MAKING WORK SUSTAINABLE

To ensure that software delivery performance is not achieved through brute force or at the expense of the mental health of your team, our project investigated both burnout on teams and how painful the deployment process is. We measured these because we know they are important issues in the technology industry that contribute to illness, attrition, and millions of dollars of lost productivity.

DEPLOYMENT PAIN

The fear and anxiety that engineers and technical staff feel when they push code into production can tell us a lot about a team's software delivery performance. We call this deployment pain, and it is important to measure because it highlights the friction and disconnect that exist between the activities used to develop and test software and the work done to maintain and keep software operational. This is where development meets IT operations, and it is where there is the greatest potential for differences: in environment, in process and methodology, in mindset, and even

in the words teams use to describe the work they do.

Our experience in the field and our interactions over the years with professionals building and deploying software kept highlighting the importance and salience of deployment pain. Because of this, we wanted to investigate deployment pain to see if it could be measured and, more importantly, if it was affected by DevOps practices. We found that where code deployments are most painful, you'll find the poorest software delivery performance, organizational performance, and culture.

The Benefits of Continuous Delivery at Microsoft

Microsoft engineering is one example of engineering teams feeling the benefits of continuous delivery. Thiago Almeida is a Senior Software Development Engineer Lead at Microsoft who drives cloud computing, open source, and DevOps practices on the Azure team. He spoke about the additional benefits of continuous delivery practices to his team, saying, "You may think that all of the benefits [are] going to your customers, but even inside of your company . . . [there are benefits]."[1] Before implementing the technical practices and discipline of continuous delivery on the Bing team, engineers reported work/life balance satisfaction scores of just 38%. After implementing these technical practices, the scores jumped to 75%. The difference is striking. It means the technical staff were better able to manage their professional duties during work hours, they didn't have to do deployment processes manually, and they were able to keep the stresses of work at work.

While deployment pain can be an indication that software development and delivery is not sustainable in your organization, it is also a concern when development and test teams have no idea what deployments are like. If your teams have no visibility into code deployments—that is, if you ask your teams what software deployments are like and the answer is, "I don't know . . . I've never thought about it!"—that's another warning that software delivery performance could be low, because if developers or testers aren't aware of the deployment process, there are probably barriers hiding the work from them. And barriers that hide the work of deployment from developers are rarely good, because they isolate developers from the downstream consequences of their work.

We often have developers, and especially operations professionals, ask us, "What can be done to relieve deployment pain and improve the work of technical staff?" To answer this question, we included deployment pain in our research in 2015, 2016, and 2017. Based on our own experiences in software development and delivery and our time spent talking to people working with systems, we created a measure to capture how people feel when code is deployed. Measuring deployment pain ended up being relatively straightforward: we asked respondents if deployments were feared, disruptive in their work, or, in contrast, if they were easy and pain-free.

Our research shows that improving key technical capabilities reduces deployment pain: teams that implement comprehensive test and deployment automation; use continuous integration, including trunk-based development; shift left on security;

effectively manage test data; use loosely coupled architectures; can work independently; and use version control of everything required to reproduce production environments decrease their deployment pain.

Put another way, the technical practices that improve our ability to deliver software with both speed and stability also reduce the stress and anxiety associated with pushing code to production. These technical practices are outlined in Chapters 4 and 5.

Statistical analysis also revealed a high correlation between deployment pain and key outcomes: the more painful code deployments are, the poorer the IT performance, organizational performance, and organizational culture.

How Painful Are Your Deployments?

If you want to know how your team is doing, just ask your team how painful deployments are and what specific things are causing that pain.

In particular, be aware that if deployments have to be performed outside of normal business hours, that's a sign of architectural problems that should be addressed. It's entirely possible—given sufficient investment—to build complex, large-scale distributed systems which allow for fully automated deployments with zero downtime.

Fundamentally, most deployment problems are caused by a complex, brittle deployment process. This is typically the result of three factors. First, software is often not written with

deployability in mind. A common symptom here is when complex, orchestrated deployments are required because the software expects its environment and dependencies to be set up in a very particular way and does not tolerate any kind of deviation from these expectations, giving little useful information to administrators on what is wrong and why it is failing to operate correctly. (These characteristics also represent poor design for distributed systems.)

Second, the probability of a failed deployment rises substantially when manual changes must be made to production environments as part of the deployment process. Manual changes can easily lead to errors caused by typing, copy/paste mistakes, or poor or out-of-date documentation. Furthermore, environments whose configuration is managed manually often deviate substantially from each other (a problem known as "configuration drift"), leading to significant amounts of work at deploy time as operators debug to understand configuration differences, potentially making further manual changes that add to the problem.

Finally, complex deployments often require multiple handoffs between teams, particularly in siloed organizations where database administrators, network administrators, systems administrators, infosec, testing/QA, and developers all work in separate teams.

In order to reduce deployment pain, we should:

- Build systems that are designed to be deployed easily into multiple environments, can detect and tolerate failures in

their environments, and can have various components of the system updated independently
- Ensure that the state of production systems can be reproduced (with the exception of production data) in an automated fashion from information in version control
- Build intelligence into the application and the platform so that the deployment process can be as simple as possible

Applications designed for a platform-as-a-service, such as Heroku, Pivotal Cloud Foundry, Red Hat OpenShift, Google Cloud Platform, Amazon Web Services, or Microsoft Azure, can typically be deployed using a single command.[2]

Now that we've discussed deployment pain and covered some strategies to counteract it, let's move on to burnout. Deployment pain can lead to burnout if left unchecked.

BURNOUT

Burnout is physical, mental, or emotional exhaustion caused by overwork or stress—but it is more than just being overworked or stressed. Burnout can make the things we once loved about our work and life seem insignificant and dull. It often manifests itself as a feeling of helplessness, and is correlated with pathological cultures and unproductive, wasteful work.

The consequences of burnout are huge—for individuals and for their teams and organizations. Research shows that stressful jobs can be as bad for physical health as secondhand smoke (Goh et al. 2015) and obesity (Chandola et al. 2006). Symptoms of

burnout include feeling exhausted, cynical, or ineffective; little or no sense of accomplishment in your work; and feelings about your work negatively affecting other aspects of your life. In extreme cases, burnout can lead to family issues, severe clinical depression, and even suicide.

Job stress also affects employers, costing the US economy $300 billion per year in sick time, long-term disability, and excessive job turnover (Maslach 2014). Thus, employers have both a duty of care toward employees and a fiduciary obligation to ensure staff do not become burned out.

Burnout can be prevented or reversed, and DevOps can help. Organizations can fix the conditions that lead to burnout by fostering a supportive work environment, by ensuring work is meaningful, and ensuring employees understand how their own work ties to strategic objectives.

As in other fast-paced, high-consequence work, software and technology is plagued by employee burnout. Technology managers, like so many other well-meaning managers, often try to fix the person while ignoring the work environment, even though changing the environment is far more vital for long-term success. Managers who want to avert employee burnout should concentrate their attention and efforts on:

- Fostering a respectful, supportive work environment that emphasizes learning from failures rather than blaming
- Communicating a strong sense of purpose
- Investing in employee development
- Asking employees what is preventing them from achieving

their objectives and then fixing those things
- Giving employees time, space, and resources to experiment and learn

Last but not least, employees must be given the authority to make decisions that affect their work and their jobs, particularly in areas where they are responsible for the outcomes.

COMMON PROBLEMS THAT CAN LEAD TO BURNOUT

Christina Maslach, a professor of psychology at the University of California at Berkeley and a pioneering researcher on job burnout, found six organizational risk factors that predict burnout (Leiter and Maslach 2008):[3]

1. Work overload: job demands exceed human limits.
2. Lack of control: inability to influence decisions that affect your job.
3. Insufficient rewards: insufficient financial, institutional, or social rewards.
4. Breakdown of community: unsupportive workplace environment.
5. Absence of fairness: lack of fairness in decision-making processes.
6. Value conflicts: mismatch in organizational values and the individual's values.

Maslach found that most organizations try to fix the person and ignore the work environment, even though her research shows that fixing the environment has a higher likelihood of success. All of the risk factors above are things that management and organizations have the power to change. We also refer the reader to Chapter 11 for more on the importance and impact of leadership and management in DevOps.

To measure burnout, we asked respondents:

- **If they felt burned out or exhausted.** Many of us know what burnout feels like, and we're often exhausted by it.
- **If they felt indifferent or cynical about their work, or if they felt ineffective.** A classic hallmark of burnout is indifference and cynicism, as well as feelings that your work is no longer helpful or effective.
- **If their work was having a negative effect on their life.** When your work starts negatively impacting your life outside of work, burnout has often set in.

Our research found that improving technical practices (such as those that contribute to continuous delivery) and Lean practices (such as those in Lean management and Lean product management) reduce feelings of burnout among our survey respondents.

HOW TO REDUCE OR FIGHT BURNOUT

Our own research tells us which organizational factors are most

strongly correlated with high levels of burnout, and suggests where to look for solutions. The five most highly correlated factors are:

1. **Organizational culture.** Strong feelings of burnout are found in organizations with a pathological, power-oriented culture. Managers are ultimately responsible for fostering a supportive and respectful work environment, and they can do so by creating a blame-free environment, striving to learn from failures, and communicating a shared sense of purpose. Managers should also watch for other contributing factors and remember that human error is never the root cause of failure in systems.
2. **Deployment pain.** Complex, painful deployments that must be performed outside of business hours contribute to high stress and feelings of lack of control.[4] With the right practices in place, deployments don't have to be painful events. Managers and leaders should ask their teams how painful their deployments are and fix the things that hurt the most.
3. **Effectiveness of leaders.** Responsibilities of a team leader include limiting work in process and eliminating roadblocks for the team so they can get their work done. It's not surprising that respondents with effective team leaders reported lower levels of burnout.
4. **Organizational investments in DevOps.** Organizations that invest in developing the skills and capabilities of their teams get better outcomes. Investing in training and

providing people with the necessary support and resources (including time) to acquire new skills are critical to the successful adoption of DevOps.

5. **Organizational performance.** Our data shows that Lean management and continuous delivery practices help improve software delivery performance, which in turn improves organizational performance. At the heart of Lean management is giving employees the necessary time and resources to improve their own work. This means creating a work environment that supports experimentation, failure, and learning, and allows employees to make decisions that affect their jobs. This also means creating space for employees to do new, creative, value-add work *during the work week*—and not just expecting them to devote extra time after hours. A good example of this is Google's 20% time policy, where the company allows employees 20% of their week to work on new projects, or IBM's "THINK Friday" program, where Friday afternoons are designated for time without meetings and employees are encouraged to work on new and exciting projects they normally don't have time for.

A point worth mentioning is the importance of values alignment and its role in fighting burnout. When organizational values and individual values aren't aligned, you are more likely to see burnout in employees, particularly in demanding and high-risk work like technology. We have seen this all too often, and the effects are unfortunate and widespread.

We think the opposite is more promising and actionable: when organizational values and individual values are aligned, the effects of burnout can be lessened and even counteracted. For example, if an individual strongly values environmental causes, but the organization dumps waste into nearby rivers and spends money to lobby their government representatives to allow this to continue, there will be a lack of alignment. This individual will likely be much happier working for an organization with a strong commitment to corporate social responsibility in green initiatives. This is an area of potential impact that organizations neglect at their own peril. By aligning organizational values with individual values, employee burnout can be reduced. Imagine the effects on employee satisfaction, productivity, and retention. The potential value to organizations and the economy is staggering.

It is important to note that the organizational values we mention here are the real, actual, lived organizational values felt by employees. If the organizational values felt by employees differ from the official values of the organization—the mission statements printed on pieces of paper or even on placards—it will be the everyday, lived values that count. If there is a values mismatch— either between an employee and their organization, or between the organization's stated values and their actual values —burnout will be a concern. When there is alignment, employees will thrive.

In summary, our research found evidence that technical and Lean management practices contributed to reductions in both burnout and deployment pain. This is summarized in Figure 9.1. These findings have serious implications for technology

organizations: not only do investments in technology make our software development and delivery better, they make the work lives of our professionals better.

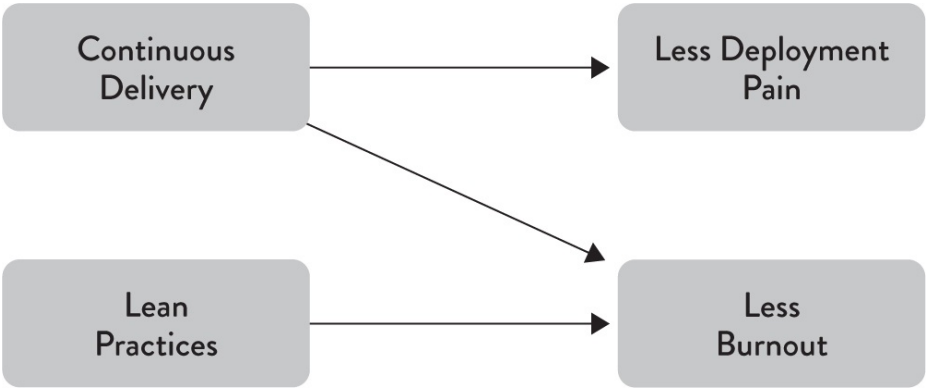

Figure 9.1: Impacts of Technical and Lean Practices on Work Life

We have discussed the important components of organizational culture and ways to both improve and measure it. We will now turn to details of identity and employee satisfaction—and what it means for technology transformations.

---

[1] https://www.devopsdays.org/events/2016-london/program/thiago-almeida/.

[2] One example of a set of architectural patterns that enable this kind of process can be found at https://12factor.net/.

[3] We note that there are other models of burnout in the literature as well; one notable example is the work of Marie Asberg, senior professor in the Department of Clinical Sciences at the Karolinska Institutet, Sweden. We focused on Maslach's work in our research.

[4] Note that postdeployment pain is also important to watch for. Broken systems that are constantly paging your on-call staff after hours are disruptive and unhealthy.