OCCAM

The Practical Guide to Implementing DevOps in Your Call Center

Work in a much more collaborative way by removing silos and aligning teams to deliver continuous improvements to your CX operations.

Making the change to a DevOps methodology within a call center cannot be done overnight.

From getting buy-in from departments, to sourcing the right technology to make it all happen, there are many steps that need to be thought through ahead of its implementation.

In this guide, we will walk you through the processes for a smooth transition and effective operation.

DevOps overview

The goal of implementing a DevOps methodology into a call center is usually to improve and shorten the system's development cycle through delivering features, fixes, and updates frequently, and where possible, automating to align with the objective(s) of the business.

The mindset of DevOps focuses on continuous testing, regular delivery and streamlining the processes between design, build, test and release phases in a software's development lifecycle.

However, DevOps is more than software and procedures, it's a mindset and culture shift. Moving from separate departments into a collaborative team, that will work together to achieve the shared objective – faster and more reliable deployments.

By adopting these practices you'll have the potential to improve customer experience with speed and quality.

Principles of DevOps

The methodology of DevOps relies on communication amongst the technical and operations teams.

Collaboration

Development and operations teams must work together and share the responsibility of the delivered release.

This requires constant and effective communication between both departments.

To achieve this:

- ☑ Agree your goals and set common objectives
- ☑ What do the end-users need?
- ☑ Produce a clear roadmap that outlines the path set to achieve your objectives

Automation

This is essential to moving to a DevOps methodology.

You will be required to automate builds, functional and regression testing, as well as being able to automate feedback on code quality and monitor the live environment.

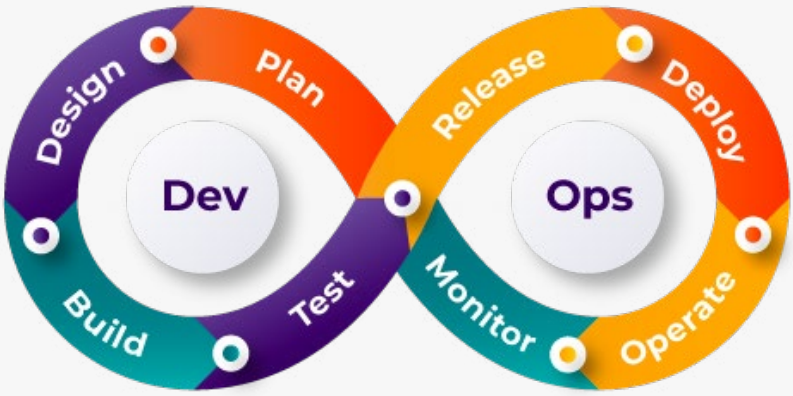
According to the State of DevOps report, high performing organizations automated:

33% more configuration management

27% more testing

30% more deployments

27% more change approval process than other teams



Continuous integration

Continuous integration helps the productivity of the developers by finding and addressing problems faster so that releases can be delivered quicker.

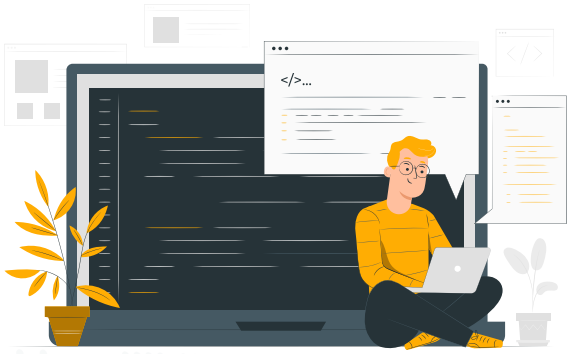
This is achieved by having a single repository for source code, automated builds, running automated testing and the team having access to the latest version at all times.

Continuous delivery

This is the ability to get software into production safely, efficiently and sustainably.

This is done by always producing code that is “ready to go into production”.

This is supported through automated testing, working in small batches and pursuing continuous improvement, helping to reduce deployment pain and higher IT performance.



Continuous deployment

Being able to deploy continuously results in better throughput and more predictable and reliable releases.

For this to happen, continuous testing needs to be in place so that you can ensure every change can be immediately tested and released without manual intervention.

Continuous testing

The practice of continuous testing relies on executing automated tests that are developed to test the end-to-end system and follow the route of the customer’s journey.

This methodology is all about testing early, faster, more often and automating wherever you can.

Automated testing will allow for UAT, functional, regression and load testing to be run regularly without putting additional strain on the DevOps team resources.

If you try and rush the implementation you will be doomed for failure.

But, if you follow our guidance and roll out your DevOps implementation in easy to manage steps, you can reap the benefits.

Where to start?

Your customer experience (CX) should be the first and foremost consideration that you plan your DevOps implantation around.

Quality in your customer service delivery cannot be compromised, so it's essential to establish early on what a good customer journey looks like.

Use experience-based suggestions from your call center agents, performance reports from your current system, as well as feedback from clients in order to plan and design products and bug fixes that will address these needs.

This approach will quickly result in positive outcomes that will justify and validate the decision to deploy DevOps throughout the business.

Starting out with this knowledge, you will be able to allocate resources proportionately, select and integrate the right DevOps tools, assign tasks to the correct teams, and streamline your production pipeline effectively.

Another benefit of starting with CX is that you will already have mechanisms in place for monitoring your CX success (NPS, CSAT etc.) and through these metrics you can quickly see the impact that DevOps can deliver.

Communication is key

When an innovation is brought into any area of the business, there will always be a level of resistance to change.

Before bringing your teams into a DevOps methodology it's important that they understand why and how this will change their day-to-day roles.

One of the biggest changes when implementing DevOps in a call center can be to your corporate culture.

If the technical department and operations teams are currently managed separately, with separate budgets and objectives, moving to DevOps will have to see a structural change if you are going to be successful.

As well as freeing up the flow of resources, you will create a much more collaborative culture by working towards customer-centric, shared goals together.

Communicating this early on in the process will make sure no one feels blindsided or at risk due to these changes being rolled out.

Focus on the pain points that the solution of DevOps is being brought in to resolve and how this will improve their work, as well as the benefits to the business and customer experience.

It's also important to highlight where any challenges may be within this change. DevOps will not mean that all manual testing will stop.

With continual testing and more resources to test all system configurations, issues or errors that may have remained unnoticed through manual testing can be identified and resolved faster, before they impact the customer's experience thanks to frequent tests, monitoring, and continuous feedback loops.

Changing culture

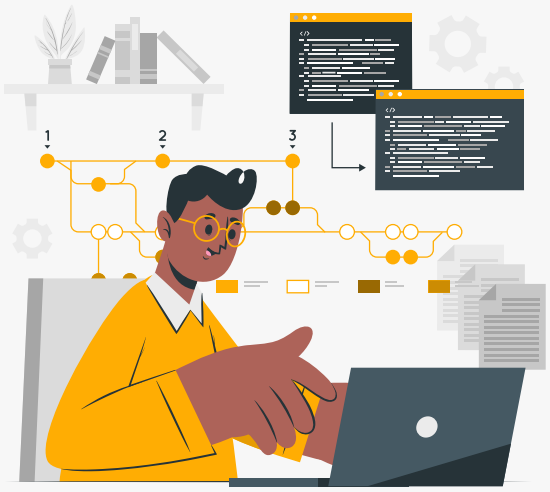
For a DevOps approach to be successfully deployed within a contact center environment, it's important to acknowledge and highlight the cultural changes that will be necessary to remove the silo mentality from the development, QA, and operations teams.

By changing the way responsibilities are split so that every member of the DevOps team is working towards the same business goal, the repetition of workload across departments will be reduced.

This will drive efficiencies when pushing a viable release into the live call center environment.

From an operations and QA perspective, they will see changes coming through regularly and should be reporting back to the development team on how these changes are affecting the operations of the business.

Creating a continuous feedback loop and sharing insights to drive the future development of the call center's environment will help to continuously improve the customer's experience and mitigate risk.



To ensure that this shift is as smooth as possible, you should look at every development innovation and how the end customer uses it.

In the past, you may have had situations where a change has taken place and the developers have built to the brief. However, how they anticipate the system being used, and how a customer actually uses it can be very different.

By always testing and working from the customer's point of view, the transition from staging into live should be as smooth as possible.

Testing individual pieces of code will now take on a more comprehensive approach in that developers will have an overview of how the code performs throughout the deployment and into the live environment.

This increased visibility will limit downtime as the DevOps team will be far more responsive to any other features in the system being affected by the changes being deployed.

Planning stage

The planning stage is where you will schedule all elements of a project and allocate resources.

To ensure speed and quality, include the following best practices:

- Get set for continuous testing by specifying which tools must be used to monitor the new deployment
- Prepare for configuration management – this refers to how your updates will be scheduled and how you will ensure consistency across all servers as you deploy the new configurations
- Launch small experiments with short feedback cycles so that your team can test and learn throughout each project

Design stage

This stage will impact everything else, so it is critical that the design of the development is done through the perspective of the customer.

This can be achieved by focusing on the common customer journey issues and identifying what can resolve them.

Having clarity at this stage, and an understanding of the anticipated outcomes, ensures that all stakeholders are in agreement and the development team has the required information to deliver the necessary changes.

Testing stage

Before the talk of continuous testing becomes a monster to be feared by your DevOps team, consider how you can automate areas of your testing process to cover more user variations automatically in minutes, instead of weeks with manual testing.

Many of the tests that are carried out before and after the deployment stage will be repetitive and easy to move from manual testing to automated.

For example, manual functional and regression testing can take weeks for your development and QA teams to complete.

However, creating scripts that can be run continuously at each change will reduce risk to your live environment by covering 90-100% of the customer journey's touchpoints, compared to an average 20%-30% with manual testing alone.

Scalability testing is another area that can be automated as you move to DevOps.

This will cut out any nasty surprises making their way into production, as you can verify the new build has not introduced any bottlenecks into the environment as you go.

This step cannot be overlooked within your DevOps process when you consider the number of options that are available to a customer when connecting with a contact center.

Test Automation Progression

1	2	3	4	5
Unit	**Integration**	**Regression**	**Performance**	**Monitoring**
Feature or focused testing on individual component	Systems communicate with one-another	You've added or updated a new feature, has it impacted any old features?	Did anything you do change how it scales? Are there non-linear bottlenecks?	Post deployment 'day 2', Ops ensure patches or config changes have not impacted CX

So, how should you implement your DevOps testing processes?

Build the foundation

To have the right infrastructure in place to implement DevOps in your call center, you will first need to have the right software in place to monitor and highlight any potential errors within your environment.

This allows repetitive tests required as a result of more regular deployments to be handled through automated testing, freeing up your developers to move into the next development sprint.

Get everyone using the same technology

To streamline your systems and processes a centralized repository for all members of your DevOps team is highly recommended.

This allows everyone to ingest information, execute tests, validate current documentation, and easily export any findings to be shared as needed.

Once in place, your call center will be able to eliminate redundant systems from your processes and individual teams to allow for a more coordinated, single platform to be used across the teams.

Standardize and reduce variability

Continuous testing allows your DevOps team to work together to create a standard customer experience testing process.

By standardizing and introducing automated testing capabilities, your contact center will benefit from a faster deployment while eradicating the chances of human error.

Once you're set-up, here are some key points to consider to help you get the most out of your continuous testing process:

1. **Functional and regression testing -** validate that the application is functioning as intended and make sure that the new changes have been effective

2. **Test in staging, verify in production -** size does matter. Your lab environment won't be as big as your production, so be sure to continue testing once in production to verify it can handle the capacity, as well as any cloud or remote-based call center resources that will be needed

3. **Test the full customer experience journey -** you will need a test sandbox instance so you can test all of the end-to-end customer journey options

4. **Test for success, prepare for failure -** understand how the environment is supposed to scale and plan testing around this. Be prepared for testing to throw up expected errors and build the time needed to fix them into your timescales

5. **Collaboration is key -** ensure you are involving subject matter experts of the full technology stack so that everyone is aware of what changes are coming, and assist on any required troubleshooting

6. **Establish performance baselines with load testing, monitor new deployments in production -** by continuously testing your environment, with every new deployment to monitor any performance issues as they arise

Once your testing procedures are set up, you can enter the phase of fine-tuning the best way to manage testing and the errors that are found.

Creating an information feedback loop is central to a successful DevOps implementation. Your development team can anticipate and resolve issues in the pre-production environment by having an accurate overview of the live environment, granting them more independence to take a proactive approach to bug fixing without relying on QA or operations.

The latter teams will be notified of changes thanks to the documentation process you have put in place.

Release stage

Once testing has confirmed that the codes perform as expected, and at scale, the hand-off process can begin.

There should also be monitoring to capture any customer experience (CX) issues as it moves into production.

Load testing should also be carried out to confirm that customers do not experience any problems by simulating real-world traffic volumes and testing across all channels in the customer journey.

Monitor stage

Continuously monitor your environment and capture any potential points of failure so that you can ensure a reliable service for the end-to-end customer journey. Anything that comes up at this stage can be used to loop back into planning and the cycle can begin again.

Pitfalls to watch out for

As with any change within a company, challenges can arise. Having the foresight of what could become a hurdle in your DevOps implementation, and having the strategies to overcome them, you'll be in the perfect position to deal with any hiccups you may experience.

Not having a lead

For your DevOps migration to be successful you will need someone to take ownership of the project to see it through and keep everyone else accountable.

Remember, this isn't just about the strategic investment but the cultural changes that will be needed too.

CX doesn't just stop at the DevOps team: sales, marketing, and customer services are also going to be affected by the changes that are introduced to your internal structure.

Whoever this role is assigned to will need to have broad authority and budget over people and processes.

Not aligning your objectives

DevOps can only be successful when teams work collaboratively.

Ensure that each area of the team's objectives align with their function within the process and complement the business objectives around customer satisfaction.

For example, the development team's objective may be around the speed or the number of fixes needed after deployment, whereas operation's objectives can be around high reliability.

Not setting clear KPIs

It's important that your KPIs work for the collaborative structure of your DevOps team and that everyone is aware of what they are being measured against.

Critical KPIs should be widely distributed across the organization so that CX can be viewed at an operational level.

Not having the right processes in place

If the implementation of DevOps ends up in late nights and long hours to get the release deployed, or worse still, to fix issues in the live environment, there has been an oversight in the processes.

This will usually come down to the testing, if testing has not been automated or implemented correctly, bugs and issues will arise later than expected and cause bigger issues.

Making sure you have an automated testing process and software that provides your team with the support needed will remove the chance of this occurring.

During the process development for your migration, you will also want to evaluate any manual processes you have in place across the teams and look for efficiencies and improvements that can be made during the transition.

Not only can this reduce the chance of human error, but you will also be likely to avoid project delays or quality issues due to lack of resources.

Not streamlining environment migration

The biggest frustrations during development and deployment tend to come from moving between different environments, from development to QA, through to staging and production.

By having a centralized repository you can avoid having to rework elements of the deployment or the introduction of bugs as code moves through the different stages.

Measuring the user experience

You will have to define a set of metrics so that you can use an automated monitoring tool to spot any changes to the user experience based on the improvements you have implemented.

By measuring and testing this regularly, you will be able to clearly see the impact your DevOps has had on overall CX and pinpoint areas that can be optimized next.

As you approach the implementation of DevOps within your call center, take the time to review each stage we have discussed above to make sure all your bases are covered, and that all members of the newly collaborated team are on the same page every step of the way.

We have worked with many organizations to help get their tool stacks set up and ready to move to a more agile development cycle.

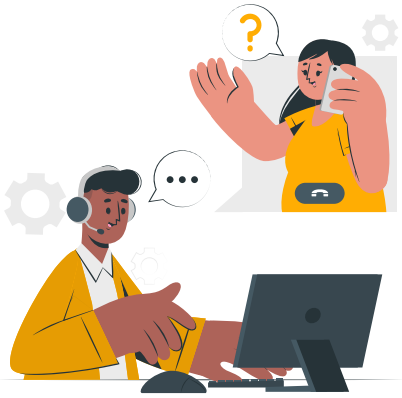
If you would be interested in finding out more about how Occam's Razor can benefit your contact center, get in touch via our website by clicking here or through our contact details below.

Facilitate the constant change of all Call centers

Request your free demo call and find out how you'll be able to:

- Create a collaborative work space
- Reduce Team Downtime
- Deliver superior CX

Book your FREE demo



Reduce your time spent manually testing

Request your free trial today and see the benefits of:

- Early identification
- Flexible deployment options
- Unlimited testing

Request your FREE trial

OCCAM

www.occam.cx