OverOps

Dev vs. Ops:

The State of Accountability

Table of Contents

Executive Summary

There has long been a divide between development and operations teams. But recently, there has been a movement, within both small startups and massive enterprise organizations alike, to break down these metaphorical walls and build bridges of shared accountability between the two functions. With the emergence of roles like DevOps and Site Reliability Engineering (SRE), we are seeing the introduction of a more collaborative approach to delivering reliable software.

Still, in the context of increasingly distributed and complex systems and tooling, when things go awry, accountability often remains unclear. In the heat of battle, when an application breaks and customers are feeling the burn, who is ultimately responsible for ensuring application reliability? Do enterprises with DevOps workflows have the right processes in place to ensure quick resolution of issues?

In our Dev vs. Ops: The State of Accountability report, we surveyed over 2,000 IT professionals around the globe to get a sense of how shared accountability affects the delivery of reliable software in a DevOps environment, and what are some of the top challenges teams face when it comes to building and maintaining quality applications.

Key Findings

DevOps — no longer just a buzzword, but still not a household practice.
The majority of respondents said that DevOps is in their roadmap. However, over 82% of organizations have only partially adopted DevOps practices (or haven't adopted any), in contrast to just 17% of respondents that claimed to have fully adopted DevOps.

Organizations are under pressure to deliver software faster than ever — which is causing their applications to break.
More than 90% of respondents are deploying code at least once a month, and over 60% are deploying code at least once every two weeks. At the same time, nearly 40% of all respondents indicated that moving too quickly is a primary reason that errors make it into production.

The road to DevOps is paved with chaos.
With most organizations in the midst of DevOps adoption, many IT professionals find themselves lacking the structure and resources they need to deliver reliable applications. Survey participants cited a lack of formal processes as the top reliability challenge for them, and also said that a lack of resources in pre-production, including tools and/or people, was a key reason for errors making it into production.

Too many organizations rely on their customers as an alerting system.
Despite heavy adoption of automation and DevOps tooling, more than half of respondents said they rely on customers to tell them about errors, and over 10% said they are notified about issues by their boss.

A lot of people are wasting more than a day per week just troubleshooting errors.
Though more than half of respondents named productivity as the primary way they measure team effectiveness, more than 25% of respondents, including both development and operations, still spend roughly one full work day per week (or more) troubleshooting errors. Another 42% of respondents spend between half to a full day of their work week troubleshooting.

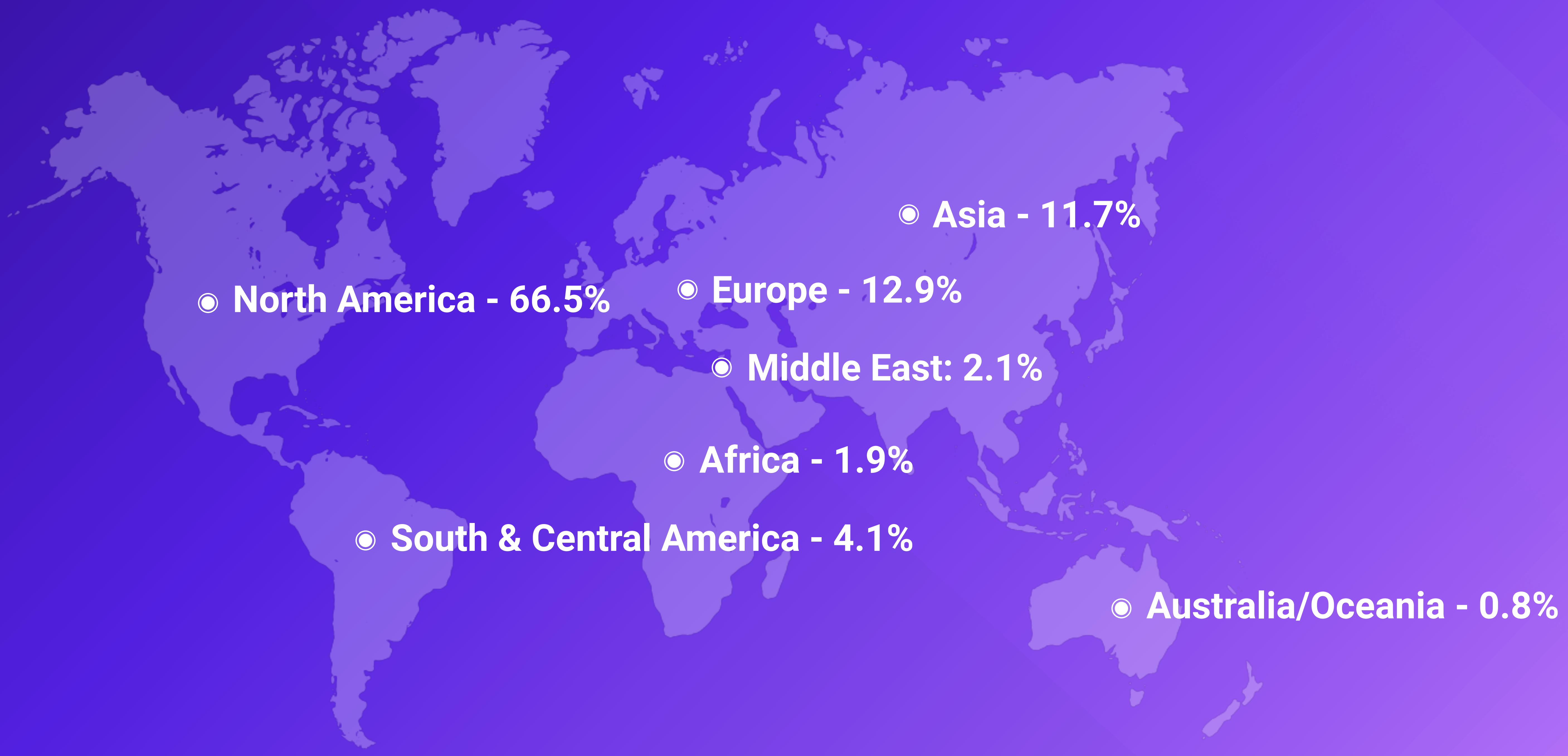
When everyone feels accountable, no one is really accountable.
67% of respondents blame their entire team when an application breaks or has an error, and 73% said that both Dev and Ops are equally accountable for the overall quality of an application. However, when everyone is an owner, it can be difficult to actually hold someone responsible. Having multiple or unclear owners was cited as the second biggest obstacle to ensuring application reliability, and respondents also noted that a lack of clarity around who is actually responsible for the quality of code is a leading cause of errors making it into production.

OverOps

Methodology & Demographics

This report is based on a survey conducted by OverOps of 2,419 IT professionals ranging from developers and QA professionals to DevOps engineers and SREs. We solicited responses through a variety of channels, including our own database, social media, developer and Ops-focused conferences, and third-party websites geared towards engineering professionals. Respondents represented a wide range of company sizes, industries and geographical locations.

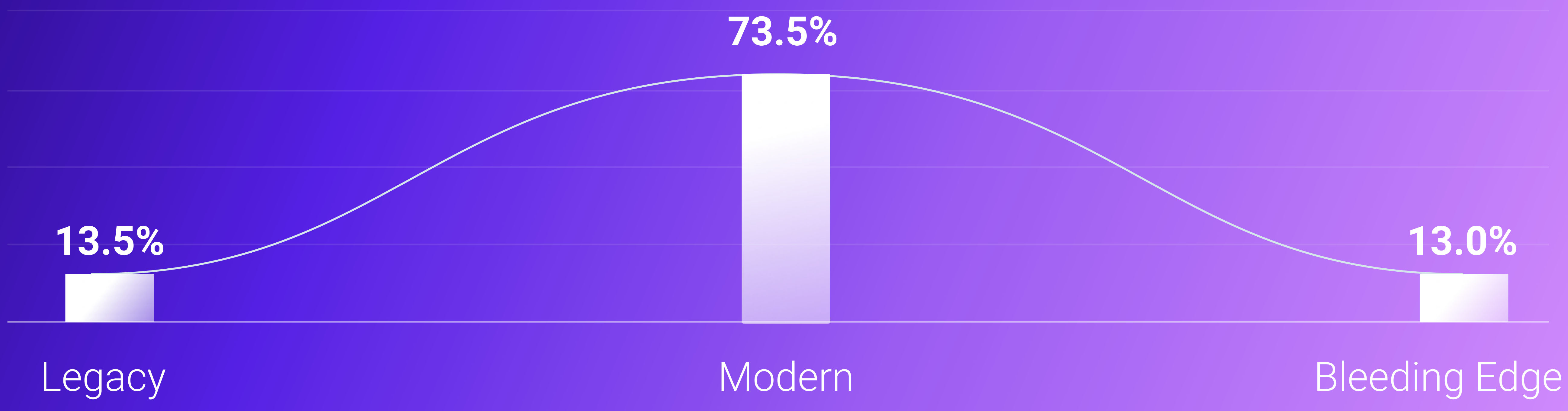
Region



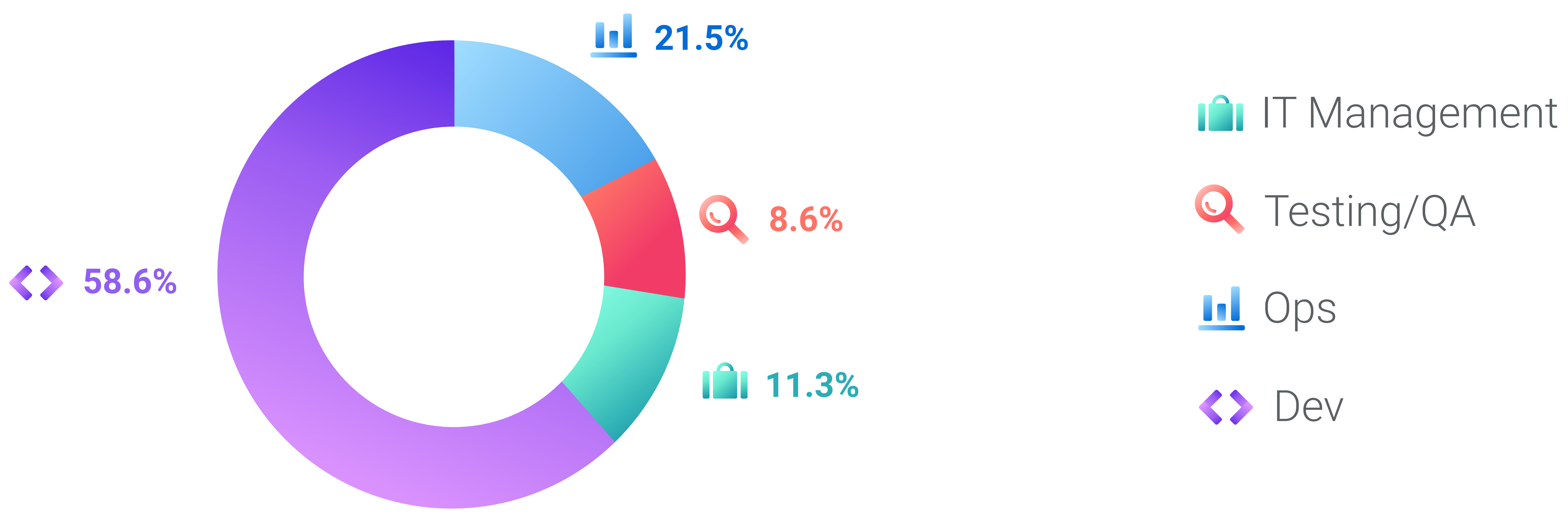
Top Industries

- Finance/Financial Services
- Telecom
- Healthcare/Pharmaceuticals
- Education
- Media & Entertainment
- Energy
- Retail
- Public Sector
- Technology
- Manufacturing

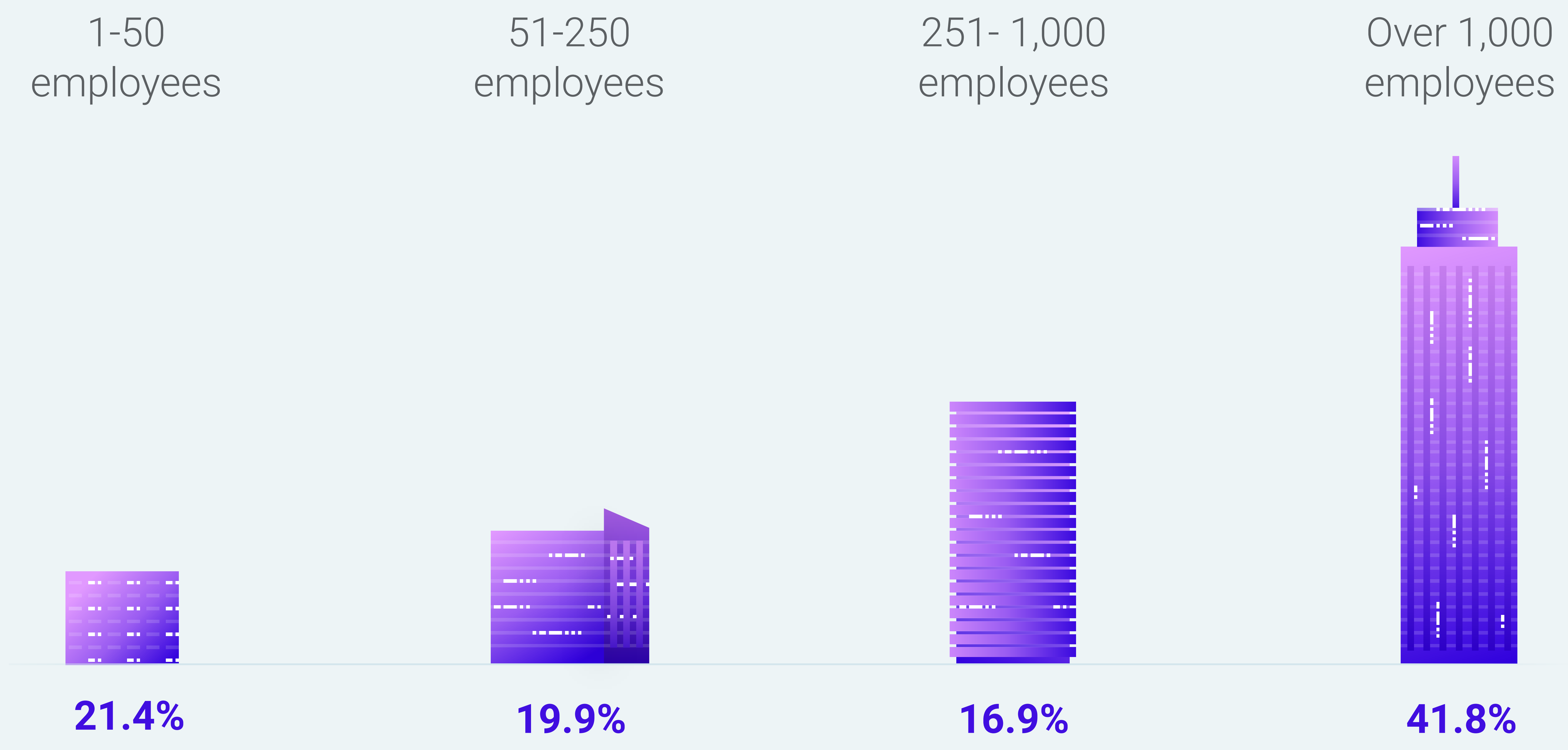
Infrastructure



Role



Company Size



OverOps

The Modern Software Delivery Lifecycle:

What are the Practices, Metrics & Tools in Today's DevOps Ecosystem?

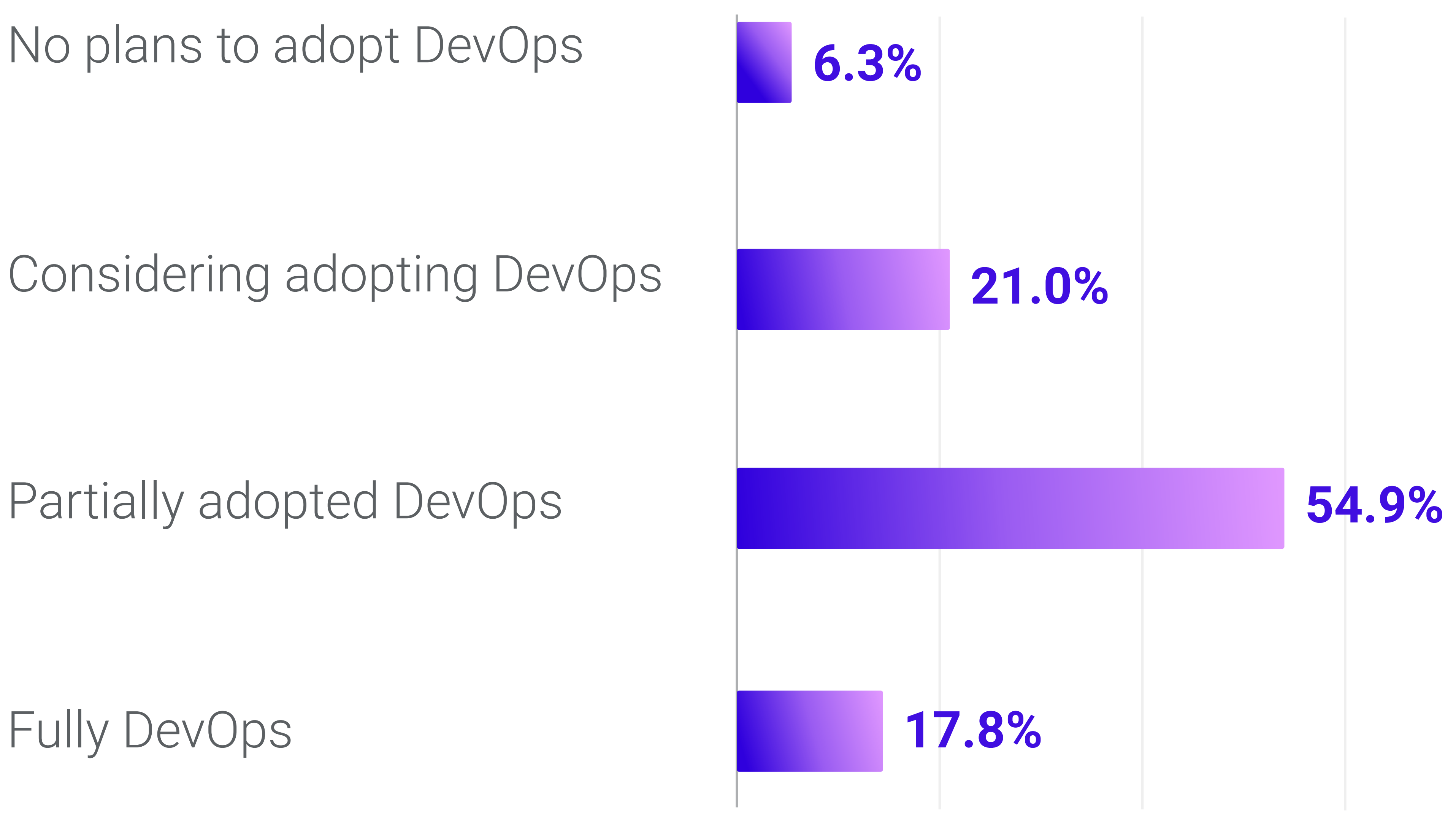
The Modern Software Delivery Lifecycle

Modern development and operations professionals rely on a variety of tools and processes to build and maintain their applications. Survey findings point to DevOps adoption as a foundational part of the toolchains and workflows used by today's developers and IT professionals.

DevOps Transformation is Underway — But Slow Going

Despite the buzz around DevOps for the past several years, very few organizations have fully embraced it. Less than 18% of respondents claimed to be fully DevOps, in contrast to the 82.2% that have only partially, or not yet at all, adopted DevOps practices and tools. Respondents indicated that the vast majority of organizations are headed in that direction, but there is still a long way to go.

Current State of DevOps Adoption



The survey also revealed that the larger the company, the greater the chance that they have at least partially adopted DevOps workflows – 78.5% of respondents from companies with 1,000+ employees versus 48.2% from companies with 50 or less. Conversely, the percentage of respondents that indicated no plans to adopt or that they're still considering adoption decreases as the size of the company grows.

17.8%

of respondents claim to be fully DevOps

82.2%

have only partially, or not yet at all, adopted DevOps practices and tools

“

The larger the company, the greater the chance that they have at least partially adopted DevOps workflows.

We're Moving Faster Than Ever (And Still Breaking Things)

Agile adoption is alive and well across all industries and company sizes. According to the survey, the majority of respondents – regardless of company size, DevOps adoption, industry or infrastructure – are running more frequent release schedules. More than 90% are deploying code at least once a month, and over 60% are deploying code at least once every two weeks. 43.8% of all respondents also noted that they align their code or feature releases with sprints.

While speed is clearly a priority amongst the development and operations communities, it introduces risks. Pressure to move quickly to deploy new features and meet sprint deadlines can significantly impact code quality and reliability. 38.2% of all respondents indicated that moving too quickly is actually a primary reason that errors make it into production. Of respondents that reported having full DevOps adoption, the percentage was slightly higher at 44.6%.

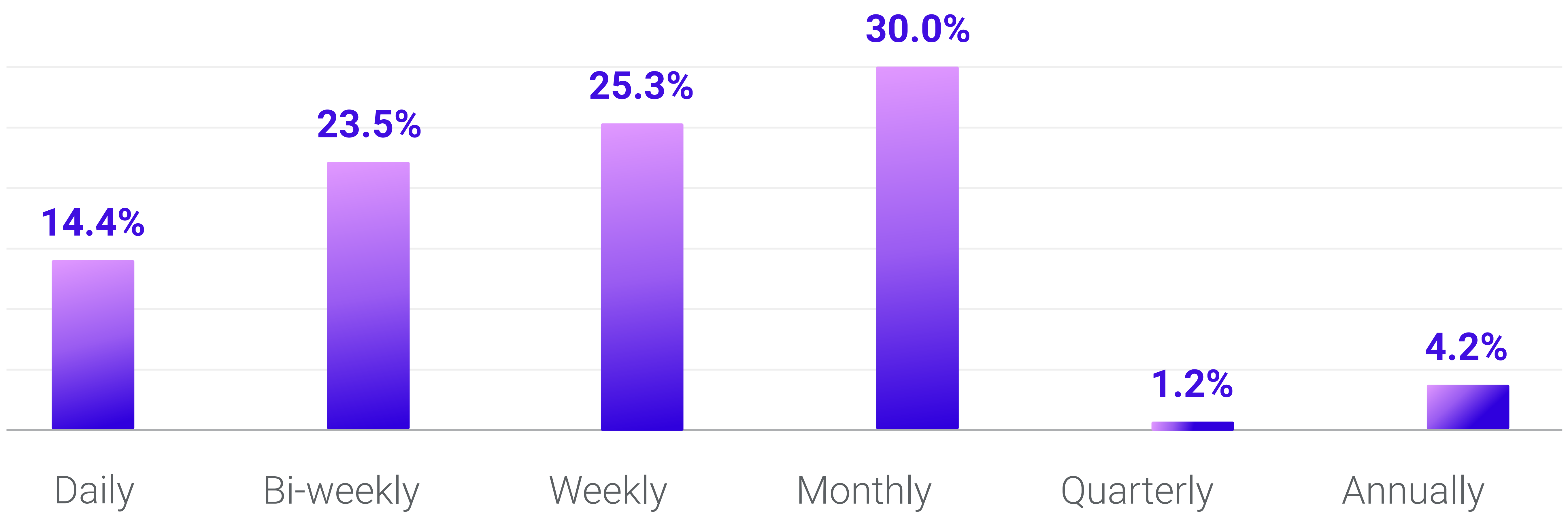
93.2%

of respondents are deploying code at least once a month

44.6%

of respondents that have fully adopted DevOps say moving too quickly results in production errors

Average Release Frequency



1.4% responded with other various time periods

Automation Isn't a Silver Bullet

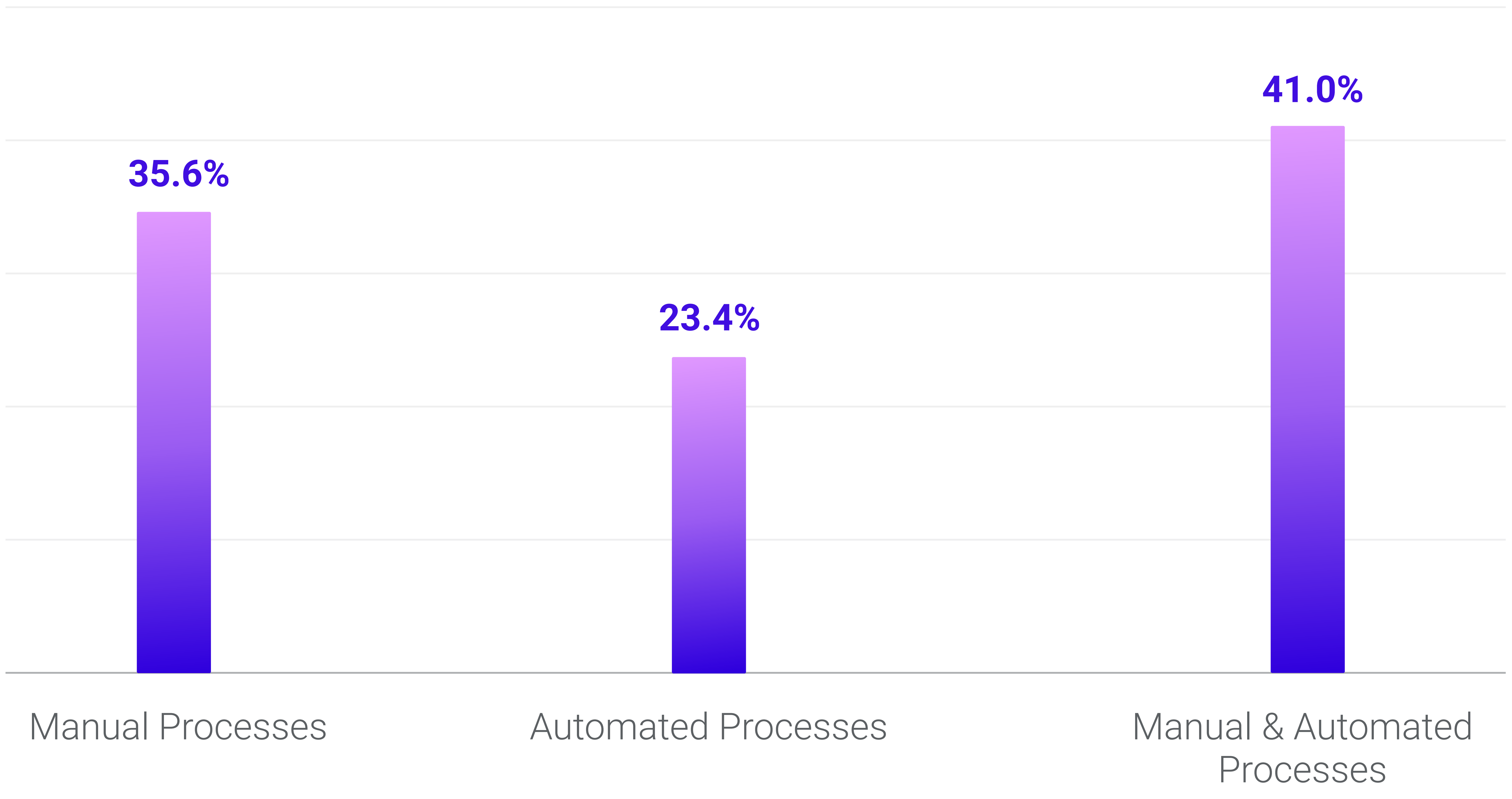
To keep up with accelerated release schedules, organizations are leveraging a wide spectrum of automation tools to support the software development lifecycle. Both development and operations teams rely heavily on automation to detect issues in production, with 62.7% of all respondents reporting they use automated tooling for error notification. Additionally, 63.2% reported that they use automated testing to ensure application quality.

Despite automation adoption, the survey also revealed that an alarming number of people still rely on manual methods. 76.6% of all survey participants said they use at least one manual process to discover errors, and 35.6% said that they use exclusively manual processes. Even worse, more than half (52.2%) specifically said they rely on customers to tell them about errors.

“

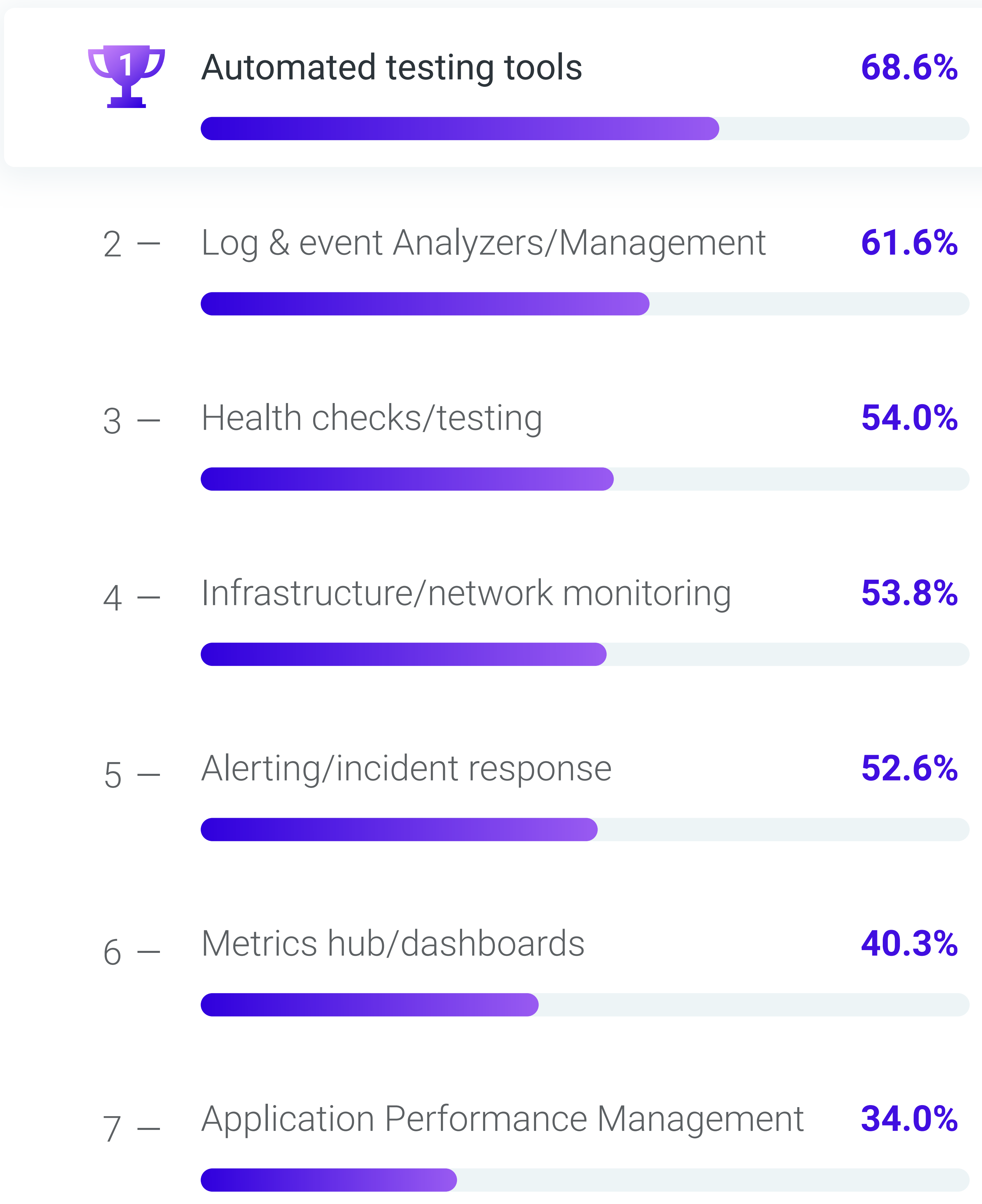
Despite automation adoption, 52.2% of respondents still rely on customers to find out about errors.

How do you discover errors in production?



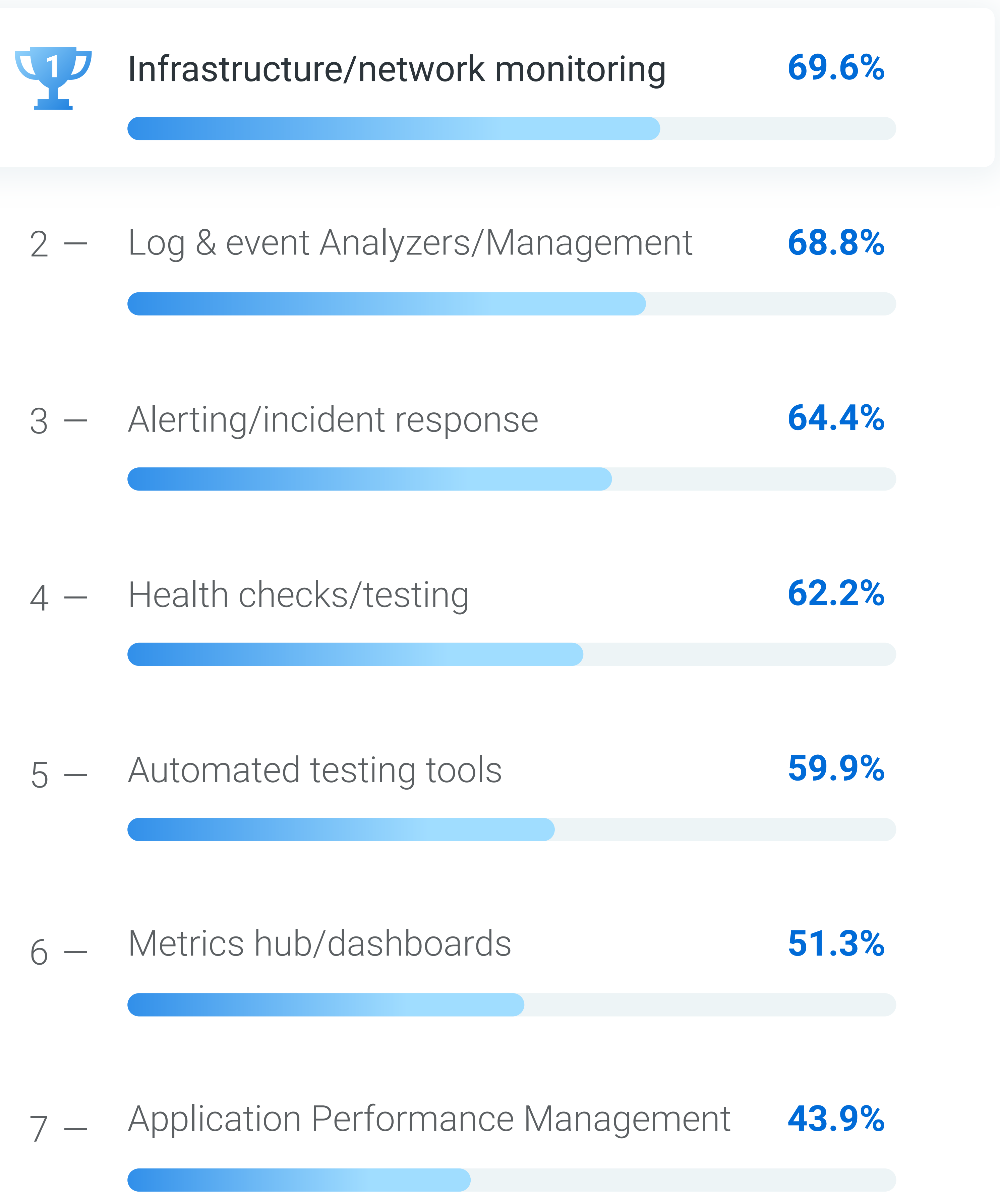
Most Popular Tools

For Developers:



1	Automated testing tools	68.6%
2 —	Log & event Analyzers/Management	61.6%
3 —	Health checks/testing	54.0%
4 —	Infrastructure/network monitoring	53.8%
5 —	Alerting/incident response	52.6%
6 —	Metrics hub/dashboards	40.3%
7 —	Application Performance Management	34.0%

For Operations:



1	Infrastructure/network monitoring	69.6%
2 —	Log & event Analyzers/Management	68.8%
3 —	Alerting/incident response	64.4%
4 —	Health checks/testing	62.2%
5 —	Automated testing tools	59.9%
6 —	Metrics hub/dashboards	51.3%
7 —	Application Performance Management	43.9%

“

Over 60% of developers and close to 70% of operations use log and event management tools.

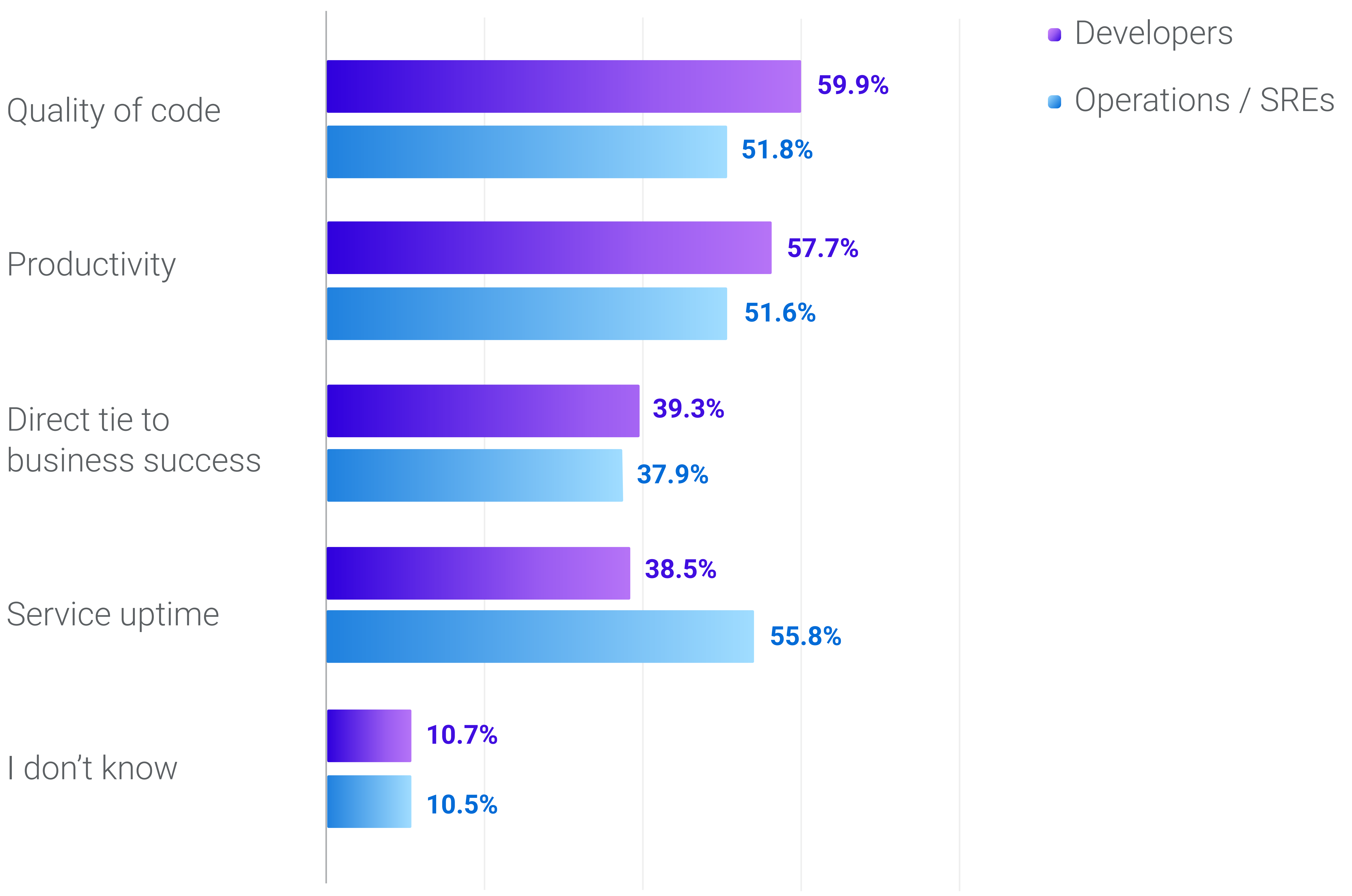
Measuring Success: All Eyes are on Code Quality and Productivity

Everyone seems to agree that code quality and productivity are important. Over half of all respondents indicated that productivity (55.8%) and code quality (56.9%) are the primary way they measure their team's effectiveness.

When it comes to measuring the success of individual teams, service uptime is the number one thing to evaluate, according to 55.8% of Ops respondents. In contrast, developers put even more emphasis on productivity and the quality of their code.

"Productivity and code quality are the top two performance indicators for developers and DevOps.

How do you measure the effectiveness of your team?



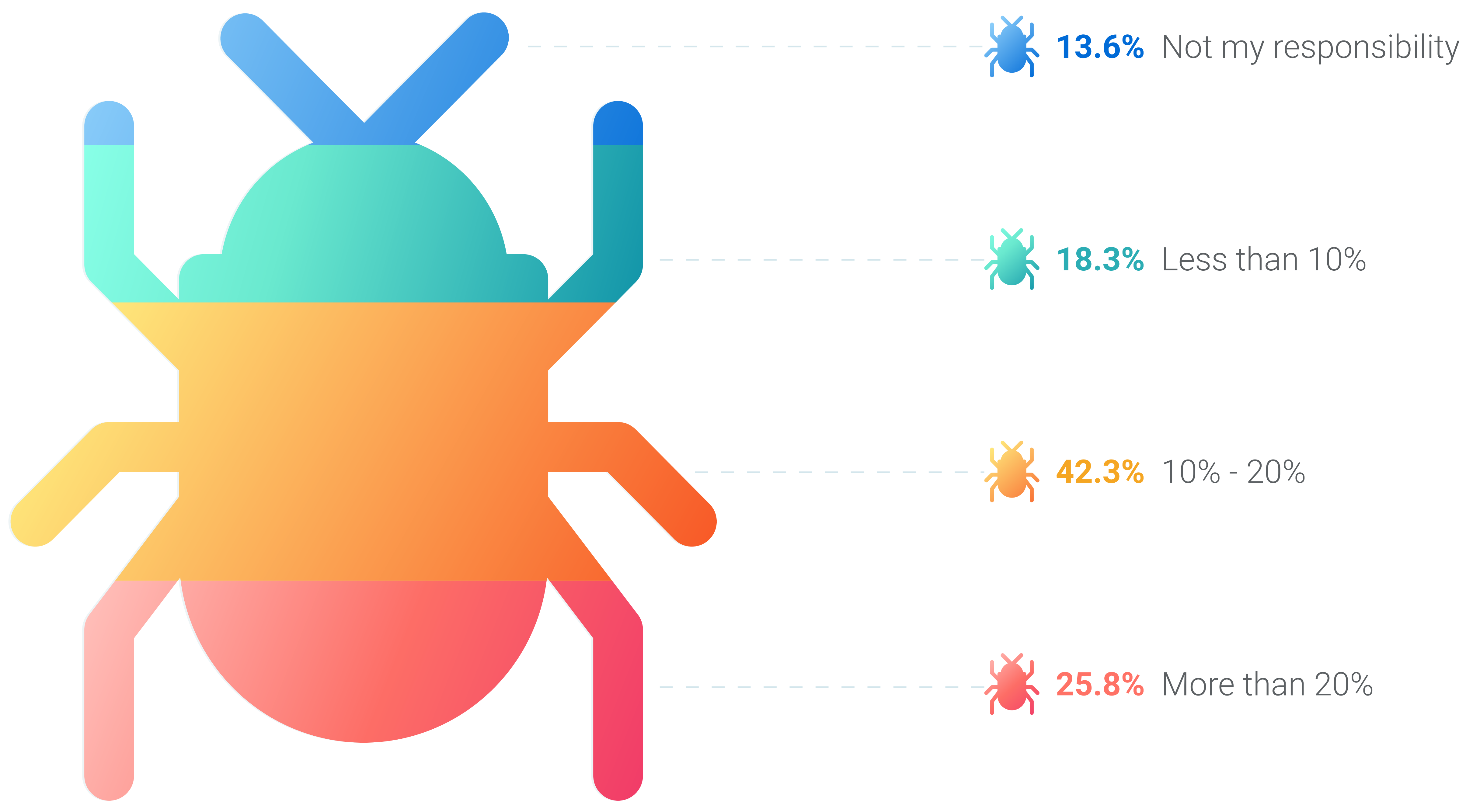
Troubleshooting Takes Time

Despite heavy automation and a strong emphasis on productivity and quality of code, more than a quarter of respondents (25.8%) still spend over 20% of their time troubleshooting code issues – this equates to roughly one full work day per week (or more) spent troubleshooting errors. Another 42% of respondents spend 10-20% of their time troubleshooting (between half and a full day of a work week). This means that precious time that could be spent on developing new features to out-innovate competitors ends up wasted on fixing poor quality code.

25.8%

of respondents spend more than a full work day each week troubleshooting issues

How much of your time do you spend troubleshooting?



OverOps

The Blame Game:

Who is Accountable for Ensuring Overall Application Reliability?

The Blame Game

To better understand how the DevOps transformation affects ownership for application reliability, the survey examined how development and operations teams view accountability. In line with the DevOps mentality, the majority of respondents believe that both teams play a role in keeping software running how it's supposed to, and the further organizations move into their DevOps transformation, the more likely they are to hold the entire team accountable. But with teams moving at breakneck pace to release code, critical tools missing, and processes and roles in flux, this collaborative approach can lead to major confusion.

When Everyone is Accountable, No One is Accountable

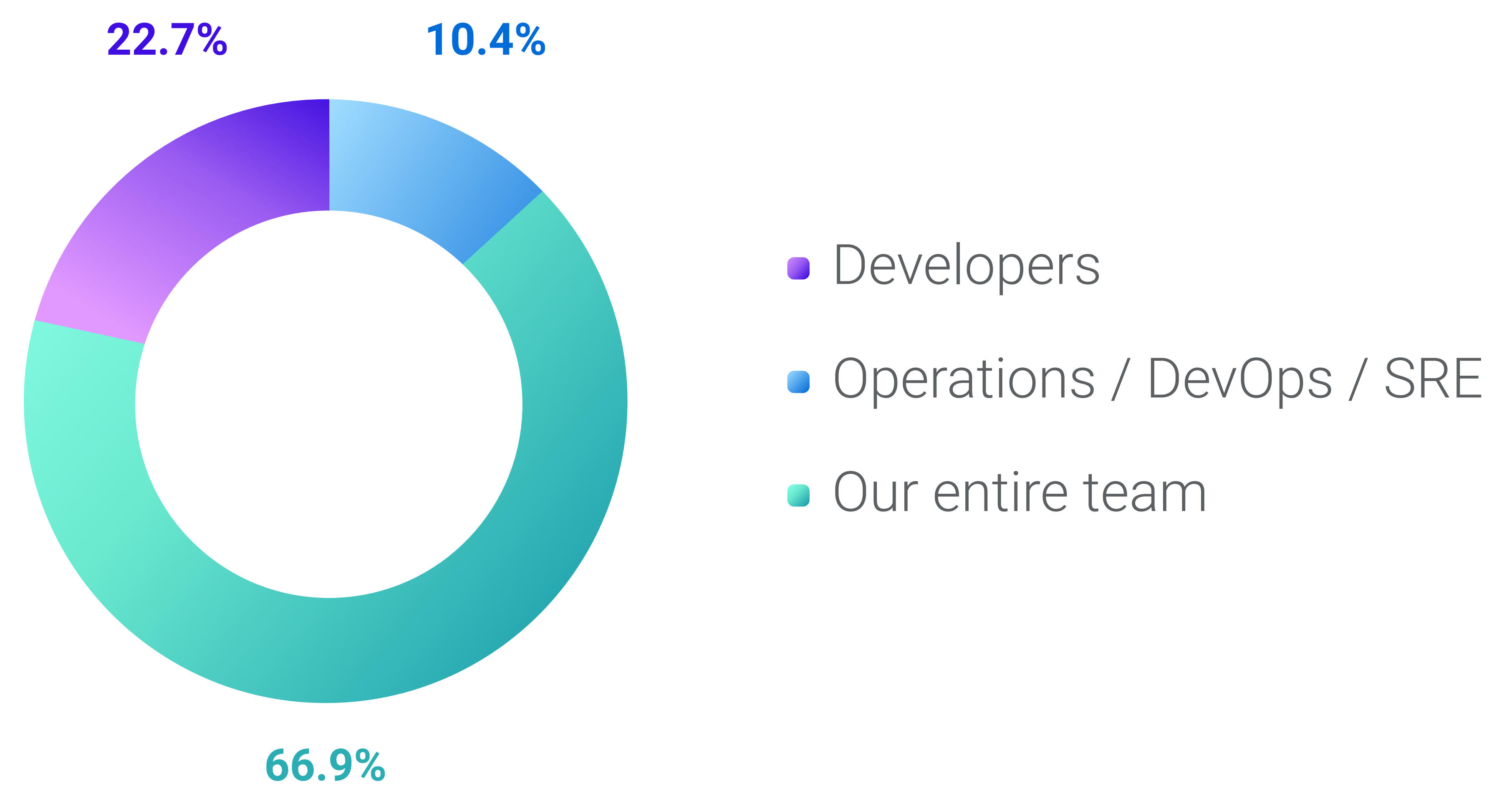
Shared accountability, while a critical part of a successful DevOps approach, can often create confusion and miscommunication. Over two-thirds of respondents (66.9%) believe that their entire team – including both Dev and Ops – is to blame when an application breaks or has an error. Additionally, nearly three quarters of respondents (73%) believe that both Dev and Ops share responsibility for the overall quality of an application.

At the same time, survey participants noted that having multiple or unclear owners and lacking clear processes were the top two challenges they face when it comes to ensuring reliability and quality of applications. On top of that, a quarter of participants (23.2%) feel that a lack of clarity around who is responsible for the quality of code is a leading cause of errors making it into production.

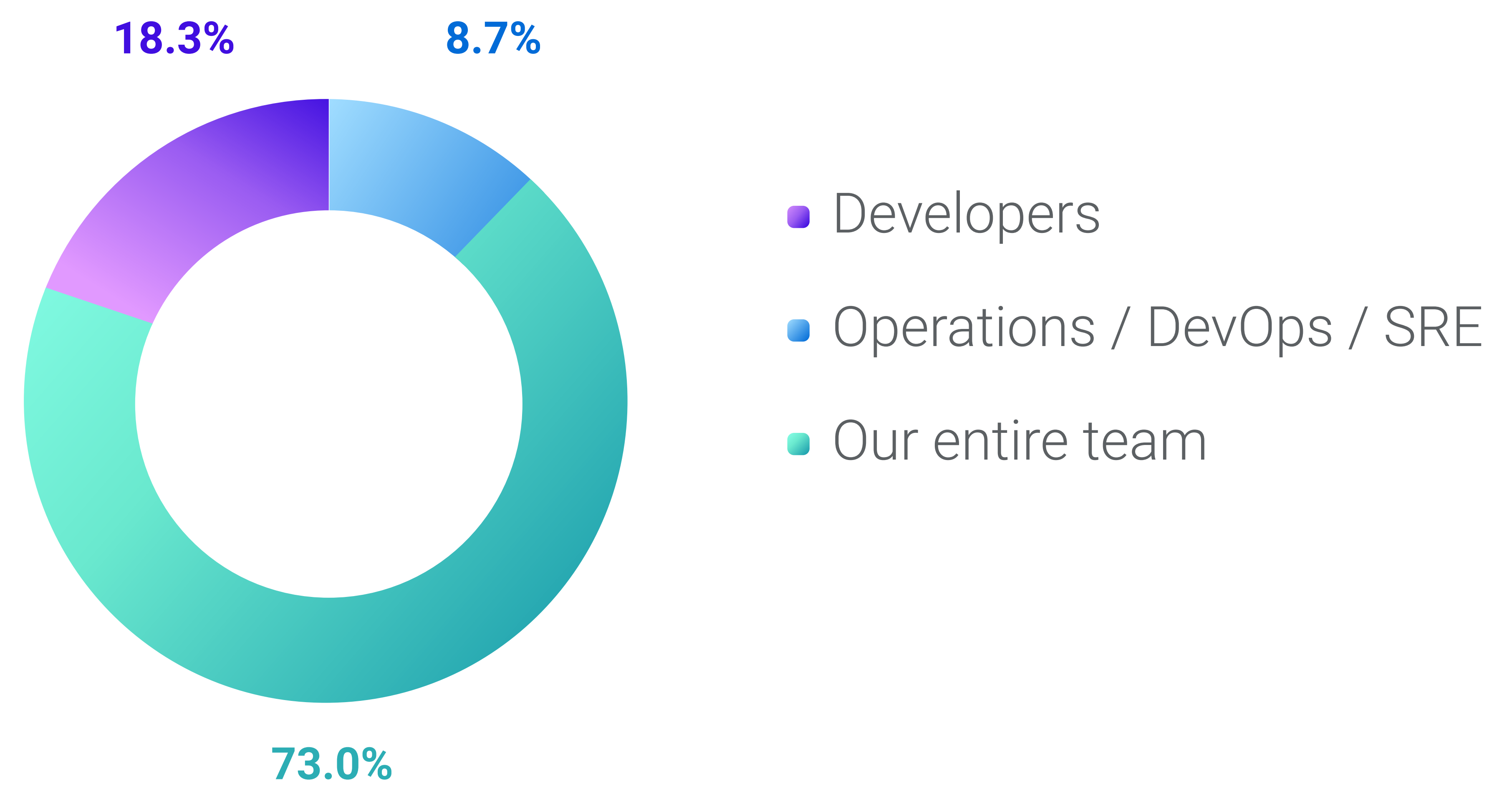
73.0%

believe that both Dev and Ops share responsibility for the overall quality of an application.

When an application breaks or has an error, who do you **blame**?



In your opinion, who is primarily **responsible** for the overall quality of an application or service?



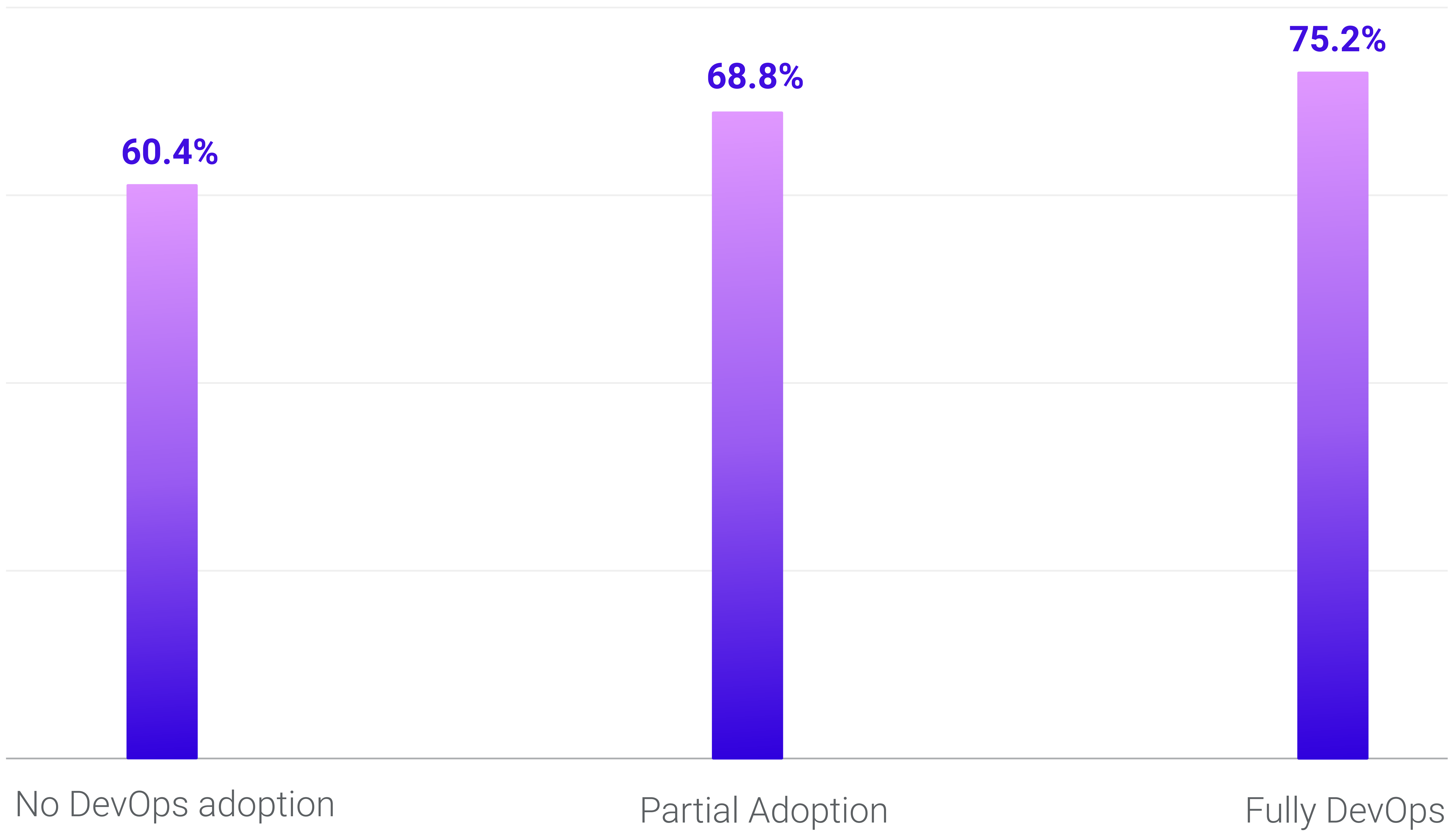
Clear Ownership and Proper Processes are Crucial

Survey participants noted that having multiple or unclear owners and lacking clear processes were the top two challenges they face when it comes to ensuring reliability and quality of applications. A quarter of participants (23.2%) also feel that a lack of clarity around who is responsible for the quality of code is a leading cause of errors making it into production.

"

The increased collaboration between Dev and Ops teams has given rise to challenges revolving around ownership for code in production.

"The entire team is to blame when an application breaks"



% of respondents that agree with the above statement according to their level of DevOps adoption

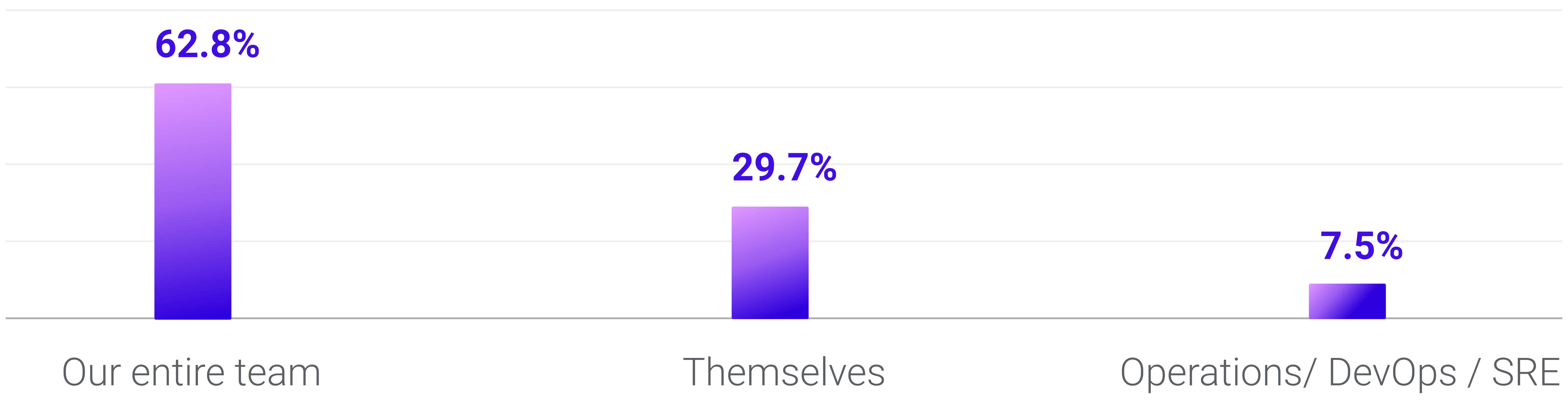
Dev and Ops are No Longer Pointing Fingers

Historically, development and operations have been known to point fingers at each other when something goes wrong – but it looks like this is starting to change. Both development and operations survey respondents indicated they are each more likely to hold themselves responsible for application quality over their Dev or Ops counterparts.

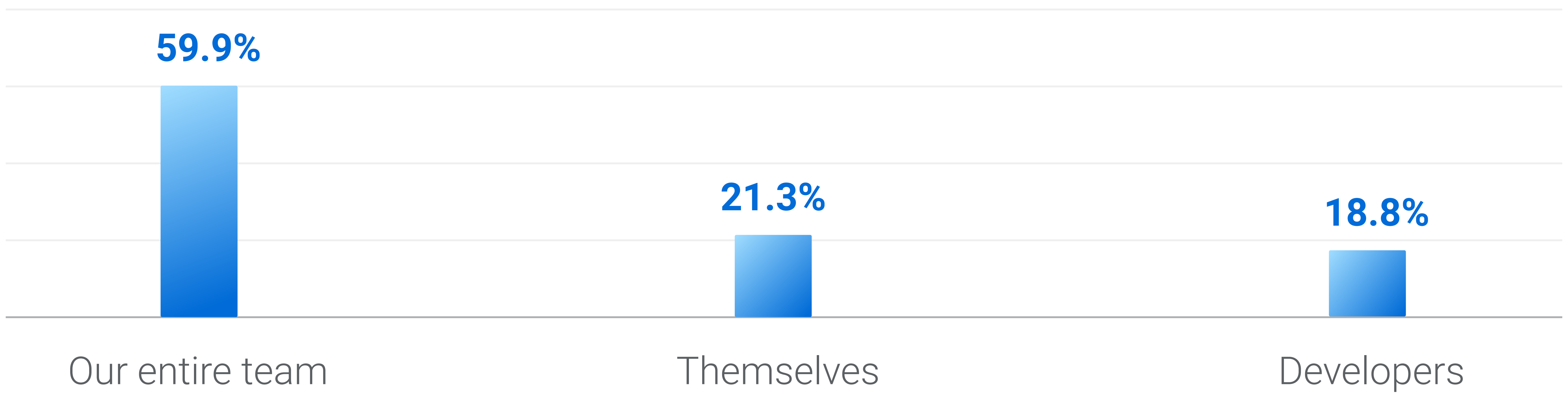
“

Both development and operations are more likely to hold **themselves** responsible for app quality over their Dev or Ops counterparts.

Who do **developers** think is primarily responsible for app quality?



Who do **operations** think is primarily responsible for app quality?



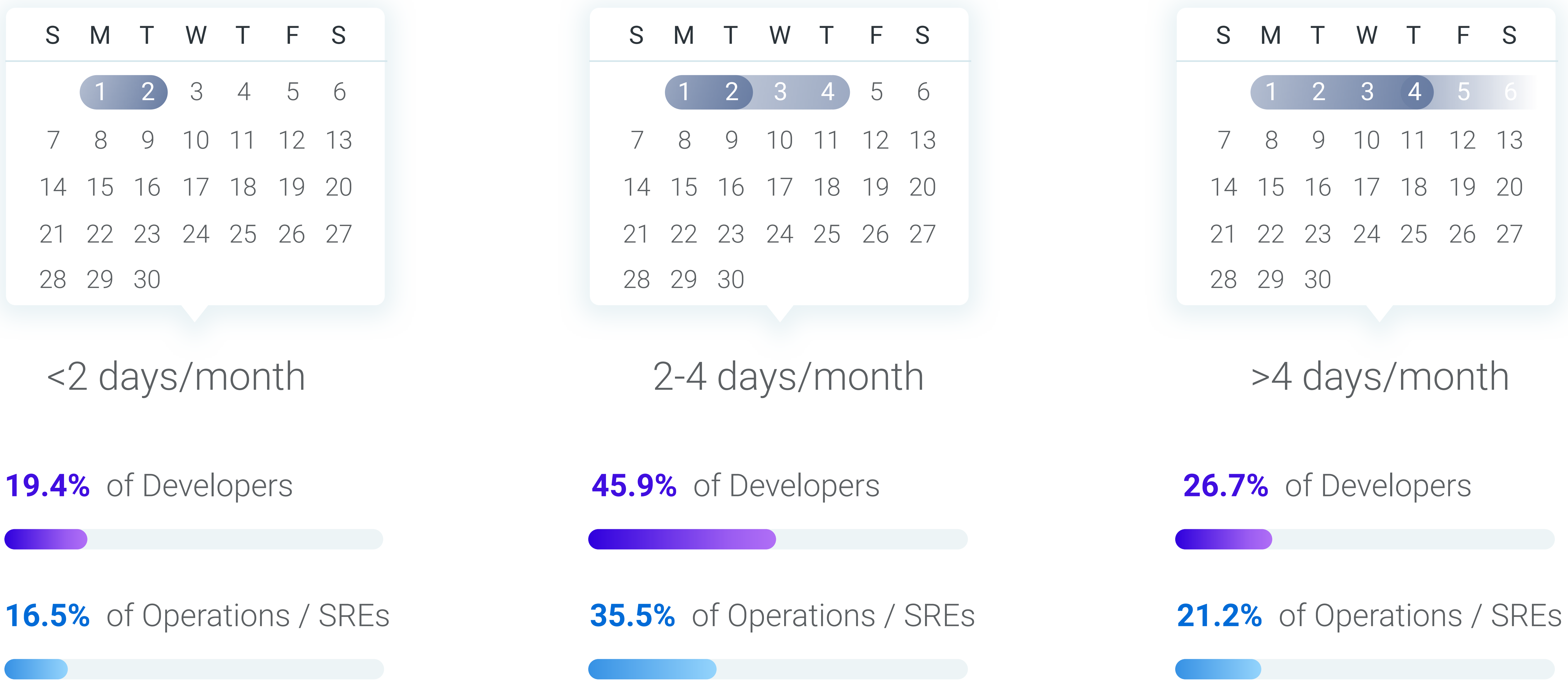
Code It, Ship It, Own It...Fix it

Although both teams claim to share responsibility for errors in production, Devs spend (unsurprisingly) the most time troubleshooting, with 72.6% reporting that at least 10% of their time goes to issue resolution compared to only 56.7% of Ops. Despite reporting a feeling of shared responsibility for errors and code quality, more than a quarter of Ops respondents (26.9%) claimed that troubleshooting isn't their responsibility.

72.6%

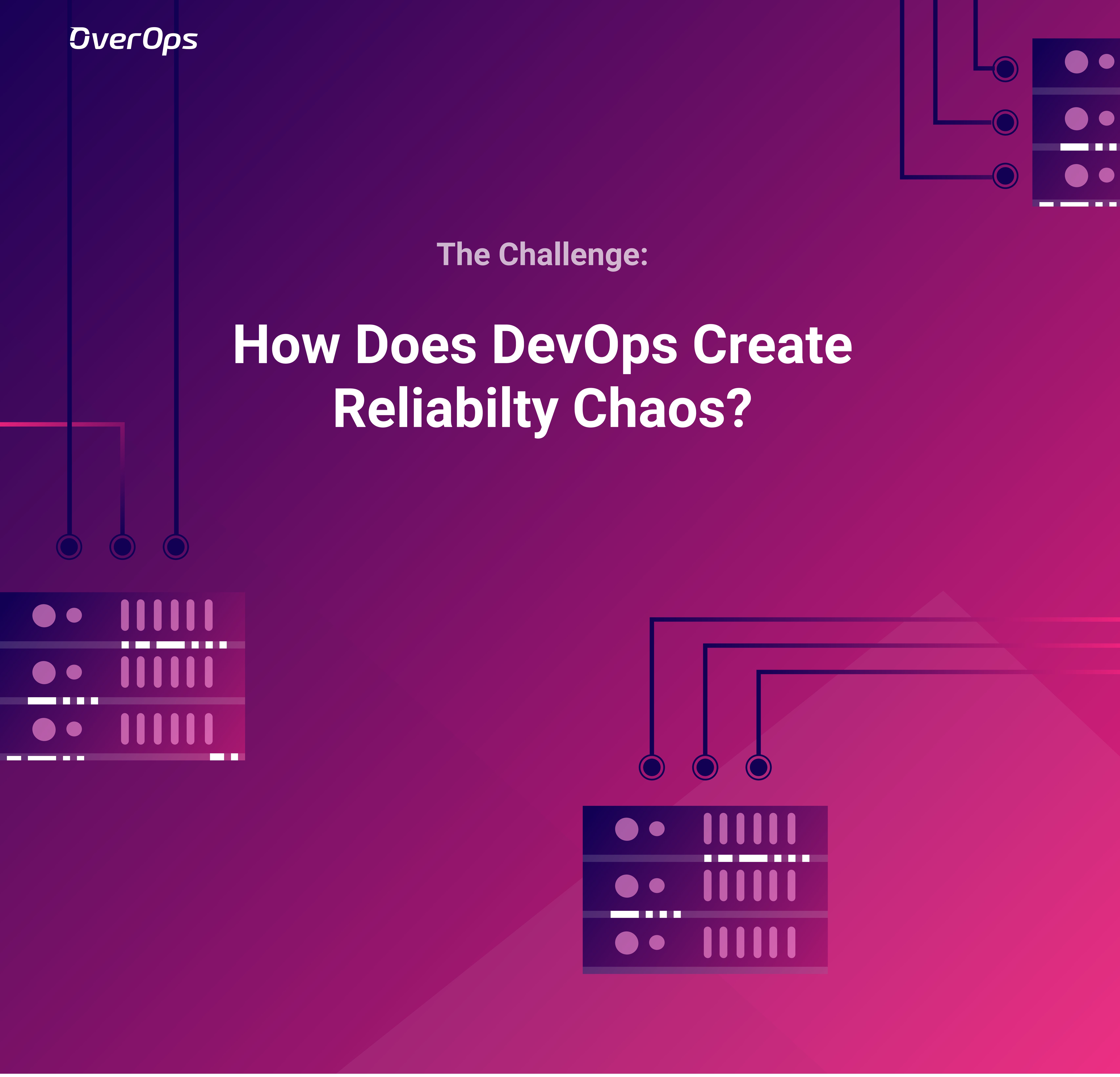
of developers reported spending at least 10% of their time troubleshooting

How much of your time do you spend troubleshooting?



	<2 days/month	2-4 days/month	>4 days/month
of Developers	19.4%	45.9%	26.7%
of Operations / SREs	16.5%	35.5%	21.2%

* 8.0% of Developers and 26.9% of Operations/SREs claimed that troubleshooting is not their responsibility



The Challenge:

How Does DevOps Create Reliabilty Chaos?

The Challenge

The vast majority of survey respondents, regardless of company size, role or industry, believe reliability is a priority. Many organizations are adopting DevOps practices in the hope that it will improve software reliability, but the pressure to move quickly and the added confusion around accountability can actually create new hurdles across the software delivery lifecycle. When error-ridden code is hastily released into production and starts making problems, joint accountability between teams can make it difficult to determine the root of the issue and who is responsible for fixing it.

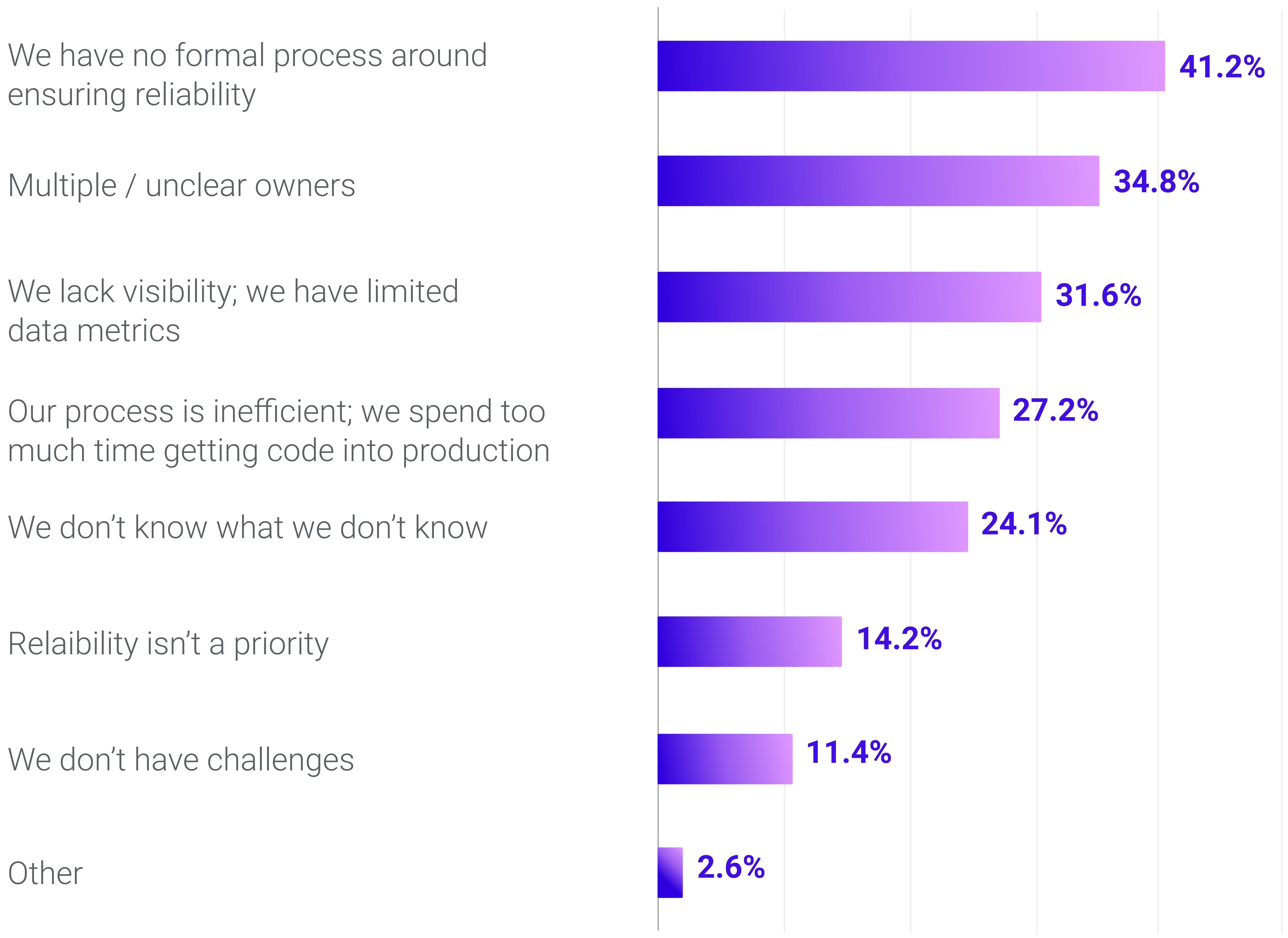
More Owners, More Problems

With the majority of organizations in the early stages or only midway through their DevOps journey, many lack a formal process around reliability, leading to confusion around the role each member of the team plays. This pattern of disorganization was even more prominent in larger enterprises than in small companies. Only 27.9% of organizations with less than 50 employees noted an issue with multiple/unclear owners, as opposed to 36.5% of organizations with over 1,000 employees, pointing to the added challenges that come with scaling.

“

As company size increases, having multiple or unclear ownership becomes more problematic.

What are your main reliability and quality challenges?



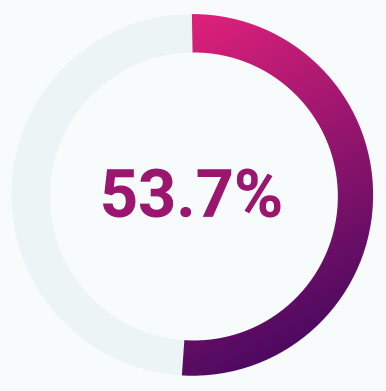
Without Proper Tools & Processes, Chaos Ensues

In alignment with the fast pace and unfamiliar territory that comes with an ongoing DevOps transformation, a common theme amongst reliability challenges was the lack of structure and resources.

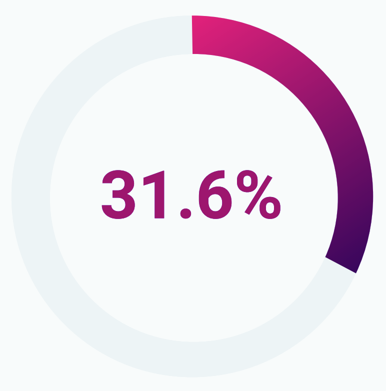
In addition to the 41.2% of all respondents that said a lack of formal process around ensuring reliability was a top challenge for them, 39.9% said that a lack of resources in pre-production, including tools and/or people, was a key reason for errors making it into production.

Without proper tooling and an efficient process in place, teams don't have the visibility they need to understand exactly what's happening in their environments. Without insight into what is going on, teams spend more time on troubleshooting and are still struggling with who to hold accountable for each issue.

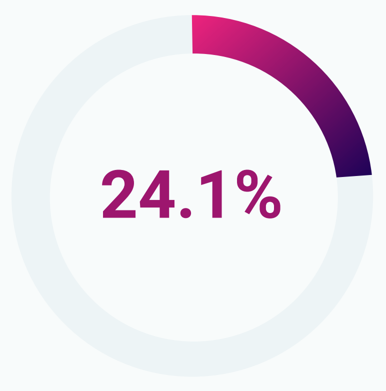
Visibility is a challenge across the board for IT professionals:



don't know how many errors their apps have in a day



reported a lack of visibility and limited data or metrics is a main challenge for them



struggle with reliability because they just "don't know what they don't know"

Conclusion

The transition to DevOps promises increased flexibility, improved operational health, and deeper collaboration across the software delivery lifecycle – but it also comes with a new set of challenges.

At the center of this DevOps adoption chaos is the evolving relationship between development and operations. Many organizations are already taking a shared approach to accountability for application health, however they still lack the tools and application visibility needed to know who is ultimately responsible for addressing and fixing each issue. As the lines between these two teams continue to blur, organizations will need to focus on adopting tools that deepen visibility into their applications. Clarifying ownership of applications and services, and avoiding the “multiple owners = no owner” syndrome is a crucial for even the most bleeding edge organizations.

The “Dev vs. Ops: State of Accountability” survey revealed that as more organizations begin the transition to DevOps workflows, defining roles and processes becomes more difficult and more important. Furthermore, businesses of all sizes are building and releasing new code and application features faster than ever before, which adds additional pressure across the entire software delivery supply chain. Organizations going through the DevOps transformation are more likely to face visibility challenges that make it difficult to maintain or improve application quality and reliability.

About OverOps

OverOps is a software reliability platform that applies machine learning to application code as it's running to automatically identify anomalies and provide code-level insights in QA, Staging and Production. By identifying when, where, and why the code breaks in real time, OverOps helps companies fix application issues before the customer is affected, as well as prevent them from being released.

When anomalous failures – such as newly introduced issues, error regressions and slowdowns – occur in any environment, a complete picture of the code is provided, including:

- Execution stack, source code and complete variable state
- Previous 250 log statements (including DEBUG- and INFO-level, even in production)
- Frequency and failure rate for ALL known and unknown errors and exceptions
- Classification of new versus reintroduced errors
- Which release or build is associated with each specific event
- Event analytics

This data allows organizations to create a culture of accountability, where every team has visibility into what went wrong, eliminating any finger pointing. Not only does access to this information greatly reduce resolution time for issues in all environments, it provides deep insight into the overall quality of new deployments and of the application as a whole. As more organizations aim to innovate faster and deliver a seamless digital experience for their customers, OverOps helps avoid costly downtime with minimal performance impact and no need to modify code.

The data collected by OverOps can easily be displayed on any other platform using OverOps native integration with tools like Splunk, AppD and Grafana for monitoring, alerting and visualization purposes.