DevOps: Bridging the gap between Development and Operations

Avita Katal
School of Computer Science
University of Petroleum and Energy Studies
Dehradun, India
avita207@gmail.com

Vinayak Bajoria
School of Computer Science
University of Petroleum and Energy Studies
Dehradun, India
bajoriavinayak@gmail.com

Susheela Dahiya
School of Computer Science
University of Petroleum and Energy Studies
Dehradun, India
sdahiya@ddn.upes.ac.in

***Abstract*— DevOps is a cooperation of advancement and activities conceived to weight on correspondence and coordination between them. The fundamental of DevOps is to assist an association with growing and exceed expectations. With its assistance an association can create programming items and administrations. The fundamental utilization of DevOps is to streamline the everyday exercises of an association and accelerate the procedure for auspicious conveyances. Organizations are concentrating on the computerization of procedures along these lines convenient conveyance and quality outcomes are accomplished. Getting the workforce prepared with the most recent innovations and getting ideal work for them have turned into the need of great importance. This survey paper puts forward the required cultural, organizational changes along with automation tools to realize DevOps. It also addresses the common objections raised again DevOps and how to address it.**

Keywords—DevOps, Developers, Operations, tools, Cultural, Organizational, Automation

I. INTRODUCTION

A typical organization or company is made up of various business units like Research and Development, Information Technology (IT), Manufacturing, Finance, Human Resource and many more. IT is the backbone of all the business units and is mostly used by them for free flow of information, communication and software delivery and maintenance required for inner workings of the units. Finance department may need customized software for easy calculation of the GST rates as per the various tax slabs, IT provides all these services. A typical IT business unit of any mid-tier Company consists of Architects, Project and Product Managers, System Owners, Developers, Quality Assurance team, Business Analysts, and Centre of Excellence. In a typical organization all these teams work at various levels of the software delivery pipeline to deliver software to the end client, which can be a third party customer or any other business unit of the organization. However, often these teams are independent silo teams with their own policies and procedures surrounding their stage of deployment time. This leads to various organizational pains like outages, slow IT, and infighting. It also leads to knowledge waste, over production waste, motion waste and many more.

Under the DevOps the goal is to add value and improve the flow in the organization. Development and operation teams are not siloed anymore, they are merged together and engineers work across the entire application lifecycle and are not limited to single function. The focus is to add customer value, eliminate waste, reduction of cycle time, and identification and removal of bottlenecks. DevOps is well explained with the CAMS acronym, Culture Automation Monitoring and Sharing. It is a cultural movement that aligns people, processes, technology, towards a common goal of eliminating waste and increasing value. Tools are used to build together a DevOps capability from an automation standpoint, also tools help to allow sharing i.e. having true feedback loops between the Developer and Operation teams. They can also be used to share ideas. Implementing DevOps can increase the company evaluation, job satisfaction, and promote a culture of sharing and learning.

II. RELATED SURVEYS

One of the essential objectives of this research was to comprehend the adoption patterns of DevOps as organizations advance. Our theory was that there are particular stages in DevOps advancement and explicit practices that empower organizations to scale accomplishment over the silo teams.

Huttermann [1] defined DevOps as a set of practices that streamline the software delivery pipeline, emphasizing the learning by streaming feedback from the production to development and improving the cycle time. However, the definition is incomplete as cultural aspects of DevOps were overlooked but, Huttermann was right to point that DevOps is not a new team or department in the organization. He refers to four key aspects in a blog post by Wilis[2]: culture, automation, measurement and sharing (CAMS). Similar works by DeGrandis[3], Humble and Molesky[4], Limoncelli and Huges[5], Loukides[6], Bang et. al[7] and Roche[8] add to the definition of DevOps. They imply that DevOps is an organizational revolution shifting mindsets from silo teams to large scale collaboration in the organizations. Some of them throw light on the key cultural aspects also like openness, communication, incentive, ownership, respect and trust.

However, most of the above literature focused on completing the definition of DevOps. Puppet Labs survey of 2012[9] taken by over four thousand IT Operation professionals DevOps enables high performance and shipped code thirty times faster and deployments eight thousand times faster than their peers. The survey also showed that DevOps increased reliability and restored services twelve times faster than the peers. In the 2014 survey[10] puppet labs showed the relationship between the organizational performance, IT performance and DevOps practices. It stated that firms with strong IT performance are twice more likely to exceed their profitability, market share and productivity goals. DevOps practices like version control and continuous delivery correlates with strong IT performance. Organizational culture is also an important aspect for both IT and the whole organization itself. Cultural practices like High-trust, shared responsibilities, cross-functional collaboration, learning from failures and new ideas strongly correlate with organizational performance and result in job satisfaction.

In the report of 2015[11] by Puppet Labs, most of the previous findings were discussed in details but they also shed light on Deployment pain points, pathological cultures, unproductive and wasteful work and critical role played by mangers in any DevOps transformation. In the 2016[12] report, focus was one of the most important aspects of DevOps culture: Quality, High-performing organizations spend twenty two percent less time on unplanned work and rework. As a result, they are able to spend twenty nine percent more time on new work, such as new features or code. They are able to do this because they build quality into each stage of the development process through the use of continuous delivery practices, instead of retrofitting quality at the end of a development cycle. Also, they showed that undertaking technology transformation can produce sizeable cost savings for any organization. Moving a step ahead in 2017 survey

report [13] they showed that loosely coupled architecture and teams is the strongest predictor of continuous delivery.

However, all these reports and literature reviews missed a key point 'what is the executive summary of the entire DevOps?'. This survey provides the required executive summary for any person who wants to implement DevOps by understanding what DevOps actually means and the tools that you may need to realize it.

III. SOLVING TRADITIONAL IT PROBLEMS VIA DEVOPS

Every Industry or company is in dire need to bring the software into the core of their business for instance, customer service is often provided through software. Companies following traditional IT practices can have silo teams at various stages of the software delivery pipeline. These teams are skill centric, made by banding together the people with same skill set in one team. This drives greater utilization, and benefit from economies of scale. However, in such typical IT environment every new feature has to pass at least 3-4 silos before being delivered to the end customer in which it spends 80% of the time waiting between silos. A DevOps organization instead has cells which consist of cross functional teams, focused on single software.

Given the modern demands the traditional IT practice involving the use of Commercial off the Shelf (Cots) software is dying and the companies are moving towards using Custom software's under DevOps , where the companies use open source software and cater it as per the needs of the organization. In the present market trends, the business objectives are changing frequently; a product developed following traditional IT practices may be missing some essential features and may not fulfill the changed business objectives as they would have changed while the product was being developed. Instead in DevOps, small versions of the products are developed and released continuously until the entire and final version comes out. This helps the team to cope up with the ever changing business needs of the client. Continuous release of product in form of versions also reduces the risk by daily integrating the code into the trunk, automated testing, ensuring all environments are in sync. In simpler terms, code is promoted from one stage to another if and only if, it is sure that it will work in the production.

In traditional IT practices after a project is completed the entire team involving Developers, testers, Project Managers move to a new project and the software developed becomes one of the many applications maintained by the company. But, when an issue occurs during the maintenance period, after the team has dispersed, there is no one to take the responsibility. This can lead to infighting and finger pointing, which could cost a potential client. A DevOps organization avoids this by understanding that failure is inevitable. So , instead of trying to avoid failure and finger pointing they try to understand the failure, put up a fairly small team together, recover fast and get the product online again.

Traditional IT companies need to develop a want for incremental values. In most of these companies, there are massive, multiyear projects that constantly miss budgets, and timelines. CIOs and CEOs have realized that speed can be a competitive advantage. Long windows of 2-3 years can lead to change in business objectives by the time the product or values are delivered. A DevOps organization understands that there's no time for long delays therefore the multiyear projects are shrunk down to small units and instead of releasing the entire software at once, versions of the software are continuously released. This can only be realized when a cultural and organizational change is achieved by realizing the values of DevOps in the organization.

IV. REALIZING DEVOPS

The whole aim of DevOps is to add value and improve the flow within the organization. It's about freeing up the resources and eliminating waste, to work systematically to get rid of any non-value-added process, to reduce the cycle time i.e. the time customer orders to the time the product is actually delivered. DevOps is also about sharing information, not only within the team but also with other teams. It works on the concept of work being pulled, not pushed and avoiding bottlenecks. In words of John Willis DevOps can be explained with CAMS acronym, Culture Automation Monitoring and Sharing. It is a cultural movement that aligns people, process, and technology towards a common goal of eliminating waste and increasing value using some accompanying technologies.

A. Cultural Change

Transforming the traditional IT organization to a DevOps Organization starts with reestablishing the purpose of the organization. Questions need to be asked that what has to be changed in order that DevOps values can be realized. Realizing DevOps may require identification of the shared goals of the organization, fair and common compensation formulae, values of the organization etc. A DevOps organization quality is prioritized over deadlines, staffs isn't considered to be easily replaceable or interchangeable rather they are considered as an asset to the organization. These cultural changes are inevitable if an organization has to realize the values of DevOps.

This all begins by empowering the employees so that they can make the required decisions to maintain the service. Like, Developers can raise alarm if they feel the quality of the product or service is being compromised at any point of time. The employees should be allowed to take the ownership. With Empowerment comes Accountability, the entire teams are to be held accountable for the product, project, entire service success, and quality. A measurement of success of application of DevOps principles is done using quality as a factor. Quality is maintained from the start itself. Quality Assurance teams and testers are not relied upon for catching defects in the software; instead the developers are held accountable for delivering good code. An environment is created where teams are committed towards excellence. In case of a failure, the team responsible should accept the accountability for the mistake and other teams should volunteer too.

This creates another great cultural aspect of DevOps, Teamwork. Teamwork is Development, Project Management, Operations, Testing, and Quality Assurance all working together both in normal and crisis times. It creates a sense of collaboration, during crisis hours; deep relationships between the members of the teams create a team by bringing individuals together to solve the problem. Teamwork helps in creating a sense of mutual respect within the organization, where each and every one is valued for the unique skill set they bring to the table.

Learning, continuous improvement is also a huge part of the cultural aspect of DevOps. A DevOps organization values

and encourages learning; this helps the individuals and teams to develop new required skills. An environment for learning can be facilitated by providing resources like online subscriptions to help teams learn, organizes conferences to brainstorm ideas, encourage public profiles and engagement where teams can learn from the public. Teams can learn from each other's mistakes, they can provide knowledge about their ongoing work to other teams and keep the curiosity alive within the organization. It helps to build trust within the organization which is another important cultural aspect. A trust between the various parts of the organization that lie on the supply chain is necessary.

B. Organizational Change

Organizational change is brought by understanding the work being undertaken by the organization, alteration of team structures, streamlining of procedures. DevOps leader, Jeff Cessna, said that nothing useful can be designed without understanding the people from whom it is being designed. A DevOps organization understands the system, the value provided by the system, and the waste being generated within the system. It understands the process, for ex. It identifies the waste involved in the process or steps that can be taken to add value to the customer. It also understands the flow between the Developers, Testers, Quality Assurance, Production and Support. The main aim is to understand the software delivery pipeline.

A bottleneck in an organization is a constraint that hinders the smooth flow of the software delivery pipeline. A bottleneck can be caused due to some technical problems, social debt, people, tools, and even outside parties. Like, In inconsistent environments, developers may not have production like environments to develop code against. The code works upon a local administrator, but fails during the production. This may be due to the change in the configuration of the environments or sheer size, this leads to a bottleneck for Developers who have to wait for Operations to provide the right environment. It also becomes a bottleneck for Deployment as Quality Assurance cannot reproduce code with defects. Therefore, Operations should maintain a similar environment everywhere, whenever demanded.

Organizational change can also be brought by altering the team structure. DevOps is reorganization, not a new team to hire. The idea of DevOps is to promote collaboration and virtual teams that work together. In a typical IT organization, after a project is deployed a skeleton team is left behind. This leads to loss of the tribal knowledge as the original team has disbanded so in face of crisis, specialists are brought in to solve the problem. This can be solved through DevOps principle, which counteracts deploy and disband model, by organizing teams purposefully around the internal products. In a DevOps organization, team of Developers, Quality Assurance and all retain some ownership of the system and service moving forward. Therefore, in crisis people who have worked upon the project and taken the accountability can be easily brought in to solve the problem.

Custom manual builds can also become a bottleneck as they require constant update. Therefore, often there is an inventory of useful features available in the source control but cannot be used in Deployment or Development phases as they need to be updated. This leads to unreliable and inconsistent long running custom builds, that can be avoided through continuous integration and delivery tools. Poor quality is another major bottleneck, in absence of accountability and shared excellence, Developers may forward incomplete code or code infested with bugs. It may be done to develop the code within the deadline or they may feel that it is the job of Testers. This makes Quality Assurance and Testers as bottlenecks and deployment fails. Therefore, rigorous integration testing is required to assure the quality and standards.

Another major bottleneck is Lack of Communication; teams may not share the objectives and plans with other teams within the software delivery pipeline. This makes the work for various parts in the pipeline to be batched up, at one point of time, Quality Assurance may have nothing to test and then suddenly they have huge amount of code to test within a short time window. Instead, if a feedback loop runs through the pipeline, Developers, Testers, Operations can evaluate the outcome of their work. They can monitor and have knowledge about what other parts of the DevOps Organization is working on.

Another possible alternative while altering the team structure can be reorient the various production lines around specifically maintained dedicated teams, instead of the teams moving from one product to another. This helps in maintaining the consistency, shared knowledge, and quicker delivery. DevOps organizational changes can be recognized when teams have higher commitment, as a part of the pipeline they understand that the ultimate customer is the end user. Development may have Operations as the customer but Operations may have end user as the customer therefore, everyone in the pipeline must be empowering the next team in the pipeline.

C. Automation Tools

The automation tools of DevOps help to realize the framework created by applying the cultural and component changes. DevOps tools automate the repetitive tasks, the feedback loop between the Developers and Operations and other teams, identification and removal of waste, and smooth flow of software delivery pipeline. It consists of collaboration tools, planning tools, developer environment tools, continuous integration tools, and deployment tools.

1) *Collaboration tools:* Collaboration is important aspect of organizational change as it helps teams to take action and thus produce results. Collaboration is regular communication of meaningful information with the decision making body. It can be encouraged through downtime exercises or by simulation of failures. Collaboration tools includes real-time chat boxes, discussion rooms, knowledge repositories and many more. All these help in avoiding knowledge waste i.e. disruption in the free flow and absorption of knowledge within the system, loss in date due to constant reshuffling, lack of communication within teams and loss of key information. It also helps to avoid waiting waste, for ex. Developers are waiting for the environment to be created or updated, Developers waiting for the code to be tested, customer waiting, and many more.

Collaboration tool like Skype and Lync, provide quick one-to-one collaboration. The video chat and conferencing engages teams together which helps in taking quick decisions or quick question and answer session. However, Skype has to be downloaded, configured, and installed on various systems, tools like CampeFire are web-based group chat tool that lets you set up password-protected chat rooms in just seconds. A client, colleague, or a vendor can be invited to chat, collaborate

and make decisions but, these have limited functionalities. Another useful way is Documentation; an accessible and updatable documentation help teams and keeps the document relevant. GitHub Wiki which comes with every GitHub repository is easily editable and helps to keep track of the commits made to the page. Like someone can store the coding standards for APIs or exceptional handling strategy. It helps to maintain to persistent information.

2) *Planning tools:* To implement DevOps values like communication, transparency, and teamwork, teams are engaged in planning. This requires formal project management tools; Kanaban Boards is one such work and workflow visualization tool that uses sticky notes on a whiteboard to communicate status, progress and issues. It can be used to share information, distribute ownership of tasks, maintain transparency, and much more. Like once a week Quality Assurance, Development, Design and Project Management, Product Management, and Operations come together and review the business objectives. Together, they prioritize the needs and put forward feedback on the priority list. This helps in maintaining the transparency, which reduces the knowledge waste and confusion about the blockers in upstream and downstream also have planning features, online view which can used to manage the product backlog, assign things to individual sprints and much more. JIRA is another important tool with Agile Roadmap planning feature that can be used to track progress and easily share plan to the stakeholders.

3) Issue Tracking Tools: Issue tracking is all about rapid response, to collect, triage, and respond to the issues such that it solves the issue raised by the subject. The subject can be a third party to the organization like the customer or it can be someone within the organization like Developers or Operations team member. Zendesk is a tool that creates knowledge base over time by collecting information from the support teams about the customer service issues they have been receiving over time. Through this knowledge base, it allows to create an online, customizable, help centre that can solve the third party i.e. customer issues. It has built in Artificial Intelligence powered Answer bot, a feature, that can send customers most relevant documents on the issue while they wait for an agent. However, when it comes to solving issues raised by people within the organization, other tools have to be looked upon. If an organization tier 1 person collects information and hands it off to another team, which puts it into their bug database, then it goes to the developers as a part of feature, all achieved by copying data multiple times. This leads to knowledge waste, transportation waste, and motion waste. Tools like JIRA, can be used to track and solve such issues that rise within the organization. It also supports continuous deployment and integration. It provides a shared list of issues on which people can interact to solve them.

4) Monitoring Tools: DevOps success potentially hinges upon the manner in which system is monitored. Like, after release it is important to have the telemetry in place to measure the positive and negative impacts of the release. This requires monitoring of the business system metrics before and after the release. There may be no negative system performance after deployment but, new customer sign-ups have almost cratered and no new orders are coming in. Monitoring the system effectively will help to know that although technicality of the system is correct, but an error has occurred in customer experience and they cannot use the system correctly. In such scenarios tools like Microsoft System Center, is able to perform group based collective management and monitoring of multiple systems in a Microsoft environment. Logstash is another important tool that integrates data from multitude sources like logs, metrics, web applications, and various AWS services all in continuous streaming fashion. Logstash filter parse each event to identify name fields and build structure for a common format that can provide easier accelerated analysis and business values. NewRelic is another tool that is built around the concepts of DevOps and has designed user interface that can be viewed by the entire team to track the changes in real-time. Through it, services can be pinged to check their health, latency, response time and thus, providing a holistic view of the service**.**

Tools like Kibana can be used to make charts, rank data, explore numbers and much more. It turns the raw data into information from which knowledge can be derived. Logstash delivers useful insights through the Elasticsearch, similarly Graphite and StatsD can be used for statistics, rendering graphs, or time series analysis. Monitoring helps in taking smart decisions using the information emitted by the system. It helps to understand the system from inside out and outside in. These tools can do monitoring of the individual components inside the system while being outside the system.

5) *Configuration Management Tools:* Configuration management is one of the most important parts of DevOps, it enforces a state to avoid configuration drift by using automation to achieve the required consistency. Like establishing a particular server state while the server is online, configurations can potentially change from server to server, when one quickly fixes them while forgetting to maintain consistency elsewhere. Therefore, automation is necessary to enforce consistency which in turn requires configuration management tools. In this scenario, tools can be used to maintain a large set of servers with repeatable and automated configuration enforcement by treating infrastructure as code that can be source controlled and deployed.

DevOps, abstracts environment like a configuration file which can be deployed easily and any changes are made systematically rather than manually. Like on a cloud platform, servers are not manually updated with a new load balancing policy rather it is performed centrally or pushed into affected servers all at once. This helps in maintaining consistency, one of the key values of DevOps. It avoids time wastage in correction of manual defects, instead of Developers tweaking individual production machines, configuration tools are used to push services along the deployment pipeline with a detailed audit trail. It enforces a systematic state and allows controlled access.

There are tools like Chef, where one can create cookbooks and recipes with Ruby and manage clusters of Linux servers from a central server or perform Chef Solo, without a centralized server, where each server maintains its own configuration locally. It also has Azure integration, thus allowing cloud integration with the tools. Another great tool,

Puppet can be used to create declarative infrastructure definitions; also the entire lifecycle can be managed starting from provisioning to runtime, deployment and reporting. Puppet is open source so it can be customized to support the needs of the organization to manage the clusters of Windows and Linux servers. It comes with lot of prebuilt modules for managing and maintaining commercial software and open source software both. However, both Chef and Puppet require pure Ruby knowledge.

Another great tool is Salt, based on Python and it can be used to set up angel list model that uses SSH instead of the idea of agents on individual boxes over the centralized server model. However, it also supports a centralized server or a master server, or one can run minions all over the windows and linux machines. It is focussed on high speed communication between the nodes over the polling strategy used by Chef or Puppet. Ansible is another great tool that uses SSH for automating configuration management, provisioning and more. It has a concept of playbook for configuring and deploying and orchestrating deployments. However, it does not come up with a lot of support for Windows. Therefore, with these tools a desired state of configuration can be achieved over the windows and Linux servers easily. Features and roles can be enabled or disabled, registers can be managed, software can be deployed and have a central configuration with a pull server and push things out to individual machines.

6) Source Contol Tools: One of the key points of Configuration management is to allow systematic changes but to ensure this controlled access is necessary. Source control is used to closely guard the software assets i.e. code and for configuration of all the environments. So, if Infrastructure configuration is performed and it passes the testing phase but fails during deployment, one can compare the configurationally changes that took place before and after the code were deployed. If something works differently than it is supposed to, source control tree can be looked upon to find by whom, when and where the change has been made that caused the behaviour in question. It provides a better audit trail, security and compliance to the environment. GitHub is the most widely used web-based hosting service for version control and Source Code Management. It provides access control and several other features like bug tracking, feature requests, task management and wikis for every project.

7) Development Environment Tools: Development Environment tools are used to accelerate social development and enforce consistency. It removes the problem of code not working in different environments, like it is working over the local machine of the developer but fails during the production. These problems can also be solved using the configuration management tools, like providing developer with the similar configurations as in the production environment. In addition to that, new tools like web-based IDEs have emerged. Tools like Codenvy, is complete browser-based development with a whole host of languages. It is useful in social development scenarios, where collaborative development like pair programming is being performed between teams.

Another great tool is Vagrant, it is a portable work environment that can be source controlled and run on any hypervisor technology. It is consistent and can be set up using a single command ‘vagrantup’. It can also be used to configure networks, set up multi-machine configurations. Like an operation team can use it to do a quick development, test locally and use the same workflow to test on the cloud. It also has Vagrant Cloud, which allows sharing of servers with base OS as Ubuntu, Windows, or Linux through a HTTP port or as a SSH client so that software can be run upon them for experimental purposes. It realizes infrastructure configuration file, as a code.

8) *Continuous Integration tool:* Continuous integration is about realizing one of the most important values of DevOps, speed. It is an incremental process to prevent the last-minute integration problems. In most of the big enterprise projects, last phase integration testing turns out to be the most painful part of the project, where all the independently developed components are taken and made to work together. It involves a lot of fixing which is inevitable. However, in continuous integration, there are centralized build engines, coupled with automated test suites, automated security inspection and other code inspection suites, that verify code quality before committing the build. This leads to continuous build, test and release which helps to realize the business objective before they change and gives an advantage over the competitors by maintaining the speed. It also removes the bottleneck at Quality Assurance where continuous integration not allows works to be piled up. The constant availability of working code and immediate feedback to developers if the build fails maintains quality assurance and reduces defect waste.

Tools like TeamCity, work well with Visual Studio is good for .Net solutions and supports NuGet. It allows testing before any commits are made to the changes and also shows progress reports. Tools like Hudson, now Jenkins, can be used also for monitoring software builds while quickly integrating changes. It can distribute builds across servers for complicated builds. Another tool TravisCI require least maintenance and integrates well with GitHub, it also supports a wide range of programming languages.

9) *Deployment tool:* The principle of continuous integration results in continuous deployment, it means that every new build goes for production immediately. Deployment tools are used to make deployment more reliable, it encourages thoughtful collaboration between the Developers and Operation and helps to make deployment an automated process. Like there may be a bug fixed or there can be a compliance and security problem that has been solved and these changes are pushed till production. Deployment tools give an audit trail from the starting till production, so that if any errors have been made, can be caught. It brings DevOps to life, where all the tools up till now come together and help to realize value to the end user.

A tool like CloudFormation, in Amazon Web Services is used to deploy environments and Elastic Beanstalk is used for code deployments. Environment deployment can be achieved via templates. Packer is another great open source tool that creates images for platforms using provisions that configure the running machine before turning them into Vagrant image or a VMware template or AWS machine image. All these templates

can be then run as a command line built which have latest code. So, instead of patching or updating a server, images can be used to simply replace them. Tools like Packer make image template creation much simpler.

Docker is another great tool; it is a Linux technology for creating isolated containers that are portable between the machines. This is OS virtualization versus server virtualisation. The Operating System now acts as hypervisor and manages the containers deployed over it. These individual containers have isolated resources, CPU, memory, network, virtual interfaces, and content isolation. This provides a higher density use of a virtual machine and since it can be used on any hardware, it is employed more and more in the continuous deployment pipeline and continuous integration pipeline where the code can be tested over different containers.

Octopus is another great tool that works well with both ASP.NET and TeamCity. It can be deployed in Azure environment and provides information related to the latest deployments made, its state and other valuable insights. It can be used to manage application configurations that are environment sensitive like IIS settings. It can introduce approvals and manual interventions as everything cannot be automated. It even allows self-service deployments with role-based access.

TABLE1. DEVOPS TOOLS AND THEIR FEATURES

Tools	**Type**	**Features**
Skype, Lync	Collaboration	Provides Chat boxes and video chat features for both one to one communication and teams
CampeFire	Collaboration	Sets up browser based, password secure chat rooms for teams within minutes
GitHub Wiki	Collaboration, Documentation	Easily updatable blog like pages that helps to familiarize any new participant with the ongoing work.
Kanban borads	Planning, Issue Tracking	Communicates status, progress and issues to the entire team using simple visualization like sticky notes on a whiteboard
JIRA	Planning, Issue Tracking	The Agile Roadmap feature of JIRA also allows to share progress with stakeholders.
Zendesk	Issue Tracking	Allows to create online, customizable, help centre and provide excellent customer service using Knowledge stored in its knowledge base
Microsoft System Centre	Monitoring	Monitors multiple system in a Microsoft environment.
Logstash and ElasticSearch	Monitoring	LogStash along with ElasticSearch integrates data from multiple sources and explores data to provide valuable real-time insights.
NewRelic	Monitoring	Interactive User Interface allows the entire team to track changes in real time and provides holistic view of the services
Kibana	Monitoring	Can rank, chart data to provide insights for monitoring the system
Graphite, StatsD	Monitoring	Provides statically rendered graphs, time series analysis and other statically relevant information by monitoring the system changes.
Chef	Configuration Management	Provides cookbooks and recipies to manage clusters of Linux or Windows servers individually (Chef Solo) or centrally.
Puppet	Configuration Management	An open source software that can be customized as per the needs of the organization using the pre-built features.
Salt	Configuration Management	Focussed on high speed communication over the poll strategy followed by other Configuration Management tools.
Ansible	Configuration Management	Makes use of SSH technology and also has features and roles that can be enabled and disabled to maintain a desired state of configuration.
GitHub	Source Control	Web-based hosting service that provides version control and source code management.
Codenvy	Development Environment	A web-based development tool that allows pair programming.
Vagrant	Development Environment	Has Vagrant Cloud that allows sharing of servers through a HTTP port or SSH client so that software can be run from anywhere.
TeamCity	Continuous Integration	Works well with Visual Studio Online and supports NuGet, allows testing before commits are made and generates reports thereafter.
Jenkins	Continuous Integration	Allows monitoring of software builds along with integrating changes.
TravisCI	Continuous Integration	Supports wide range of programming languages and integrates well with GitHub
CloudFromation	Deployment	Amazon Web Service based environment deployment solution, its variant Elastic Beanstalk is for code deployments.
Packer	Deployment	Uses images to create a server, and templates can be run as a command line built
Docker	Deployment	Creates isolated containers on OS thus allowing higher density use of Virtual Machines.
Octopus	Deployment	Works well with ASP.NET and TeamCity.

IV. MYTHS ABOUT DEVOPS

Although change is inevitable yet it is not easily accepted so every organization will raise some objections before employing DevOps within its software delivery pipeline. With the advent of Cloud Computing, most of the IT business units of the organization are being outsourced as executives don't think IT has value and are thus decreasing investment into it. However, in such scenarios one needs to incubate the ideas of DevOps more into the organization so that IT can deliver customized software that can be useful to other business units. Executives need to realize that they require a cultural and organizational change to build the trust back between the various units of the company and remove such myths. The various other myths are-

- ***Ignorance***- organization often sees DevOps only as Quality enforcement and object to its implementation on the grounds that they already have Quality Assurance and Release team. This is because they have a wrong perspective about DevOps, DevOps is a mind shift, cultural and organizational shift. It is not merely having separate disciplines, DevOps has feature teams, software teams, and service teams that are focussed on customer service and have been embedded in various disciplines.
- ***Security***- It is a myth that if Developers touch the production, information security problems and code security problems can arise. However, this can be solved easily using DevOps, DevOps focus a lot on quality, it is one of the main criteria for having organizational change and quality can reduce the security vulnerabilities. DevOps push code faster through the pipeline and pushing code quickly means that vulnerability can actually be patched faster. Also, automated mirror test environments provide security testing before production. All this is realized using DevOps and removes the question of security.
- ***Remote teams***- Most of the times DevOps teams are not locally available, it can be an off-shore team. It can be tricky, but with help of right collaboration tools and shared culture and vision it can be made possible. Instead if a major part of software delivery pipeline is outsourced or developed under a partnership agreement by some third party, it reduces the control of the organization over the particular part of the pipeline.
- ***Impact of DevOps on Operations***- DevOps does not adversely effects the Operation team; DevOps means both Developers and Operations work jointly to deliver software. Operations still has to provide quality environments to Developers, Testers, Production and others. Operations have to configure, monitor, maintain and tune the environments. DevOps is all about Developers and Operations working together, understanding their part in their software delivery pipeline, creating feedback loops and optimizing each other. Developers cant expected to construct network topology or troubleshoot all server OS issues but, it does change their role by adding more value added roles.
- ***Legacy Systems***- It is often argued that Legacy systems cannot support the new tools and provide the required changes. However, DevOps does not only mean automation tools, with legacy systems one can build up the necessary flow leading up to the legacy systems to release stuff effectively, efficiently test and verify the code changes. Continuous integration, scripted deployments can still be performed and in DevOps culture Service teams that have shared knowledge and shared ownership of the service can be formed.
- ***Lack of Knowledge***- A big objection can be that none of the teams know about DevOps. The answer to this is, learn. DevOps is not always hardcore technical skills, it is better communication, teamwork and continuous improvement. It is also a skill set that has to be developed so that organization can improve its software delivery pipeline and most importantly customer service experience.

CONCLUSION

DevOps is a cultural and organizational change that can be incorporated within the environment by focusing on why it is needed, changing accountability, trusting each other to work towards a shared objective, identifying bottlenecks, creating dynamic teams, changing the mode of thinking, rearrange more around services rather than pump and dump strategy and delivering persistent values. With DevOps mindset employees are happier and systems have lesser defects that can be fixed faster using the automation tools like collaboration tools, issue tracking tools, source control and management tools.

REFERENCES

[1] M. Huttermann.: DevOps for Developers, vol. 1. Springer (2012)

[2]J.Willis: What devops means to me, July 2010. http://www.getchef.com/blog/ 2010/07/16/what-devops-means-to-me/

[3] DeGrandis, D.: Devops: So you say you want a revolution? Cutter IT J. 24(8), 34–39 (2011)

[4] J. Humble., J. Molesky : Why enterprises must adopt devops to enable continuous delivery. Cutter IT J.24(8), 6–12 (2011)

[5] T. A. Limoncelli, D. Hughes, L. L. Lisa themedevops: New challenges proven values. Login 36(4), 46–48 (2011)

[6] M. Loukides: What is DevOps? O'Reilly Media, Inc. (2012)

[7] S.K. Bang, S. Chung, Y. Choh, M. Dupuis : "A grounded theory analysis of modern web applications: knowledge, skills, and abilities for devops." Proc. of the 2nd Annual Conference on Research in Information Technology, RIIT 2013, pp. 61–62. ACM, New York (2013)

[8] J. Roche: Adopting devops practices in quality assurance. Communications of the ACM 56(11), 38–43 (2013)

[9] Puppet Labs and IT Revolution Press. 2013 state of devops report (2013). https:// puppetlabs.com/wp-content/uploads/2013/03/2013-state-of-devops-report.pdf (Accessed 19 December 2018)

[10] Puppet Labs, IT Revolution Press, and Thoughtworks. 2014 state of devops report (2014). http://puppetlabs.com/sites/default/files/2014-state-of-devops-report. pdf (Accessed 19 December 2018)

[11] Puppet Labs, IT Revolution Press, and Thoughtworks. 2015 state of devops report (2015). https://puppet.com/resources/whitepaper/2015-state-devops-report. (Accessed 19 December 2018)

[12] Puppet Labs, IT Revolution Press, and Thoughtworks. 2016 state of devops report (2016). https://puppet.com/resources/whitepaper/2016-state-of-devops-report. (Accessed 19 December 2018)

[13] Puppet Labs, IT Revolution Press, and Thoughtworks. 2017 state of devops report (2017). https://puppet.com/blog/2017-state-devops-report-here. (Accessed 19 December 2018)