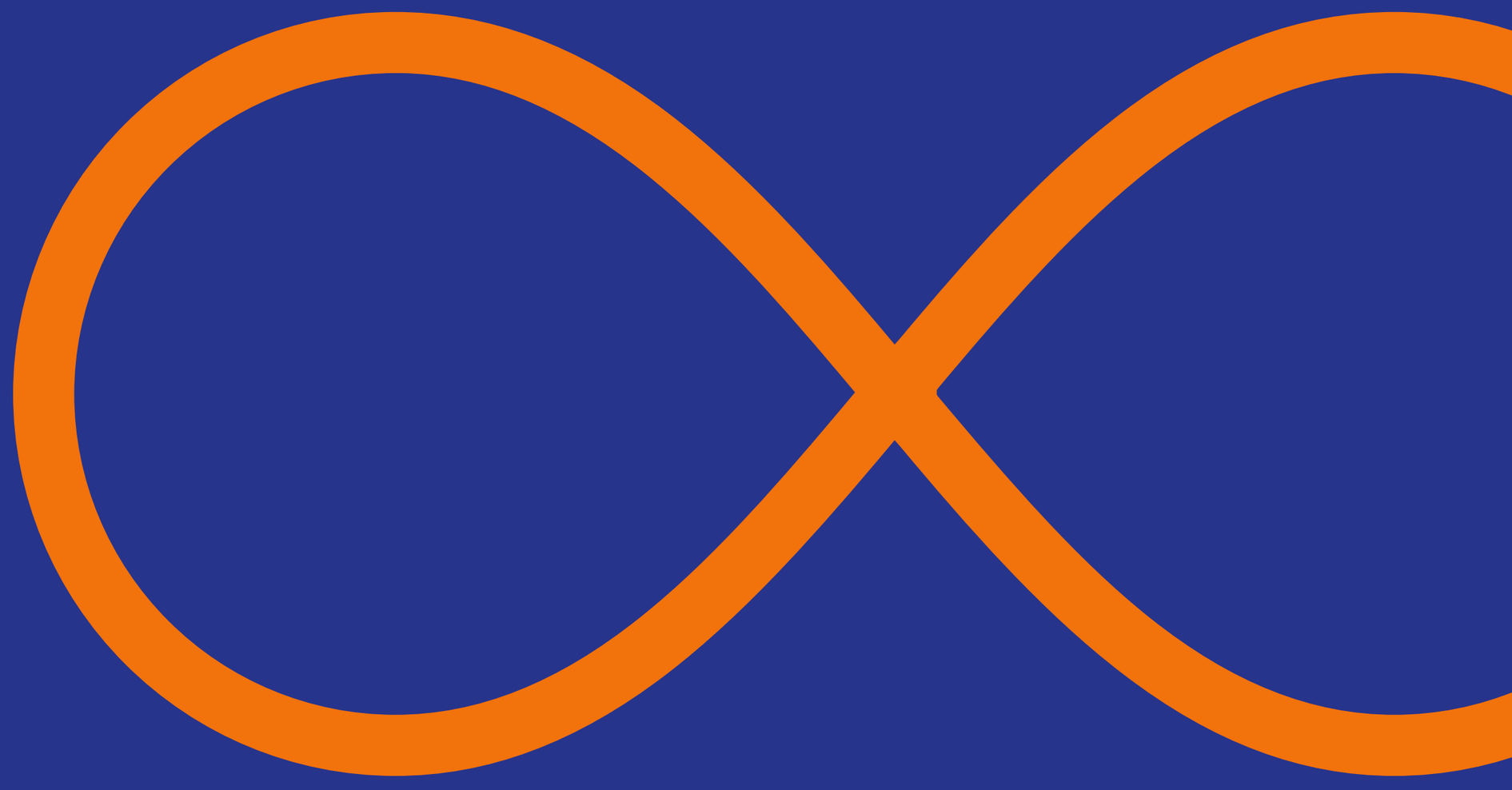
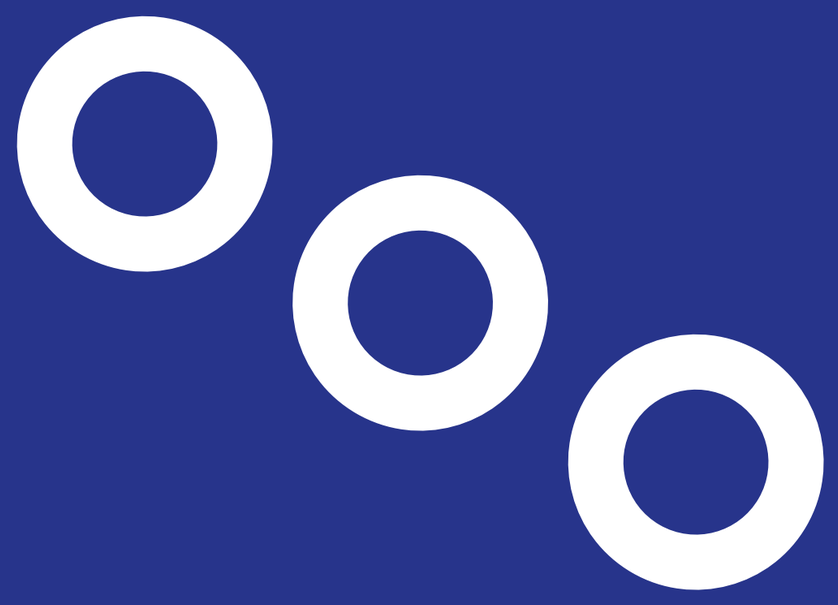
DevOps Handbook Companion Guide

How knowledge sharing helps organizations follow the Three Ways of DevOps





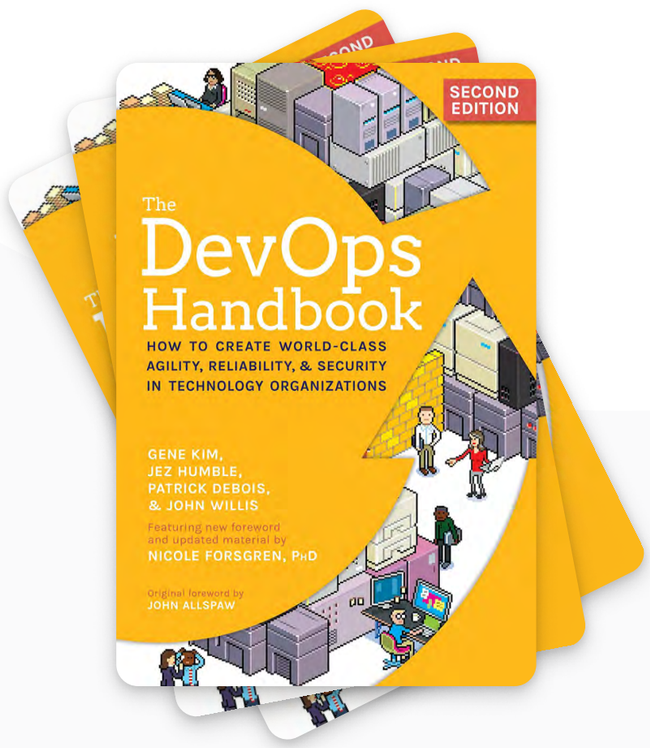
stack overflow FOR TEAMS

Overview of the DevOps Handbook

Now in its second edition, *The DevOps Handbook* has provided countless organizations and practitioners with the theory, principles, and practices to help companies realize the benefits of successful DevOps initiatives. Since its publication in 2016, the handbook has sold a quarter of a million copies and been translated into 11 languages. Coauthors Gene Kim, Jez Humble, Patrick DeBois, John Willis, and Nicole Forsgren, PhD, provide compelling guidance and case studies to demonstrate how real-world companies are executing and benefiting from DevOps adoption.

In this unofficial companion guide, we'll explore what's new in the second edition and how the right approach to knowledge sharing and collaboration can support the Three Ways of DevOps and enhance business outcomes. We also sat down with coauthor Gene Kim to talk about platforms that support DevOps projects and improve developer experience and performance.

With more than 90% of IT organizations exploring or implementing DevOps projects to accelerate solutions delivery, plenty of organizations are looking for developer-first platforms that can support DevOps. Keep in mind that this companion guide doesn't address every chapter of the handbook, and it's not a replacement for the handbook: it's a closer look at how the right developer-focused platform can help organizations fulfill the promise of the Three Ways of DevOps.



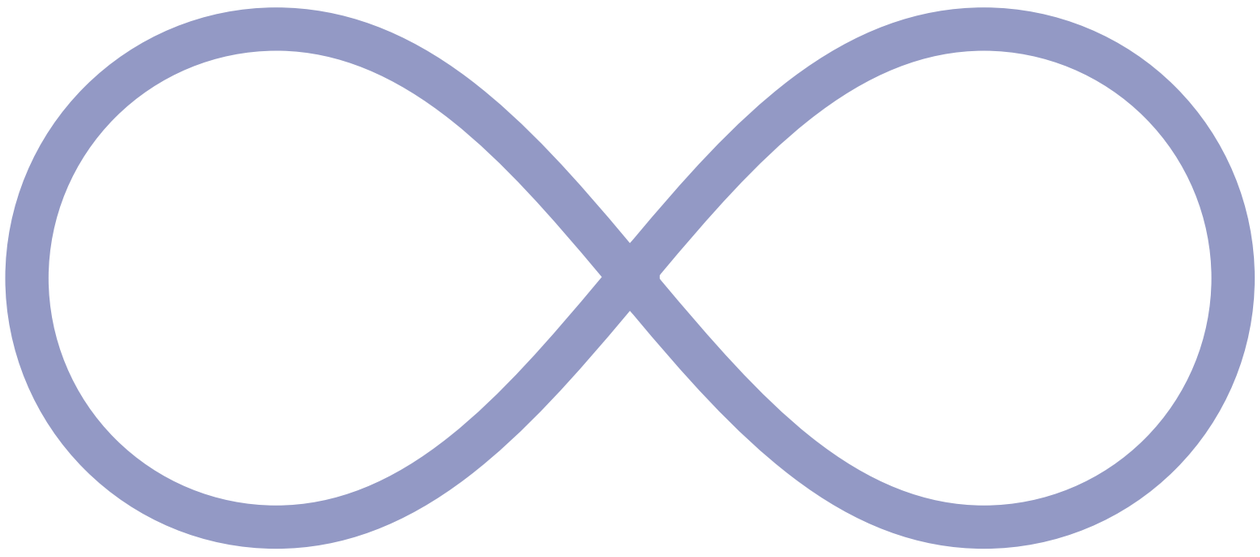
Gene Kim

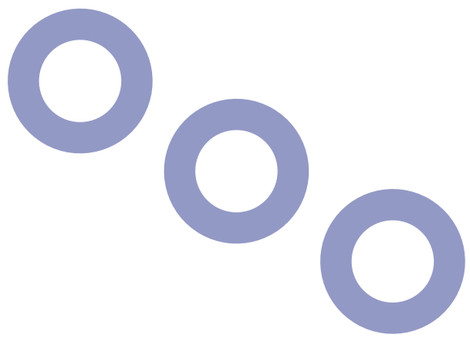
Coauthor, The DevOps Handbook

What's new in the second edition?

The updated and expanded edition of The DevOps Handbook includes:

- 15 new case studies to illustrate DevOps principles and practices in real-world contexts, from Adidas and Fannie Mae to American Airlines and the US Air Force
- A new foreword and research from Dr. Nicole Forsgren, Partner at Microsoft Research, Head of the Developer Velocity Lab, and a lead researcher on the State of DevOps Reports
- Updates to the main text where new research, findings, and experiences have shaped how DevOps is understood and implemented
- New resource sections following each chapter to guide further learning
- Updated afterwords from each coauthor





The Three Ways of DevOps

The DevOps Handbook refers to the principles underpinning DevOps as the Three Ways: Flow, Feedback, and Continual Learning and Experimentation.

Throughout this companion guide, we will highlight how continual learning cultures and knowledge sharing provide the foundation for strengthening your DevOps practice. Now, let's explore the DevOps Handbook.

The First Way:

The principles of flow

The First Way of DevOps is concerned with *flow*: "the fast and smooth flow of work from Development to Operations in order to deliver value to customers quickly" (*The DevOps Handbook*, p. 19). Improving flow by making work visible and building processes that keep defects from being passed along to downstream work centers is crucial to achieving DevOps outcomes.

Go with the flow

Developers often talk about being in the *flow*, meaning to be focused on complex cognitive work without being interrupted by distractions or the need to context-switch. Context-switching can be a result of attempting to multitask on too many projects, interruptions from texts, instant messaging, email alerts, or "shoulder-tapping" from coworkers .

Interruptions are a nuisance for technology workers. Although the immediate consequences of these distractions may be invisible, their cumulative impact on productivity can be enormous (p. 21).

Did you know?
Employees lose 23 minutes every time they are interrupted.

Source: Fast Company, 2008

As we've added more collaboration tools that come with higher expectations for an immediate response, it's become harder to achieve a state of flow. Kim acknowledges that interruptions from email and chat platforms effectively form "this hyperactive hive mind, where it's really almost impossible to get into a sense of flow." We might think of the First Way of DevOps as an organization-wide state of flow, in which tasks and information flow quickly and without friction from development to operations.

As You Prepare to Implement the First Way, Ask Yourself:

- Where and how are your teams being taken out of their flow state?
- Can you see the ripple effects of this in your efficiency and productivity metrics?

Make work visible

How does your team spend time?
Regardless of what time or project tracking system you use, it's important to track both actual, uninterrupted hours you spend on projects and other tasks like meetings and emails that may be interrupting focus.

DevOps success depends on an organization's ability to make the technology value stream visible: to see where work is flowing without friction and where it's piling up or stalling out.

Many teams use visual aids like kanban boards or sprint planning boards, both physical and virtual, to overcome this hurdle.

In trying to uncover "invisible work," ask yourself:

- Are you scheduling 100% of everyone's time? Is there any buffer in the time estimates to account for unexpected non-hands on keyboard time?
- Can you see which people are unofficial helpers with other projects, because they are either subject matter experts or because they are highly tenured?
- Can you see which people are struggling and might need more help or time?

Reduce handoffs with self-service platforms

The First Way also calls for reducing the number of handoffs necessary to move code through the technology value stream. With multiple departments and stakeholders at work on everything from functional and integration testing to environment creation, server and storage administration, security, and more, the number of handoffs—and, thus, the potential for errors—can quickly get out of hand.

Every team involved represents another opportunity where communication might break down or the flow of work might be interrupted. Some knowledge, inevitably, is lost with every handoff, so that over time people working on the problem can lose sight of crucial context or what strategic goal the work supports. To add to the complexity and confusion, various teams probably use different systems for ticketing, project management, documentation, and communication.

DevOps principles recommend that organizations manage these problems by reducing the number of handoffs involved in each project. This isn't a one-step process: it requires organizations to build platforms and restructure teams in such a way that engineers can self-service instead of depending on other people for building, testing, and deploying software.

To streamline handoffs and reduce friction, ask yourself:

- Where are your communication or knowledge breakdown points?
- Can you think of an example of when crucial context was lost because of loosely structured handoffs?
- What was the impact on the individuals on your team?
- What was the impact on the business?

Putting the First Way into practice

1. Reduce distractions and interruptions that pull DevOps team members out of their flow.
2. Improve the visibility of work to eliminate hardships in the value stream, including time required to research, locate, or recreate solutions others have already reached.
3. Reduce knowledge loss caused by unnecessary handoffs.

A common knowledge base, like Stack Overflow for Teams, reduces context switching for both knowledge seekers and the experts throughout your organization. Look for integrations with other collaboration tools that your teams use to increase engagement without taking them out of their workflow. For example, chat integrations with Slack and Microsoft Teams allow people to search for solutions, ask questions and provide answers without switching apps.

The Second Way:

The principles of feedback

The principles of the First Way enable the rapid flow of work from left (development) to right (operations), while the principles of the Second Way focus on creating fast, reciprocal, constant feedback from right to left. The goal is to create a safer and more resilient work system. Absent or lacking feedback leads to poor outcomes: missed opportunities to spot and fix problems before they damage the business. In our conversation, Kim put it more plainly: “Learning without feedback is flipping impossible.”

Better feedback leads to better outcomes

The Second Way of DevOps recommends that organizations “shorten and amplify feedback loops” so teams can see problems as they happen and share that insight with everyone in the value stream (p. 223). This approach allows teams to identify and mitigate problems early in the SDLC—with luck, well before a catastrophic incident like the Knight Capital failure.

Moreover, the Second Way requires building a work system in which knowledge acquired downstream in operations is integrated into the upstream work of developers and product managers. This is how problems, potential issues, and revealing patterns lead to learning and improvements.

The handbook recommends that DevOps-minded organizations “create a process that allows everyone to get feedback on their work” and “makes information visible to enable learning” (p. 223).

Assess how well you're overcoming silos

What metrics should organizations consider when evaluating how effectively they're eliminating silos? Kim recommends focusing on code deployment lead time, which simultaneously predicts how effectively your team can build/test/deploy and how quickly developers can get feedback to improve code quality.

"Code deployment lead time is a great proxy for many things"

Gene Kim

"Code deployment lead time is a great proxy for many things," says Kim, "but especially for ineffective silos, because if a code deployment has to span 20-30 different silos, of which if anything goes wrong it might take add another week, you're looking at code deployment lead times measured in months." Kim calls this metric "one of the best measures and indicators that every technology organization needs to focus on. Are developers getting feedback within seconds or minutes or, worst case, hours…or are they discovering the error only nine months later during integration testing, where cause and effect has basically vanished?"

Swarm to solve problems and build more knowledge

Part of the process of shortening and amplifying feedback loops to maximize opportunities for improvement involves "swarming" issues to fix them immediately, rather than creating workarounds or waiting to solve the problem. The goal of swarming is to contain problems before they have a chance to spread, and to diagnose and treat the problem so it cannot recur.

The handbook cites the famous Toyota Andon cord as the ur-example of swarming: "In a Toyota manufacturing plant, above every work center is a cord that every worker and manager is trained to pull when something goes wrong [a defective or missing part or a delay in completing a task]. When the Andon cord is pulled, the team leader is alerted and immediately works to resolve the problem."

If the problem can't be resolved in a short, specified time window, the entire production line is stopped so that everyone can focus on fixing the problem.

Swarming might seem to contradict common management wisdom, which dictates that local problems should not disrupt global operations. But swarming is crucial to enabling the Second Way of DevOps because it enables learning. Swarming "prevents the loss of critical information due to fading memories or changing circumstances. This is especially critical in complex systems, where many problems occur because of some unexpected, idiosyncratic interaction of people, processes, products, places, and circumstances" (p. 38). As time passes, it becomes increasingly difficult to reconstruct exactly what was happening when the problem arose—whereas swarming freezes the problem in time so that it can be examined in its full context.

The physical Andon cords used by Toytoa don't help remote and hybrid teams very much, but there are virtual versions of raising your hand and saying "help" before an issue becomes a catastrophe. Intelligent swarming means getting the people who are best-qualified to solve the issue working on the issue as soon as possible—wherever they are located. In this model, the person who takes ownership of the issue often owns it until it's fully resolved. They may well solicit help and feedback from others in the process, but they won't escalate the problem to somebody else and forget about it.

Ask Yourself:

- Have you had a time when teams needed to swarm to solve an idiosyncratic interaction or another complex challenge? Where did you capture the learnings and outcomes for the future?
- How are you making lessons learned from problems being solved discoverable?

What is an Allen Curve?
The Allen Curve is a graphical representation that reveals the ponential drop in frequency of communication between engineers as the distance between them increases. It was discovered by Massachusetts Institute of Technology Professor Thomas J. Allen in the late 1970s.

Putting the Second Way into practice

1. Build feedback loops that integrate downstream learning into earlier development.
2. Empower employees to quickly get the assistance they need, from peers and experts throughout the company.
3. Look for ways to build connections across remote and hybrid teams to flatten the Allen Curve.

The right knowledge sharing and collaboration solutions can facilitate feedback on approaches before they're implemented, and connect colleagues with upstream and downstream stakeholders.

The Third Way:

The principles of continual learning and experimentation

The Third Way is a set of principles and practices designed to build a culture of continual learning, experimentation, and improvement. Fostering a dynamic learning culture helps organizations create valuable products and solutions more quickly.

Empowering individuals to discover and share knowledge allows teams throughout the organization to benefit from those individually obtained insights. Simply put, one person or one team might have discovered or designed a solution to a problem that also plagues others throughout the organization—but if that knowledge can't be captured and disseminated, its value is limited.

As many organizations have shifted to working remotely, managing knowledge has become both more challenging and more crucial.

The handbook lays out three practices that organizations should implement as part of the Third Way:

- Enable and inject learning into daily work
- Convert local discoveries into global improvements
- Reserve time to create organizational learning and improvement

Enable and inject learning into daily work

The DevOps Handbook emphasizes that organizations in the technology value stream should strive "to create a high-trust culture, reinforcing that we are all lifelong learners who must take smart risks in our daily work" (p. 45). When individuals, teams, and organizations are encouraged to learn from their mistakes as well as their triumphs, it's easier to identify and reinforce great, workable solutions.

"The only way to win in the marketplace is by out-learning the competition,"

Gene Kim

Kim emphasizes that the best way to succeed in a competitive market is to out-learn and out-experiment everybody else, acquiring more knowledge in the service of delivering more value to customers, more quickly. "The only way to win in the marketplace is by out-learning the competition," Kim says, referencing a quote from systems scientist Peter Senge, author of *The Fifth Discipline: The Art and Practice of the Learning Organization* (2006). "That means you learn faster and you learn more."

Ask Yourself:

- How are you enabling your team to learn every day?
- What's the risk to your business if the competition is out-learning you?
- Do you know how your teams prefer to learn? What are they learning in their spare time, away from work?

Dynamic learning equals resiliency

Complex systems require self-diagnostics and self-improvement to stay responsive, reliable, and resilient. When it comes to complex systems, it's impossible to predict every possible outcome of each action we might take. Of course, static tools like checklists and runbooks help to structure our understanding of complex systems, but unexpected, potentially catastrophic accidents remain a possibility.

To work safely and effectively within complex systems, organizations need to embrace a learning system that empowers organizations to understand their mistakes and turn that hard-won insight into improvements that keep problems from recurring. One of the significant advantages of resilient organizations is that coworkers can learn from others' missteps, borrow and adapt their solutions, and build in optimizations that avoid triggering the same problems in the future. When someone comes up with an elegant, eminently workable solution, future folks can find that solution immediately, as soon as they encounter the need.

Making solutions accessible and implementing systems to capture individual knowledge as it's produced is an important step toward building resiliency into your organization. Again, working within a complex system requires constant self-diagnostics and self-improvement, areas that allow us to detect and fix problems, then translate those solutions to organization-wide issues. Organizations that develop the ability to detect and solve problems—and make those solutions available throughout the community—are resilient, capable not only of solving problems but of using them to improve performance across the board.

One key to developing organizational resilience is making problem-solving and crisis response a habitual part of the culture.

Ask Yourself:

- Do you consider your team to be resilient? What's preventing them from being resilient?
- How do you normalize troubleshooting, problem-solving, and mistake-making so that recognizing and responding to crises becomes habitual?

Convert local discoveries into global improvements

Resilient organizations need to be able to find local solutions and translate them into global improvements. Teams with resilient, high-trust cultures can translate local learnings into global improvements.

Dr. Ron Westrum, an expert on organizational culture, performance, and safety, observed that, with healthcare organizations, a “generative” culture was a top predictor of patient safety: “Generative organizations are characterized by actively seeking and sharing information to better enable the organization to achieve its mission. Responsibilities are shared throughout the value stream, and failure results in reflection and genuine inquiry” (pp. 47-48). The same holds true for technology companies: “a high-trust, generative culture also predicted software delivery and organizational performance in technology value streams” (p. 48).

Improve culture to boost morale and retention

Global improvements don't just have a positive impact on your bottom line; they also contribute to higher job satisfaction and lower burnout rates among your employees. As *The DevOps Handbook* puts it, "Continual learning and experimentation do more than just improve the performance of our systems. These practices also create an inspiring, rewarding workplace where we are excited to work and collaborate with our peers" (p. 46).

How does psychological safety enable high performing teams?

Learn more

Research from the State of DevOps Report reveals that people who work at companies that implement Third Way practices are more than twice as likely to recommend their organization to friends, report higher job satisfaction, and experience lower levels of burnout. Citing McKinsey research, the handbook also reports that "culture—which includes psychological safety, collaboration, and practicing continuous improvement—is a key driver of developer velocity and organizational value" (p. 46).

Against the backdrop of the Great Resignation, many organizations are asking themselves what they can do, beyond straightforward comp adjustments, to attract and retain the best development talent. One clear step is to invest in solutions to build a gratifying, generative, knowledge-obsessed culture.

Reserve time to foster organizational learning and improvement

The DevOps Handbook emphasizes that creating a culture of continual learning, experimentation, and improvement requires not just buy-in from leadership but also leadership's commitment to prioritizing learning and creative problem-solving. Great teams and organizations don't grow out of leaders who make all the right decisions, every time. Instead, a leader's role is to create the conditions that will allow their teams to excel.

Says Kim, “It’s incumbent on leaders, especially in the age of the Great Resignation, to really unleash the creativity and the full potential of their teams, where everyone is bringing their best to work. They feel engaged. It challenges them. They get an intrinsic sense of reward out of the work…leaders have a great deal to do with to what degree members on the team feel that, and if you don’t have that, there’s not a lot keeping you there.” Developers, like all employees, vote with their feet, and if they’re not happy, they’ll join the millions of other people walking away from their old jobs.

Kim says that leaders must state clearly that learning is valuable—and walk the walk. One, freely sharing knowledge, tools, and troubleshooting techniques demonstrates that learning is a core principle of your organization. “Two,” says Kim, “it’s super handy to be able to pick up tips and tricks from others. That means learnings are not just local and trapped to the person, to the team, whatever. Instead, [knowledge] really should be radiated out as quickly and widely out as possible. Better yet, if it’s so good, then it actually gets embedded in the tools you use so that you don’t… require a workaround all the time.”

Choosing the right strategy for continual learning and knowledge sharing

A culture of continual learning and knowledge sharing doesn’t happen accidentally. It needs the wholehearted support and careful attention of leadership, along with enabling platforms that remove barriers between knowledge seekers and experts.

“Platforms for developers are so important”

Gene Kim

Empower developers with platforms that prioritize their needs

Platforms tailored to developers’ needs allow them to be more productive, focused, and happier in their work. “Platforms for developers are so important,” explains Kim, because they “allow developers to be productive, to focus on what needs to be done… platforms are all about removing toil, simplifying how much of the world [developers] actually need to understand to do what they do.” Protected from what Kim calls “the idiosyncrasies of the real world,” insulated from distraction, developers are better-equipped to do their jobs—and more likely to be happy doing them.

The right platforms support the developer experience, which is a top concern for developers evaluating new job prospects in the midst of the Great Resignation. In Stack Overflow’s pulse survey of more than 500 developers, more than 53% said prioritizing the developer experience makes a current or future employer more attractive. Salary transparency was the second-highest priority (41%), while 40% wanted opportunities to learn from people outside of their team. Thirty-five percent prioritized a structured onboarding experience, and 33% wanted their employer to make it easy to identify experts within the company.

Pick a solution that transcends silos

Most of us in tech have encountered the frustration and communication breakdown that can occur when teams use radically different tools, languages, and terminologies. For an organization to reap the rewards of a continual learning culture, seeking and sharing knowledge can’t be ineffective or burdensome. That’s why a developer platform that doesn’t offer knowledge portability won’t solve your problems.

“You want portability between teams,” Kim explains. “You want enough uniformity in tools and norms so that you can switch between teams somewhat easily. The opposite is when everything is so idiosyncratic that just because you work on one team in an organization doesn’t mean you know anything about what it would take to work on another team.”

Conway's law states that an organization's design system will mirror its communication structure. This law informs a core value within DevOps: that architecture is a top predictor of performance. Says Kim, "One of the most amazing discoveries to me in the State of DevOps research was to what extent architecture is one of the top predictors of performance." He measures performance as "to what degree can a team independently…develop, test, and deploy value to customers, to what degree can they do their work without a lot of fine-grained communication coordination with people outside the team, to what degree can they deploy their service on demand, even independent of services it depends upon."

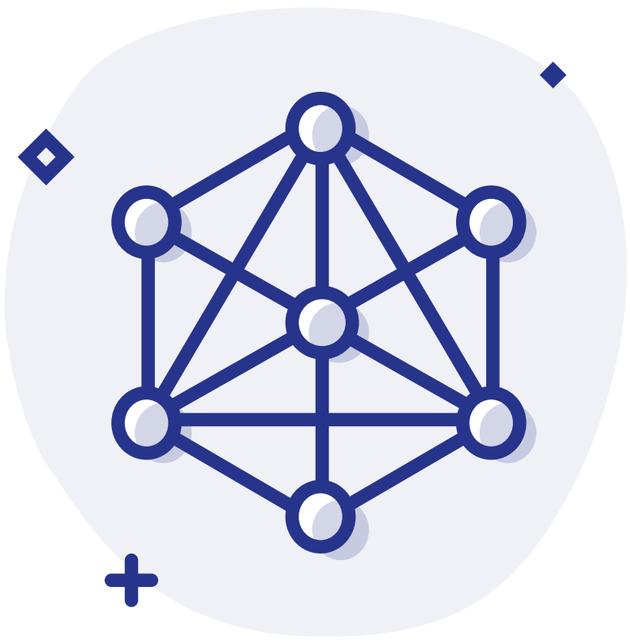
As the number of functionalities increases, so does the need for cross-team collaboration that transcends silos. Enabling this collaboration and communication becomes a key mandate for leaders.

Ask Yourself:

- Are feedback and handoff knowledge part of your developer platform?
- Has scaling technical teams created new knowledge silos?
- Are you capturing new learnings from each launch, and sharing those with less experienced team members?

Putting the Third Way into practice

1. Create an environment that provides daily learning opportunities.
2. Build a culture of psychological safety where employees feel safe making mistakes, learning from them, and sharing their experiences so the whole organization can learn.
3. Commit to prioritizing learning time for your teams.
4. "Enable every new learning to be incorporated into the collective knowledge of the organization" (p. 334).



Appendix A:

How a knowledge sharing platform enables the Three Ways

Stack Overflow for Teams is a developer-first knowledge sharing and collaboration platform that integrates with development and engineering workflows to support DevOps projects.

In line with the Three Ways, Stack Overflow for Teams enables:

- The fast, seamless flow of work from development to operations, in order to deliver more value to customers (The First Way)
- Rapid feedback to increase software quality, reliability, and safety (The Second Way)
- The continual creation of individual and collective knowledge to produce value more quickly and cheaply (The Third Way)

Suggested reading/viewing:

- We already use Stack Overflow's public site. Why do we need Stack Overflow for Teams?
- Stack Overflow Vs Stack Overflow for Teams - What's the difference?

Help your teams stay in their flow for better business outcomes

The high cost of context-switching is one of the reasons organizations are looking for platforms that help developers locate knowledge quickly, without having to interrupt their colleagues. Because 82% of developers already work with a tab open to Stack Overflow’s public platform, it’s the natural place to get their questions answered quickly with minimal disruption or application switching.

Stack Overflow for Teams' common knowledge base reduces context switching for both knowledge seekers and the experts who are documenting solutions. In addition, chat integrations for platforms like Slack and Microsoft Teams allow people to ask questions and provide answers without switching apps.

It can be hard to visualize the often-invisible costs of frequent interruptions to search for knowledge, but Stack Overflow for Teams can help.

Case Study: Progressive Insurance visualizes knowledge flows between teams.

In a recent webinar, members of Progressive's Open Source Office described how they leveraged the flexible API from Stack Overflow for Teams to visualize how knowledge flows between departments and gain more insight into trending questions, topics, and concerns. "The immediate story that [Stack Overflow for Teams] told us, right after we made it go live, was that we were already breaking down the walls between our domains in terms of communication," said Open Source Developer Advocate Michael "d00d" Parkins.

As a self-service knowledge management platform, Stack Overflow for Teams helps organizations reduce knowledge loss or avoid it entirely by capturing individual learnings and radiating this knowledge throughout the organization. The platform makes knowledge easily accessible to people seeking the same information in the future and embeds the best solutions in the tools developers use every day.

Case Study: At Expensify, Stack Overflow for Teams gives developers the confidence to work on any project.

"It gives an open opportunity to everyone who has the knowledge to share it," said software engineer Ira Praharaj. Engineers across the company estimated they were saving between two and three hours a week that had previously been devoted to hunting down information they needed.

Increase software quality, reliability, and safety

Stack Overflow for Teams helps organizations enable fast feedback and feedforward loops across departments. The transparent and easy to discover knowledge sharing format is visible to all departments and invites early interaction and collaboration to eliminate issues and surface better solutions.

Not unlike the Andon cord, Stack Overflow for Teams allows users to quickly and easily solicit input and assistance from colleagues, ideally before the issue becomes a significant blocker for the whole team.

Additional reading: Stack Overflow for Teams helps transcend swarming and move toward a dynamic, powerful knowledge loop.

The platform allows developers to solicit immediate feedback and assistance from their community, even connecting them with company experts outside of their normal silos, while at the same time preserving critical knowledge and essential context. This way, when the same problem or a similar issue pops up, developers have the solution immediately at hand.

Create a developer-first culture: focus on continual learning

Stack Overflow for Teams helps normalize troubleshooting, problem-solving, and mistake-making so that recognizing and responding to crises becomes habitual.

Stack Overflow for Teams gives organizations the ideal framework for building a generative culture in which users actively seek and share information to improve efficiency and performance. A purpose-built platform puts the developer experience at the center. Creating opportunities to learn from folks outside the team and uncovering SMEs throught the organization.

Stack Overflow for Teams enables learning (and teaching) that fits seamlessly into developers' existing workflows and leverages an already-trusted platform for developer knowledge. The familiarity of Stack Overflow increases adoption, reducing the risk associated with adoption of unknown platforms, and meets developers where they already are.

Case study: 84.51 is enabling learning and growth for everyone.

Something interesting happened as employees began sharing on Stack Overflow Teams. Data scientists showed off their deep technical knowledge, answering questions on the nuances of PySpark calculations before the designated experts. Junior developers came out of the woodwork to answer questions and contribute. It wasn't just the universally-known SMEs who were answering questions. Anybody who had knowledge was sharing it. "You start to identify who the real subject matter experts are," said Wones. Not only do Wones and his team have an easy channel to ask questions of the SMEs in the entire company, but those questions and answers remain easily accessible for the next person.

Appendix B:

Platform Evaluation Checklist

What should you look for in a platform that enables the Three Ways? The right platform should:

- ✓ Support cross-functional collaboration as DevOps moves closer to the business
- ✓ Capture local knowledge and helps apply it to global problems
- ✓ Convert tacit, institutional knowledge into explicit information that anyone can access
- ✓ Improve productivity and reduces distractions by allowing employees to ask and answer questions without disrupting their established workflows
- ✓ Surface subject matter experts (SMEs) throughout the organization by connecting questions with the right people to answer them
- ✓ Reinforce a culture of continual learning and experimentation where employees feel supported, inspired, and not terrified of making mistakes
- ✓ Build connections across remote and hybrid workforces

stack overflow
FOR TEAMS

To learn more visit
stackoverflow.co/explore-teams