

8 Barriers to DevOps Adoption

By Amit Naudiyal

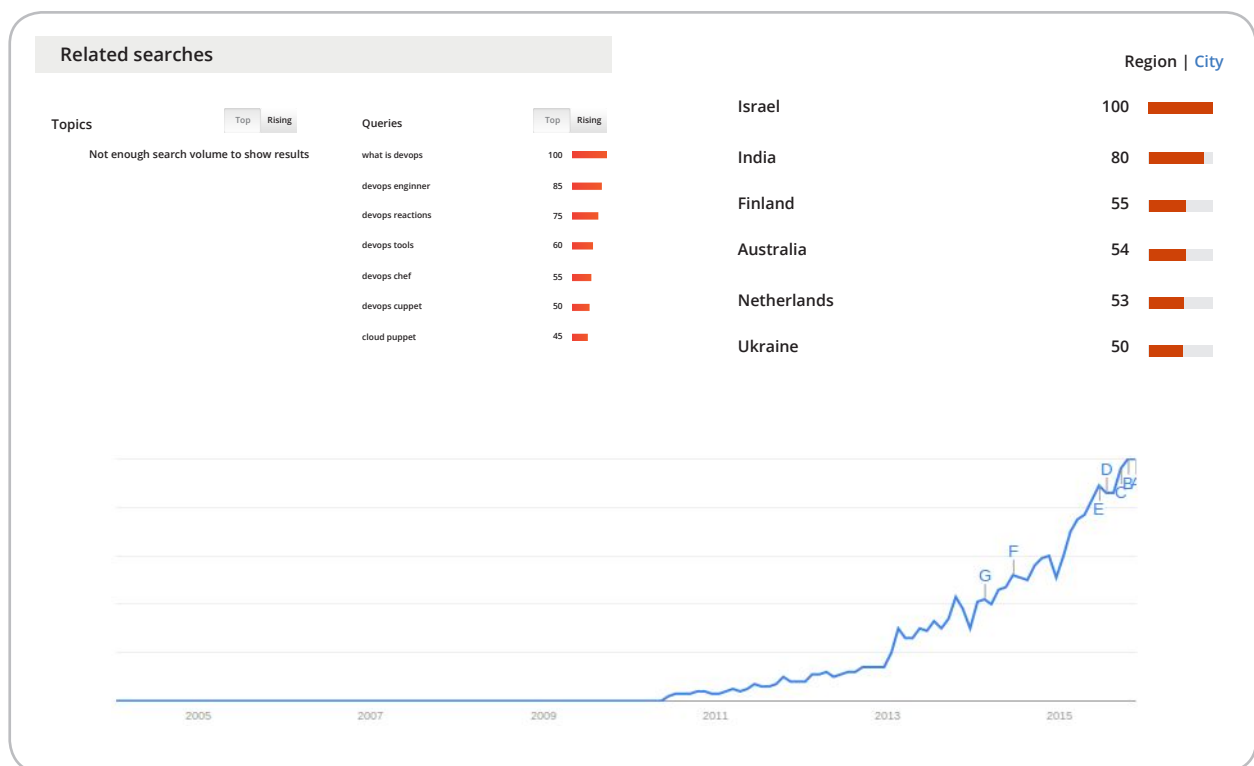


CONTENT

1. Introduction	3
<hr/>	
2. Practices and Principles of DevOps	4
<hr/>	
3. Barriers	5
3.1. Lack of Experimentation	5
3.2. Legacy Systems	6
3.3. Resistance to Change	6
3.4. Fear of Failure	7
3.5. Lack of Buy-in	7
3.6. Silos	8
3.7. Old Habits	8
3.8. Unclear Value	9
<hr/>	
4. About the Author	9
<hr/>	
5. About TO THE NEW	10

1. Introduction

Every organization, may it be a startup or an enterprise, wants to adopt DevOps strategies in their culture, everyday routine and the way they think, act and evolve. It is not surprising that there is 'a noise in the DevOps Signals' coming from every small, medium and big organizations. The term DevOps is gaining a lot of attention worldwide, and this fact is corroborated by Google's claim that it has been searched a lot. Some stats are shown below:



Courtesy: Google

There are many misconceptions about the term DevOps. Many understand the concept pretty correctly but find it hard to implement on their existing IT Infrastructure and organizational behavior..

There exists a wall of confusion, which is transparent yet hazy in demarcating the boundaries between Development and Operations. Due to this ambiguity in understanding, even those who claim to have adopted DevOps as a full fledged strategy fail to understand the distinction between two interconnected concepts i.e., development and operations and often misinterpret the way the other one works.

**I want
Change!**



Development

WALL OF CONFUSION

**I want
Stability!**



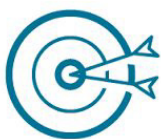
Operations

Development team works with deadlines and pressure to develop faster and better applications with top quality. Often, they find the operations team inefficient and outdated taking a lot of time to prepare test environments. They profess the operations team need to work quickly just like them and thus not become a barrier for their innovation and agility.

On the other hand the development team is seen as the one without an idea about production environment and their applications just do not work efficiently because of their poor development skills, unless they go through regression testing on a series of environmental changes. The operations team would hardly trust a new release as it may put the existing functionality at high risk.

Both these domains and their ideas of thinking and functioning in silos is broken and integrated with the concept of DevOps to ensure frequent, stable, reliable and high quality application/software deliveries. It makes everyone responsible for everything while working together towards a common goal. One cannot claim to have adopted DevOps entirely unless they have a fair understanding of its four pillars : Consistency, Agility, Quality and Scalability.

These four pillars form the basis for the implementation of the practices and principles of DevOps.



Consistency



Agility



Quality



Scalability

2. Practices and Principles of DevOps

Automation: Anything repetitive should be automated. This becomes valid for all of your DevOps activities; be it coding, recoding, testing, deploying or monitoring. A sufficient amount of knowledge, discussion, testing and analysis is required for anything to be automated and once it is implemented you become 80-90% more efficient.

Continuous Integration, Delivery & Deployment: To increase code quality, regularly test your code and automatically get reports, you must start using Continuous Integration that gives you faster, efficient and error-free codes. Continuous Delivery and Deployment ensure decreased code review time and testing time, increased code coverage and efficiency in moving codes from staging to production or other environment.

Version Control: This is quite a common term now and one of the most integral parts of coding. You will anyway need to version your codes to track their changes, do diligence checks, restore them to older state(s), collaborate them between teams and obviously back them up centrally.

Logging, Monitoring and Reporting: If you really want to see what happens when your code runs and simultaneously what happens in & around your system, in your infrastructure & network, you will need LMR. This will give you a complete and a clear picture about what, who, when and how of your implementation.

Configuration Management: Treating your Infrastructure as a code gives you an ease to provision and deploy your servers and applications, configure and tune them for the best possible output, all in a single click or sometimes even without it. This along with automation and version control can help you setup LMR for every possible component.

Cloudification: Sounds new ? But, it's 'The Internet'. It is just any device or application moved to some invisible place that serves you without any downtime. Now you do not have to purchase a device, get some consultation to configure and then feed it regularly. You just need to click on a service and start using it. As cool as a breeze !!

Learn and Experiment: L&E is the key for an organization to improve and learn from failures, identify the best possible tools, practices or processes, bring innovation and improvement in the system. This protects DevOps from unsought risks and helps it rebound to its original state much faster.

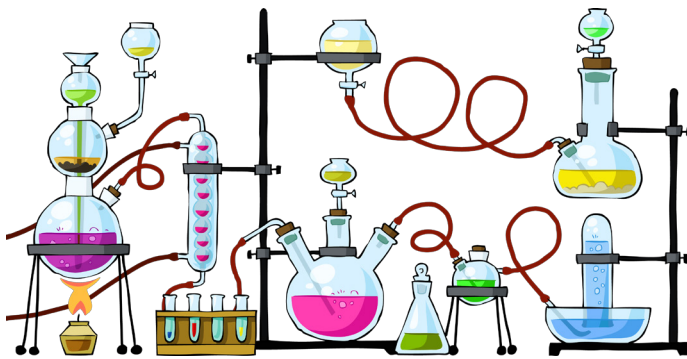
Buy-in Strategy: This thing surely cannot be missed!. Management buy-in is equally important as Employee buy-in. Employee buy-in signifies the commitment of employees towards the vision and goals of the company. Management buy-in is when the stakeholders know the need and importance of above DevOps practices and approaches along with those who execute it.

These tools or approaches will guide you through the path that will give you faster surface to develop, test and deploy the codes, implement requests rapidly resulting in happy customers. You will also have the flexibility to launch and deploy a resilient infrastructure any day, anytime with a single click of button for multiple environments that gives an easy management.

Since it is a complete change in the methodology of the work-think-act approach, people do find some obvious obstacles or barriers that prevent them from going ahead with adopting DevOps. The intent of this paper is to discuss these barriers in detail.

3. Barriers

3.1. Lack of Experimentation



As aforementioned, experimentation builds healthy cultures in winning organizations. However, there are still many organizations that do not experiment at all. They feel better to pay a consultant to make company decisions or to set strategies on the basis of their own experiences and intuitions rather than conducting an experiment to find new results.

“Experimentation requires short-term losses for long-term gains”.

Thus, the need of the hour is to start experimenting and test all the hypothesis you hear or see to adopt DevOps practices and principles. This will give you more confidence on the possibilities of the outcomes than taking suggestions from a consultant.

3.2. Legacy System



Legacy systems (Infra and applications both) are certainly the best systems that many organizations have relied upon as they run their basic essential functions. Some outdated platforms still survive because of the risk and cost associated with replacing them, even if they get the required skills and support.

These systems were not created, keeping DevOps practices in mind, so they cannot easily fit into the DevOps approach. Upgrading these systems also seems daunting since budgets are confined to 'the show' and 'something new'.

The need of the hour is to escape from Alcatraz !! Replace them with or adopt another system that is built to support DevOps practices or move to some external service provider (mainly SaaS) to decrease software development, maintenance efforts and to support agility.

3.3. Resistance to change



With the growing technology, customer needs, economy, opportunities and challenges it becomes inevitable to change and adapt. However, people or the organization need a reason to 'change'. They do not want to be in a state of surprise with the change.

Their resistance to change does not want them to deviate from a set of routines and comfort since they are familiar with how things work. For them, **If It Ain't Broke, Don't Fix It.**

The need of the hour is to learn to accept change as a supporting arm rather than something to be avoided. If you stop experiencing different things, you become stagnant and you will stop your own growth. But at the same time, it becomes important to recognize when and where you need a change.

3.4. Fear of Failure



“70 percent of all changes attempted in organizations fail” and the reasons are lack of knowledge & skills, hidden conflicts & legacy culture. But that is again a statistical term. Organizations do have a fear of imbalancing a stable system by implementing a change and so it becomes nearly impossible for them to think about a complete system and a methodological change.

The only one stopping you is You! Failure is just a unit of measuring your attempt of an action; it never stops you from making another one. The only way to overcome fear is to ‘Live the Fear’, obviously taking along the alternatives, benefits of attempts and contingency plans with you.

3.5. Lack of Buy-In



Although management buy-in is important at different levels of the organization, this DevOps transformation must have buy-in at the leadership level. If you lack this buy-in, there is always a risk of being derailed before it even begins or eventually

a drastic decline in later stages. There should be a willingness at management level to implement changes.

The need of hour is to start involving the management or leadership in early discussions or meetings. Their attention and approval is required for all the decisions and actions you take to adopt any change. Bring facts and data on the table, make delivery on time, keep all the answers ready and go get it !

3.6. Silos



Some organizations work in silos; their different groups or teams do not share information, tasks, priorities, tools etc. with each other. They don't even understand how another team works or functions. They usually avoid, shift or otherwise escape blame during times of crisis. Such structure prevents organizations from growing at faster pace and breeds a narrow mindedness among the workforce.

Break the Silo!! Have large scale employee buy-in to give them an understanding of the organization's goal, team or group objectives & combined initiatives. There must be a unified leadership team that can drive the workforce towards achieving a unified goal.

3.7. Old Habits

**old
habits
die hard**

As discussed already, there exists certain comfort with all the procedure we follow each day and when we find a better and efficient solution, old habits do not let us adapt them easily. Similarly, if you think, you will be able to cross and raise above the barriers, you would require to implement changes in your traditional ways of software building and delivery. The old habits of counting a bug, deploying it on machines in different environments stop the adoption of DevOps practices.

New tools cannot fix old habits. Keep your previous experiences aside when you go for a change. Otherwise, you cannot adopt new things. Set your own goals to achieve milestones by creating a plan, being disciplined and keeping a check on your track.

3.8. Unclear Value



The buzzword DevOps has different understanding in different people. The true value of DevOps is not understood by many people. Believing DevOps as an independent team does not add any value to the organization as it creates a silo again. The failure to understand the purpose of DevOps as to deliver qualified software quicker, does not lead to the transformation desired by an organization.

The need of the hour is to: understand that DevOps transformation does not mean to create a new team. Rather, it is a change in your current way of thinking, planning, collaboration and execution to get a faster, durable, resilient, flexible service or product.

Organizations and the workforce that have the responsibility to adopt and implement DevOps must identify their barriers and start accepting and implementing this 'methodology change'.

I believe you would have found this whitepaper useful and will be able to identify and act on any of these barriers that exist in your organisation today.

Please share your feedback/comments/suggestions with me @ amit.naudiyal@tothenew.com.
Good luck!!

About the Author

Amit Naudiyal

Senior DevOps Engineer
TO THE NEW

Amit is a Linux expert and a subject matter expert. He has vast experience working as a DevOps consultant cum engineer in administration, architecture setup, server handling, security and virtualization. He is passionate about cloud automation and technologies and has smartly automated his usual tasks and processes. He loves playing Counter-Strike.

About TO THE NEW

TO THE NEW is a digital technology company that builds disruptive products and transforms businesses. We leverage the power of experience design, cutting-edge engineering, cloud and analytics led marketing to enable digital transformation.

Our passionate team of 750+ people includes passionate technologists, digital analytics experts, video specialists and creative mavericks who have transformed businesses of more than 300 companies spread across 30 countries worldwide. We take pride in our culture which is driven by passion for making an impact through technology.

Explore More Resources



**DevOps Practices
and Principles to
Improve IT Efficiency**

[Download Whitepaper](#)




**4 Key Tools to
Monitor Docker**

[Download E-Book](#)



 info@tothenew.com

 www.tothenew.com

Lets Connect

