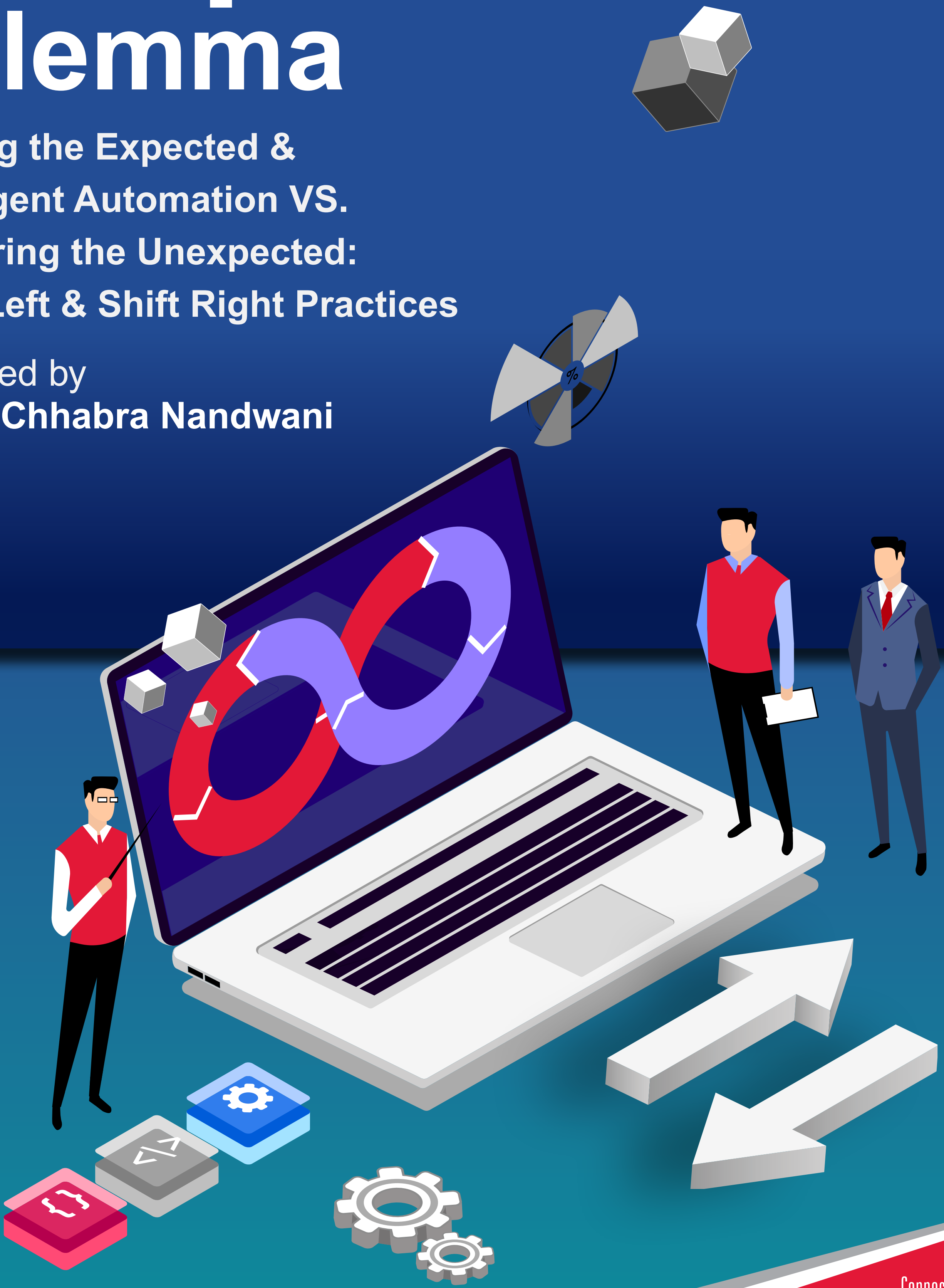
Tech Mahindra

DevOps Dilemma

Testing the Expected & Intelligent Automation VS. Exploring the Unexpected: Shift Left & Shift Right Practices

Authored by
Anjali Chhabra Nandwani



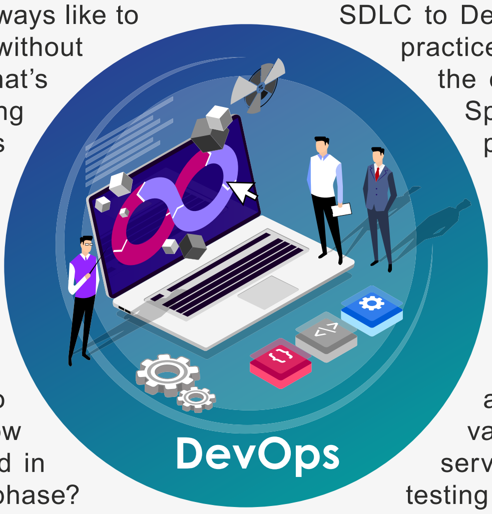
Connected World.
Connected Experiences.

Abstract

Are you worried about your organization’s ability to cope up with complexity of delivering at high velocity with excellent quality in multi-speed IT landscape and hybrid environments? Read some thoughts here about some Quality Engineering paradigms in DevOps world that we, Digital Assurance Services- Tech Mahindra, have implemented successfully with our customers

DevOps

DevOps - an art (combination of philosophies, methodologies & tools) of delivering products and services at a higher velocity as compared to traditional SDLC methodologies – now, I always like to add a statement that Velocity without quality is of little use – and that's where the question of testing comes up. There has always been a confusion (to say the least) about the role of testing or testers in the DevOps world. While "Continuous Testing" and "Quality Engineering" are some phrases that have taken front seat when it comes to DevOps – one always wonders where to draw the line. How to assess how much QA is actually needed and in which phase or is there really a phase? While "Shift Left" is what everyone is talking about what does that actually entail? And then comes the phenomenon of Testing the Unexpected, Testing in Production, Exploratory Testing etc. i.e. "Shift Right"



With all the consulting effort in my experience at Tech Mahindra across multiple enterprises as they go through that transformation journey from traditional SDLC to DevOps and while we try to align the practices to meet difficult demands from the complexities arising in a true Multi-Speed IT environment – I have built a perspective that follows.

Shift Left is something that is simpler to implement. IT community has put in lot of effort in terms of setting up guidelines, tools and frameworks for early validation and verification. More unit and component level testing, early validation of services, techniques like service virtualization for early integration testing are practiced. However, it is more about "Testing the Expected" and does very little for real time feedback and exploring and base-lining the undefined and unknown. However, this still remains an integral part of Continuous Testing framework.

I have read learnt, experienced and implemented many paradigms of Intelligent Automation – but if I have to define simply then the features that one should expect from automation framework in DevOps world are:

Agility to Adapt to Multiple Tools

Intelligent automation framework will not force a specific tool on the developer and testing community – instead would align with multiple tools used across the enterprise. Many times I am asked if multiple tools imply anti-standardization? My answer to that is as far as all developers / SDETs / QA community are using the same framework it does not matter which execution engine they use. For the enterprise there should be a standard automation framework across Unit / Component / Component Integration / System Testing with integrated reporting. That's how we at Tech Mahindra have developed our Intelligent Automation Framework. Point to note here is that multiple tools does not imply unlimited tools – it is in fact a set of tools that are able and enough to cover the enterprise technology landscape from automation perspective.

Ability to Overcome Application Changes

This is a vast subject in itself and there are multiple ways we have achieved this. While auto-healing scripts is a word many use, eventually I believe it is the framework design that leads to scripts that are easy to maintain and adaptable to application changes. Also headless automation (services layer automation) is the key to achieving this efficiency.

Promotes Early / Progressive Automation

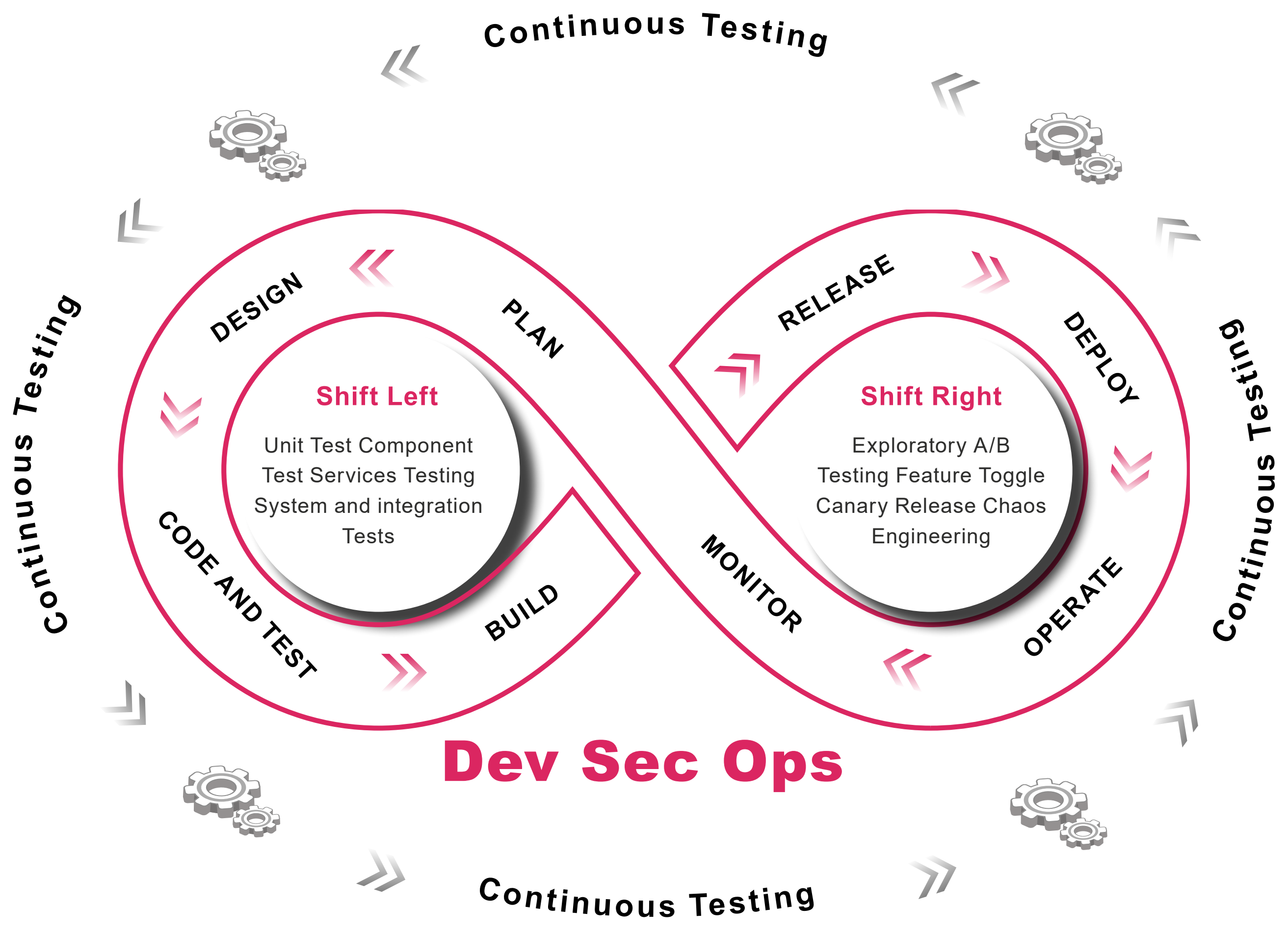
That's what Shift Left is all about. Automation framework plays a major role here. The key consideration is how much unit, component and component integration automation it can support.

Helps in building regression through progression

Very detailed subject in itself but bottom line is that the in- sprint automation shall help in building the end to end test scripts as well.

Supports Hybrid Infrastructure & Varied Testing Personas

Question to ask here is that does the framework support write once and run in many environments (on premise/cloud, Web / Mobile, multi browser) with minimal configuration changes – if yes, it is indeed intelligent automation framework.



In Nutshell

Progressive / In-sprint automation and Shift-Left is the backbone of Continuous Testing that is mandatory for "validating the expected" for successful DevOps and Intelligent Automation Framework is what makes it possible. We have implemented different flavors of such Intelligent Automation Frameworks leading to 20-25% savings on total cost of quality ownership for some large enterprises

- **So where do we put the money for Quality Engineering in DevOps scenario Shift Left or Shift Right Testing? Where is the ROI**
- **While Shift Right can get more real time feedback defect fixes when in production and late in game can be very costly**
- **At the same time just relying on early validation of expected results can lead to ignoring that unexpected that can cause major disruption at some point**

Now let's look at the concept of Shift Right in context of services and APIs testing

There is a testing pattern in which before replacing an existing service with a newer one in production we replicate and direct the real time production (with older version of service) request traffic to a staging environment (with newer version of service). Then results from new service are validated against production results. This is defining the unknown and unexpected as to how the new service behaves with the production request data. This is a neat way of Shift Right Testing – without breaking anything in production as such and without spending too much money on creating huge amount to test data to mimic all production scenarios.

Another interesting concept is Chaos Engineering

The idea behind this is fairly simple – Can we avoid 100% of failures in production in complex distributed and hybrid environments – application issues, environment issues, service failures, configuration issues etc.? If not, then why not induce controlled failures and see how resilient the product or application is under the failure circumstances. Typically organizations use this to experiment how a distributed system behaves in rough production conditions. Chaos Monkey is a system that was originally designed by Netflix to induce pseudo failures or sudden termination of services in the system architecture to "explore the unknown and undefined" system behavior under failure scenarios. This helps to proactively identify issues and fix them. Very neat technique and we will see lot more adoption

The last concept I will discuss about on Shift Right Testing is the A/B testing

This is slightly different in terms that it is not to test the application behavior as much as it to analyze the consumer patterns or behavior. Basically two stable versions of same application with some modifications (to analyze the patterns) are deployed in production and traffic is divided between the two versions – example is with one of our customer's eCommerce site – we applied this to see if one version of site with different way of displaying search results leads to more purchases? Interesting – isn't it?

There are a few more scenarios and use cases for Shift Right Testing – however mostly similar concepts with varied techniques.

Conclusion

In my view, both Shift Left and Shift Right Testing concepts are important and required – how much of each is what needs to defined and implemented as it varies for enterprises based on business scenarios, priorities, technology landscape etc.

It’s been a great learning and experience so far while working on defining these strategies for some of our esteemed customers as they progress on their Digital Transformation road-maps and DevOps adoption. There is lot more to explore – looking forward to this journey of exploring that “unknown and undefined”!

Anjali is a technology leader with 20 years of testing & QA leadership experience in Information Technology Services industry, with specialization in optimizing Software Development Lifecycle (SDLC) and Quality Assurance Services.
She has worked with large enterprise customers ranging from Banking and Financial Services, Telecommunication, Technology to various verticals. She has proven track record on DevQAOps Transformation initiatives, Consulting for DevOps and Agile Transformation initiatives, Automated Delivery Pipeline (CI/CD/CT) framework design and set-up for large enterprises, QA Community of Practice (COP) set-up, DevQAOps Program Management and Governance. In her role, she has worked with diverse IT teams globally to optimize SDLC with usage of Predictive Analytics, Machine Learning and RPA. She comes with in-depth understanding of Integrated SDLC Automation experience along with framework design / tool selection and ROI analysis.

Anjali Chhabra Nandwani
VP & Head Digital Assurance Services
Tech Mahindra Americas
LinkedIn

www.techmahindra.com

connect@techmahindra.com

www.youtube.com/user/techmahindra09

www.facebook.com/TechMahindra

www.twitter.com/Tech_Mahindra

www.linkedin.com/company/tech-mahindra