06_introduccion_a_devops

October 15, 2020

1 Introducción a DevOps.

2 ¿Qué es DevOps?

- Es una contracción de "Operaciones de Desarrollo".
- Muchas personas y organizaciones tienen su propias definiciones sobre "DevOps".
- El término "DevOps" fue utilizado por primera vez en 2009 por Patrick Debois.

3 Definición de "DevOps".

"Conjunto de prácticas enfocadas a reducir el tiempo entre el registro de un cambio en un sistema y el momento en el que el cambio es puesto en un entrono de producción garantizando la calidad de éste".

4 DevOps puede considerarse como una 'cultura'.

Una cultura en la que las organizaciones de desarrollo de TI se enfocan en crear grupos de trabajo mutidisciplinarios que facilitan la comunicacion, control, seguridad, calidad y agilidad del ciclo de vida del software.

5 ¿Conceptos ligados a DevOps?

- Desarrollo ágil.
- Cómputo en la nube.
- Infraestructura como código.
- Automatización.
- Microservicios.
- "Full stack" developer.

Desarrollo ágil.

Aún cuando es posible hacer DevOps con metodologías más convencionales, DevOps nació de los conceptos de agilidad.

- Se basa en ciclos de desarrollo cortos con objetivos y entregables especificos .

- El producto (software) evoluciona conforme los ciclos avanzan.
- Los grupos de trabajo son multidisciplinarios y por lo general autogestionados.
- Sin embargo, DevOps puede funcionar con otras metodologías.

6 ¿Cuál es la promesa de DevOps?

6.1 Autoservicio.

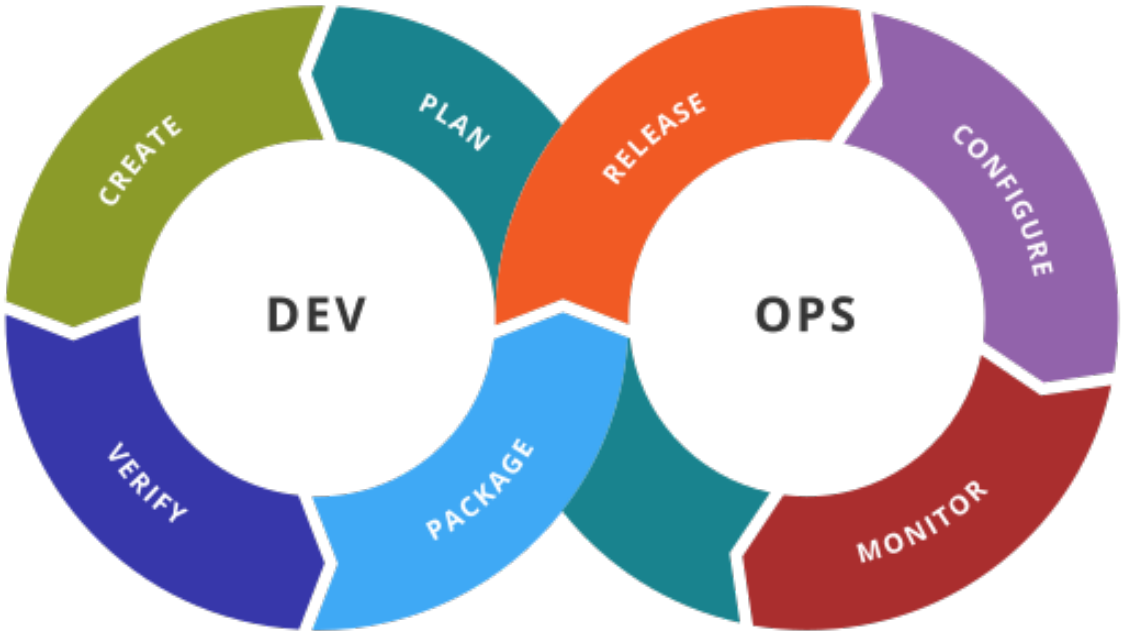
6.2 Autogestión.

6.3 Entrega e integración continua.

6.4 Colaboración.

6.5 Escalabilidad.

7 La cadena de DevOps.



Fuente: Wikipedia

8 Componentes en el desarrollo (ágil) de un producto de software.

- Control de versiones.
- Gestión y aprovisionamiento de entornos.
- Gestión de artefactos.
- Automatización.
- Pruebas y QA.
- Integración.

- Entrega/Despliegue.
- Comunicación.

9 El "stack" de desarrollo de este curso.

9.1 Control de versiones.

- El sistema de control de versiones más popular actualmente es *Git*, el cual permite crear repositorios distribuidos de forma local o remota.
- Con base en *Git* es posible acceder a servicios que ofrecen repositorios remotosm tales como:
  - *Github*.
  - *BitBucket*.
  - *GitLab*.

Cabe hacer notar que la mayoría de los servicios de repositorios remotos han evolucionado para ofrecer sus propios "stacks" de DevOps.

9.2 Aprovisionamiento de entornos.

Los entornos en DevOps pueden ser aprovisionados ya sea mediante:

- Máquinas virtuales, los cuales son sistemas que pueden emular en diversos grados sistemas de hardware.
- Contenedores, los cuales permiten crear diversos "ambientes" que contienen su propia estructura de archivos y bibliotecas dentro de un sistema anfitrión. En esta ocasión se utilizará *Docker* de forma básica.

Uno de los fines últimos de DevOps es ofrecer "Infraestructura como código" (IaC), de tal forma que los entornos virtualizados o contenerizados puedan ser repoducibles, administrables y orquestados mediante scripts que definen desde un servidor hasta un cluster de servicios.

9.3 Gestión de paquetes y binarios.

Un proyecto de desarrollo de software rara vez puede ser contenido en un único archivo binario que no haga uso de una biblioteca de recursos y dependencias.

En el caso del entorno de *Java* existen herramientas capaces de automatizar de forma eficiente el cálculo de dependencias y la construcción de binarios. Estas herramientas en muchos casos pueden ser utilizadas concurrentemente.

En este curso se explorarán las herramientas:

- *Apache Ant*
- *Apache Maven*
- *Gradle*

9.4 Gestión de artefactos.

Un "artefacto" es un recurso que forma parte de un proyecto de desarrollo de software y que a difrencia de un repositorio de código, el cual está conformado primordialmente por archivos de

texto, puede consistir en:

- Documentación del proyecto.
- Archivos de configuración.
- Bilbiotecas.
- Imágenes de máquinas virtuales o contenedores.
- Scripts de pruebas.
- Bases de datos.
- Binarios de diversas índoles.

Durante este curso se explorará el uso de *JFrog Artifactory*.

9.5 Automatización.

La herramientas basada en Python más popular para realizar operaciones de automatización de la configuración genérica es *Ansible*.

- Esta herramienta es capaz de realziar operaciones automatizadas en uno o cientos de servidores tan solo con Python y SSH.

9.6 Pruebas.

El desarrollo orientado a pruebas (*TDD*) es una disciplina que forma parte esencial de DevOps. La única forma de contar con procesos de desarrollo ágil eficientes y seguros es mediante el diseño y automatización de pruebas de distintas índoles una vez que el código está listo para ser integrado a un proyecto.

Este curso está enfocado a explorar las bases de *TDD* e integrarlo como parte fundamental del "pipeline" de desarrollo.

Las herramientas de pruebas que se explorarán e integrarán en este cursos son:

- *JUnit* para realizar pruebas en la JVM.
- *Selenium Web Driver* para pruebas de aplicaciones web.
- *Postman* para desarrollo y prueba de web API.

Cabe aclarar que una vez estudiadas estas herramientas de forma aislada, se explorará la forma en la que estas pueden ser integradas en el proceso de creación de artefactos y binarios tanto con *Gradle* como con *Jenkins*.

9.7 *CI/CD.*

La integración, entrega y despliegue continuo (*CI/CD*) del código a un entorno de producción de forma inmediata es el fin último de las metodologías de desarrollo ágil y de DevOps. Todos los esfuerzos y herramientas de una organización ágil de desarrollo de software se enfocan en impementar de forma exitosa un pipeline eficiente y seguro de *CI/CD*.

La herramienta de CI/CD de código abierto mas popular es *Jenkins*, la cual es una herramienta de automatización de proyectos de software, que incluye una gran diversidad de plugins que le permite automatizar la creación de binarios, realización de pruebas y distribución de artefactos de forma flexible y compatible con una gran cantidad de "stacks".

9.8 *DevSecOps.*

La promesa de *CI/CD* implica que el código y la integración de este a un proyecto de software sea seguro. *DevSecOps* nació a partir de la necesidad de desplegar aplicaciones y sistemas seguros desde el código.

Durante este curso se explorarán las herramientas.

- *SonarQube* para validar la calidad y seguridad del código.
- *AppScan*.

9.9 En caso de dudas.

Exsite una enorme cantidad de herramientas y frameworks que pueden ser aprovechados para DevOps y a partir de ahí pueden diseñarse una inmensa cantidad de "stacks". Conocer todas las herramientas, frameworks y stacks es imposible.

Sin embargo, el sitio *StackShare* es un lugar por el que se puede empezar a evaluar, comparar y conocer sobre la mayoría de estos recursos con la colaboración de una gran comunidad de colegas.