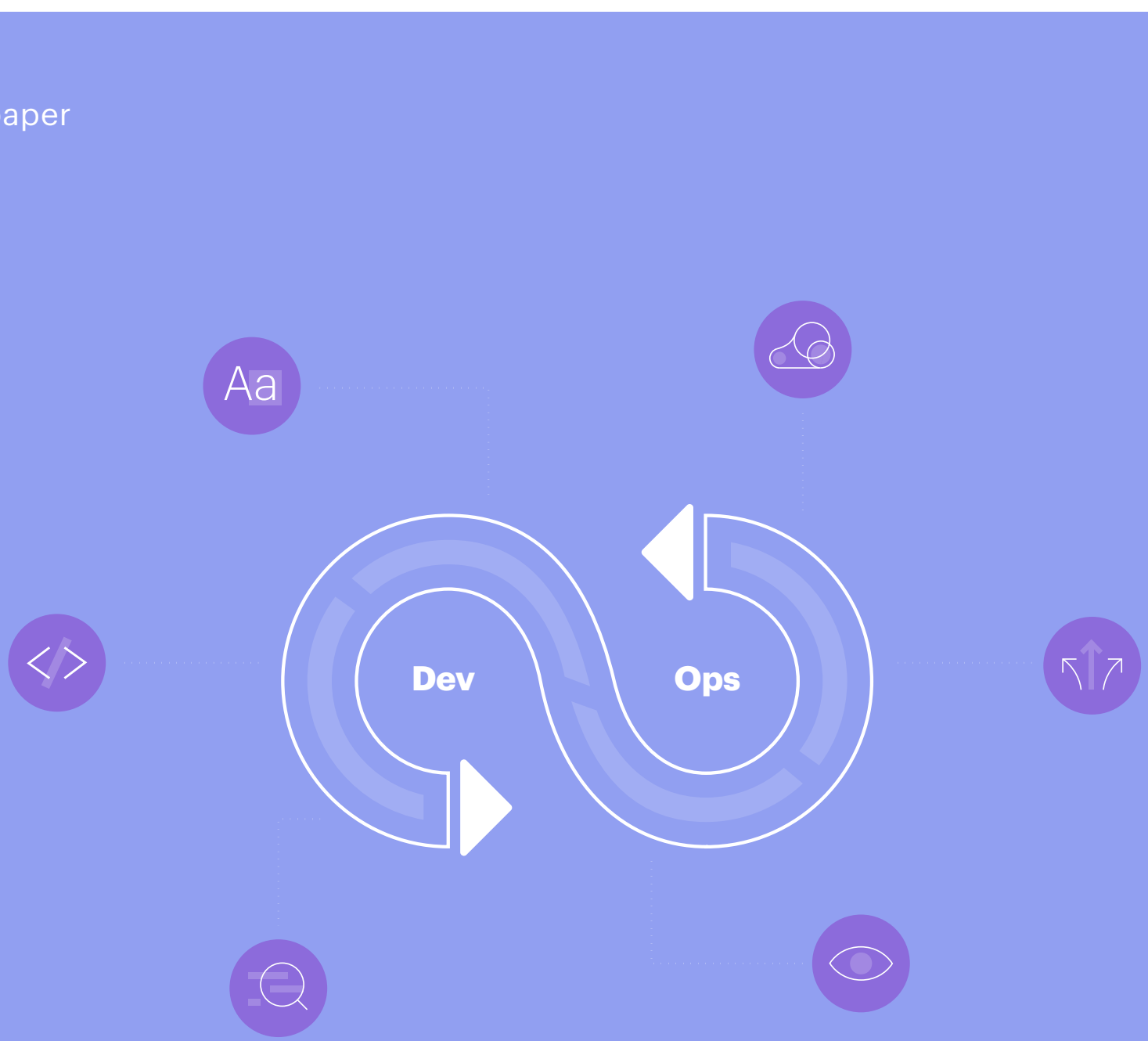
DevOps:
Implementation Guide

Whitepaper



GeneXus™
by Globant

Today's market calls for the development and release of software functionalities to become increasingly agile.

To fulfill such requirement with an ongoing delivery of value, processes –in addition to tools and working methods– must be subjected to assessment and changes by means of a method, pertaining to software engineering, called DevOps.

The name, invented by Belgian-born technology consultant and programmer Patrick Debois, is a combination of the terms "development" and "operations" that summarizes the benefits of dismantling, with the implementation of agile tools and processes, the barriers that exist between development and operations teams.

In 2009, the word gained significance worldwide with the launching of an event known as Devopsdays which was organized by Debois), that had widespread effects on social networks and captured the attention of the industry's major players like IBM.

Below is a detailed description of the changes required in organizations willing to implement a DevOps culture, including the processes and tools to be used for ensuring the delivery of an enhanced product.

Reasons for getting started in the DevOps world

Technology companies are aware of the significance of having agile and ongoing releases of their products in the market. However, they face difficulties in finding the best way to solve situations such as:

- the barriers existing between different work teams;
- the smoothness of internal cooperation;
- the best use of resources available;
- streamlining the production of applications;
- reducing time of failure in services;
- increasing automation; and
- computer security.

DevOps is the working method for solving many of the aspects mentioned above.

With a group of specific practices, this methodology brings a novel way of thinking and approaching situations in order to reduce barriers between people, while improving automation of processes throughout the value chain.

«DevOps focuses on principles, practices, and results. For a shift towards DevOps, it is of the essence to ensure that teams are well prepared, in addition to understanding everything that surrounds that culture, and adjust processes accordingly bearing in mind actual needs, to end up with the installation of the technology that aids in implementing these practices»

DevOps Consultant, Genry Leiva at the Doing DevOps with GeneXus webinar that also included Software Engineer Enrique Almeida.

During an [interview by Devopslatam](), Debois affirmed that **the key to team integration is understanding problems that colleagues from other areas might encounter.** “Sometimes, we think that others bring their problems on us and consider them opponents, but it’s not so. We must be comprehensive and capable of discussing ways to make things work out better. Understanding your own problems and the problems of other is the first step to take. Afterwards you can start aligning the business and improve processes, and there small victories will be noticed. It’s important to build a chain with such achievements, to be aware of what will be in your hands or what will start to happen.”

the key to team integration is understanding problems that colleagues from other areas might encounter.

Initial recommendations

· Be patient

It's important to understand that it's not possible and you should not change everything from the start.

· Define an iterative and incremental process

It's convenient to start with small and scarcely ambitious objectives with the idea of assessing results to start improving.

· No need to undo current development processes

DevOps practices may be implemented alongside the work that is done within the organization.

· No silos

Teams doing work separately and independently should be avoided.

· Shared responsibility

A sense of group responsibility must be fostered towards maintaining the product and building new projects.

· Collaboration

Meetings are a good integration method for individuals to become aligned once they all know what everyone else is working on. This makes project planning and preparation easier.

· Empathy

Diverse opinions can lead to disagreement and conflict, and the best way to solve those situations is for the parties to talk in order to comprehend all arguments and determine if it's possible to reach a solution for the various teams. In the end, it's a matter of being empathic and working together.

· Think about the system

It's important to have a global vision of everything happening. It's not about everyone fixing their own part. Because systems are complex, they must be considered globally.

Getting started

Companies might encounter difficulties in adopting the DevOps culture. To get a clearer idea on how to get started, consider the following basic method in the implementation of DevOps projects:

Stage 1 | General analysis

- ☑ Evaluate the organization's **maturity model.**

- ☑ Create a proposal for process adjustments, including the metrics to be measured.

- ☑ Develop a training plan for the teams.

- ☑ Review the tools available in the organization, and their scope and limitations, since the objective is actually not adding new technology but rather making the most of the resources at hand.

Stage 2 | Work plan

- ☑ Define a work team with members from different areas. This group will begin the evangelization process and the changes in paradigm for the organization.

- ☑ Create a Minimum Viable Product (MVP) and an action plan. At this point, the first practices to be implemented should be defined, regardless of whether it is a small project or a project in a specific development area.

- ☑ Carry out training and workshops, including those not detected in Stage 1 but appearing in this stage.

- ☑ Implement all tasks necessary for the MVP.

- ☑ Provide feedback to the staff involved.

Stage 3 | Review of results and plans for following project

- ☑ Analyze results in accordance with the scope proposed.

- ☑ Provide feedback on the adoption of these practices and the results obtained.

- ☑ Plan the next practice to be included.

Get on it!

1. Teams

In many companies, it's common for teams to work separately from one another, so, unification is one of the main changes required.

"Unified teams are possible regardless of whether they work at the same location or not. In either case, the teams must be available to work jointly and while connected as a sort of tribe," explained GeneXus Technical Support Manager Silvia Keymetlian.

The dictionary of the Royal Spanish Academy defines Tribe as a group whose members share customs they have in common. In the tech industry, tribes are organized teams with 5 to 100 members (and most ideally not more than 100), who work in a connected way towards a common interest.

Tips for organizing teams:

Think big. DevOps teams are made up not only by people from the development and operations areas, but also from testing, design, and IT security, among others. Such mergers give way to names like:

- **DesignOps**

Integration of the design team with the development team.

- **DevSecOps**

Integration of the computer security, development and operations team.

Testing. It's ideal do testing at all stages in the project. It should be borne in mind that some methodologies are based on test-driven development to start developing from tests.

Pair programming. This common and effective practice consists of developing the system in pairs (both code and tests). This is also useful for revising code.

2. Processes

Architecture. To determine the type of architecture that should be used, it's necessary to consider the pros and cons implied in each option. We mentioned that markets call for quicker software releases, so, **working with monolithic architectures is not ideal** in such cases.

"When you need to make changes in a functionality or move a field on a screen, then the whole system must be released and that could prove problematic. It may also happen that another part of the application is still not ready when it's time to release those changes, so, for example, those whose part is ready must wait for the rest of the team. This causes chaotic and conflictive cycles that are not at all agile. Therefore, application architectures now tend to be modularized, making the system and the architecture as disconnected as possible," said Keymetlian.

Infrastructure. "The cloud is the most used resource these days. A good advice for managing the infrastructure is to write a script with all the requirements for building the server and then execute it every time that such environment must be created. This helps avoid human errors and guarantees that the configuration is done in the same way.
Once the system's architecture has been defined, it is necessary to consider things that must be done daily to achieve an agile process, in addition to turning automatic as many tasks as possible to make things flow towards the agile delivery we seek."

Analysis. With the app in production, an analysis must be carried out with the purpose of learning and assessing ways to improve the code.

"It's important to understand the code's behavior during production. Despite the numerous tests done on the app, it is possible that a traffic peak was never reached, so this information could be obtained during the production stage," added Silvia.

Metrics. Measurement results will indicate the aspects to be adjusted in the plan. Some of the variables to measure are:

- the term implied for delivery of the changes;
- he frequency of deployment;
- the time required to restore the service in the event of a failure. At this point, you must act quickly overcome problems and immediately release the fix, which must go along all steps in the pipeline;
- failure rate; and
- work satisfaction.

Tests. Tests should be carried out throughout the process. Automated tests will aid towards the ongoing delivery of products, including speed and quality. In the case of failures, then that information will be available before the app goes to production.

3. Automation

When the processes are already defined, all possible tasks must be automated so that everything flows and agile deliveries are achieved.

Automation transfers to computers all the repetitive tasks that don't call for human thinking and this enables the team to focus on problems that come up and their possible solutions. Automations help ensure that not a single step is missed.

Automation transfers to computers all the repetitive tasks that don't call for human thinking and this enables the team to focus on problems that come up and their possible solutions.

The following are some of the tasks that should be automated in order to define a pipeline:

- ☑ Code
- ☑ Build
- ☑ Test
- ☑ Integrate
- ☑ Deploy
- ☑ Release

Continuous Integration. It consists of automated integration and continuous validation. For example, the code must be tested individually first; and then, if it works well, it may be integrated in the whole system. This method reduces the probability of errors and minimizes possible conflicts.

Continuous Delivery. It's important that automation range from implementation all the way to production cutover and the preparation of the release.

Continuous delivery is achieved by automating the different stages of the development process, which leads to a package with everything necessary for the app to go into production. The only thing done manually at this stage is copying the package, with just one click, to the server where the application will be made available.

DevOps. This stage is reached when the process includes the app's operation and monitoring. This generates the feedback information for the plan, as well as the actions to be taken.

DevOps in the GeneXus world

Many organizations –like Microsoft, Amazon and Google– have used DevOps for increased agility and for enhancing the release of their applications. GeneXus has also joined this idea by making changes in working methods and in its releases, seeking to provide higher value to the solutions created by the community.

GeneXus has included DevOps in the different stages of testing, static code checking, deployment tasks, and the monitoring of apps in production, among others.

DevOps is capable of covering more parts of the app's extended development cycle, and it's applicable to both large development teams and small groups of developers.

In line with the premise of automating everything liable of being automated, GeneXus has available a wide range of tools that allow constructions, deployments, API checks, performance checks, unit tests, integration tests, and much more.

Having a single reference Repository minimizes manual errors and leads to an optimal level of traceability capable of identifying, from the version installed to the fixes and data of changes made, as well as unwanted changes.

GeneXus has included DevOps in the different stages of testing, static code checking, deployment tasks, and the monitoring of apps in production, among others.

Basic recommendations for doing DevOps with GeneXus

Use GeneXus™ Server, the Software Configuration Management tool to work with GeneXus that also enables the unification of all automation tasks associated with DevOps.

Make sure that the entire DevOps team has knowledge about testing processes.

Automate tests from the beginning of the development process.

Carry out functional tests.

Boost agility, automation, security, testing and monitoring by using tools such as GeneXus Server, GXtest, Modules, Log API, Deployment Units, MS Build, Jenkins, and Docker Containers.

GeneXus tools for DevOps

The following are the tools offered by GeneXus for automating the entire DevOps process:

GeneXus™ Server

The product that automates the integration of knowledge while improving teamwork capabilities and with no additional integration costs (enables the inclusion, in the project, of managers or the client involved for them to supervise the system's development).

This tool offers the possibility to create and monitor Continuous Integration Pipelines that allow for the overall integration and automation of the Continuous Integration process.

GeneXus™ Server stores complete information on changes (when and why they were made) in a historical log that aids in understanding the evolution of objects, and in identifying changes that could have caused a specific error, with the possibility of a quick solution to it.

Modules

Modules constitute the first thing that is found when building Knowledge Bases (KB). They help organize so developers may know which modules they must work on. Later, this functionality allows them the flexibility to pack and distribute the modules, making architecture changes easier, when necessary, by shifting from a monolithic system to other options.

Import of external designs

For scenarios requiring fully customized experiences for a very particular reality, the functionality of importing external designs from Sketch and Figma has been enhanced to allow designers to work in their favorite tool and reduce friction to get from the design to a pixel-perfect implementation.

GXtest

GXtest is a product developed to design, automate and execute functional tests in web and mobile applications developed with GeneXus. The best part of it is that GXtest users need no programming knowledge, because GXtest is intuitive and very easy to use.

Containers

GeneXus offers the option to deploy in Containers and in other environments like IBM, Amazon, Google, Azure, SAP, Dockers, and Kubernetes, to help make the app's deployment much faster and without the installation complications of installing a server. All this is programmable and automatable, and thus easy to include in the DevOps process.

Deployment Units

This is a deployment made from the information contained in an object. Even when having only one Knowledge Base (KB), this functionality facilitates the creation of other deployment units for when the time comes to migrate from a Monolithic Architecture to a Micro Services Architecture. It is possible to have Deployment UNIT for batch processes, for mobile services, for the frontend, and for the backend.

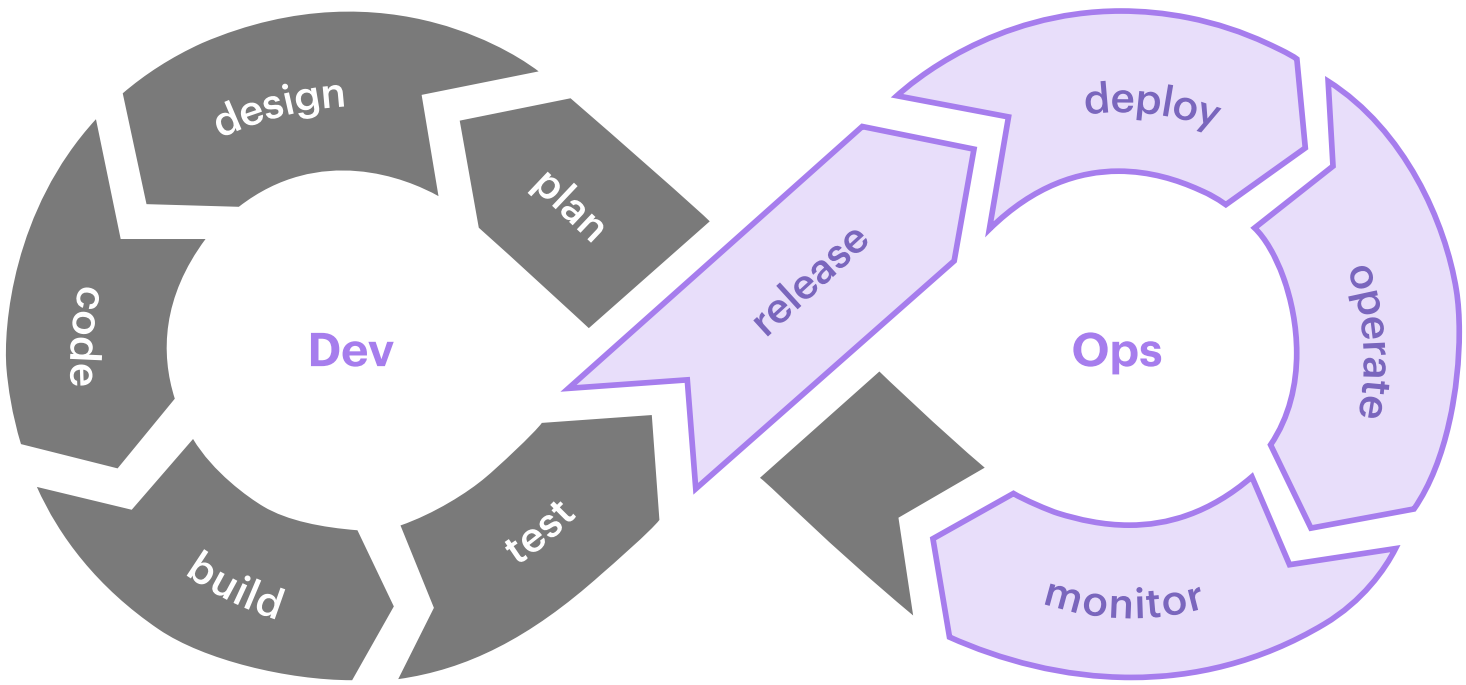
Log API

Allows the user to monitor the application by recording (on the Log files) one or several of the processes that are running.

With this, information is obtained regarding the app's behavior, which in turn provides feedback information for the plan, allowing for everything to flows permanently.

SSO GAM

SSO GAM enables the use of a single user login between two different web apps (this avoids a new request for login every time).



Find out how GeneXus can do the same for your company.

Contact Us

info@genexus.com

GeneXus™

by Globant

MONTEVIDEO - URUGUAY	Av. Italia 6201- Edif. Los Pinos, P1	(598) 2601 2082
CIUDAD DE MÉXICO - MÉXICO	Hegel N° 221, Piso 2, Polanco V Secc.	(52) 55 5255 4733
MIAMI - USA	8950 SW 74th Ct, Suite 1406	(1) 201 603 2022
SÃO PAULO - BRASIL	Rua Samuel Morse 120 Conj. 141	(55) 11 4858 0300
TOKYO - JAPAN	2-27-3, Nishi-Gotanda	(81) 3 6303 9381
	Shinagawa-ku, Tokyo, 141-0031	(81) 3 6303 9980