Master the skills of QAOps

Contents

As the industry is changing constantly, and these changes require the system to be more efficient, scalable, and, of course, reliable so the quality expectations are also increasing day by day.

Explosion of Ops

Recently, the IT world has been experiencing an explosion of different terms related to operations. The good old days—when the global order was defined around a rule of thumb and IT as separate from business—are gone, never to return. Dozens of 'Ops' crowded the sphere of software testing: starting with trendy DevOps
The successful application of DevOps – combined practices of software development and IT operations – triggered the rise of similar practices such as DataOps, ArchOps, DevSecOps,..AIOps, BizDevOps, CloudOps, DevOps, ITOps, NoOps, etc. All of them complement the DevOps methodology aiming to optimize the SDLC (software development life cycle).

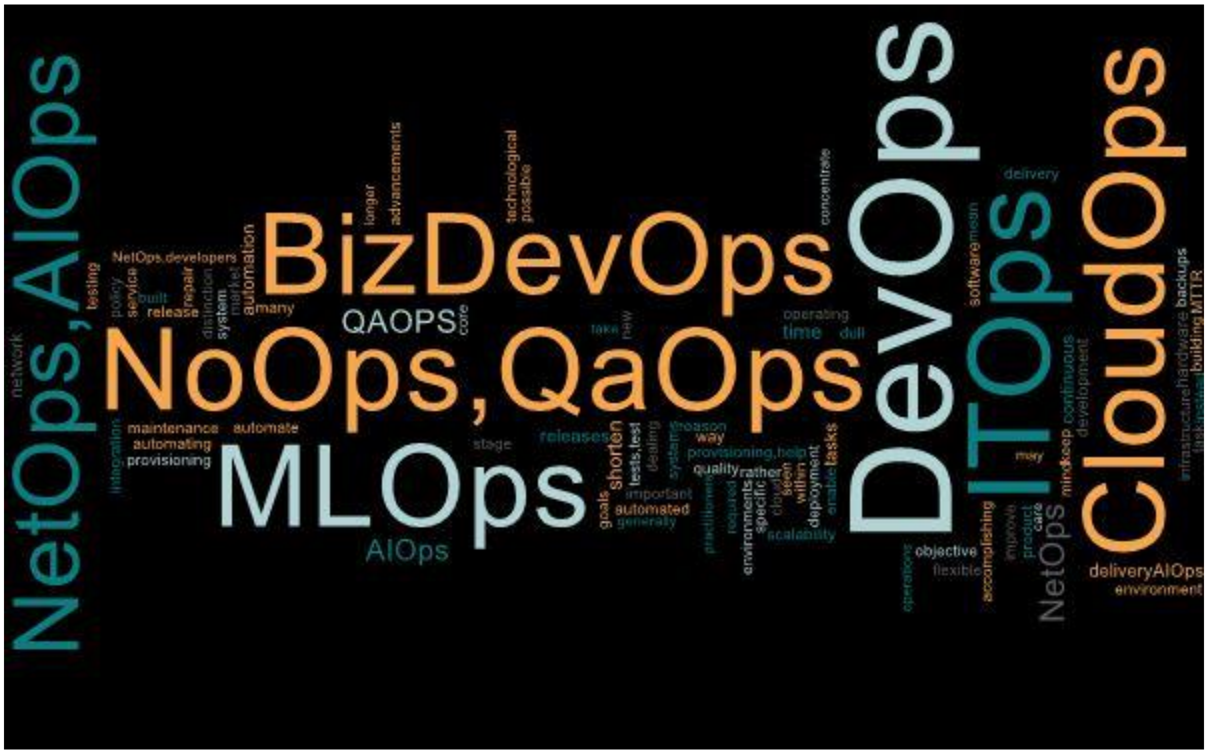


Fig 1. Explosion on Ops

What's main common thing about all Ops frameworks are recommend to automate of tasks to reduce the human error and also collaboration of various departments and all motivated to act as a single team to work towards the common goal and faster deliveries and scalable deliverables.

How are they connected with QAOps and what does this notion mean? Let’s discuss in article.

Fig2. What’s main common thing about all Ops frameworks

QAOps Introduction

It’s very important to see product from QA point of view, and thus very important that Qa should be involved in every phase of software lifecycle development. The core idea behind QAOps is to increase direct collaboration between developers, testing engineers & ops instead of having them work in isolation.

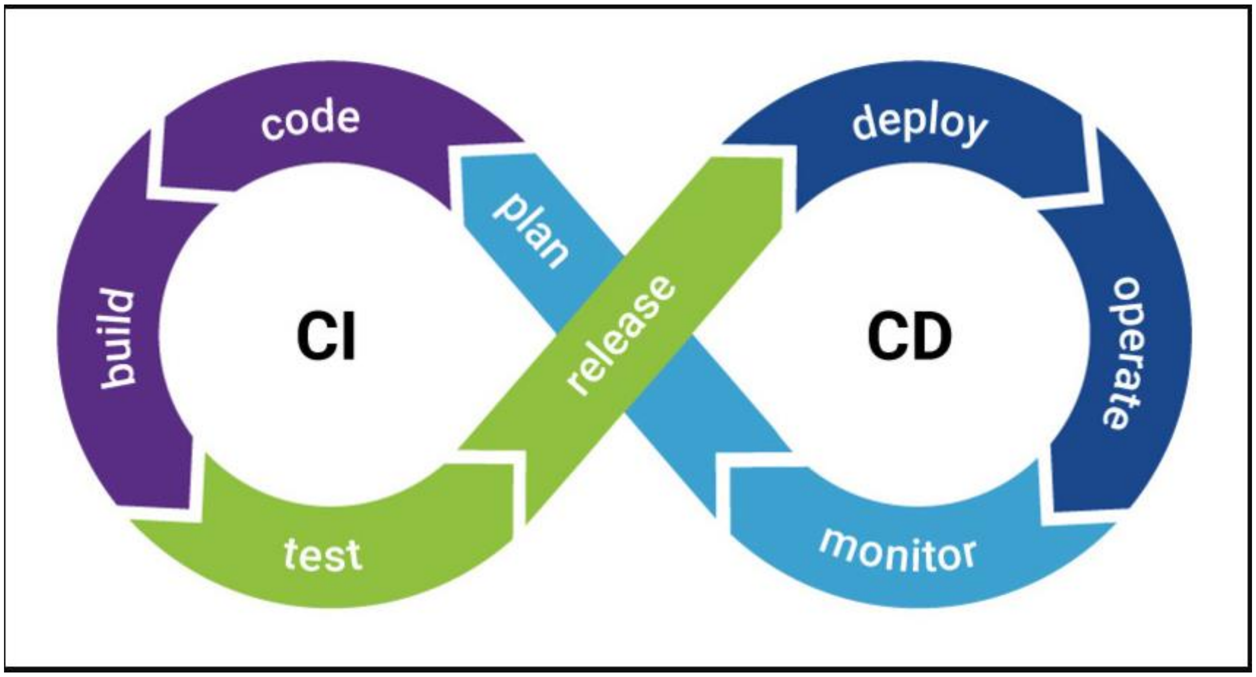
Product teams are often pushed to build and release faster. But sometimes this accelerated delivery comes at the expense of quality. So I say that sacrificing quality is just a short-sighted delivery strategy that almost always ends up causing more damage in the long run.

QAOps is the practice of integration of quality assurance activities into CI/CD workflow. It means that QA specialists should interact with the dev team, operations engineers, and other experts involved in the SDLC.
For me DevOps is the delivery of application changes the speed of the business.,

QaOps is something similar to DevOps as this framework uses the basic approach and DevOps mindset, a mindset of direct collaboration work as an integrated team towards common goal. And this mindset is very much important

CI/CD, DevOps, QAOps - how they coexist

As I mentioned before also QAOps means integrating QA into the CI/CD pipeline.



This CI/CD forms the very foundation of what we call the DevOps pipeline and its resultant approach. Continuous Integration (CI) is the first phase of the pipeline and involves:

1. Integrating code and content changes from multiple sources rapidly into a single main build of the application; and then,
2. Executing various automated testing to verify the functionality of that build. The execution of such can collectively be referred to as Continuous Testing (CT).

Continuous Delivery (CD) is the natural follow on process of continuous integration. It involves the packaging of an application with all its deliverable artifacts together for delivery to end users.

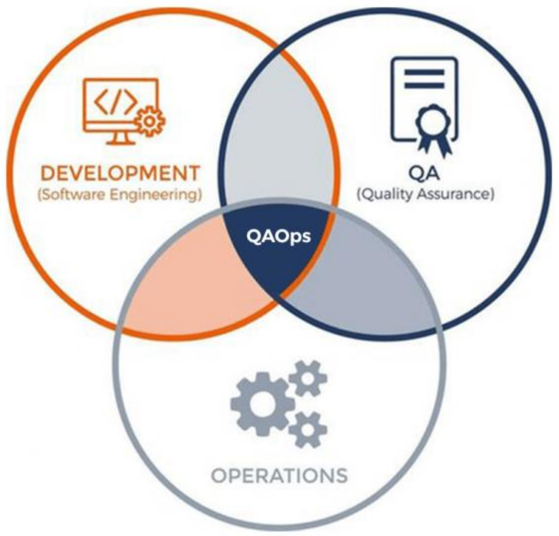
Continuous Deployment is the third and final phase of this pipeline and is dependent upon the successful execution of both the prior CI and CD phases in that order. This phase involves the automatic launching and distribution of the packaged and acceptance-tested application to the targeted end users.

DevOps, in turn, sits atop, and is dependent upon, the CI/CD approach. It goes some steps further however and incorporates cultural philosophies, practices, and tooling. When these items all work in concert as designed, they facilitate significant increases in an organization's ability to:

Deliver high performing product and/or service to clients in much faster pace than pre-DevOps.

QAOps Definition & Implementation

QAOps Definition



The commonly accepted **definition of DevOps** is the **delivery of application changes the speed of the business**

Two key principles of QaOps

- The QaOps refers maintaining Software quality by approaching it with Devops mindset.

- **QaOps framework** increases the **collaboration** between the QA engineers & developers. Qa engineer developers, IT Ops engineers and everyone else involved in the CI/CD pipeline. In other words, QA should not exist in a silo.

QAOps Implementation

QAOps is the operations of running and orchestrating QA across the CI/CD pipeline It prescribes the QA process, rather than being a set of isolated activities performed here and there, to become an integral part of the DevOps process.

QAOps, in its current incarnation, can be successfully executed through the holistic adoption of Test Automation, Test Parallelization, Test Scalability, and what we can refer to as Holistic Connectivity & Collaboration. Let us take a quick look at each of these.

Automation testing

Automation is a central tenet of QAOps. It involves using technology and tools to mitigate the otherwise manual efforts expended upon repeatable testing and test-supporting activities that are characterized by:

- Automated build\deploy cycles involving-
- Environment allocation
- Code analysis, unit testing, & deployment
- Functional testing
- Performance testing
- Security analysis

This results in freeing up QA practitioners so their time can better be spent on more important activities such as:

- Test design, analysis, and optimization
- Defect analysis and processing
- Participation in the continual improvement of product and process

Test automation also accelerates the testing feedback loop which fulfills the cadence requirements of an effective QAOps setup. This can be accomplished via the following:

- The automation of repeatable testing (like Regression). Regression tests are usually large in number and are to be executed repeatedly, especially before major releases. Automating these tests removes the burden from the manual testers but also empowers the project to execute far more often than if performed manually.

Tests for New Functionality can be automated once they have been identified as suitable for automation. Ideally, this will occur in the current sprint and be incorporated into the ‘living’ regression in time for the next sprint.
The automation of Smoke and/or Sanity Tests. These tests can be subsets of regression or separately formed tests altogether. They are utilized to provide very quick feedback to all in the QAOps pipeline as to the viability of a particular build or release. Such quick feedback will facilitate quick action and turnaround.

The automation of Unit Testing. Unit testing should typically be done by the Development team as the first check that their created code components (or units) behave as intended. A suite of automated unit tests should be executed whenever a new build is created.

Automated testing is the perfect vehicle for efficient testing across Multiple browsers & O/S Combinations. Vast efficiencies, time, and cost savings are evident by employing this method versus trying to do this manually.

Parallel Testing

Next is parallel testing, It involves executing instances of the same tests, different tests, or any combination thereof across a multitude of target browsers and/or operating systems.
For example, rather than executing the automated regression suite first on O/S#1 and then O/S#2 immediately afterward, Parallel testing would, instead, prescribe the concurrent execution of the automated regression suite on both O/S. Another example could see the breaking up of the regression suite into smaller, disparate, test suites which could be, once again, executed at the same time. This would drastically reduce overall total testing time, making parallel testing an ideal component of QAOps.

Facilitating such parallelism in terms of target test environments, however, might require greater resourcing in terms of hardware and infrastructure. An alternate and cost-effective solution that is well suited to the QAOps methods could be cloud-based testing environments that could be accessed as required.

Scalability testing.

We all know that business growth is good for the company. When you develop a product and people start liking it, you will have to scale it. Scaling the product includes adding more features and making the existing ***features better.***

Scalability testing is the testing that is done against the capacity of a network, system, or process to adapt to expected and sudden changes. Applications must handle large increasing changes in data volume, simultaneous users, and other workloads. Scalability tests can ensure that the applications are ready for such situations prior to going live.

Scalability tests are designed to be scalable with the CI/CD pipeline and the results of such will allow the QA engineers to find and fix the performance issues of the application. Some of the key measures (or attributes) of importance – thought not limited to – in this testing are response time, throughput, CPU usage, memory usage, and network usage.

Holistic Connectivity

and last, but not least, is integration with Dev, Ops and QA to achieve common goal as integrated team.

Here we are talking about The idea of improved collaboration, from a QA point of view, is made possible and realized through the following components:

- The usage of test automation tools – for Unit, Regression, API, Performance, and other types of testing – is noted given the automation-centric ideas behind QAOps.

- The usage of QA lifecycle and supportive tools are vital to enable critical activities like test design, planning, execution, and defect processing. Using tools that are not designed specifically for testing artifacts and tasks will lead to inefficiencies and a sub-optimal pipeline if any pipeline at all.
- Once these necessary QA lifecycle and supportive tools are in place. it is important to integrate the communications between them and the other Application lifecycle tools to realize a Tooling Ecosystem. This integrated Ecosystem entails the following points:
    - Seamless transfer of important project information across tools to ensure real-time availability for the intended target audience. An example of this would be the ability to update a User Stories' Acceptance Criteria and then have this information automatically update the automated tests related to that Story (that may reside as Selenium Code or other open source or commercial automation tool). This reduces onerous manual activities.
    - Ease of linkage with regards to artifacts. For example, linking a test case to its corresponding Story, Requirement, Test Case runs, and Defects. This facilitates a greater breadth and depth of resultant metrics and reports. An obvious example of this is the ability to run a test coverage report that displays how much of the given requirements (or Stories) are covered by the existing test suite. The explicit linkage between the test and the requirement provides the necessary basis.
    - The enhanced employment of metrics to aid decision-making. Such can entail (but is not limited to):
        - Applying risk and priority levels at the story level to drive the level of testing needed for that story and its features.
        - Setting test coverage targets and then actively measuring against those targets at key intervals.
        - Adjusting test coverage as per changing functionality and its associated risk level.
- All QA tasks and artifacts must be planned, executed, and measured like other project tasks. For example, all tasks and tests spawned from a User Story or Requirement should be processed like any other artifact in the project within the Tooling Ecosystem. QA tasks and activities should appear in task/asset lists or Boards as pieces of work that have a status, an estimate, an assignment, and any related info. The main idea is that all

project tasks and artifacts should appear in one place and be given appropriate, if not equal, consideration in terms of importance.

Holistic Collaboration

Collaboration: QA practitioners must fully participate in and contribute to a project just like Development and Operations. QA must work closely with other groups to ensure optimal execution of the pipeline. These activities can include:

- Participation in planning, daily stand-ups, retrospectives, and other relevant project activities.
- Cross-training (if required) in some Development and Operations skills (and likewise for Dev and Ops).
- Cross-tasking (when possible) – allowing QA to undertake some Development and/or Operations tasks (and likewise for Dev and Ops).

When QAOps Should Be Used

Always. That's the short answer to when QAOps should happen during the production process. This means staging, development, release, and even after when you update, improve or modify the app or software. Even if your entire application lives in a cloud environment, Quality assurance has to be part of the entire operation. That is precisely why it is so good at releasing the bottle neck that generally occurs with testing. When you continuously check for the small bugs throughout the entire process, those small bugs don't turn into major bugs.

QAOps is especially useful when specific types of testing are needed. For localization, QAOps is almost non-negotiable. When you are designing software or apps for a specific location or culture, you need to know that it actually fits in with that culture. If it doesn't, your app will fall short. QAOps allows you to connect with testers in or near that location so you can really see if your software will connect or just be plain confusing.

QAOps can also be extremely useful in regression testing. In other words, you have some previously developed software and you need to quickly release a software enhancement, patch or configuration change. To ensure that the update is released in a timely manner, QAOps can help check for any new faults that may have been created by the new information being added. If you are using an agile project management process, regression testing can be viewed as something that causes unnecessary overheads. With the affordability and efficiency of QAOps, that concern is gone. And it is for this very reason that companies like Openlink are integrating QAOps into their development process–it allows them to provide customers with the best software, faster.

QAOps Industry Case Study - FaceBook

Do people really use this?

So the answer is yes. Facebook is one of the best examples of how QAOps just works. I don't work for Facebook, but this is something I read, Facebook's login feature enables users to log in to millions of apps and websites with their already created Facebook identity and privacy controls. In 2014, Facebook decided to migrate to Facebook Graph API v2.0 and enforce Login Review for all apps. To ensure that this migration went smoothly, Facebook wanted to test out the new version on the five thousand largest apps. Unfortunately, in-house they could only handle testing it on five hundred apps. So they chose to outsource. By outsourcing testing, they had all five thousand apps tested in a month and were able to identify and address critical problems with more than nine hundred apps–a feat that would have been impossible by solely relying on their in-house testing team.

So QAOps can work for the giants of technology, what about the small ones? QAOps can scale up or down to fit any business size. This includes agencies who are outsourcing providers for software development, such as DotCom Development.

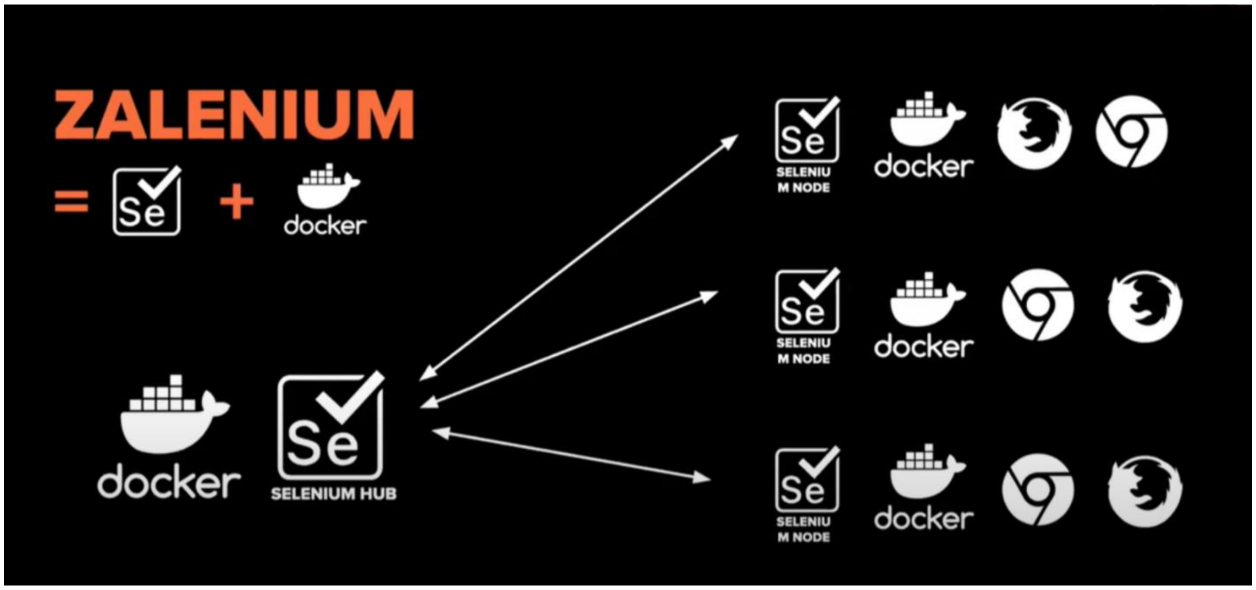
QAOps GlobalLogic Case Study

But let's take an example from our companies and vision. You all know that GlobalLogic as a company and delivery in agile manager is our core competency.
I'l walk you through the challenges we are facing in one of the projects while we are trying to move to QAOps and why to Qaops.
Its web-based application, and we are having an automation framework using a protractor in

JavaScript flavor, using Azure Devops to manage our CI/CD pipelines and monitoring, where we are doing schedule and automated deployments from Dev -integrated env -> QA env and post that manual promotions.
We have around 10000 test cases out of which we have automation coverage of 60-65%, we are using FDD (Feature Based Delivery) with TDD flavor.

In integration env we are ruing our sanity suite which takes only 15mins (its in it very small time we execute close to 300 testcases) to run on pipeline and take 10mins for our automation engineer to analyse if it a produce issue or script is failing as we are have configured AI reporting dashboard. so it helps us to categrorise the test failures and which makes our life more easy.

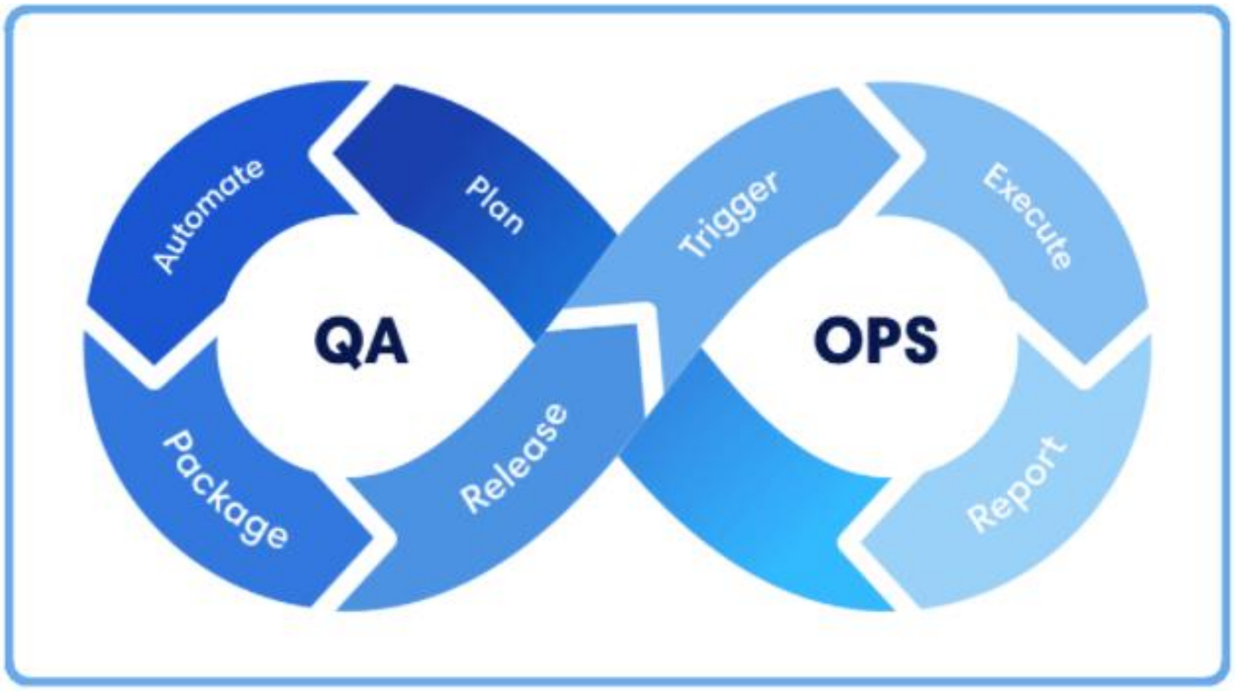
Our regression that to impact regression runs in 60mins or less than 60mins sometimes (3000+ testcases) How we achieve this by parallel testing now there is another challangecomes up as our testsuties size changes as per the changes coming ot Qa env so we dont want have fixed number of instance to run parallel.
We found out the tool called Zalenium. What's this Zalenium.Its nothing with Selenium grid with dockers



Setting up your own Selenium grid, I say that might not be so hard. The challenge here comes when you start using it to run a lot of tests on it. This sometimes could cause environmental issues and instability. So here comes the Zalenium. The interesting part here is that we can scale on demand so we are able to create several Selenium nodes. Isnt it amazing This is how qaops works, now we have capability of doing everyday deployments to our UAT env where our customer uses our project.

As tools are involved and we want ot auotmate every task, and we want ot practise this on daily basis. The QAOps framework is a process. It's a process that the QA team is practicing on a daily basis. and to make it successful we need to train all our QA to achieve this. As pleased, the implementation of QAOPS framework is defined as an infinite process of learning and putting in use the best practices of QA practices throughout the following phases.

QAOps Phases



And I would like to highlight here all the phases that we are currently following in our company to implement QaOps. So I would say **1st phase is plan** . All the initial things. Start with the plan .This step of planning will include and describe the strategy and the testing objectives that we want to achieve, as a part of this exercise we will come up with a document which we call a, test plan document.

Second, I would say test development, and this is our main activity and our biggest challenge as QA engineer. In this phase, we perform most of the activities that were defined in the previous phase of planning

Here our testing activities are divided by,
Let's say designing of test cases as per the requirements. Have Automation framework Alot of tools are available t like Selenium with all the programming languages JavaScript, but you can use it other Robot framework. If you are working in a BDD way you can also use cucumber testing. And so when it comes to performance testing, we use a meter or load UI or any other tool for mobile testing. We mostly use APPIUM, but not only, and when it comes to API testing, we used to select, postman or SOAP UI other tools.

Number three, I would say automate. And in order to meet the criteria of higher efficiency, we need to find a way to automate all previously created related jobs. Also, this is the first phase that makes QA Ops Framework different from the other traditional approaches in software testing. So for this, we are using tools like Apache Jenkins as your develop Maven and so on. Azure devops

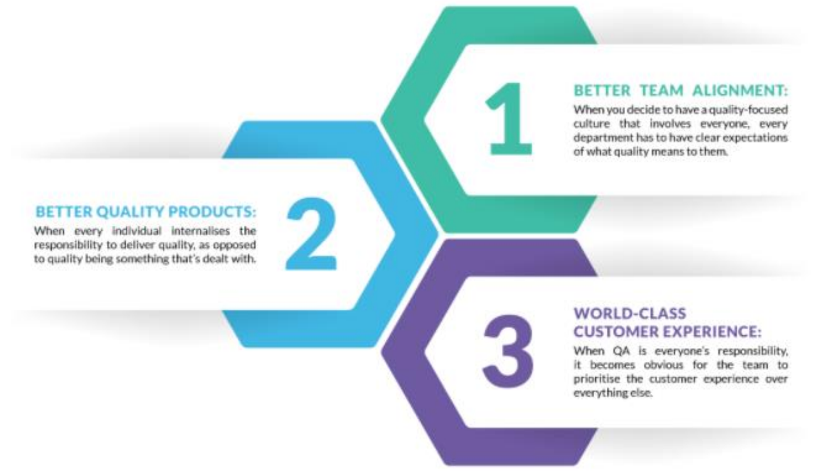
Number four is a trigger automating the job as this is not sufficient, do believe that triggering the right tests at the right time is also very important. Liek gave example we can use parallelism and also impact regression techniques.

Number five is execution. The objective of this phase is to perform real-time validation before the code goes further in pipelines depending upon the success / failure of executions.

In this phase the automated test scripts from different types of testing and executes . And in order to increase the efficiency and to reduce the costs of the automatic test combined with manual testing for verifying the final results in EndToEnd testing, at the same time , Its important to keep versioning system provisioned for from automatic tests scripts too, like we have in any other project development we do.

Last but not the least, I say it's a report and this step happens in all previous phases somehow. For example, when we open a back ticket that we have found bug during our testing, we report some anomaly in the system. However, from a client perspective, it's very important to have an overall picture of everything that happened previously and for this purpose. Tools like Jira to generate and create reports are really useful.

Benefits of QAOps



BETTER TEAM ALIGNMENT: When you decide to have a quality-focused culture that involves everyone, every department has to have clear expectations of what quality means to them.

BETTER QUALITY PRODUCTS: When every individual internalises the responsibility to deliver quality, as opposed to quality being something that's dealt with.

WORLD-CLASS CUSTOMER EXPERIENCE: When QA is everyone's responsibility, it becomes obvious for the team to prioritise the customer experience over everything else.

And here I highlighted some advantages of QAOps framework and why it matters. And of course, I mentioned about faster development cycle which I described it in the success story. It keeps the testing team continues to engage. This is really important for quality assurance team.

And I would say that quality assurance is more than development. Quality assurance people are involved in every stage of the software development and after release. And also collaboration. This is really important. And without it, this framework obviously would not work. And last, I would say, is a higher level of quality and liability

Future of QAOps

The natural progression of QAOps evolution is likely to be enhanced by the following factors:

Predictive Analytics is something that could be facilitated by the collection of data that is inherent when an integrated and effective Tooling Ecosystem is in place. The metrics generated therein could form baselines in terms of testing or project performance precedents across different type of measures such as:

- Deployment Frequency
- Post-Production Incidents
- Defect Escape Rate
- Defect & Error Rates
- And more…

These baselined or precedent measures could be used to guide other projects of like size and content both within, across, and beyond an organization.

Machine Learning, a technique of Artificial Intelligence (AI), is considered the natural next step of Predictive Analytics. The incorporation of AI would potentially enhance the analytics in place. Machine learning would be able to make assumptions, test, and learn autonomously. As a result, it could provide even deeper insight into the data at hand.

QAOps for Software Quality: Recap

Quality is imperative to the success of your application. As developing software becomes easier, faster, and cheaper, battling the competition requires more and more high-quality products. While using some quality assurance practices will help you polish the app, only a comprehensive and systematic approach will enable your company to gain an edge on the modern market.

Regarding quality assurance and testing in the context of the whole development, cycle is beneficial because it can uncover issues before they sink deeper into the app's architecture. QAOps also strives to make the complete development cycle faster. By implementing this approach, your business can achieve the impeccable quality of software while reducing the costs and time required to build the apps.