

Brief: Put Developers On The Front Lines Of App Support

DevOps Culture Begins With Visibility And Responsibility

by [Kurt Bittner](#) and [Eveline Oehrlich](#)

with [Christopher Mines](#) and Dominique Whittaker

WHY READ THIS BRIEF

The separation of development (Dev) and operations (Ops) has created inherently dysfunctional relationships that mere good intentions cannot mend. At the root of the problem are their different objectives: Developers are rewarded for making new things but not on quality or operational stability, while operations is rewarded for operational stability and predictability. Within these narrow purviews, each regards the other as a barrier to success, and neither is rewarded for achieving the customer experiences that drive business results. This brief report outlines a first step for application development and delivery (AD&D) professionals in creating a culture of shared success: putting developers on the help desk, where they will create a critical connection between their work and the customer experiences that result.

SEPARATION OF DEV AND OPS CREATES DYSFUNCTION

Separation of Dev and Ops may have made sense for AD&D pros at a time when releases happened only every 12 to 18 months. But in a world of continuous delivery, the separation robs developers of important feedback on the quality of their work and simultaneously saddles operations with problems it didn't create. Intense Ops frustration results when developers are rewarded for delivering applications that behave poorly in production. At the same time, intense Dev frustration results when Ops' inability to respond puts business commitments at risk. As delivery speeds increase to satisfy customer demands and meet competitive challenges, segregating Dev and Ops actually creates a great deal of suffering:

- **When Dev cares only about coding, Ops and customers suffer.** Developers are frequently measured only on getting software to run on their workstation or in a test environment. They are often incented to focus on implementing new features, with an assumption that if something runs in the testing environment, it will run in production and that the things that already run will keep running. The unfortunate reality is that testing is not the same as production; testers are not the same as real customers; and even when new features work, they may break old ones. When developers lack a full life-cycle view, Ops usually pays the price, but the real losers are customers who experience unstable, ineffective software that doesn't meet their needs.
- **When Ops cares only about stability, innovation and customers suffer.** Ops is often rewarded for maintaining stability and minimizing change, but change is essential to innovation. When Ops is overly cautious, everyone loses. Catastrophic incidents may be avoided, but without innovation, the competitive vigor of the company gradually seeps away. Innovation and stability are not incompatible, but they require both Dev and Ops to change the way they work together to improve quality throughout the development cycle.



Headquarters

Forrester Research, Inc., 60 Acorn Park Drive, Cambridge, MA, 02140 USA
Tel: +1 617.613.6000 • Fax: +1 617.613.5000 • www.forrester.com

DevOps Is A Culture, Not Just A Structure

Organizations that create DevOps functions or teams miss the point: It's culture that matters, not structure.

"The essence of DevOps is to design a system in which people are held responsible for the consequences of their actions — and indeed, one in which the right thing to do is also the easiest thing to do." (Jez Humble, vice president of engineering at Chef)¹

Creating a DevOps culture requires organizations to:

- **Make both Dev and Ops aware of each other.** This means rotating developers through operations, or having joint Dev and Ops daily stand-up meetings, or running lunch-and-learn sessions, or just going and grabbing lunch with someone working in a different functional silo. Or all of the above. Above all, get Dev and Ops talking to one another.
- **Make Dev and Ops responsible for the consequences of their actions.** Put developers on customer support calls; make them feel customers' pain resulting from poor releases. Measure them on successful releases, not just features delivered. Measure them on reliability, security, and scalability.²

"Development and operations teams have very different priorities, and you can see that divergence show up in their behaviors. Developers tend not to be as concerned about how hard their apps are to deploy because they're concerned about moving quickly, and ops teams tend not to move as fast because they're concerned about stability and security. To get past this, companies have to find a way to align the interests of these two teams so they aren't sacrificing their own success in order to help the other team succeed." (Luke Kanies, CEO and founder of Puppet Labs)

- **Measure everyone on customer success and business value delivery.** Beyond the consequences of actions, measuring everyone using the same criteria creates common purpose. The most meaningful measure for any organization is customer value delivery: Are their desired outcomes being achieved? Are customers more satisfied with your solutions today than they were yesterday?

PUTTING DEVELOPERS ON CALL CREATES URGENCY AND INSIGHT

Developers mislead themselves when they think that passing all tests means their software is good. Most testing is spotty at best and gives an incomplete picture of application health. Customers are the ultimate arbiters of success, and they experience software only when it is running in production.

“In too many organizations, the developer’s job ends in the test environment. If it worked there, then it’s the operations teams’ job to figure out how to make it work in production. With enough testing and automation, this can work because you can ensure that the test and production environments are basically equivalent, but rarely is this actually the case. A major source of conflict then shows up when the operations teams can’t reproduce the experience developers had in a much more limited and very different test environment.”
(Luke Kanies, CEO and founder of Puppet Labs)

Putting developers on the frontlines of customer support calls gives them the critical insights they need to build better applications that serve customers better. The benefits extend beyond just happier customers:

- **Supporting applications expands knowledge.** Many older applications are held together by a thin thread of tribal knowledge, and that tribe dwindles in number every year. Rotating call duties among developers forces them to learn the hard way, which is, in fact, the only way learning ever happens. Over time, developers learn about aspects of applications they would never otherwise see, and they learn about what makes applications brittle. This helps extend the life of older applications and makes new applications even better.

“Production systems used to be a black box to most people on our team. Only a few developers had any insight whatsoever.” (Nick Goodman, director of platform engineering at Bunchball)

- **Supporting applications creates ownership and responsibility.** Developers are accustomed to making changes without consequences; development and testing environments are forgiving and safe. When those same changes reach production, they encounter unpredictable reality. When developers never see these consequences, it’s easy to become careless. Putting developers on call exposes them to critical feedback.

“If you write code, you’re on call. If you want to take vacations, write good tests.” (Mario Cruz, CIO at Choose Digital)

Once they become aware of the consequences of applications behaving poorly in production, they approach development in a different way:

“Metrics, instrumentation, alerting, logging and error handling suddenly become a natural part of your coding workflow. You start making your software more operable, because you’re the one who has to run it. While software should be easy to operate in any circumstances, it commonly isn’t. When you’re the one having to deal with production issues, that suddenly has a very different appeal.” (Mathias Meyer, Travis CI)³

■ **Supporting applications causes proactive quality and customer-satisfaction thinking.**

Understanding how customers experience their applications makes developers more sensitive to making that experience the best it can possibly be. It also tends to increase job satisfaction when developers know their work is valued. Over time, production incidents decline and customer satisfaction increases as application quality increases.

“Now that the entire team is on call, our front-end developers are really excited when an alert goes off because they want to be the first one to hack it. And it is making our developers better because they have a holistic view of our system.” (Nick Goodman, director of platform engineering at Bunchball)

- **Supporting applications forces thinking end-to-end.** As developers become more sensitive to what it takes to support an application, they start to modify their applications accordingly, instrumenting them for monitoring and incident diagnosis, simplifying application startup and shut-down, and improving operational support capabilities. Application support costs go down, while the stability and reliability of the applications goes up.

“The more time that developers spend working in production on operations issues with operations staff, the more that they will learn about what it takes to design and build a real-world system. Hopefully they will take this and design better, more resilient systems with more transparency and more consideration for support and admin requirements.” (Jim Bird)⁴

RECOMMENDATIONS

LET DEVELOPERS SUPPORT THEIR OWN APPS TO START YOUR DEVOPS JOURNEY

Even if you do nothing else, putting developers on call will help to improve your results. AD&D pros should start the journey with these steps:

1. **Choose an application with business-critical application.** Nothing gets everyone's attention like real customers with real problems. Internal applications don't count; they will never create a sense of urgency.
2. **Web or cloud applications are the best candidates.** Web or cloud technologies allow customers to experience changes and fixes at once. Immediate bi-directional feedback provides instant and important insight that developers need to get better.
3. **Measure everyone the same way.** Don't reward Dev for delivering new features and reward Ops for keeping things running. Competing goals are at the heart of the problem. Measure everyone on customer satisfaction, which is influenced by new features but also by how well the entire solution works.⁵

4. **Rotate on-call responsibilities, and make everyone take a turn.** Don't dump being on call on the new hires or on noncritical resources. Everyone needs to experience the pain of being on call for the organization to learn. And make sure everyone expects the escalations: "If your code breaks, expect a call."

SUPPLEMENTAL MATERIAL

Companies Interviewed For This Report

Atlassian	Puppet Labs
Bunchball	Rally Software
Chef	ThoughtWorks
Choose Digital	Travis CI
Etsy	VictorOps
Onshape	

ENDNOTES

- ¹ Source: Jez Humble, "There's No Such Thing as a 'Devops Team'" Continuous Delivery, October 19, 2012 (<http://continuousdelivery.com/2012/10/theres-no-such-thing-as-a-devops-team/>).
- ² Source: Jez Humble, "There's No Such Thing as a 'Devops Team'" Continuous Delivery, October 19, 2012 (<http://continuousdelivery.com/2012/10/theres-no-such-thing-as-a-devops-team/>).
- ³ Source: Mathias Meyer, "On Pager Duty," Paper Planes, January 2, 2013 (<http://www.paperplanes.de/2013/1/2/on-pager-duty.html>).
- ⁴ Source: Jim Bird, "Developers working in Production. Of course! Maybe, sometimes. What, are you nuts?" Java Code Geeks, January 21, 2014 (<http://www.javacodegeeks.com/2014/01/developers-working-in-production-of-course-maybe-sometimes-what-are-you-nuts.html>).
- ⁵ To better understand which performance metrics I&O organizations must have in their balanced scorecard to transform both the IT and BT agendas, see the January 16, 2015, "[Shift From Chronic To Supersonic Performance With The I&O Transformation BSC](#)" report.