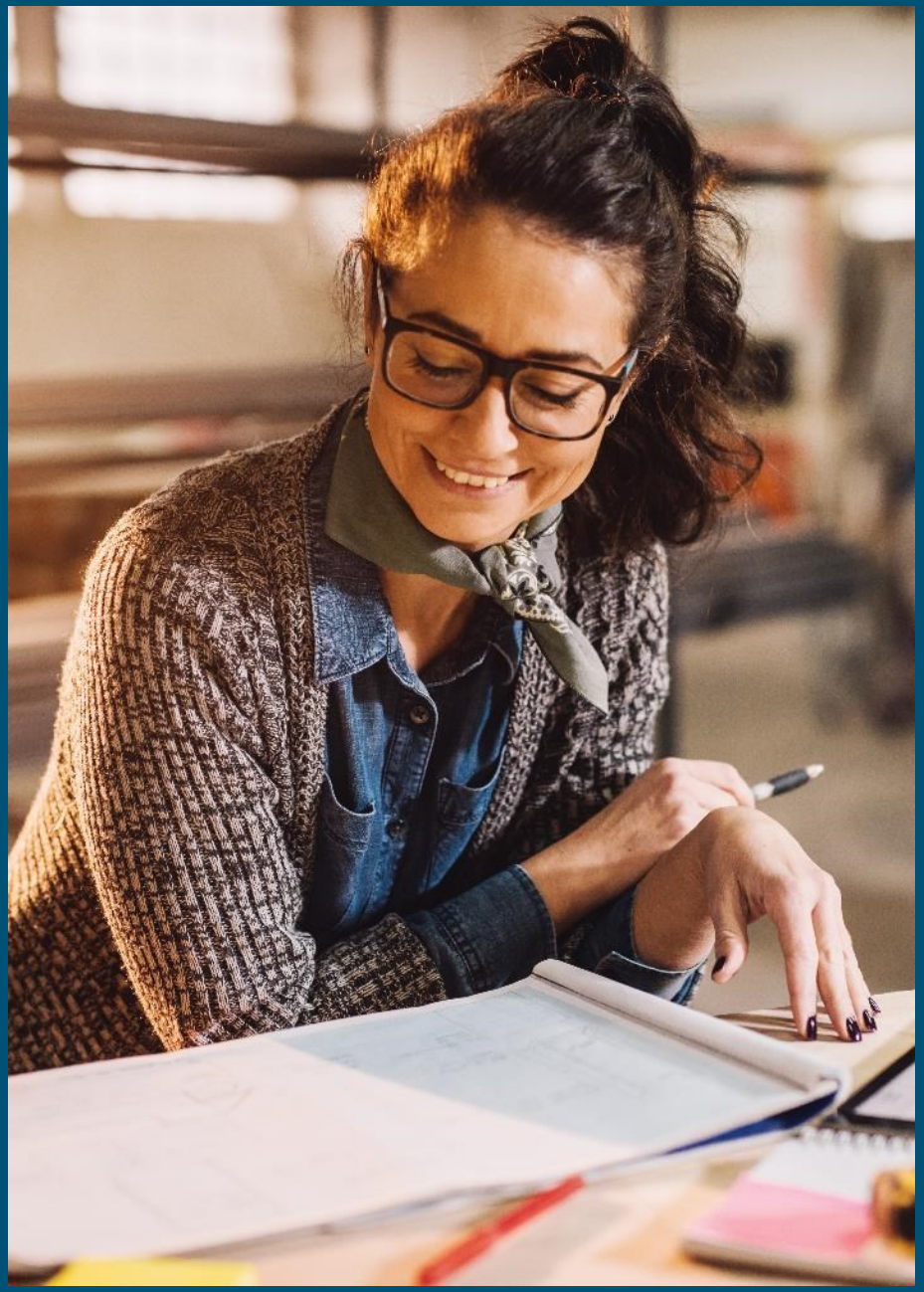
Driving a Cultural Transformation towards service ownership with PagerDuty

Best practices for introducing "You build it, you run it" at your organization

In collaboration with

PagerDuty

Table of contents



Digital transformation: To the cloud and beyond

Today, technical leaders across industries are accountable for delivering high-quality, always-on customer experiences through digital services—especially since downtime, outages, and slowdowns can negatively impact their company's revenue and brand. According to IDG's 2020 State of the CIO Executive Summary, CEOs and boards are designating CIOs as the leaders of digital business transformation projects.

To achieve digital transformation, more technical leaders are shifting to the cloud—but with hybrid and multi-cloud environments becoming increasingly common, they face added complexity to their organization's infrastructure management. As a result, teams can become decentralized into lines of business, each with their own toolchains and workflows—making visibility and collaboration difficult.

As systems and teams get more complex, there is no way to efficiently manage everything in a purely centralized fashion. A centralized approach can be especially painful when it comes to incident response. Siloed systems and teams operating in traditional models can create a domino effect that negatively impacts the customer experience and puts businesses at risk.

In working closely with its customers, PagerDuty found that organizations using a centralized incident response system typically saw 30 minutes to detection and four hours to resolution. With PagerDuty, detection time shrank to five minutes and time to resolution was reduced to under two hours.

The case for service ownership, or “You build it, you run it”

With the increased reliance on digital services across all aspects of life today, there’s more at stake when things go wrong. And things will go wrong—service failure is an inevitable part of operating with technology and systems. But how a company responds makes all the difference when it comes to the customer experience and, ultimately, company revenue. In order to streamline the incident response process, businesses need several things:

- Visibility into the incident to properly acknowledge and triage.
- Intelligent routing and context so that the right teams are mobilized with the right information.
- Focused orchestration to make the incident response as swift as possible and to contain impact.

Along with methodologies like agile and DevOps, service ownership has been a key opportunity for technical organizations that need to adapt to new cloud and hybrid environments.

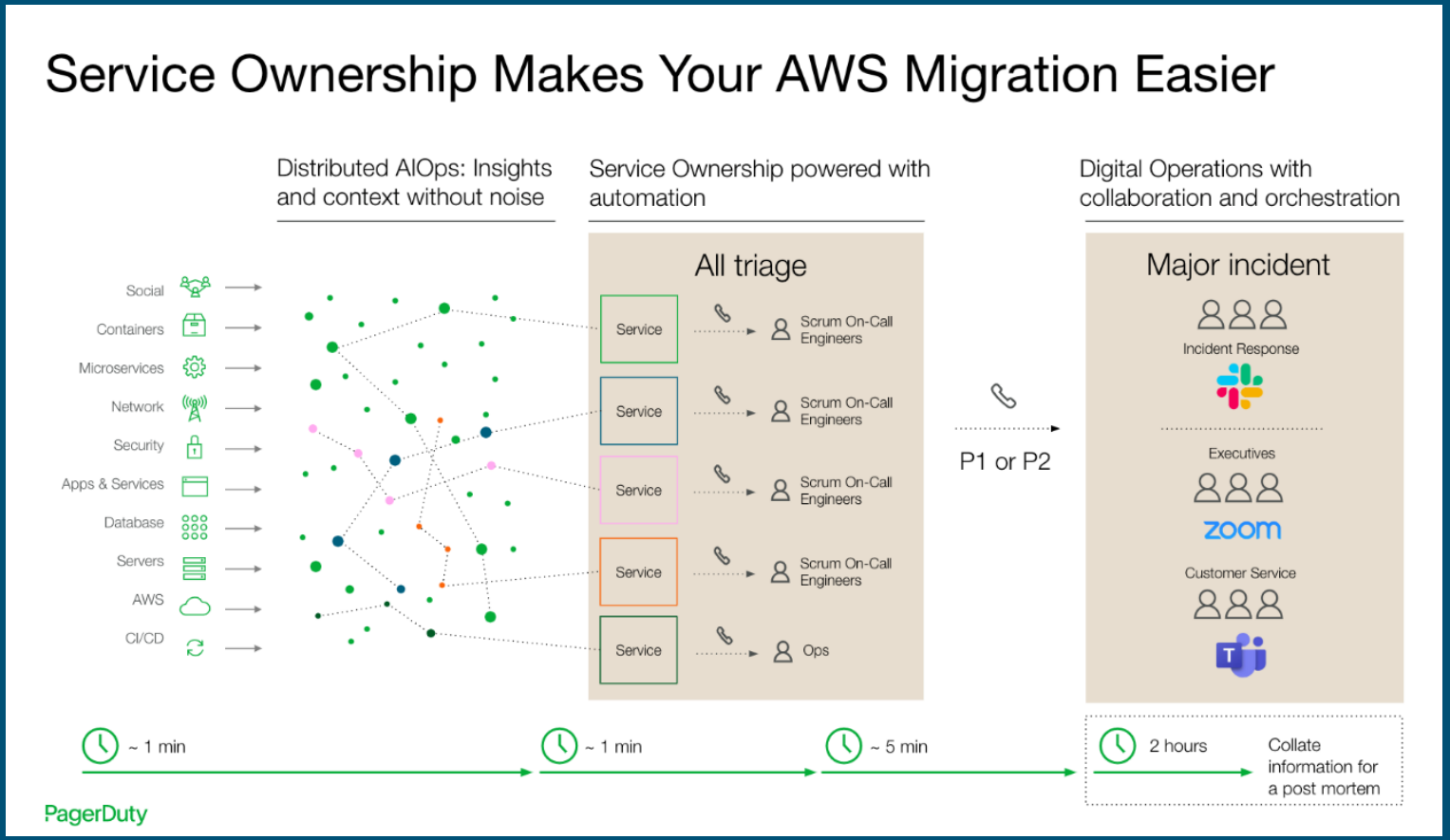
What is service ownership?

Service ownership is an operating model where the people closest to the technology from a design and implementation perspective are responsible for the entire product development lifecycle. “You service it, you own it” is a popular phrase associated with this methodology.

Service ownership, in its simplest form, empowers engineers to be responsible for their code in production, leading to higher-quality software. Realigning teams to this model gives engineers more control over their work, creates accountability for the quality of their code, and results in faster, more orchestrated responses to incidents and downtime.

What is a service?

A service is a discrete piece of functionality that provides value and is wholly owned by a team.

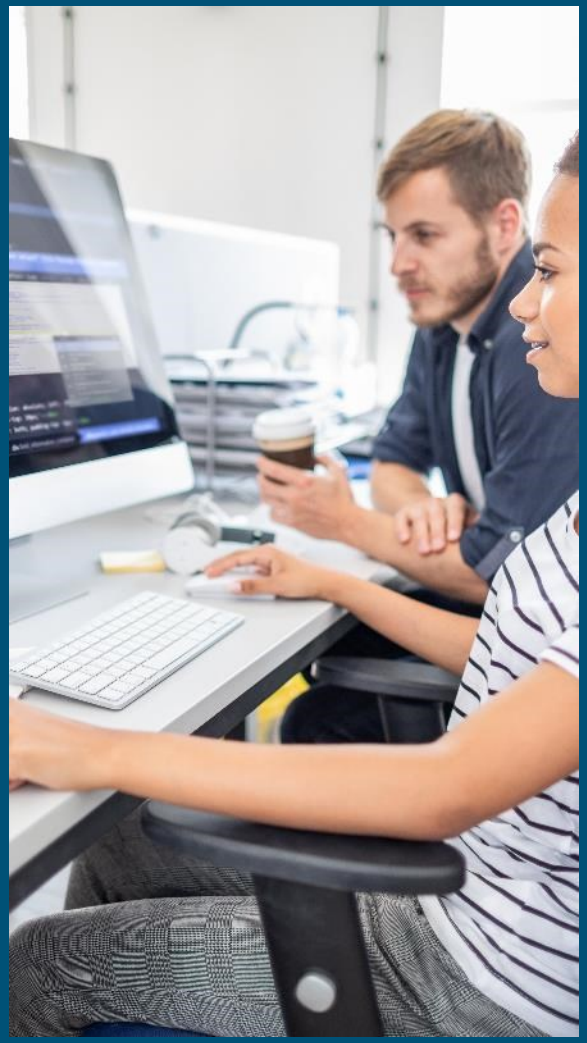


Three key benefits of having developers service their own code

1. **Developers who own their code have a greater impact on the end-user experience**. Service ownership closes the gap between the purpose and impact of a developer's work for both the business and customers. They enjoy more autonomy and responsibility without feeling like they're being restricted, which in turn allows them to focus on building great products. Happier employees result in stronger output—which affects a business' bottom line.

2. **Developers who service their code are more accountable for it in production, which can drive down mean time to resolution and ensure service continuity amidst organizational changes.** When developers are responsible for fixing issues in their own code, a quality control loop is paired with a personal incentive to make sure the code always works. This has another effect: driving down the mean time to resolution. A key feature of service ownership often involves putting teams on call for the applications and services they own, putting developers first in line when things go wrong. Faster resolution isn't only good for minimizing impact to the customer and the business, it is also beneficial for the technical teams themselves. When it's clear who owns the code, fewer staff are required to troubleshoot an issue.

3. **Developers who service their code experience more job satisfaction.** Engineers are, by nature, people who enjoy solving problems, and empowering them to find their own answers provides autonomy and professional development. A key element of job satisfaction for engineers is having opportunities to advance their skills—and allowing them to problem solve will keep their skills up to date.



Navigating a cultural shift

For an organization to successfully adopt service ownership, it's not enough for engineers to simply take responsibility for their code—a cultural shift toward blamelessness is also required. Blamelessness eradicates a culture of fear and instead embraces risk—and inevitable mistakes—as a positive pathway to learning. Come to this shift from a position of compassion, shared responsibility, and trust. Individuals need to be empowered to make decisions in an efficient manner and should not fear being blamed for an incident.

Making a cultural shift to blamelessness requires organization-wide buy-in for service ownership to be truly embraced. Such a cultural change takes time and requires buy-in from the top down, so starting small by bringing this shift to one team will set the example upfront that incidents are not causes for blame. This approach can help set your team up for success.

A note for gaining buy-in from the greater organization

For companies transitioning to service ownership from a traditional operating model, there will likely be resistance from a central operations team.

As with all change, there's a fear of what the tradeoffs may be. Early adopters of service ownership tend to navigate the following topics with their internal partners:

Visibility and Control
When presented with the idea of service ownership, IT partners may have a knee-jerk reaction and think, "If I do something differently, I will suddenly lose visibility into X." Contrary to what they may think, when moving workloads to the cloud, central IT organizations often find that they don't have less visibility. Rather, the nature of how they maintain visibility and pipeline control changes from their previous setup. It also helps them keep up with the pace of change and offers some solutions for consolidating or reducing noise.

Productivity
How much time do your teams spend on manual tasks? How often do they have to dig through old or outdated documentation and figure out how to address incidents in an ad-hoc manner? Having tools, particularly ones that leverage automation to unify processes instead of passing playbooks back and forth will help streamline and reduce manual work. This will give your teams time back to focus on innovation and driving the business forward. It encourages buy-in particularly as an organization scales; automation lets teams scale remediation efforts without increasing workflow disruption, allowing more time for innovation.

Security and Governance
Every company is going to approach this a little differently, but it's possible to find a middle ground for building processes that address valid security and governance considerations. From the start, it's important to emphasize that these efforts require shared responsibility and trust. One advantage of AWS is that it offers developers a large library of tools to integrate security and governance practices from the ground up—making it a strong option for a culture of service ownership.

Getting started: Seven best practices for activating service ownership

Based on the joint methodology of AWS and PagerDuty, there are seven ways organizations can get started with service ownership.

1

Stay agile

Agile methodologies are essential when implementing a culture of service ownership. Work is often unpredictable and agile methodologies can help teams stay the course toward long-term goals, even when the unexpected occurs.

2

Set yourself up for success

Greenfield projects eschew the past and start free from inherited technical and organizational debt. It's simpler to begin anew and design a world free from legacy restrictions. However, if the goal is to prove organizational value, it may better serve a long-term transformational initiative if organizations opt for an achievable brownfield project.

Here's a simple test: Are you likely to hear the phrase, "That's never going to work here," in your organization if you propose something new? If so, consider selecting a more achievable brownfield project to start with.

3

Don't play the blame game

Mistakes will be made. Digital transformation and service ownership are about learning new models and adapting them to the needs of the organization. Individuals should be empowered to make decisions and to experiment, without the fear of blame or retribution if they've made the wrong choice. Instead, errors should be framed as a learning opportunity.

A culture of psychological safety has a strong connection to high performance among software delivery teams. For in-depth tips on creating a culture of blamelessness, read PagerDuty's Postmortem Ops Guide.

4

Practice, practice, practice

As with any new skill, practicing in low-risk settings builds the confidence teams need when incidents happen. For teams that are new to owning production services, simulating incidents is a great way for them to become accustomed to managing those situations when they occur in real life.

5

Right-size your teams

In the early days of Amazon, Jeff Bezos set a rule: Teams shouldn't be larger than what two pizzas can feed, no matter how large a company gets. Setting this rule of small teams meant individuals spent less time providing status updates to each other and more time actually getting stuff done. It also gave team members more time to focus on continuous improvement.

6

Empower teams

Don't just give teams responsibility without authority. To truly own their code, they need to be empowered to make changes without waiting for top-down direction or permission. Another way to empower teams is to make sure they aren't stifled by process; if they truly own the service then they should make the decisions about what languages and tools they use—and even what features are included.

7

Enable teams

Provide a platform that teams can build on top of. AWS two-pizza teams are capable of "you build it, you run it" because they have the full support of the AWS Cloud. This allows teams to focus on adding business value as opposed to the heavy lifting of managing and maintaining servers. Leveraging the cloud also provides a safe space for engineers to experiment and upskill themselves.

Service ownership is a shared responsibility

Service ownership is a shared responsibility across cross-functional parts of an organization. Engineers who write code aren't the only people accountable for ownership of a service. It requires whole-organization collaboration.

For more information and resources, visit PagerDuty's service ownership page.

To see how organizations can empower their developers to service their code, visit https://www.pagerduty.com or start a 14-day free trial today.