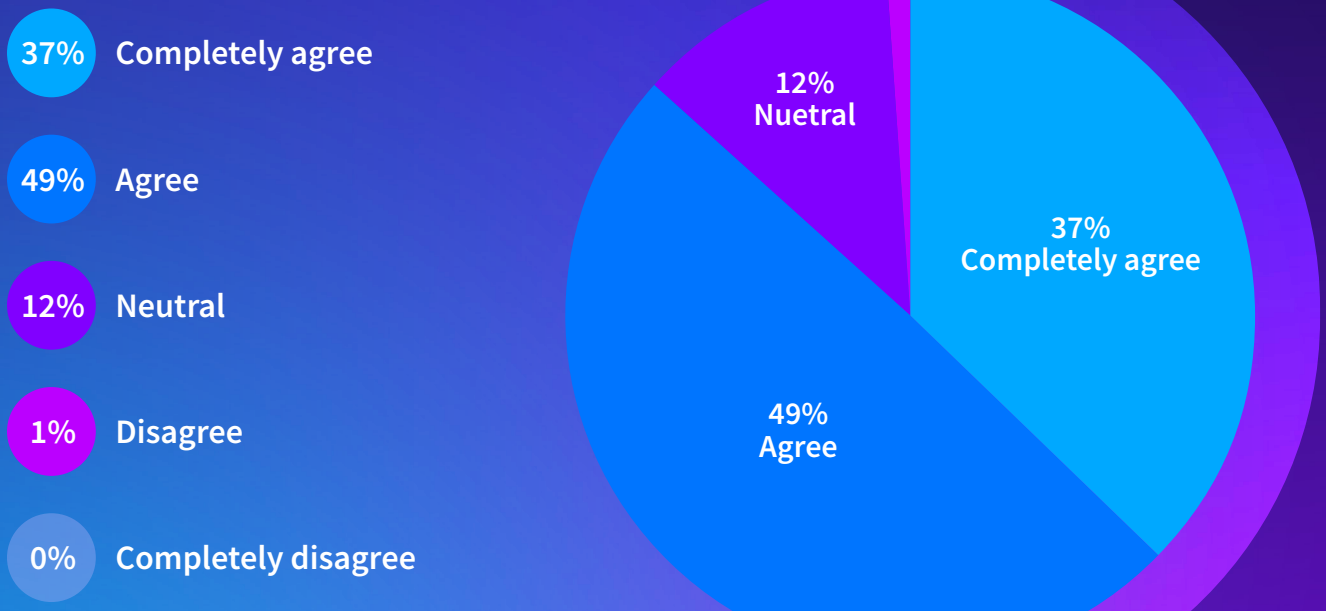
2

To what degree do you believe software **feature management** is critical to developing and releasing successful digital experiences?



How important is feature management to delivering successful digital experiences? A remarkable 92% of respondents believe that feature management is critical to achieving this objective. There seems to be clear consensus on this matter.

3 To what degree do you believe software **feature experimentation** is critical to developing and releasing successful digital experiences?



The related question, “To what degree do you believe software feature experimentation is critical to developing and releasing successful digital experiences?” garnered similar results. Just under 13% disagreed or were neutral on the issue. Forty-nine percent agreed and 37% completely agreed. Together, that’s 87% of respondents saying they think feature experimentation is important when it comes to releasing successful digital experiences for software users.

4

Right or wrong, **how successful do you think your company is** with software feature delivery and management?

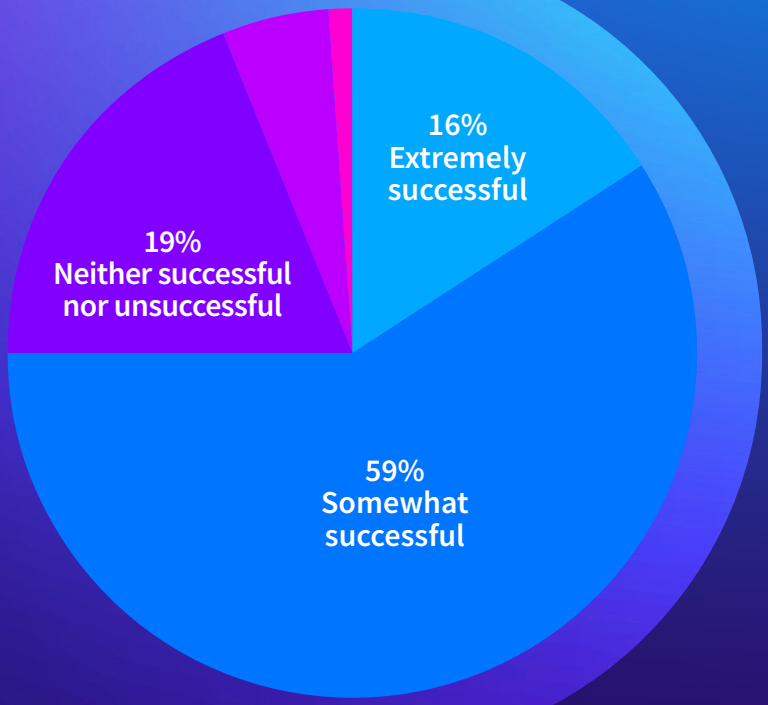
16% Extremely successful

59% Somewhat successful

19% Neither successful nor unsuccessful

5% Somewhat unsuccessful

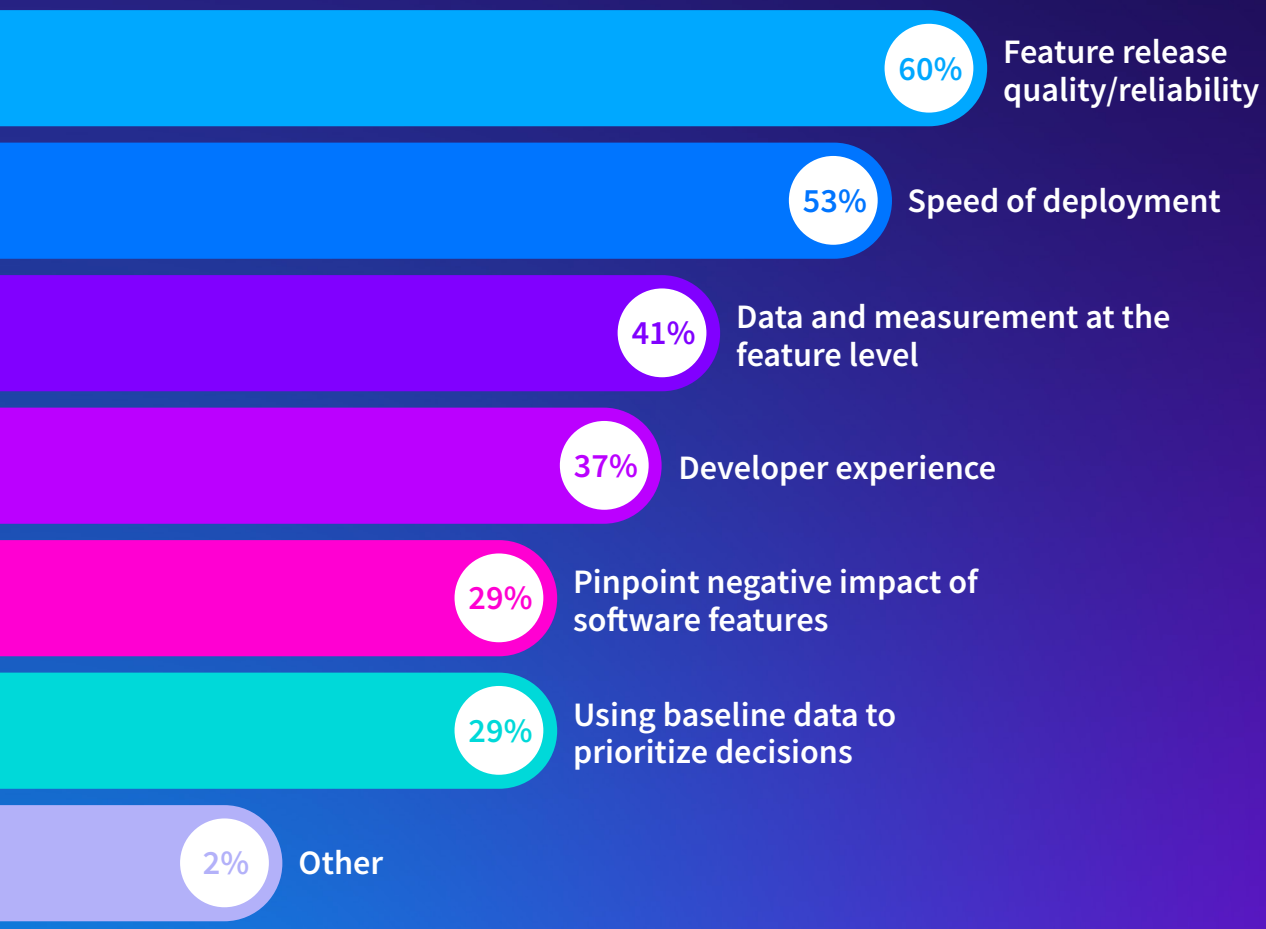
1% Extremely unsuccessful



The consensus about the importance of software feature management and experimentation notwithstanding, respondents did not view themselves as entirely successful in software feature delivery and management. Asked "Right or wrong, how successful do you think your company is with software feature delivery and management?" 59% rated themselves as "somewhat successful." A quarter said they were either "extremely unsuccessful" (1%), "somewhat successful" (5%) or "neither successful nor unsuccessful" (19%). Sixteen percent said they were "extremely successful."

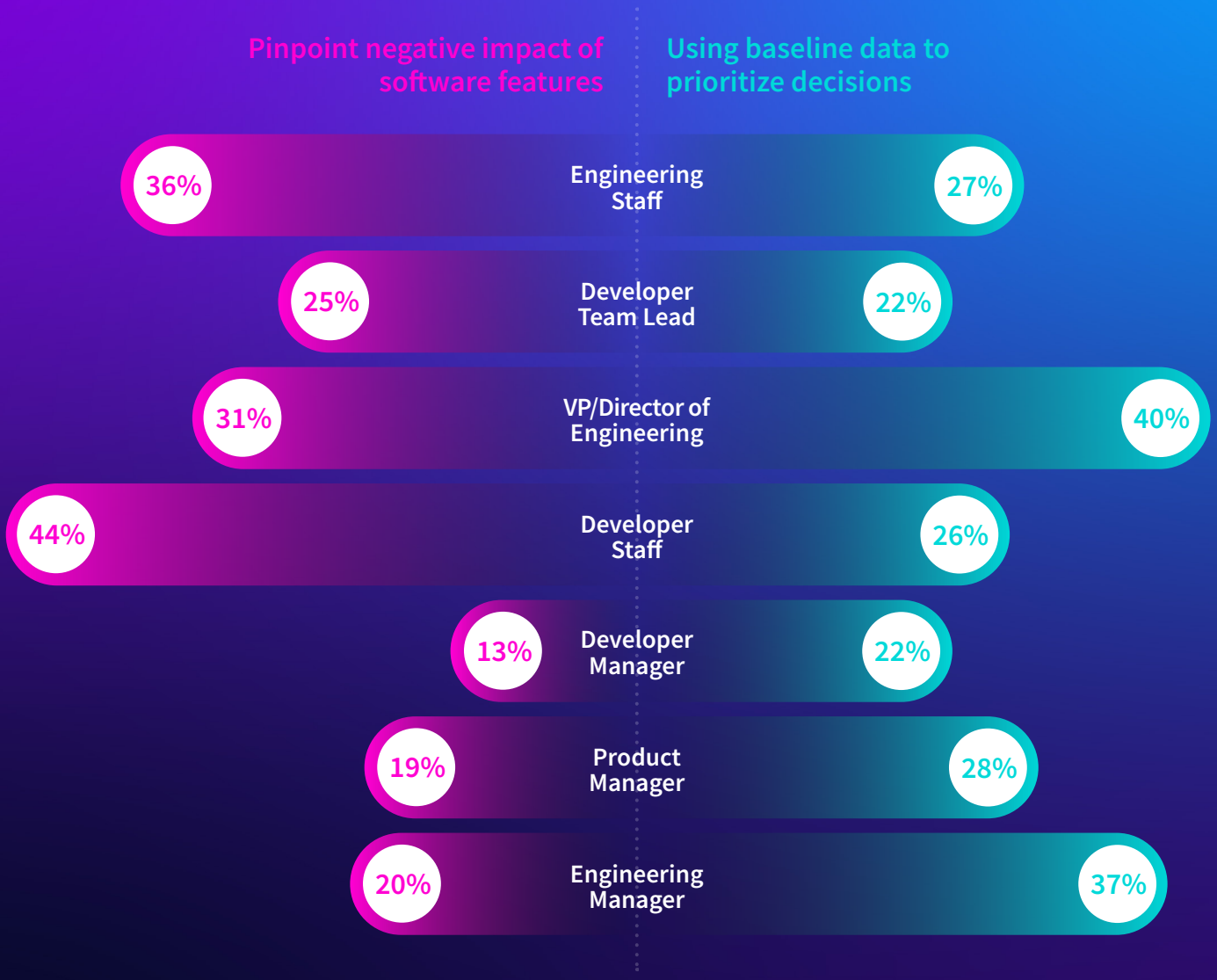
What can be made of the fact that nearly six in ten respondents considered themselves "somewhat successful" at feature delivery and management? It could be viewed as a positive finding. Combined with the "extremely successful" respondents, that's three quarters of the group experiencing some level of success in this important workload. Conversely, 59% seeing themselves as "just" somewhat successful shows room for improvement.

5 Which of the following are the **top areas for improvement?**



Where is there room for improvement? Sixty percent of respondents said software feature release quality/reliability was a top area for improvement. The next highest choice for improvement was speed of deployment, cited by 53% of respondents. Indeed, only about half as many respondents sought improvement in feature impact and the use of data to drive decisions about features, which are core to feature management and experimentation.

6 Which of the following are the **top areas for improvement?** Broken down by **job category:**



Going further, the data reveals possible organizational immaturity. Figure 6 breaks down the question "Which of the following are the top areas for improvement?" by job category. Just 13% of development managers chose "Pinpoint negative impact of software features," versus 44% of developer staff. There's a discrepancy between management and staff on the issue, showing the presence of silos within product development teams. Similarly, while 36% of engineering staff chose pinpointing negative feature impact, only 20% of engineering managers made this choice.

A comparable disparity shows up for "Using baseline data to prioritize decisions." This choice of what can be improved was selected by 40% of VPs/directors of engineering, but 27% of engineering staff and 26% of developer staff. Managers apparently see the problem as a more serious issue than staff does. A more mature organization, one that had come to a consensus about the importance of feature management and experimentation, would likely see fewer mismatches between priorities of room for improvement.

HOW RESPONDENTS ARE ADOPTING FEATURE MANAGEMENT AND EXPERIMENTATION

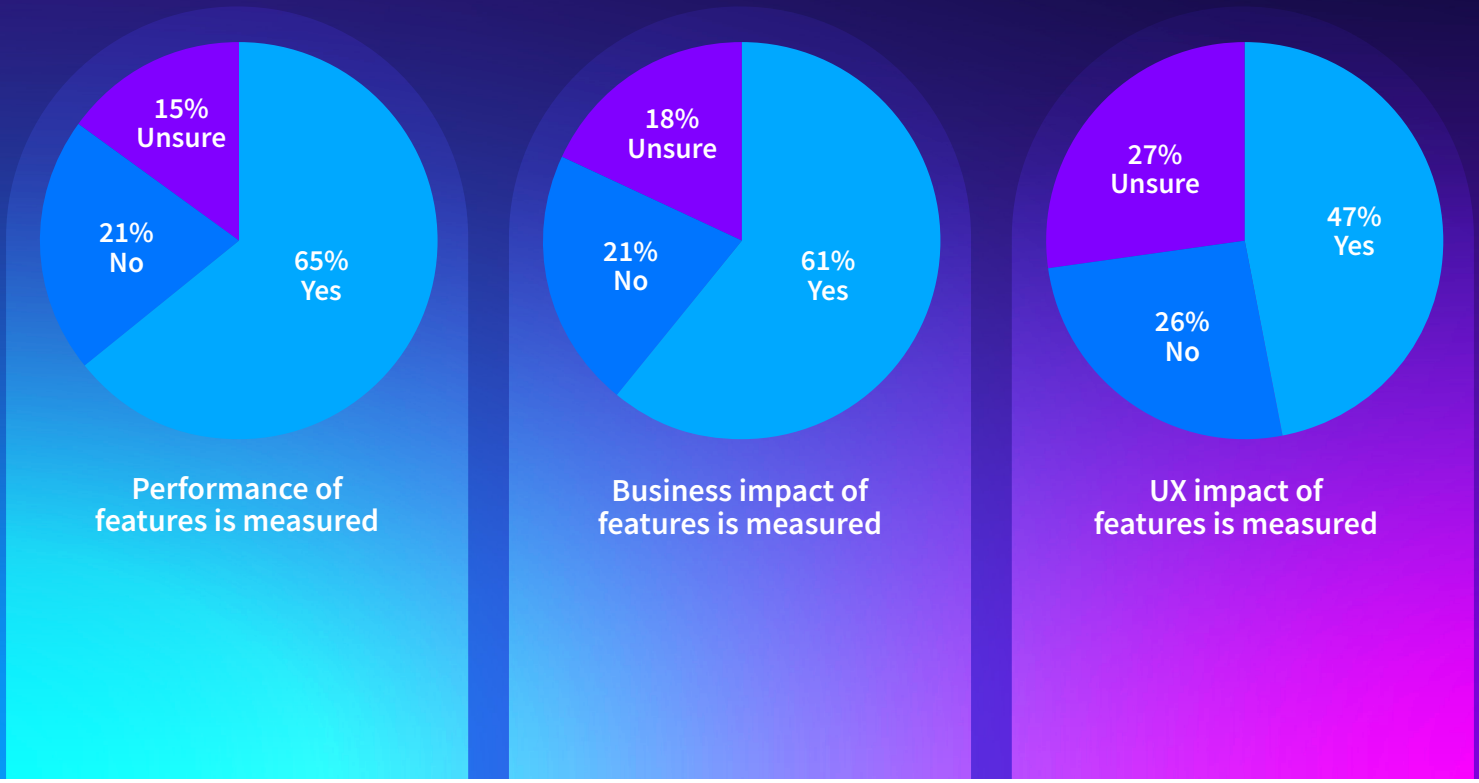
The state of feature management and experimentation reflects a field in flux. Respondents are making the effort, up to a point. At the same time, they are dealing with the consequences of incomplete feature management and experimentation practices.

FEATURE-LEVEL MEASUREMENT

When respondents were asked "How would you describe feature-level measurement at your organization?" 65% said yes, they were measuring performance of software features. Sixty-one percent were measuring the business impact of software features, and 47% were measuring the user experience (UX) impact of software features.

Their responses could refer to the impact of features at the application level, like how a new feature affected UX for the app as a whole, or feature-level performance. This latter case refers to how a specific feature, such as a new shopping cart, affects business outcomes.

7 **How would you describe software feature-level measurement** at your organization?

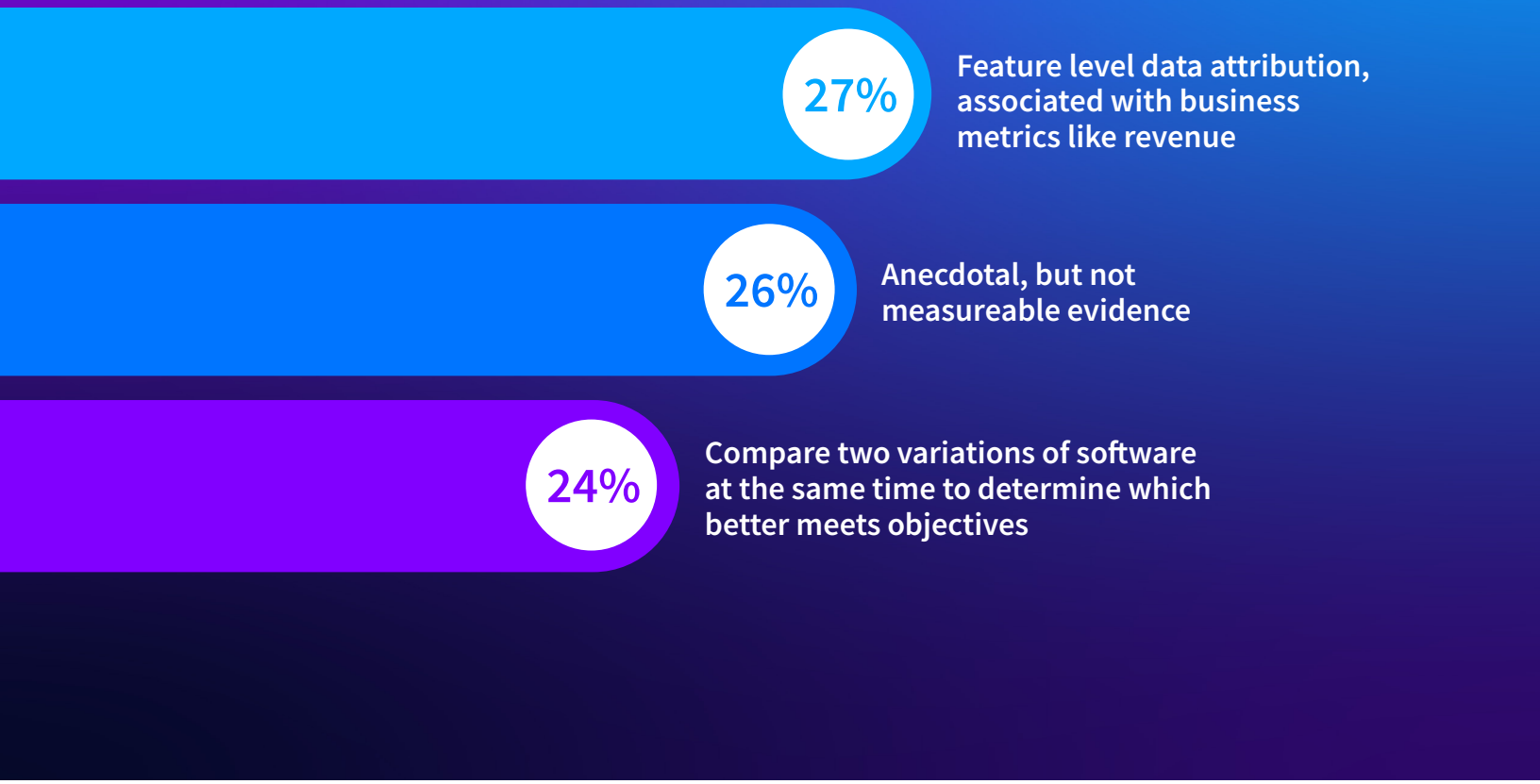


The relatively high rates of measuring feature performance and business impact (65% and 61%, respectively) match the low percentage of respondents who felt there was "room for improvement" in pinpointing the negative impact of software features and using baseline data to prioritize decisions (Figure 5).

FEATURE-LEVEL EXPERIMENTATION

Feature-level experimentation is less common than feature-level measurement. In response to the question, “How would you describe software feature experimentation at your organization?” just 27% said they experiment with feature-level data attribution that is associated with business metrics such as revenue. Twenty-six percent engage in anecdotal feature experiments, using no measurable evidence.

8 **How would you describe software feature experimentation** at your organization?



The fact that fewer than 30% of respondents are engaging in feature experimentation opposes the finding that 87% of respondents also think feature experimentation is critical for successful digital experiences. Recognizing the importance of experimentation does not necessarily translate into embracing it. This may relate to organizational inertia or a lack of maturity in feature management and experimentation. Not every organization is ready. People may lack skill sets and tooling. Experimentation may not be a priority for budget allocations, even if stakeholders recognize its importance for success with applications and the broader business.

FEATURE MANAGEMENT AND EXPERIMENTATION CAPABILITIES

More than a third of respondents currently have some feature management and experimentation capabilities. Figure 9 shows the percentage of respondents who chose “Implemented” as the status for each of the software feature management and experimentation capabilities listed. For example, 39% of respondents have implemented the ability to “automatically notify teams responsible for a problematic software feature to speed resolution.”

9 Percentages of respondents who have implemented various software **feature management and experimentation capabilities**

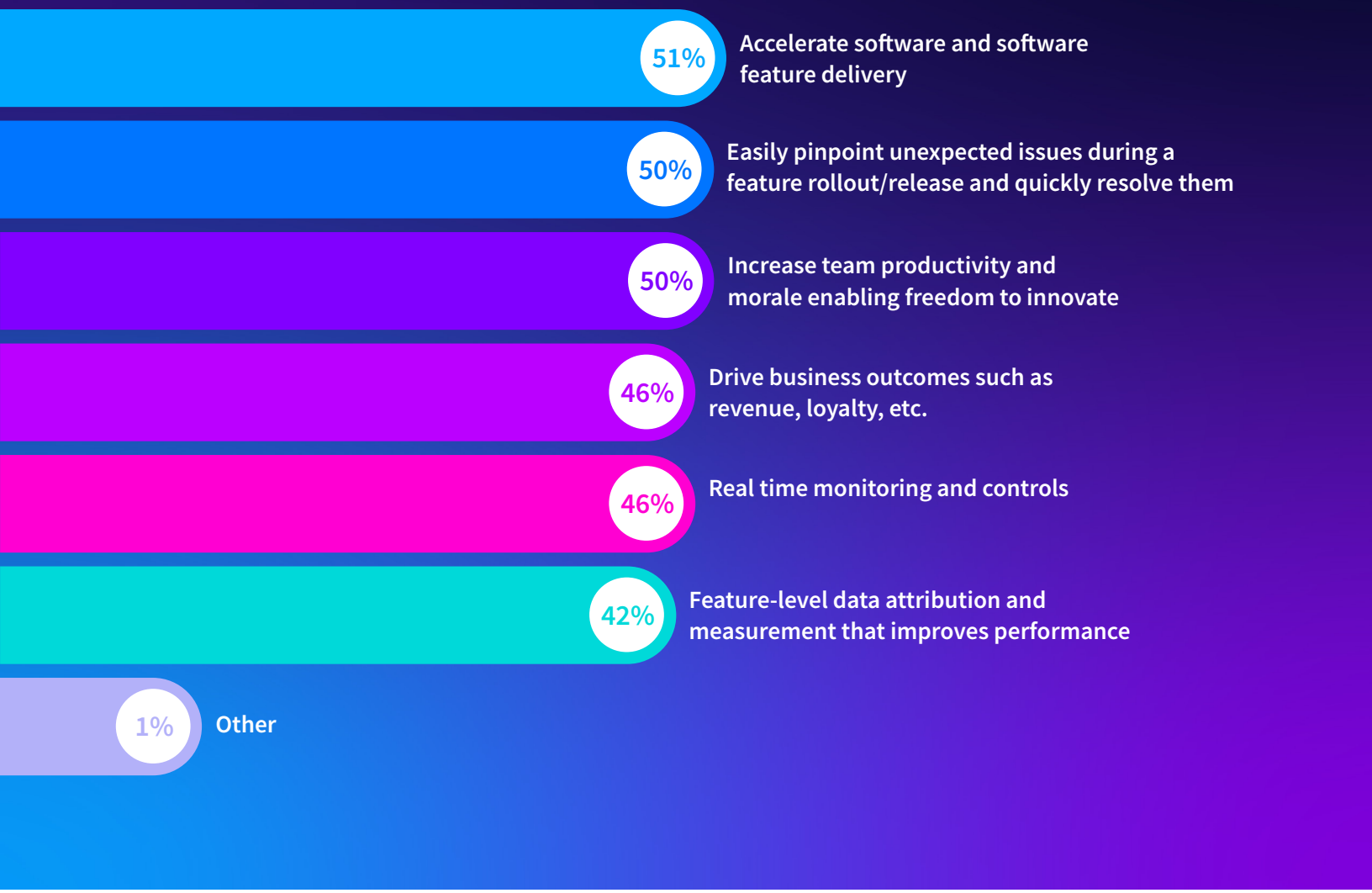
FEATURE DELIVERY PRIORITIES

Feature delivery is where "the rubber meets the road" in feature management and experimentation. Development teams and stakeholders in engineering and product management are under pressure to get features out into production as quickly as possible, but without compromising quality. Respondents spoke to these connected issues.

Asked, "Which of the below are important priorities related to software feature delivery?"

51% chose "Accelerate software and software feature delivery." Half selected "Easily pinpoint unexpected issues during a feature rollout/release and quickly resolve them." Speed and quality are both priorities — but potentially in conflict with one another. To this point, half of respondents also chose "increase team productivity and morale, enabling freedom to innovate" as a priority.

10 Which of the below are **important priorities related to software feature delivery?**

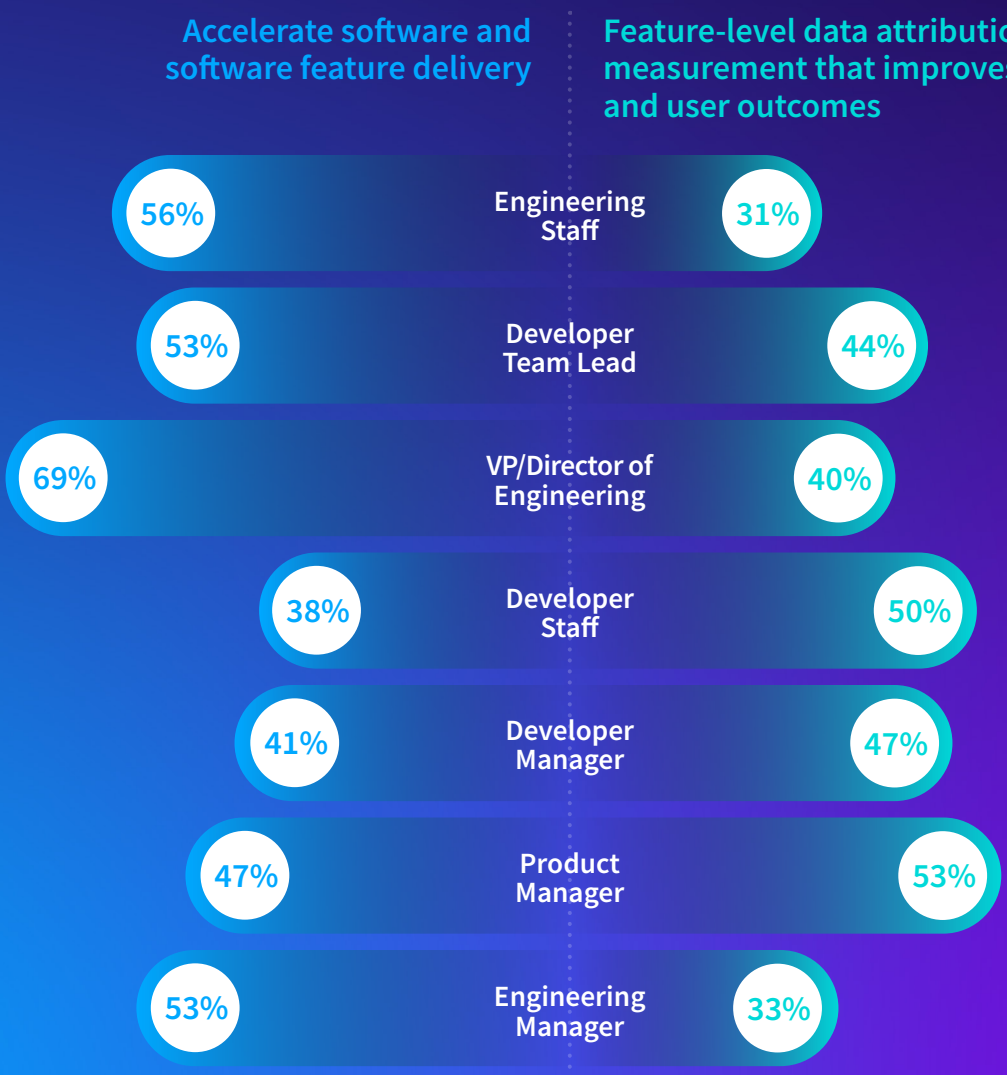


Breaking the findings down by job category provides a glimpse of organizational maturity for feature management. Sixty-nine percent of VPs/directors of engineering and 53% of developer team leads said that to "Accelerate software and software feature delivery" was a priority. Fifty-six percent of engineering staff selected this as a priority. However, just 38% of developer staff and 41% of developer managers had the same view.

This divergence, where a greater proportion of executive-level employees prioritize the acceleration of feature delivery than developers do, suggests that these two groups are not aligned. Achieving acceleration in feature delivery is a priority that needs to be shared by everyone if it's going to work.

11

Which of the below are **important priorities related to software feature delivery?** Broken down by **job category:**



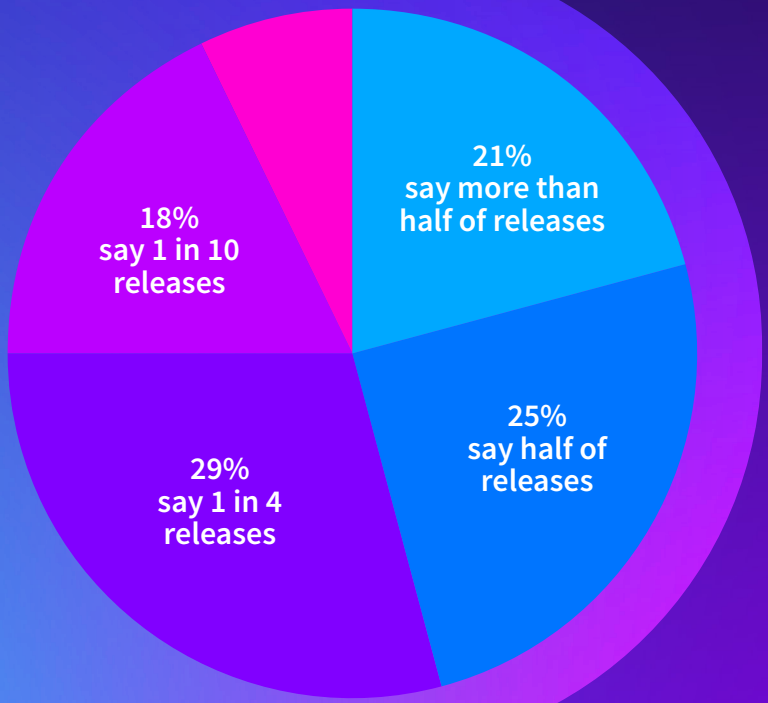
FEATURE RELEASE PERFORMANCE MEASUREMENT

Respondents were asked what percentage of their feature releases were measured for performance. Forty-six percent of respondents indicated that they measured performance for 50% or more of their feature releases. At the same time, 54% measured performance for fewer than 25% of their feature releases.

There is certainly room for improvement for this latter group. These results signify that respondents cannot measure the performance for half or more of their releases. That lack of knowledge can leave the door wide open for a negative outcome.

12 What percentage of your feature releases do you have **performance measurements** for?

- 21% More than half of releases
- 25% Half of releases
- 29% 1 in 4 releases
- 18% 1 in 10 releases
- 7% 1 in 20 releases

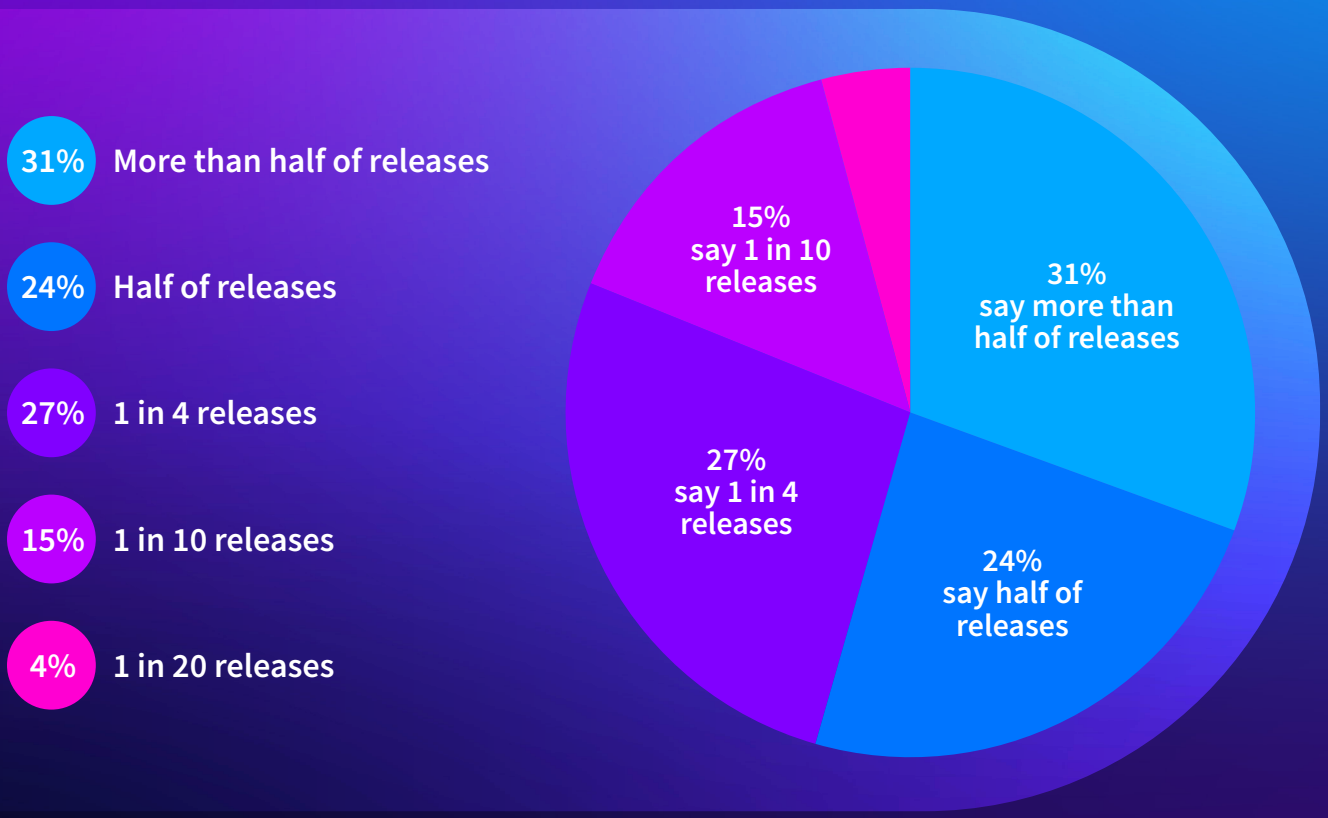


MEASURING BUSINESS IMPACT FOR FEATURE RELEASES

A similar split emerges with the question, “What percentage of your software features and software development has a measurable impact on the business?” Fifty-five percent of respondents say that 50% or more of their features have a measurable impact on their business—meaning they have the ability to measure that impact. Forty-five percent felt that less than a quarter of features had a measurable business impact.

The data points to a division in maturity with respect to feature management and experimentation. About half of the respondents are with organizations that measure business impact of feature releases. These are mature organizations. The other half are not. A mature organization should be able to measure business impact, such as by connecting event data to key performance indicators (KPIs). For example, adding a “Remember me” button in an e-commerce store might drive revenue growth because it makes shopping easier, at least in theory. A mature organization will be able to track whether adding such a button helps with revenue.

13 What percentage of your software features and software development **has a measurable impact on the business?**



14

What percentage of your software features and software development **has a measurable impact on the business?** Broken down by **job category:**

Cutting the data by job category shows another marker of organizational immaturity and silos. As the figure shows, while 40% of developer team leads say over 50% of their features have measurable impact on the business, just 16% of engineering staff, 16% of VPs/Directors of Engineering and 11% of developer staff have the same view. They're talking about the same features. How could perceptions be so divided? One answer is that they simply don't know, or, that organizational silos do not share data about features' impact on the business. Or, they lack the tooling to do so.

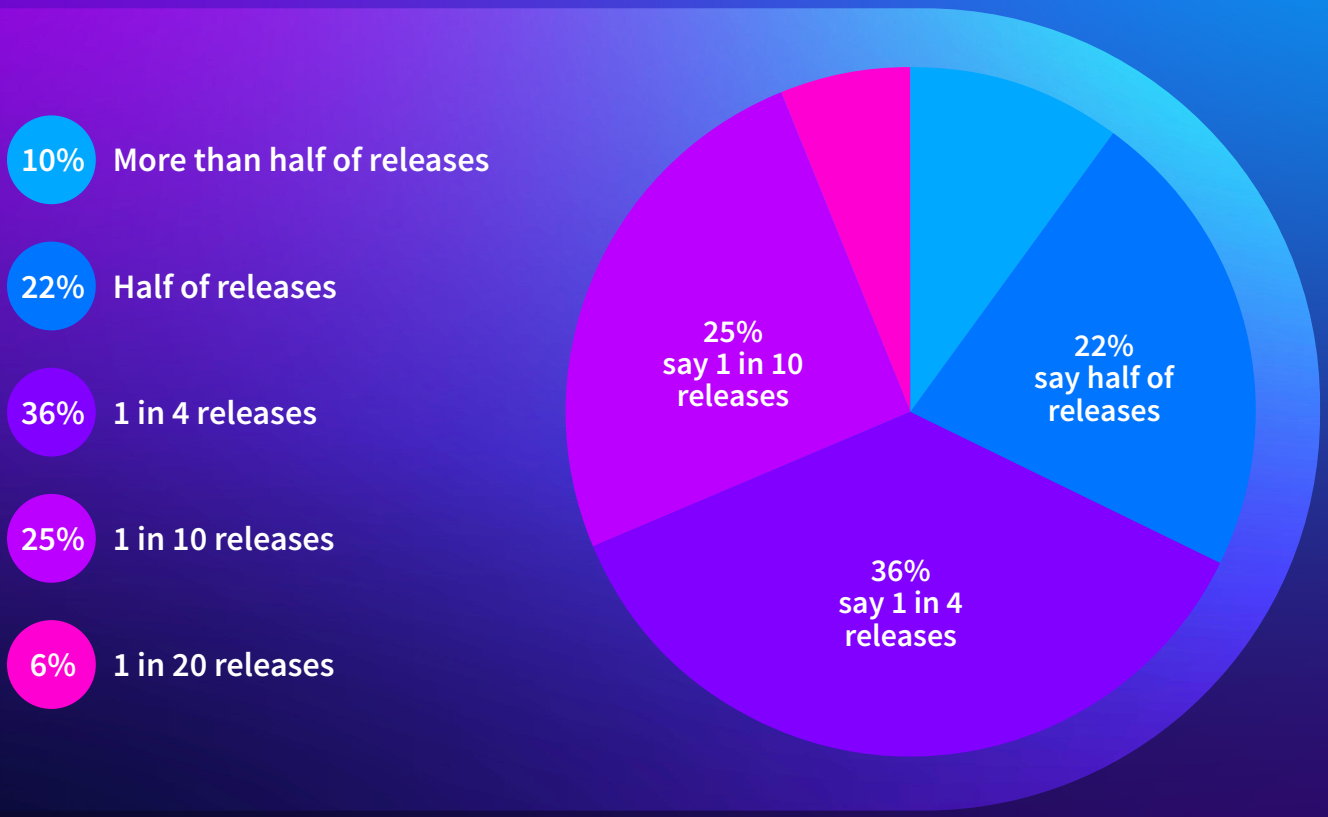
TECHNICAL DEBT

The survey queried respondents about technical debt, which refers to extra rework caused when developers choose to employ an easy, fast solution over a more durable approach that would take longer. Like financial debt, technical debt requires periodic repayments of time as the original poor choice ages — making it harder to add new features to an application. Technical debt is relevant to feature management and experimentation because excessive technical debt will slow down a development team's ability to innovate and experiment with new features. They'll be tied up fixing what they did wrong before.

Asked, "What percentage of your software engineering effort results in technical debt?" 31% said that less than 10% of their engineering effort resulted in technical debt. Thirty-six percent said that 25% of their efforts led to debt, while 32% said that 50% or more of their efforts ended up with technical debt.

What do these numbers mean? It's encouraging that two-thirds of respondents only had technical debt in 25% of their engineering. They're not drowning in technical debt, so to speak. That said, for the other third to be saddled with technical debt by 50% or more of their engineering effort is problematic for them.

15 What percentage of your software engineering effort **results in technical debt?**



16

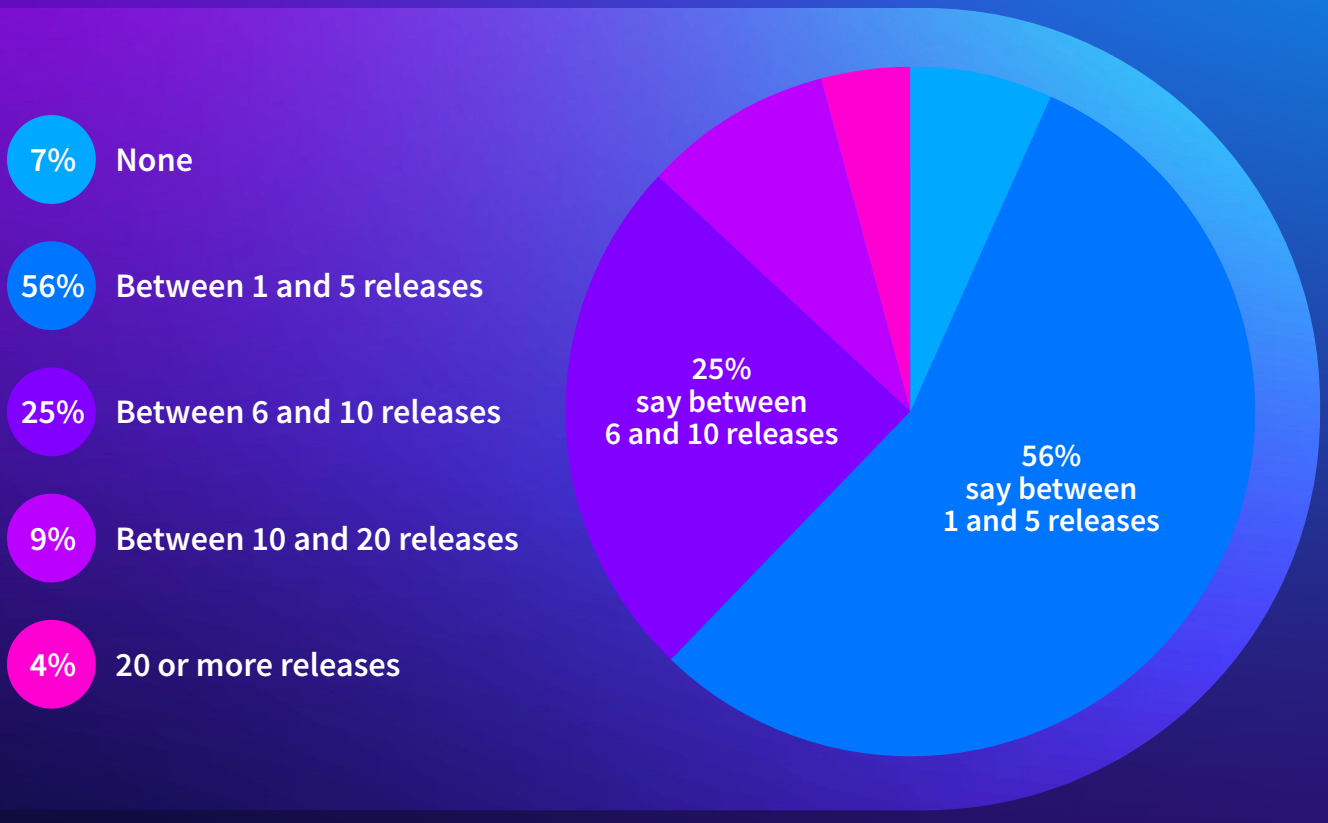
What percentage of your software engineering effort **results in technical debt?** Broken down by **deployment frequency:**

Interestingly, the percentages of technical debt in engineering did not change much when sorted by frequency of deployment. One might think that organizations with a slower releasing schedule, such as those that deploy new software between once per month and every six months, would have lower levels of technical debt compared to fast-paced groups that put out new features on demand, perhaps multiple times per day. They don't. This suggests that everyone is doing it equally well, or badly.

DEALING WITH PERFORMANCE ISSUES

Technical debt is not the only issue that can come up in software releases. Deploying new code is going to cause trouble at least some of the time. Asked "How many releases led to production software issues that had a material impact on the business and/or user experience?" over half of respondents said between one and five releases. A quarter of respondents said between six and 10 releases. Over 10 releases was relatively rare.

17 How many releases **led to production software issues** that had a **material impact on the business and/or user experience?**



CHANGES RESULTING IN FAILURE

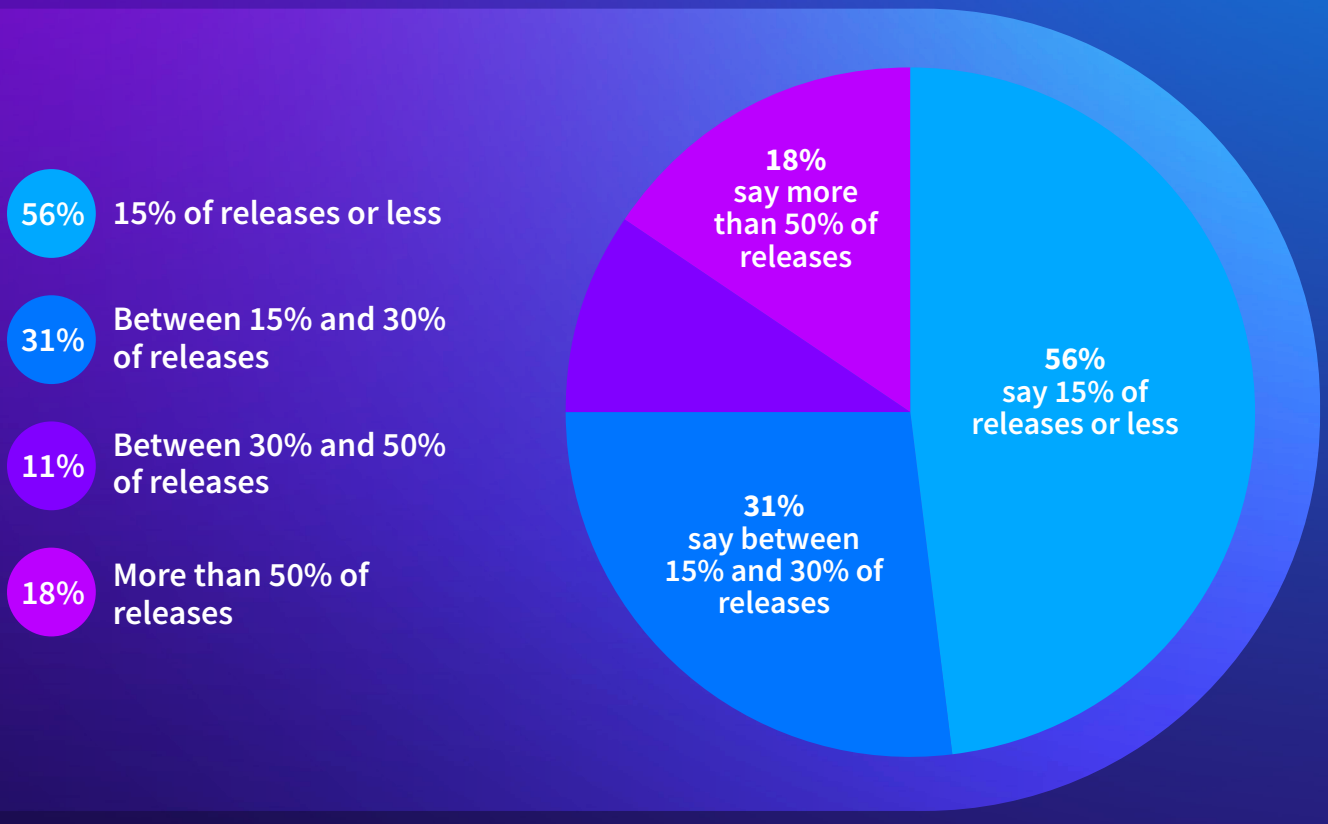
In some cases, a change to a software application results in degraded service. This kind of change failure is not a major problem for survey respondents, with 56% saying that they only had change failures between 0% and 15% of the time. Thirty-one percent said they had change failure 15% to 30% of the time, and 11% said change failures occurred between 30% and 50% of the time.

Still, any failure is a source of trouble and stress, so it's an area for improvement for everyone.

Fixing such problems is of course a high priority, with 58% of respondents saying their mean time to repair (MTTR) was less than a day. For 21%, the MTTR is less than an hour. A slightly smaller number (19%) had MTTRs of less than a week.

18

Change Failure Rate: What percentage of changes to production or end-users results in degraded service?

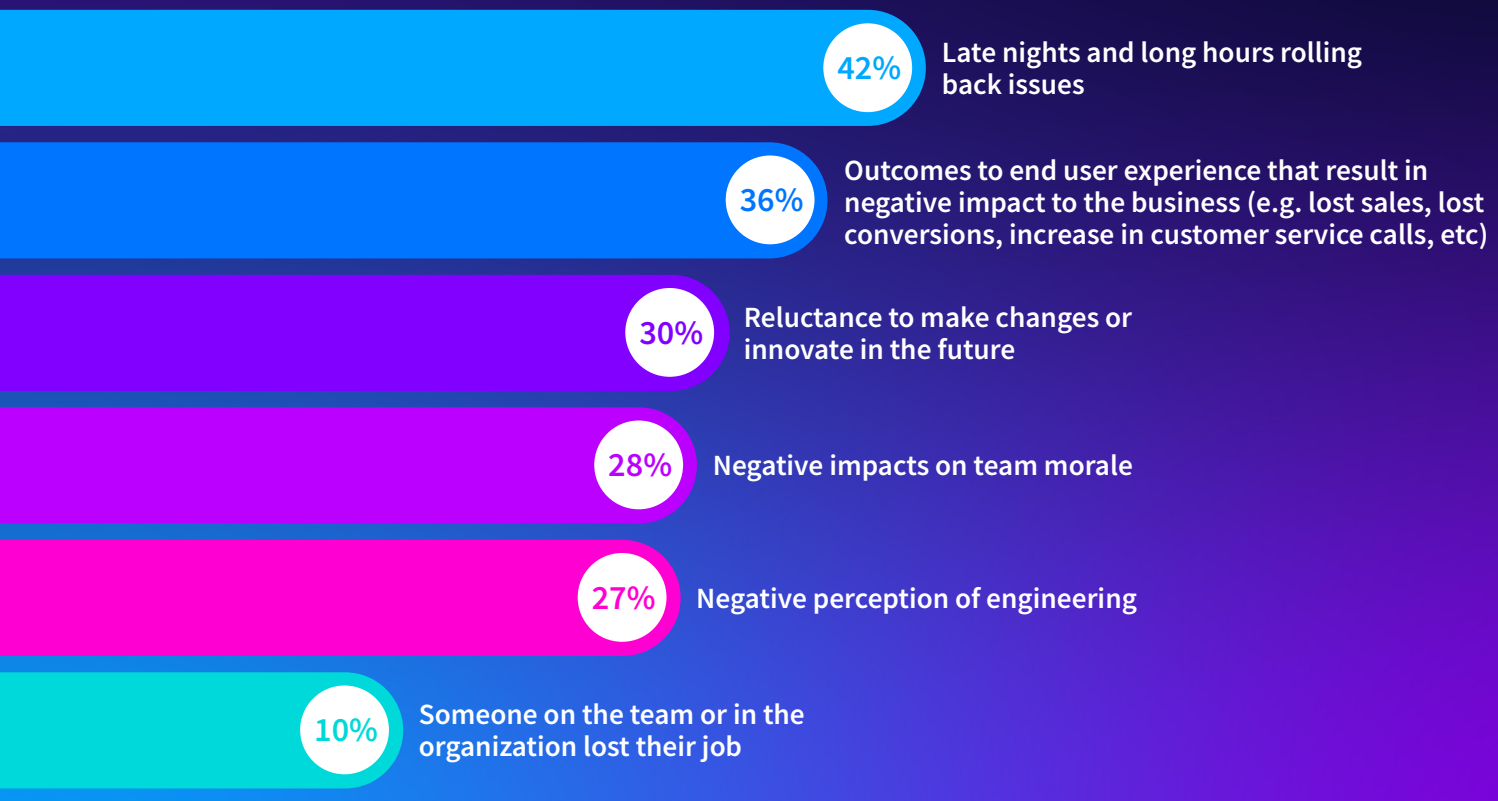


RESPONDENTS REPORT SERIOUS CONSEQUENCES FROM PROBLEM-CAUSING FEATURES

Even if MTTRs are fast, issues in software feature releases can have many consequences for a software organization. At a minimum, feature problems become team problems. Forty-two percent of respondents cited “Late nights and long hours rolling back issues” as a consequence. Twenty-eight percent felt that feature issues negatively impact team morale, while 10% said that someone on the team or in the organization had actually lost their job over an issue with a software feature release.

Twenty-seven percent indicated that issues in software feature releases create negative perceptions of engineering. Thirty percent said that problems with software feature releases made teams reluctant to make changes or innovate in the future. This is a highly problematic state to be in — where fears of negative repercussions for software changes create hesitation when introducing new features.

19 Which of the following, if any, **consequences have you or your organization experienced** from software feature release issues?



End users suffer, too. Thirty-six percent of respondents said that problems affecting user experience led to real negative business impacts like lost sales, lost conversions, more calls to customer service and so forth.

WHAT RESPONDENTS LOOK FOR IN FEATURE MANAGEMENT AND EXPERIMENTATION TOOLS

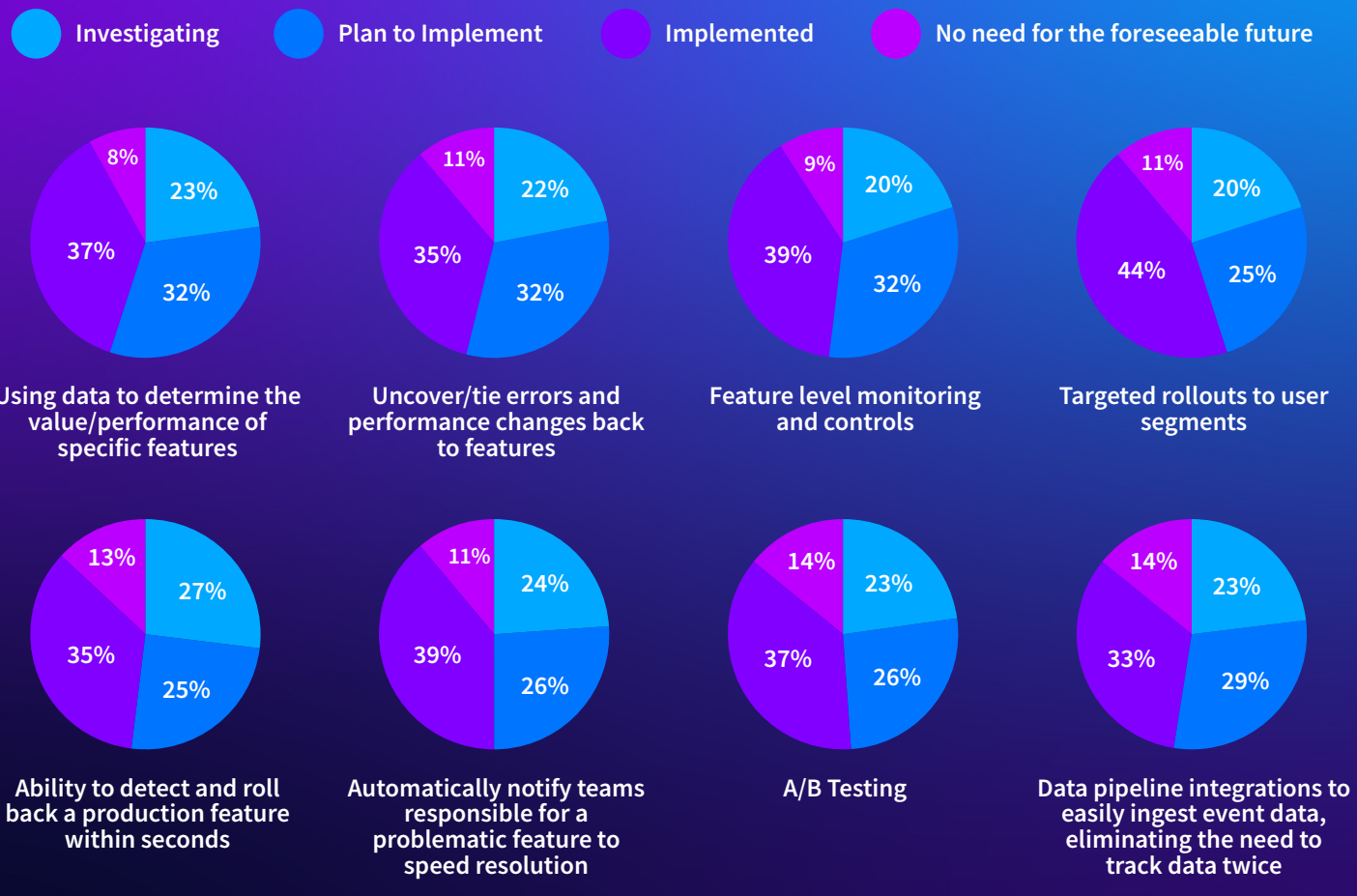
When it comes to actual implementation of feature management and experimentation capabilities, some respondents are already using tools. As Figure 9 showed previously, over a third of respondents have already implemented a range of capabilities. Forty-four percent have implemented the ability to do targeted feature rollouts to user segments, for example. Thirty-seven percent are using data to determine the value or performance of specific features.

Most respondents, however, are not there yet. For each capability shown in Figure 20, roughly half are either investigating or planning to implement. Thirty-two percent plan to implement feature-level monitoring and controls, while 20% are investigating the capability. For A/B testing, 26% are planning and 23% considering. For uncovering/tying errors and performance changes back to software features, 32% are planning and 22% are investigating.

These figures are at odds with the findings shown in Figure 2, where 92% agree or completely agree that feature management is critical to successful digital experiences, and Figure 3, where 87% agree or completely agree that feature experimentation is critical. If that high a proportion of respondents consider feature management and experimentation critical to successful digital experiences, why do only about a third actually do it, and about half are looking into it?

20

What is **your organization's status** for these software feature management and experimentation capabilities?



21

What is **your organization's status** for these software feature management and experimentation capabilities? Broken down by **deployment frequency:**

The answer is not clear, but there are several possible explanations. One is simply a lack of organizational maturity around feature management and experimentation. Inertia is a potential problem, as well. If an organization isn't ready to address the factors that contribute to its immaturity and invest in the right technology, it will lag in implementation. While having the right platform makes a big difference, it is also necessary to bring stakeholders together to agree on policies, procedures, and workflows. This takes time and effort.

Alternatively, interest in feature management and experimentation may depend on factors like deployment frequency. Figure 21 breaks down organizational status for some of the feature

management and experimentation capabilities from Figure 20 by deployment frequency. The slower organizations, those that deploy new features between once per month and once every six months, have high rates of "No need for the foreseeable future." This makes some sense, given that slow-moving organizations may feel less pressure to stay on top of feature delivery.

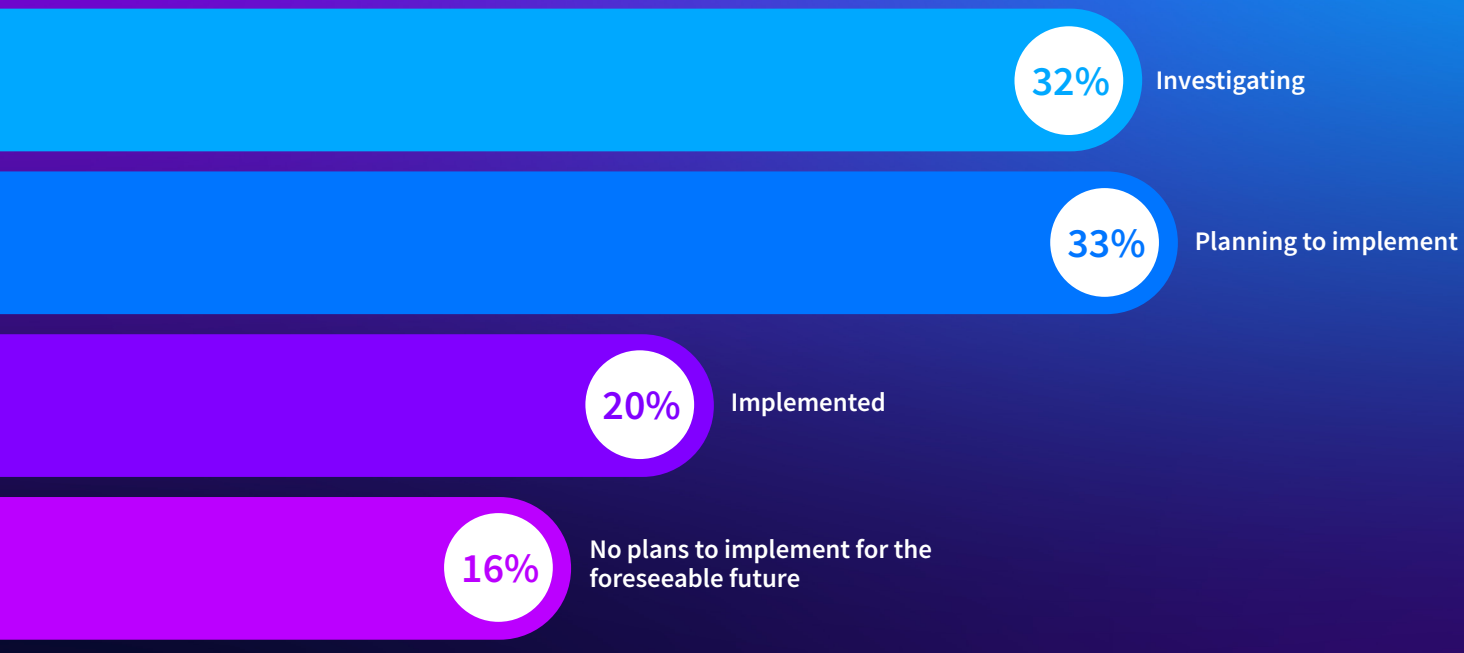
For example, 18% of those in the one-to-six-month deployment category do not see a need for the ability to detect and roll back a production software feature within seconds. In contrast, 13% of those in once-per-week to once-per-month do not see a need, as do 10% of the once-per-day to once-per-week. The teams with a faster schedule are less likely to see no need for a feature management capability like that.

The other capabilities in Figure 21 show a similar pattern. Those in the one-to-six-month deployment category have the highest percentage of respondents seeing "no need" to explore these other capabilities. Most of the rapidly releasing organizations do recognize the need for disciplined feature management and experimentation. The small minority who do not, may be fortunate to have avoided disaster to this point, but their time will run out. When it does, their next step will be to move into one of the slower-moving groups or adopt dramatically improved practices to maintain their pace.

ADOPTION OF FEATURE MANAGEMENT AND EXPERIMENTATION PLATFORMS

Organizations that want to mature in feature management and experimentation have the option of acquiring a platform built for these purposes. Of the two, feature management has a higher rate of implementation, with 20% of respondents indicating that they already had a feature management platform versus 14% for feature experimentation platforms.

22 How would you describe **your organization's adoption of feature management platforms?**



23

How would you describe **your organization's adoption of feature management platforms?** Broken down by **deployment frequency:**

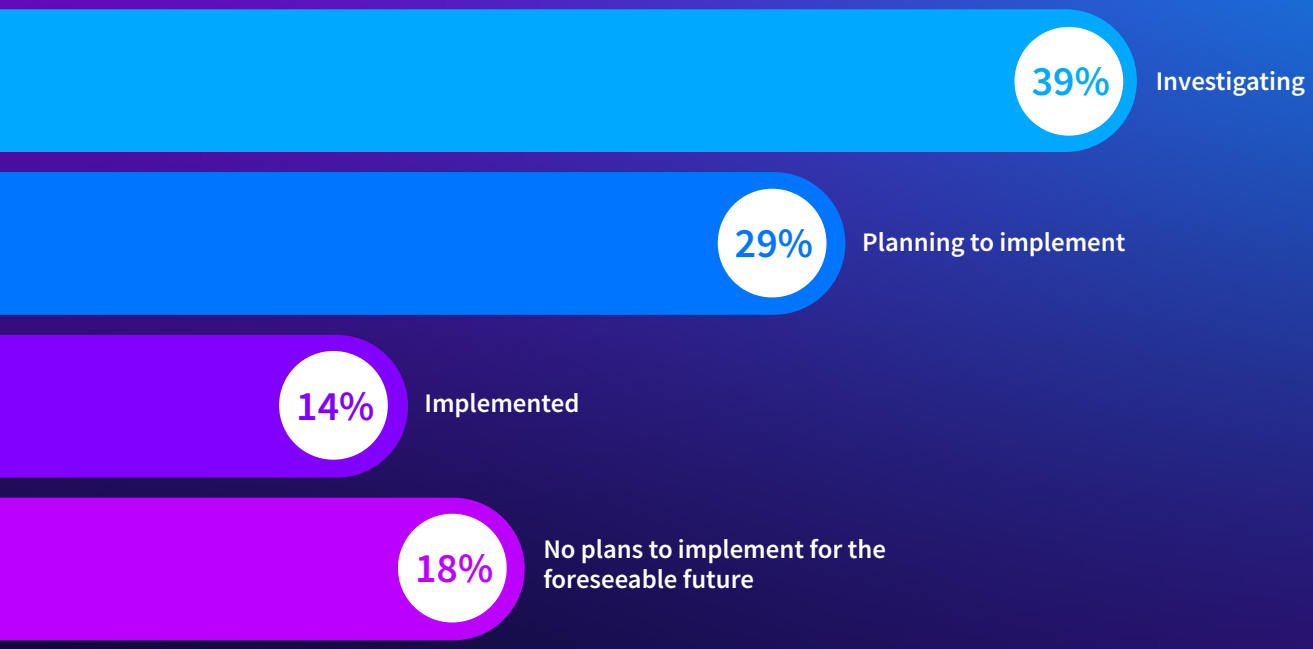
Experimentation platforms appear to be earlier in the consideration cycle. Thirty-nine percent of respondents are investigating experimentation platforms (Figure 24), while 32% are investigating feature management platforms. Twenty-nine percent are planning to implement experimentation platforms.

Breaking the responses down by deployment frequency reveals differences in organizational approach. The organizations that deploy on demand have the highest proportion of respondents investigating a feature management platform, and the lowest planning to implement. This may be an indication of fast-moving organizations relying on homegrown solutions. Although there is inertia against migrating, they likely recognize the need to investigate a purpose-built tool that has long term potential and less technical debt.

Further to this point, on-demand organizations also had a relatively high rate of "No plans to implement in the foreseeable future." Eighteen percent of respondents from on-demand organizations answered this way, followed by 12% of once-per-day to once-per-week deployers and 10% for once-per-week to once-per-month deployers. These latter two types of organizations are moving sufficiently fast that feature management could benefit their processes and quality. Not surprisingly, organizations with slow-paced once-per-month to once-every-six months deployment, have the highest rate of "no plans," at 27%.

24

How would you describe **your organization's adoption of feature experimentation platforms?**



The combined numbers are close, however. Sixty-five percent of respondents are either investigating or planning to implement a feature management platform. Sixty-eight percent of respondents are either investigating or planning to implement a feature experimentation platform.

25

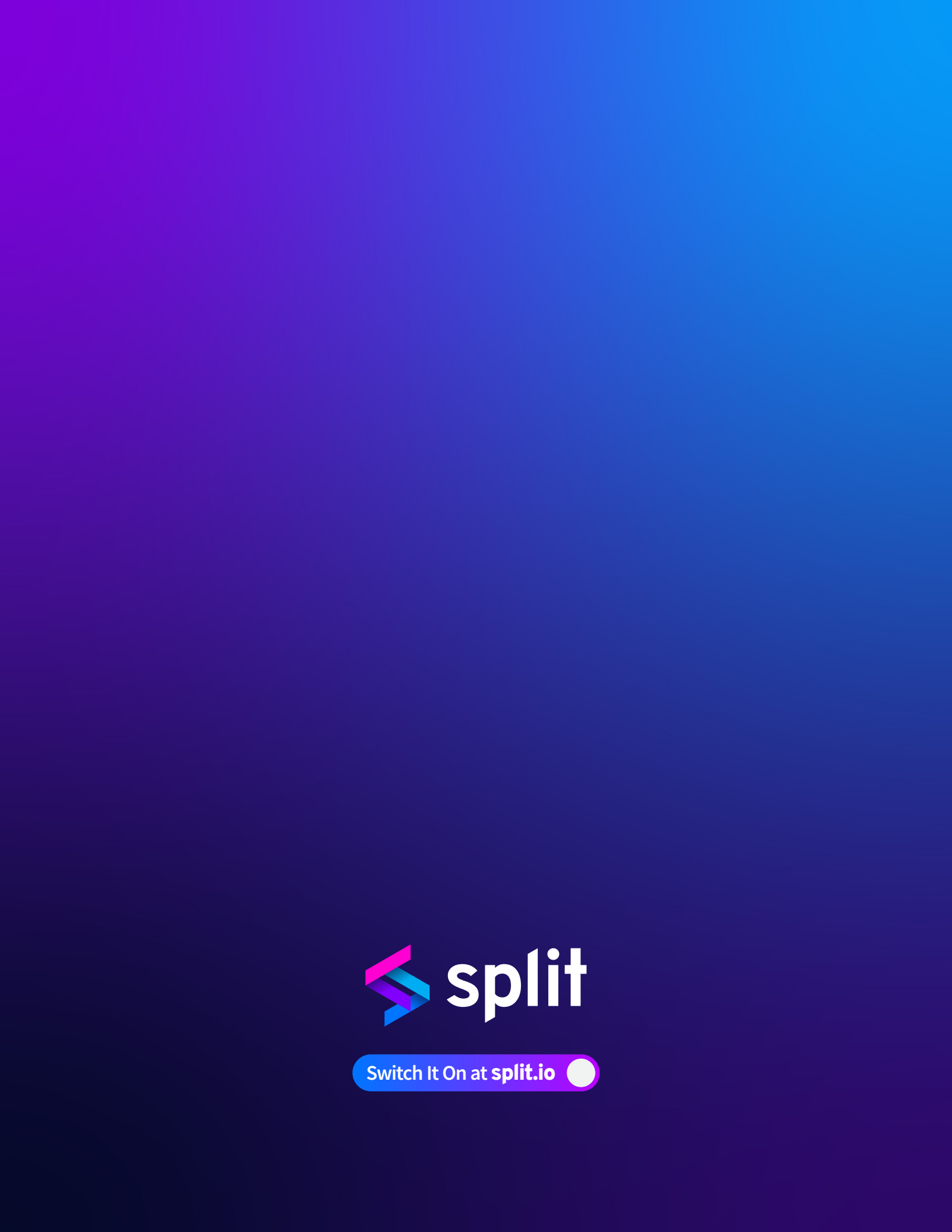
How would you describe **your organization's adoption of feature experimentation platforms?** Broken down by **deployment frequency:**

Looking at the status of feature experimentation platform adoption by deployment frequency, the data again reveals on-demand organizations lagging their slower peers, with 10% having implemented, vs. 19% of once-per-day to once-per-week deployers and 14% for once-per-week to once-per-month deployers. On-demand organizations do seem quite interested, however. They have the highest rate of "Planning to implement" feature experimentation platforms.

CONCLUSION

The survey provides useful insights into the state of feature management and experimentation. It highlights a range of maturity levels in two essential areas of software development and releasing. The mix of maturity levels is evident in the misalignments between the recognition of the importance of feature management and experimentation and respondents' lack of proportional follow through in implementation. The actual use of platforms for feature management and experimentation shows even less progress.

It's a situation with its share of nuances and surprises. For example, organizations that deploy software features on-demand tend to be less engaged in feature management and experimentation than their slower peers. The data does show strong intentions to adopt feature management and experimentation, however. Software organizations seem to be moving in that direction, and the time has come to act if they want to keep pace with competition from digital leaders. They must overcome immaturity if they want to release features faster while controlling their risk — a key success factor for competitive differentiation in the software field.


split
Switch It On at split.io