Culture Eats Strategy for Breakfast

Five Unique Skills of DevOps Leaders

Introduction

It is time for a different type of leader, one with the skills and knowledge to manage in a rapidly changing environment, keeping the goals of the organization in mind. It is time for leaders whose skills focus on DevOps transformations. Here we will discuss several innovative models and approaches to help leaders transform their teams and their organizations.

When New Ways are Needed

We all knew the problems that were there. We all felt they were intractable. Software projects ran for months or years and then over-ran release dates due to a variety of issues. We lived with these issues for years, knowing that there had to be a better way, but not knowing where to turn.

And then Agile Development came along, and improved things. But Agile was simply the beginning. Soon we had the entire line of DevOps practices and principles to take Agile to the next level. Shorter bits of work, frequent deployments, automation of the build and deployment chains. After years – even decades – of struggle, we found a better way.

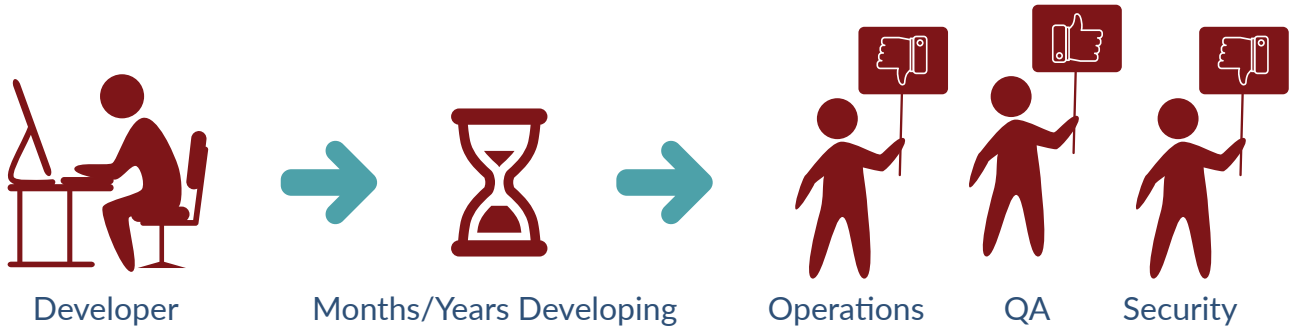
That better way also changed IT culture and longstanding practices that were no longer suitable for the modern pace of business. DevOps transformation is not just about technology and methodology; it's about people and a better way of working. The idea that Developers sit in one place, busily turning out code, while operations waits for it to be complete before testing and deploying was one of the weaknesses of the old system. The changes required to get to Continuous Integration and Continuous Delivery made code chunks smaller and operations/testing an integral part of the early development process. Small teams, iterating frequently, achieving small and more quickly attainable goals became the norm.

Significantly, DevOps doesn't change IT delivers, it changes how IT delivers it. This means that how delivery is managed must change also. The opportunities this presents make it a great time to be a leader in IT, and an even better time to be a DevOps transformational leader.

Key Differences Between DevOps and Traditional IT Management

In the traditional – commonly referred to as waterfall – method of the Software Development Lifecycle, Development, QA, Security, and Operations were completely separate areas that worked on large projects serially.

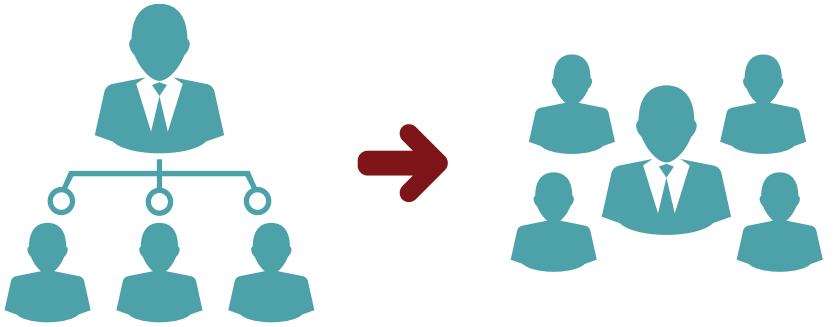
Developers would design and develop an application over months or years, bringing the other groups into the project only relatively far along in the process. Many teams would not involve QA, Security, or Operations until the development phase was “finished”. The problem was that without the other three groups’ input, the project wasn’t actually finished. It lingered, It lingered, often long after the application was supposed to be live in production, finding and fixing issues that the other three groups found.



DevOps conquers this problem by shifting former downstream activities and teams earlier into the development cycle while automating repetitive tasks that are normally time consuming. This approach streamlines a complex process by putting checks in early and not requiring as much manual intervention.

DevOps changes team dynamics, too. Where there used to be separate line managers for each of the given areas of IT (the silos), a servant leader is now responsible for members of each cross-functional product team, for enabling them, and for helping them work together toward the shared goal of providing value to the business through software.

Transformational Leadership inspires others and clears the way to a new way of thinking and behaving. Leaders find out what is needed to get the job done, and works to make certain roadblocks are removed and requests are expedited. While people management and sanity checks are still a part of leadership, enabling becomes the focus.



As we move up the management chain, the focus is still away from silos and onto cross team functionality. Higher-level IT management needs to make certain the teams are meeting their goals while ensuring that cross-team trends can be used to increase productivity and respond even more quickly to the needs of the business. A new tool or process can be tried out on one team, and cross-team management can then plan to move all teams toward using the tool or process if it is successful. Much like piloting projects in the old model, but more ingrained, and offering more corporate standardization.

1. Culture Change Happens. Strong Leaders Embrace it.

Managing change in a way that maximizes the benefits and minimizes the risks, both in tools and culture, is a critical skill of the DevOps leader. Given the right awareness, tools, and skills to manage such change, DevOps leaders can guide the team down a smooth path to transformation. Transformational leaders are open to a changing mindset and are willing to unlearn or relearn how they and their teams should work.

It's Not All Tools

DevOps aficionados generally fall into one of two camps - Automation or Culture. The people in the Automation camp see the improvements offered by increased automation as the primary focus of DevOps activities because they free up workers to focus on problems and higher level thinking instead of redundant tasks. The Culture advocates believe that the benefits of culture change and the increased communication/collaboration that results is the key to successful DevOps. Experience in the field shows us that no matter which camp you are a member of, eventually and inevitably culture change will happen if DevOps is to be successful. The processes for doing software development and deployment change significantly, and the involvement of different groups is fundamentally changed. How the culture transforms is largely a direct result of the leadership that guides the change.

Manage Change for the Best Result

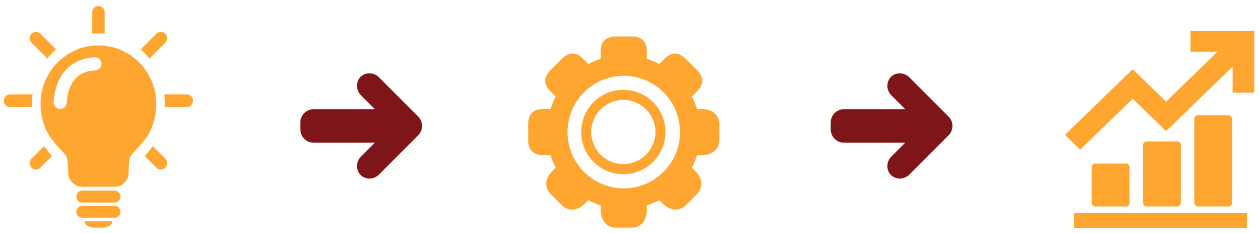
A strong DevOps leader can help shape the direction of the resulting cultural transformation. And strong leadership takes some planning. You can't automate cultural change and you can't mandate people to think or act a different way. You can plan for more interaction between developers and operations to encourage better understanding and collective solutions. Teams will be multi-talented, composed of specialists from multiple disciplines who will work together on common goals, language and practices. Cross-team reuse, of both tools and code, will need to be facilitated by managers with a broader view of the DevOps culture. People will follow leaders who inspire them.

Streamlining Takes Communications

The act of finding the bottlenecks in IT systems requires understanding between normally disparate groups of staff. Whether management is prepared or not, communications between those groups will increase, as will understanding. But this communication needs a champion, because it does not grow evenly throughout the teams. Someone must actively break down barriers and encourage communications. This is one of the key points of DevOps. A better understanding of what causes issues that cross functional boundaries is a necessary precursor to eliminating those bottlenecks. And a strong DevOps leader spends time and effort enabling communications across the team.

2. Determine Your Value Streams

Knowing at a granular level what is actual work, and what is lost time waiting for actual work to occur is critical to streamlining IT operations. A strong DevOps leader needs to understand how to map value streams from idea to realization. The leader then needs to understand how to maximize the value stream and create the type of changes necessary to keep delivering value on an ongoing basis.



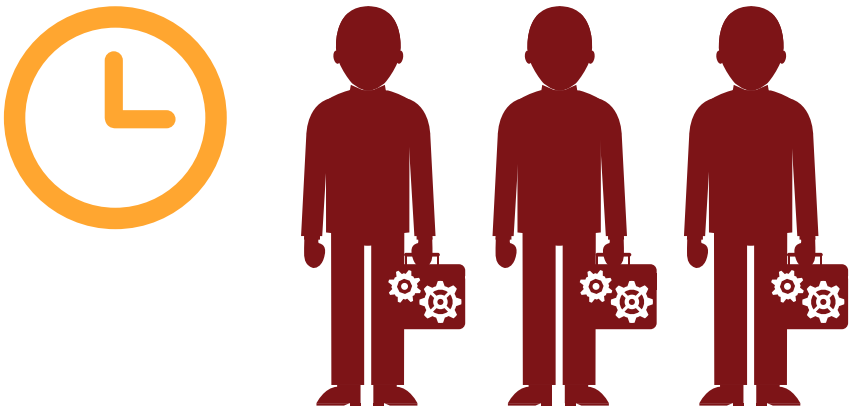
Value Streams: A Holistic Approach

Value stream mapping is a lean manufacturing approach that in DevOps usage attempts to optimize the flow of IT processes. Think of it as the time measurement of the Software Development Life Cycle (SDLC). The idea in terms of software shops is to determine where time is lost because processes aren't streamlined, or because there is manual intervention, or there is a pause waiting for information about the project. Consider the case of an updated project needing a specific library that Ops doesn't find out about until deployment fails – Value Stream Mapping would identify that problem.

Value Stream Mapping covers the entire SDLC, giving a much more comprehensive view of potential issues and bottlenecks than the traditional IT silo approach generally offers. It also identifies where and when each team or competency contributes to the delivery of customer value in the form of software.

Find What's Important

Mustafa Kapadia, Developer of DevOps Institute's DevOps Leader Certification Course, likes to compare Value Stream Mapping to a line at the bank. You can wait for half an hour to conduct 30 seconds of business. That waiting time is what Value Stream Mapping identifies as waste in the SDLC, allowing staff to focus on reducing or eliminating the parts of the SDLC from coding through deployment that hold everything up waiting for the small important bit of work. A DevOps Leader understands how to identify and mitigate waste in the value stream.



Measuring: Visibility and Traceability

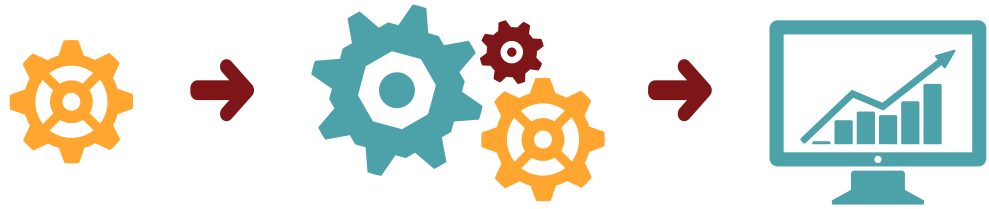
In order to understand where these places are in the process, those who know the details must be available and communicative at the outset, and the ability to map the flow of software through processes must be granted to the team. Executive sponsorship is imperative at this point, because there are always going to be some percentage of employees and managers that find issue with going through the value stream mapping process. The reasons are varied – some teams will believe they already know the weak points in their process, others will prefer the process they know to a stark look and attempts to improve the process, yet others see little value in mapping the streams, and lost time doing so. As the parts of the processes that cause the largest delays are identified, the ability to monitor them as they are improved is imperative. Inserting logging information to the process flow, and monitoring for items that could be improved more or new bottlenecks that appear because old ones have been cleared will offer the broadest level of improvement opportunities (see point 5).

3. Beyond Crossing Silos

One of the hardest parts of managing change can be improving communications and remolding IT to meet the goals and needs of the organization. There are examples and processes available to aspiring DevOps leaders to help guide their transformation. DevOps leaders will see the opportunity these changes represent, and move to make the DevOps structure and function effective.

Silo-Busting is a Start, Not an End

The point of increasing communications and breaking down traditional IT silos is to improve the flow of software development and deployment. But increased communications is not enough, as DevOps is a developmental journey that seeks to improve the overall process and the resulting products. In short, the goal of DevOps is to better serve the business. The output of implementing DevOps is an IT or Product group that creates desired features faster, creates less defects entering into production, and delivers more reliably. The primary beneficiaries of these improvements are those who count on IT to make the business run – business leaders. But do not underestimate the benefits to IT. Less busywork and bug fixing, less waiting around for that one step to occur, more work on new projects because the old are more stable than in the past. These all improve the daily work life of IT.

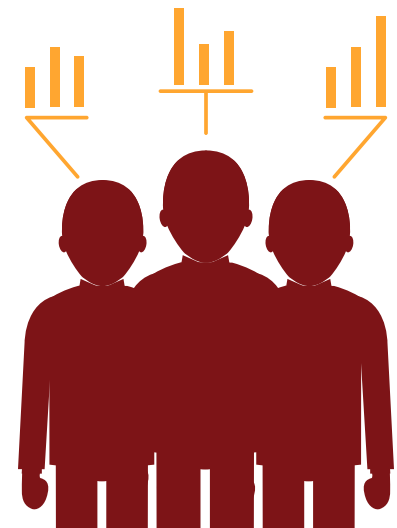


The output of implementing DevOps is an IT or Product group that creates desired features faster, creates less defects entering into production, and delivers more reliably.

Understanding the end goal is important when you consider Conway's Law. Conway's law was created in the 1960s, and essentially says that an organization is constrained to create a system that models their communications structures. From Conway's law, it follows that in order to make a software delivery/deployment mechanism that meets the needs of the business, the organization of IT must be optimized to avoid the very bottlenecks that interfere with its value chains, and must be aligned on business needs. The communication structure must mirror the desired system.

Squads, Tribes, Chapters, and Guilds

Following through on Conway's Law, the organization of IT should reflect the needs of the product and application portfolios that IT is creating. Spotify uses a model that re-organizes the traditional way of structuring IT with the goal of making better software. The overall essence of the Squads, Tribes, Chapters, and Guilds model is that a squad focuses on one area or platform, seeing to, say, the Android client for an application or group of applications. Squads are the equivalent of teams in the traditional IT model. A group of squads that share focus – like all of the UI teams for a family of products – taken together are a tribe. They share issues, and can share processes and ideas. A Chapter is what traditional IT would call the SMEs: The people who have in-depth knowledge of a tool or a specialized competency and can again share knowledge about how to streamline things or new ways to achieve goals. And finally, their unofficial Guilds are groups of people that share an interest. Guilds can focus on anything relative to the company's goals, and allow for information sharing about specific topics that may be outside their normal Squad/Tribe/Chapter.

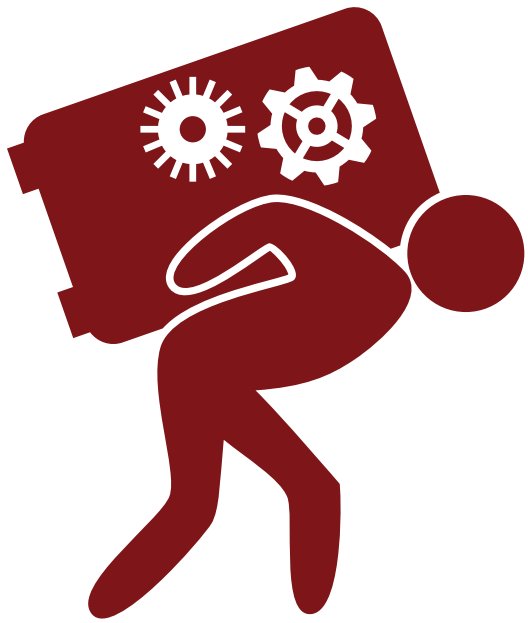


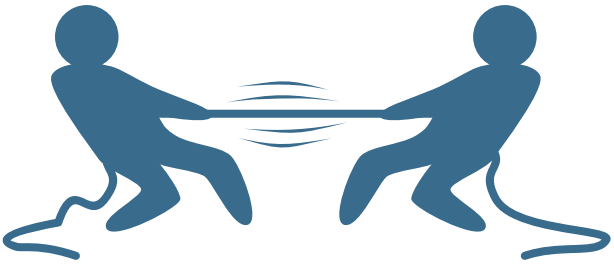
Getting Value From a Diverse Organization

The key to Spotify's organization is that they push decision making and responsibility down to the squad level where possible. Teams are given broad directives and then expected to keep the goals of the organization in mind while resolving issues that crop up on the path to meeting those goals. Flexibility in organization, and autonomy of groups make this structure adaptable to change in the same manner that the business needs to be adaptable to market fluidity. It allows IT to pull in the direction of the business, rather than being that oar dragging a bit behind the boat, pulling it off course with slow response times and inadequate movement. In the DevOps model, if midway through a major project the business environment changes, and business leaders need to change the software being delivered, it is far easier to make the change in stride and truly support the needs of the business than it is in monolithic development/deployment methodologies. Fixing critical issues is also easier. Not that debugging suddenly becomes easier, but there are less fixes occurring, so a critical issue gets immediate attention without other errors having to be ignored.

4. Managing Culture Change

Dealing with the roadblocks any organization presents to change is a key responsibility of the DevOps leader. Change resistance is not something new, nor is it unexplained. DevOps leaders must understand the mechanisms of change resistance, and how to overcome such resistance. To a DevOps leader, the challenge of convincing people that the changes are good for the entire organization is daunting, but an opportunity. The DevOps Leader is as much a guide as a servant leader.





Managing Change

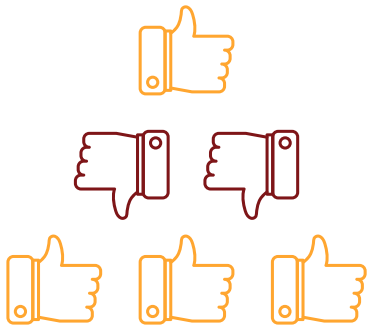
Managing in times of change is never easy. No matter how worthy the goals and how seemingly comprehensive the planning, there are always unexpected consequences and surprising resistance to the new environment. This is a known and well-studied quantity, with Harvard Business Review discussing resistance to change as early as 1969. It is, to some extent, human nature to maintain the status quo, even if that status quo is not working. The idea of the known problems with the existing environment being manageable versus the unknown is one of the more obvious sources of resistance.

When that change is to both the culture and structure of an IT organization, it will take work to make the change effective. Executive sponsorship is critical, as is a willingness to bring problems out into the light, examine them, and find the best possible organizational arrangement to address those issues.

Culture change is a bit of a different creature. Organizational change can be managed by fiat if absolutely necessary, but fiat is never the recommended route. It is much better to get buy-in and input from those that are impacted by the change. But with culture change, even the possibility of change by fiat is absent. The manager in charge of change must be willing and able to convince people that there is a benefit to the entire organization if they increase communications and interaction across the breadth of IT. People have to want to discuss issues and come to a better resolution, and that takes effort on management's part. Communication and collaboration takes time and effort to become mature and regular.

Resistance to Change

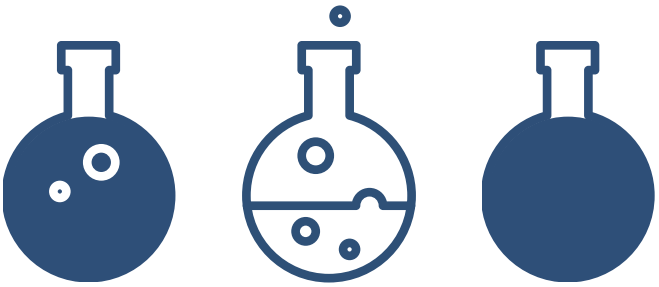
There are a variety of incarnations of resistance to change. At the top level, executives set the tone of the conversation and greatly impact its likelihood of success by their active values. Active values are what they show by word and deed in small, daily interactions. This is opposed to espoused values, what they claim to support and intend. Part of the politics of any organization makes this separation necessary, but both active and espoused values impact the chances of success of any major project, and the executive sponsoring a move to DevOps must be prepared for some speed bumps and failures along the way and committed to some ambiguity as the transition occurs. Those responsible for day-to-day tasks generally see the possibilities of DevOps quickly, as they find that issues everyone knows about will get addressed directly and they'll have more ability to talk to those who also touch the final product.



Resistance most often comes from middle managers who are uncertain of their ongoing role in cultural and organizational transformation. They are used to managing like-skilled teams. They have goals and deadlines that must be met, and upending both team structures and culture will interfere with their ability to deliver. The middle managers need to evolve into DevOps Leaders and that requires training, coaching and communication on how to manage cross-functional, self-organizing teams. Managers need to understand the benefits so that they can effectively lead. They will also need to understand the process and timelines associated with transitioning to the new way of working.

Three Phases of Change

Kurt Lewin's change model identifies the three stages of change as Unfreeze, Change, and Refreeze. For moving to an agile/DevOps environment, we can pin specific actions to these three states that will help us toward our goals.



Unfreeze

- Gain Executive Sponsorship
- Identify what isn't working in the current environment
- Set goals/timelines for existing applications or application groups that will stress what isn't working
- Talk with people to understand goals and concerns around the existing system

Change

- Start making changes to structure
- Talk with people often, involve them in the process, and address their problems/concerns
- Make certain the facts of the changes and the items still being worked out are widely understood
- Empower teams to adapt as needed to meet expectations

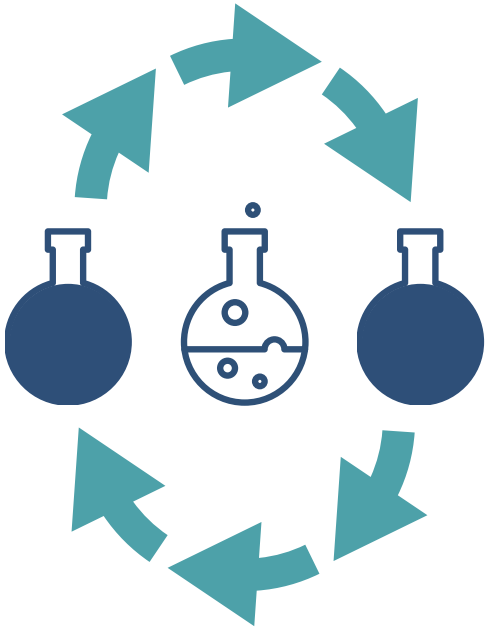
Refreeze

- Formalize the changes in more than just structure
- Encourage squads, tribes, clans, and guilds to interact and reinforce the changes
- Call out those who excelled at adapting and met or exceeded expectations for the changes

This list can be expanded upon, and probably should be customized for the given organization, but it is a starting point to begin implementing change.

5. Iterations

DevOps is not a project. DevOps is a journey. DevOps leaders must be prepared for the long haul, willing to implement in steps, and improving overall IT effectiveness with each new iteration. Seeing the positive changes of each iteration is the goal – and one of the rewards - of the DevOps leader.



This is DevOps, you didn't do it all in the first pass. Lewin's three stages are just a start. The entire organization likely didn't go on hold while the perfect IT structure to meet all of the organizations' needs was slowly and painfully built up. Indeed, that would be the antithesis of DevOps. DevOps is about manageable iterations, and what form that takes varies from organization to organization. Some start with a single application or application portfolio, some change the entire IT organization but prioritize so that the most impactful items are changed in the first go-around, and then another iteration is planned.

Even if the first iteration comes close to meeting all the disparate needs of an organization, the environment – from tools to team makeup to new processes is constantly in flux, and future iterations should be planned to make certain things stay on track. Culture change takes time.

As The Phoenix Project portrayed, there are historical animosities within IT that the first iteration will need to begin breaking down. That makes the first iteration of change the hardest, and future iterations are a little easier because those walls will have been gradually eroded until broken.

Continuous Improvement Applies to Culture Too

And making adjustments to culture matters too. In fact, those historical animosities are mostly cultural. Since it is unlikely the first pass at setting up squads, tribes, clans, and guilds was 100% perfect, looking at them again (and again) is useful. Monitoring the cultural fit as compared to the goals of the organization should be an ongoing task for managers. There will always be snags and pitfalls that are human in nature rather than technical, and those are sometimes the hardest to overcome. Subsequent iterations of cultural transformation should focus on what is off-kilter in the culture at the time of the iteration, and avoid rehashing old issues that have been resolved. Culture change must be institutionalized, one iteration at a time.

In the end, the success of a major DevOps transformation is very heavily dependent upon culture change, and cultural change requires a shift in the way people think and behave. People have to be given time and guidance on new ways and their benefits. Keeping a constant eye on communication structures and bottlenecks (Conway's Law) as well grooming innovative approaches to team organization (the Spotify model) should help people through the transformational process.

Tools Help Shape Culture

Tools have a lot to do with how culture develops in an IT organization. Everyone there is present to get a job done, and the tools in use influence the way that both culture and structure evolve. The most obvious example of this two-way interaction is Jenkins. Jenkins enables nightly builds, automated testing, and many requirements of both Agile and DevOps methodologies. It crosses boundaries and increases the speed of feedback to developers, giving them the opportunity to fix problems with code while the code in question is still fresh in their minds. It also helps standardize deployments, making it easier for Development and Operations to work together. Considering tools and their respective capabilities should be part of any discussions or planning around cultural considerations.

Summary

While it can be a bit daunting and definitely a challenge to lead a team or entire IT organization through a DevOps transformation, the goals are good ones, and tying IT delivery timelines closer to the needs of the business will reduce a long-standing friction between these business leaders and IT. It is crucial for DevOps Leaders to have the knowledge and fortitude to see the change through, and enjoy the benefits to the entire organization that IT realignment offers.

The ideas explored in this document are similar to – and inspired by – the much more thorough treatment of DevOps leadership given by the DevOps Institute's DevOps Leader Certification Course. This course will guide DevOps leaders through innovative approaches such as Value Stream Mapping and the Spotify Squad model in order to lead teams through the cultural transformations required for modern, fast paced IT environments, whether a manager, subject matter expert or consultant, everyone is a DevOps Leader.

DevOps Leader (DOL)℠

Innovative Strategies for Leading Cultural Transformation

DevOps INSTITUTE
THE CONTINUOUS LEARNING COMMUNITY

COURSE OBJECTIVES

The learning objectives for DevOps Leader (DOL) include a practical understanding of:

- The Golden Circle
- Understanding organizational culture and organizational change
- Conway's Law and its influence on DevOps and systems thinking
- Strategies for leading cultural transformations
- Evolving silos into flat, team-based organizations
- Managing conflict
- Creating feedback loops
- Creating learning environments
- Avoiding change fatigue
- Communication and collaboration strategies
- Meaningful metrics
- Resourcing for DevOps
- Demonstrating DevOps Return on Investment (ROI)
- Critical success factors
- Getting started

Find an Education Partner Today

DevOpsInstitute.com