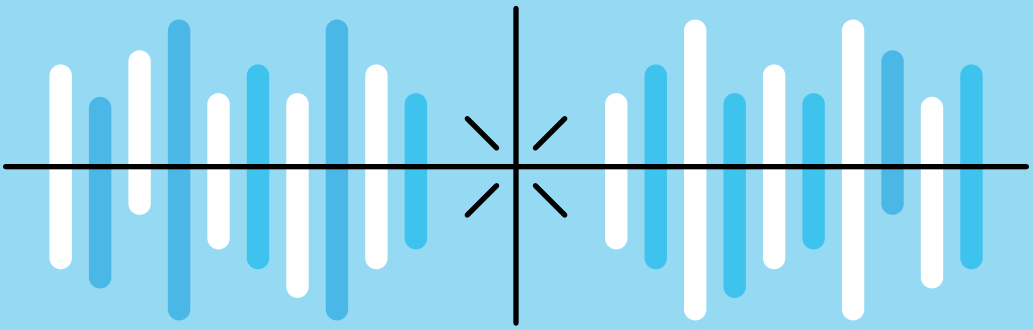
HARVARD BUSINESS REVIEW



Managing Risk in Modern Software Delivery

HBR Analytic Services **PULSE SURVEY**

Sponsored by

LaunchDarkly

In the Age of AI, Software Delivery Needs a Rethink

Software now defines how every business operates, from how products are built to how customers are served. As teams ship their products faster than ever, the risk of disruption due to bad releases grows. A single release can trigger downtime, break key workflows, or expose customer data. The challenge isn't just moving fast; it's staying in control while doing it.

We sponsored this research project because the industry is facing a fundamental shift in how software must be delivered. The old way—manual rollbacks, reactive monitoring, and large-scale "big bang" launches—no longer works. These approaches can't keep up with today's highly distributed architectures, the constant integration cycles, and the growing unpredictability introduced by artificial intelligence (AI), which is now both generating code and running inside production environments.

How you release software is no longer a technical detail; it is a strategic capability that directly impacts revenue, customer trust, and competitive edge. Success now depends on a modern approach: releases that are intentional, observable, and built for real-time adaptability.

With the help of LaunchDarkly, teams are already decoupling deployment from release, experimenting with AI safely, and managing risk with precision. This shift isn't theoretical; it's a new operating model helping software teams ship faster, learn faster, and stay in control. That's not just a technical advantage—it's a business imperative.

Managing Risk in Modern Software Delivery

In today's highly competitive business environment, companies are ideating, innovating, and releasing software at a faster cadence than ever before. But the heightened frequency of software releases produces increased risks, and problems with software releases across organizations seem to be more prevalent than one may expect, as made evident by the July 2024 CrowdStrike outage and Sonos' faulty app update two months earlier.

IN APRIL 2025, Harvard Business Review Analytic Services conducted a survey of 236 members of the *Harvard Business Review* audience who are familiar with their organization's management of its software releases. Just over half of the respondents (56%) report that their organization experiences errors or issues with its software releases on at least a monthly basis. Given the ubiquity of software in essentially everything relevant to businesses in the digital age, effectively managing the risks associated with software releases has become critical.

"Most organizations realize now the importance of strong risk management due to the increasing complexity of software as well as the impact of poor software releases," says Hassan Ennaciri, senior director at Gartner Inc., a research and advisory firm based in Stamford, Conn. "The business relies on software, so a strong risk management strategy for releases is very important."

When software releases go wrong, companies face both external and internal impacts. Externally, poor software releases can lead to dissatisfied customers, a diminished brand reputation, and revenue loss. And internally, team morale, productivity, and efficiency suffer, while costs rise when developers spend hours firefighting and troubleshooting an issue, taking valuable time away from building the next innovation. The bottom line: An airtight approach to managing software-related risk is essential for businesses to compete and succeed.

HIGHLIGHTS

71%

of respondents agree their organization should improve its approach to software release risk management.

54%

agree that dealing with software release issues is a significant pain point for their organization's developer teams.

6%

say their organization can detect software release errors or issues in real time.

Due to rounding, some figures in this report may not add up to 100%.

> **“Poor software releases kill productivity because developers and support engineers are just fixing issues; they don’t have time to create new features. It’s really just firefighting.”**
>
> Hassan Ennaciri, senior director, Gartner Inc.

Yet many approaches are less than airtight—many organizations still rely on legacy processes and technologies, which often include homegrown tools, that are inadequate to meet the demands of current software development practices. “There’s a wide range of maturity with respect to how prepared organizations are to manage risks in their software releases. Although some companies have deeply integrated risk management frameworks in place, many organizations are still early stage and have a lot of ad hoc approaches,” says Ennaciri. “They rely heavily on manual processes, and their plans for rolling back faulty releases are significantly less mature. The technologies they use, budgets, legacy systems, and skills all play a role in holding these organizations back in their risk management.”

This report will first examine the detriment faulty software releases pose to organizations as well as the central role deployment style plays in the performance of software releases. The report will then explore current strategies for managing risk in software releases, where these approaches may be falling short, and opportunities for organizations to strengthen their risk management practices.

The Pain of Poor Software Releases

An effective software risk management strategy may first require companies to take a more proactive approach. “One thing I see in common among organizations is they’re very often reactive in their risk management,” says Ennaciri. “They only act when things break, and risk management is more of an afterthought. They don’t really take risk management seriously until things happen and impact their business.”

Releasing software is a notoriously stressful experience even when it goes well. And when software releases go awry, organizations suffer significant consequences. Internally, poor or failed software releases take a toll on employees. A little over half of survey respondents (54%) agree that “dealing with software release issues is a significant pain point for my organization’s developer teams.”

The resulting detriment to team morale can be significant. “Engineers like to release their code—I think there is a point of pride of seeing your stuff out in production,” says Josh Nykamp, senior director of engineering at Boston-based gambling company DraftKings Inc. “So, if you’re not getting it out there or you’re constantly rolling it back, employee morale goes down.”

“I think what all engineers hate the most is being on call—if a release goes out at night and you get woken up by your phone because an issue came up, that’s a miserable experience,” adds Nykamp. “Every engineer has probably worked a job where they got woken up at night a lot at some point in their career.”

When issues with releases happen, engineers spend considerable time and effort to resolve the problem, which often involves a rollback that reverts an application to a previous stable version. For most organizations, rollbacks are a time-consuming manual process, according to Shahryar Shaghaghi, a professor of professional practice and the program director of the Master of Science in technology management program at Columbia University’s School of Professional Studies. “Rollbacks are not automated in most organizations, so teams need to go through a pretty tedious process to bring an application back to the original [or] previous state,” he says.

The bottom line is problematic software releases can tank employee productivity and innovation. Frustration with constantly troubleshooting failed releases can also negatively impact employee retention in the long run, requiring the hiring and training of new developers. “Poor software releases kill productivity because developers and support engineers are just fixing issues; they don’t have time to create new features. It’s really just firefighting,” says Gartner’s Ennaciri. “There can be a lot of burnout and a real frustration with the constant distraction of having to fix all these problems. And of course, these issues make it hard to keep good people—retention becomes a big problem.” The survey results reflect this sentiment: The business area most cited as moderately or greatly disrupted because of software release errors or issues

is team productivity and efficiency, selected by 61% of respondents (excluding those whose organization never or very rarely experiences software release issues). FIGURE 1

Problematic software releases also have consequential external impacts. In customer-facing contexts, poor or failed software releases diminish customer satisfaction and trust, which translates to lost revenue. The stakes are higher considering the centrality of digital products to many businesses today. In a growing number of situations, a customer's only touchpoint with a company may be through an app, constituting their entire brand experience, so it's increasingly imperative for businesses to deliver seamless interactions on that front. "The financial impact of poor software releases can be huge, especially these days where many companies heavily rely on revenue from digital products," says Ennaciri. "When your product is down, you're not making revenue and you're losing money."

Consumers have no shortage of choice in today's saturated market, so companies that release buggy software risk seeing their customers take their business to any of a myriad of competitors. "Downtime from poor or failed software releases directly translates into costs—customers would be turned off, and they may just say, 'I'll go to another provider that is more reliable,'" says Columbia University's Shaghaghi. "If you're looking at customer-facing applications, you're dealing with reputational damage and a real loss of trust and loyalty."

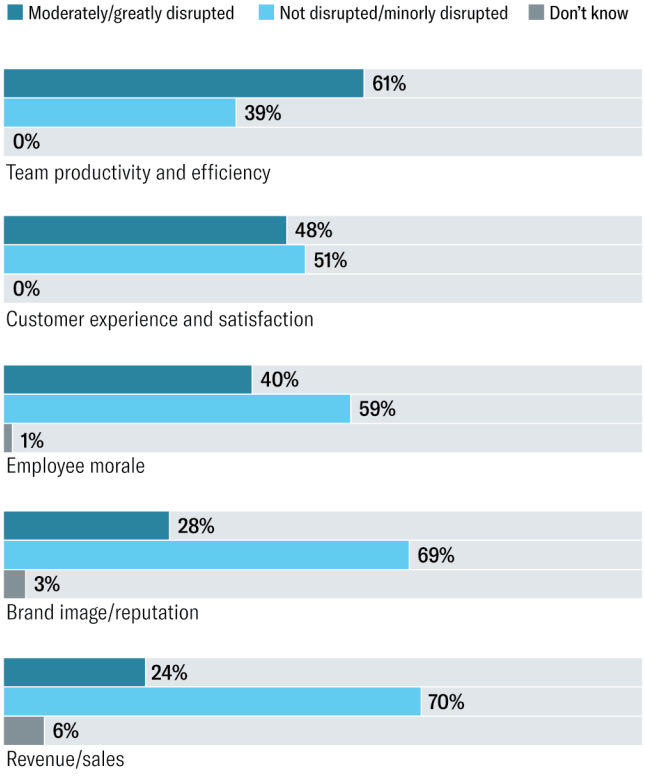
The extent of that risk is apparent in the survey results: Customer experience and satisfaction is the second most-cited area that respondents say has been moderately or greatly disrupted at their organization due to software release errors or issues, selected by 48% (again excluding those whose organization never or very rarely experiences software release issues).

FIGURE 1

Negative Impacts of Faulty Releases

Both employees and customers feel the impact of poor releases

To what extent have the following areas of your organization been disrupted because of software release errors or issues?



Base: 217 respondents, excluding those who report their organization "never/very rarely" experiences software release errors or issues.

Source: Harvard Business Review Analytic Services survey, April 2025

More Essential Risk Management

In today's age of artificial intelligence (AI), having solid risk management for software releases will likely become even more essential. In the survey, half of respondents agree with the statement "Increased use of AI has created more potential risk in software releases." As AI becomes embedded in more production applications—across chat experiences, personalized recommendations, and decision-making logic—the risk surface area for software releases is expanding dramatically, and the frequency of releases is increasing, as well. The advent of AI introduces a new class of release risk: AI-powered features can behave unpredictably, have inconsistent outputs, and suffer from model drift. And given that these features are now more commonly embedded in applications, the potential impact of faulty releases on customers is greater and the stakes higher.

> **“Downtime from poor or failed software releases directly translates into costs—customers would be turned off, and they may just say, ‘I’ll go to another provider that is more reliable.’”**
>
> Shahryar Shaghaghi, professor at Columbia University’s School of Professional Studies

In another vein, Ennaciri highlights the potential risk attached to leveraging AI to generate code, particularly with respect to hallucinations, though he notes these risks can be addressed through robust testing. “Whether it’s a developer or AI writing the code, as long as organizations have ways to test everything and mitigate risk automatically, the potential impact to users can be significantly minimized,” he says. Ennaciri also underscores the importance of creating governance around how AI is used in the software development process and training engineers on how to use it properly.

The repercussions of poor software releases underscore the need for a solid strategy for minimizing their likelihood of occurring. Unsurprisingly perhaps, many organizations seem to recognize that need: 54% of survey respondents say software release risk management is an extremely high or high priority for their organization, and 29% of respondents say it is a moderate organizational priority. Just 14% say it’s a low priority or not at all a priority.

Despite that degree of prioritization, organizations’ efficacy at managing software release risk may leave something to be desired. Thirty-nine percent of respondents rate their organization’s approach to software release risk management as extremely or very effective, and 71% of respondents agree with the statement, “My organization should improve its approach to software release risk management.”

Pitfalls of Big Bang Deployment

Proactive risk management for software releases starts with how the software is deployed. A wide range of deployment types exists, but a general distinction can be made between “big bang” deployments, where new software is released to all users at once, and phased deployments, where new software is slowly released to users in phases. Big bang deployments were more popular in the past, but over time, phased deployments have become more commonly used, as reflected in the survey results. Among those surveyed from the *Harvard Business Review* audience, 48% say their organization primarily does phased deployments, 32% do an equal mix of big bang and phased deployments, and 17% primarily do big bang deployments.

According to Shaghaghi, the growing popularity of a phased approach to software deployment stems from the greater agility phased deployments offer organizations to deliver on customer demands. “Over the years, to be more agile and adapt to customer feedback, most companies started to shift toward the agile methodology and phased rollouts over the big bang approach for their software releases,” he says. “Especially now with the cloud and [software-as-a-service] environments, you can’t afford to wait for a big bang—you’re constantly updating and enhancing the feature functionality of the software.”

And given the pace and dynamism of today’s business landscape, constant innovation is imperative. “Back when the big bang approach was more commonly used, companies had months or even years to work on a large-scale application and roll it out. Customers were more receptive to that, but things are different now since today’s customers are used to the speed of cloud computing,” Shaghaghi says. “Time to market is key. If companies don’t move fast enough with improving feature functionality, their competitors will swoop in and gain a greater share of the market.”

Within engineering teams, the big bang approach has numerous drawbacks. The pain of big bang releases was directly felt by the team at Jackpocket, a digital lottery app that was acquired by DraftKings in 2024. Jackpocket previously conducted on average two big bang releases every month, according to DraftKings’ Nykamp, Jackpocket’s vice president of engineering prior to its acquisition. These releases would contain a high volume of code, the result of weeks of work, and if the code contained any bugs, as was often the case, implementing hotfixes and rolling back the release proved to be painful and time-consuming.

“Larger releases are hard to deploy; it’s difficult to merge all that code together, and it’s also harder to test and do quality assurance,” Nykamp says. “There’s a lot of problems that come with taking a big amount of code and shoving it into

production. And when you're releasing a larger amount of code, mistakes tend to be more serious. It's a lot easier to fix one day's worth of work versus several weeks'."

The engineering team at Hireology Inc., a Chicago-based HR and recruitment platform, had similarly negative experiences while conducting big bang deployments, according to Scott Gainous, Hireology's vice president of engineering. "There was a lot of effort that went into building a release with many parallel pieces of work that were happening in isolation. It created issues where, when we tried to merge it all together, the code wouldn't compile; it wouldn't run effectively," Gainous says. "A lot of the team's time was spent just trying to sort that out, which creates a lot of frustration when that leads into late nights or weekends and just not accomplishing the things you set out for the day."

Besides producing internal problems, big bang releases can impair organizations' ability to gauge customer preferences and needs. "You're not getting to test how your customer feels with big bang releases. You're making a lot of assumptions, putting six weeks of work into one feature and rolling it out. What if customers don't even like it? Then those weeks of work go to waste," Nykamp says.

Without a clear sense of customer preferences, companies can end up releasing software with little or no business purpose. "A negative side of larger releases is you end up spending more time doing things that aren't really focused on the business outcomes of shipping new product features; you're focused more on just how to move stuff through the machine. When the release goes awry and has to be rolled back, and you wait two weeks to try again—it just slows down the process, and customers aren't getting what they're needing," says Hireology's Gainous.

Benefits of Phased Delivery

Adopting a phased, gradual approach to releasing software allows organizations to avoid the pitfalls of big bang releases and lays the groundwork for a solid risk management strategy. With phased delivery, only a portion of the customer base is exposed when incidents do occur, minimizing impact while developers work to fix the problem.

In fact, phased delivery is becoming the industry standard. "In the big picture, phased rollouts are becoming the preferred approach, especially when organizations are making significant updates to complex systems and large applications," says Ennaciri. "I think a lot of companies have learned big bang deployments where everything goes live are very risky, given how critical their digital products are to the business." He clarifies, however, that although the growing popularity of phased rollouts is the general trend, big bang deployments still have their place in certain contexts, often for smaller applications.

Gainous also emphasizes the benefits of gradually implementing changes through phased software delivery with respect to mitigating risk and minimizing the likelihood of errors. "Philosophically, I'm a big believer that smaller changes are less risky, and they have better focus from an engineering peer review standpoint," he says. "I'm a firm believer in not creating these big ceremonious releases that tend to get filled with bugs."

Nykamp cites numerous benefits from implementing progressive rollouts of new product features after moving Jackpocket away from big bang releases, particularly with respect to keeping costs to a minimum in the case of a failed release. "Progressive rollouts are useful, especially when you have big audiences. If you want to test out a feature and make sure it works well, you can roll it out to 5% of your audience, and if something goes wrong, you can just turn it off," he says. "That's really helpful when you're trying new things—you can test your assumptions before you sink a bunch of time or money into it."

To take fuller advantage of phased delivery, organizations can target releases to specific types of customers. Building on the benefits of progressive rollouts, segmentation allows companies to gain valuable insights from users through strategically releasing software to set groups. Testing in production is a key context for segmentation, where internal

users can be the first to try out new changes before updates are released to any customers. "Progressive rollouts allow software to be incrementally deployed to targeted users. Perhaps you want to release to a specific user community that's more tech-savvy and can adapt to the release more easily, and once they have the chance to test it and provide feedback, you release to the broader audience," says Shaghaghi. "A progressive approach not only minimizes the blast radius but also enables real-time feedback from customers."

Gainous similarly highlights the value of segmentation and targeted releases in the context of aiding innovation and improving software through collecting user feedback. "When we ship software, we give customers who are excited to try new things and engage the first chance to test out new features before we ramp it out to everyone. What we've found is there's a lot of positive customer engagement in that," he says. "It makes the feedback cycle and engagement much more valuable, because these customers are the people who most want to use new features, so they're going to give more meaningful feedback that allows us to make improvements before we scale up to general availability." Only 39% of survey respondents whose organization has software release monitoring tools say their tools have the capability to do controlled targeting of releases to subsets of users.

Beyond Deployment

However, managing risk for software releases goes beyond the approach to deployment. An effective strategy also includes practices for testing, issue detection, monitoring, and rolling back releases. Although organizations seem to generally recognize the overall importance of risk management, there's room for improvement in several key areas.

Testing new software is one area where organizations can fall short, as the manual approach still used by many organizations may be insufficient, Shaghaghi warns. "All software releases need to go through various testing cycles, including unit tests, integration tests, regression tests, and performance tests. If testing is very manual, you may skip through some tests in pushing to get a release out, which leads to defects in the software down the line," he says. And ultimately, a heavily manual approach negatively impacts time to market. "Although we've made significant improvements with automated testing in the last decade or so, many small to midsize companies still do a lot of manual testing, which doesn't scale well with modern release velocity requirements," he says.

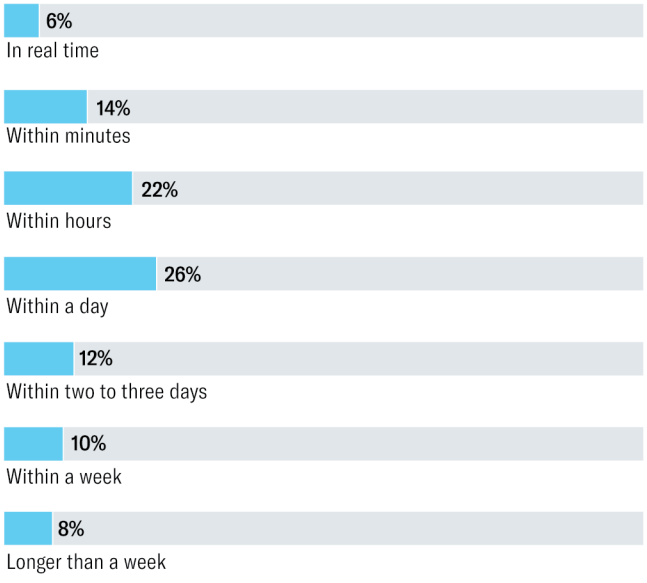
The speed of detecting problems with releases is another key component of risk management. Rapid detection is essential to speedy resolution of software release issues, and being able to detect errors or issues with releases in real time is, of course, the most optimal. However, only 6% of survey respondents report having this capability, and almost a third (30%) of respondents say their organization takes longer than a day to detect errors or issues with releases. FIGURE 2

FIGURE 2

Absence of Real-Time Issue Detection

Few organizations are capable of detecting software release issues in real time

In your estimation, when there are errors or issues with a software release, how long does it take your organization to detect them?



Base: 236 respondents. Not shown: 2% don't know.

Source: Harvard Business Review Analytic Services survey, April 2025

Monitoring the performance of releases, by employing key metrics such as latency and error rates, constitutes another essential element of risk management. "You need to have monitors—it's a way of knowing the health of your application and you can also see if you mess something up with a release," says Nykamp. "Monitoring allows you to take preventative action and hopefully catch stuff before it affects any of your users."

Overall, organizations seem to recognize the importance of monitoring: 91% of survey respondents' organizations have some form of monitoring in place to track the performance of their software releases. When it comes to whether monitoring tools are automated or manual, the group is split: 22% say their organization primarily uses automated monitoring tools (with real-time insights and alerts), 21% are using primarily manual monitoring tools, and the bulk (47%) sit in the middle using an equal mix of automated and manual monitoring tools.

Despite the degree of prioritization that seems to be placed on monitoring, companies may have room to grow in this area. "Organizations lack real-time visibility into the impact of their releases," says Shaghaghi. "Typically, there's a soft application support group that sits out there to address issues, but there's no streamlined function to establish that visibility for most organizations. Their approaches are more or less ad hoc."

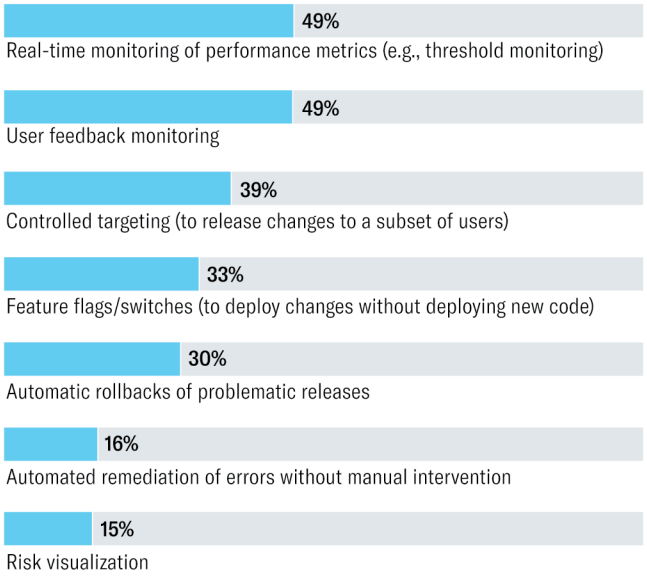
According to Ennaciri, although organizations are investing in modern observability solutions to gain insight into the performance of their releases, adoption of tools for automated rollback releases is significantly lower. Current observability tools provide considerable visibility and some level of insight into the source of errors, as well as the resulting customer and business impacts, but do not take action to remedy issues. Shaghaghi highlights the rollback process as an area ripe for improvement, citing rollback automation that prevents prolonged outages from faulty releases as a promising avenue. "For many organizations, software releases are rolled back manually, which is not only time-consuming but also leads to downtime," he says.

FIGURE 3

Components of Software Release Management Toolkits

Monitoring and targeting figure most prominently in organizations' release management capabilities

Which of the following capabilities do your organization's software release management tools have, if any? Select all that apply.



Base: 228 respondents, excluding those who report their organization has minimal or no monitoring tools in place to track the performance of software releases. Not shown: 14% don't know, 3% none, 0% other.

Source: Harvard Business Review Analytic Services survey, April 2025

He adds that very few organizations have a solid automated rollback process in place. In the survey, less than a third (30%) of respondents report being able to automatically roll back problematic releases. FIGURE 3

Recognizing the Need

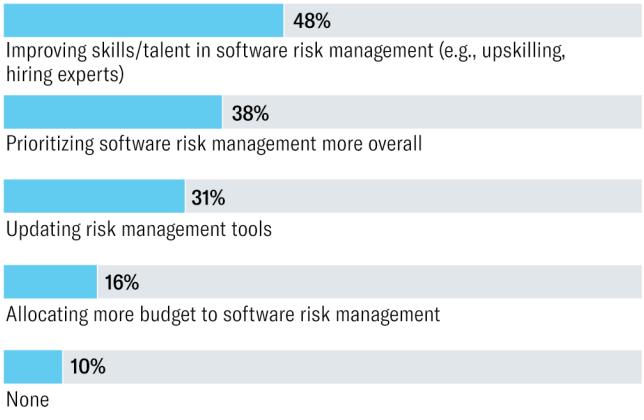
Fortunately, organizations recognize the need to bolster their approaches to risk management. In the survey, respondents report that their organizations are improving their strategies in various ways over the next 12 months, chief among them improving skills/talent in software risk management (48%) and prioritizing software risk management more overall (38%), followed by updating risk management tools (31%). Just 10% of the respondent pool say "none" to improvements being made in this area over the period. FIGURE 4

FIGURE 4

Steps to Strengthen Approaches

Companies are taking steps to improve their software release risk management

Looking ahead, in what ways is your organization improving its software release risk management over the next 12 months, if at all? Select all that apply.



Base: 236 respondents. Not shown: 14% don't know, 3% other.

Source: Harvard Business Review Analytic Services survey, April 2025

According to Ennaciri, organizations are actively investing in the various components of their risk management toolkits. "We're seeing strong investment in testing, test automation, and clear rollback plans to recover when software fails. Organizations realize that with more frequent releases, failure is going to happen. How quickly can we detect that failure and roll back?" he asks. "And finally, companies are investing more in monitoring and observability. They want to see how well systems are working, so if there are any issues, they catch them before the users do and fix the problem before the impact becomes bigger."

But companies may also need to take the softer additional step of facilitating greater internal collaboration among the teams involved. "When you look at the software value stream, there are many teams involved—you have product teams, development teams, security teams, infrastructure teams—but there's still poor communication and collaboration across those groups," Ennaciri says. "But if we want to push software quickly, we need to make sure we are collaborating and have a shared sense of accountability for the reliability of releases."

Conclusion

Software is more important now to businesses than ever, which means that effectively managing risk in software releases to ensure seamless delivery has become even more imperative. Importantly, AI has introduced a new class of risks in software releases, with respect to both its utilization in code generation and its amplified presence in production applications, further highlighting the need for solid software release practices.

Given the tangible pain and negative impact of faulty software releases, particularly with respect to team productivity and customer satisfaction, efforts to minimize both the frequency and impact of software failures should be a high priority for organizations with top-down support and buy-in from all key stakeholders. These initiatives should include specific KPIs and measurable goals to assess the efficacy of release management strategies.

Promoting successful, error-free releases starts from the means of software deployment used. A phased, gradual approach to releasing software can help organizations avoid the pitfalls of big bang releases, which are more prone to errors simply because of their size, and serve as groundwork to reduce risk ahead of a full production release. By coupling progressive rollouts with segmentation capabilities, organizations can reap the benefits of intentionally targeting their releases to enable productive user feedback and heightened innovation, which can translate to revenue growth and better business outcomes overall.

Building on the groundwork laid by phased delivery, organizations can then effectively harness various technologies to address other pieces of risk management, including tools and frameworks for testing, issue detection, monitoring, and release rollbacks. Approaches to tie monitoring to automated rollbacks of problematic releases, effectively bridging the gap between awareness to active remediation of issues, can prove particularly useful. Ultimately, a comprehensive framework for software-related risk management utilizing these key components can turn risk reduction into a strategic advantage, as well as allow organizations to address the novel risks AI poses as it is increasingly embedded in applications and the software development life cycle itself.

"Because of the negative impacts of poor software releases, there's a huge interest and growing awareness in the adoption of crucial practices like progressive rollouts and robust testing," says Ennaciri. "A lot of organizations are trying to build that maturity in their releases and be better prepared."

METHODOLOGY AND PARTICIPANT PROFILE

Harvard Business Review Analytic Services surveyed 236 members of the *Harvard Business Review* audience via an online survey fielded in April 2025. Respondents qualified to complete the survey if their organization developed and managed any of its own software and they were familiar with their organization's management of its software releases.

ORGANIZATION SIZE

- 28% — 10,000 or more employees
- 36% — 1,000–9,999 employees
- 10% — 500–999 employees
- 17% — 100–499 employees
- 9% — 50–99 employees

SENIORITY

- 24% — Executive management/board members
- 43% — Senior management
- 20% — Middle management
- 13% — Other grades

JOB FUNCTIONS

- 20% — IT
- 19% — General/executive management
- 9% — Engineering

All other functions less than 8% each.

INDUSTRIES

- 17% — Technology
- 14% — Financial services
- 11% — Manufacturing
- 10% — Government/not-for-profit
- 8% — Health care

All other sectors less than 8% each.

REGIONS

- 48% — North America
- 20% — Europe
- 17% — Asia Pacific
- 8% — Latin America
- 8% — Middle East/Africa

VISIT US ONLINE

hbr.org/hbr-analytic-services

Harvard Business Review Analytic Services is an independent commercial research unit within Harvard Business Review Group, conducting research and comparative analysis on important management challenges and emerging business opportunities. Seeking to provide business intelligence and peer-group insight, each report is published based on the findings of original quantitative and/or qualitative research and analysis. Quantitative surveys are conducted with the HBR Advisory Council, HBR's global research panel, and qualitative research is conducted with senior business executives and subject-matter experts from within and beyond the *Harvard Business Review* author community. Email us at hbranalyticservices@hbr.org.

CRE66040725