

Semana Cuatro

Fecha	13 - 17 marzo 2023
Nombre	Olivia Yuyu Maceda Pérez

Ejercicios Xideral

- **Explicar qué es y qué consiste DevOps?**

DevOps combinación de los términos ingleses development (desarrollo) y operations (operaciones), designa la **unión de personas, procesos y tecnología para ofrecer valor a los clientes de forma constante.**

DevOps permite que los roles que antes estaban aislados (desarrollo, operaciones de TI, ingeniería de la calidad y seguridad) se coordinen y colaboren para producir productos mejores y más confiables. Al adoptar una cultura de DevOps junto con prácticas y herramientas de DevOps, los equipos adquieren la capacidad de responder mejor a las necesidades de los clientes, aumentar la confianza en las aplicaciones que crean y alcanzar los objetivos empresariales en menos tiempo.

Métodos de DevOps

-**Scrum.** Scrum define la forma en la que los miembros de un equipo deben colaborar para conseguir entre todos acelerar los proyectos de desarrollo y control de calidad.

-**Kanban:** Kanban prescribe que el estado «en curso» (WIP, del inglés «work in progress») de un proyecto de software debe controlarse en un tablero Kanban

-**Agile:** se estipulan ciclos de desarrollo de software más cortos en lugar de los tradicionales métodos de desarrollo «en cascada» que se prolongaban en el tiempo.

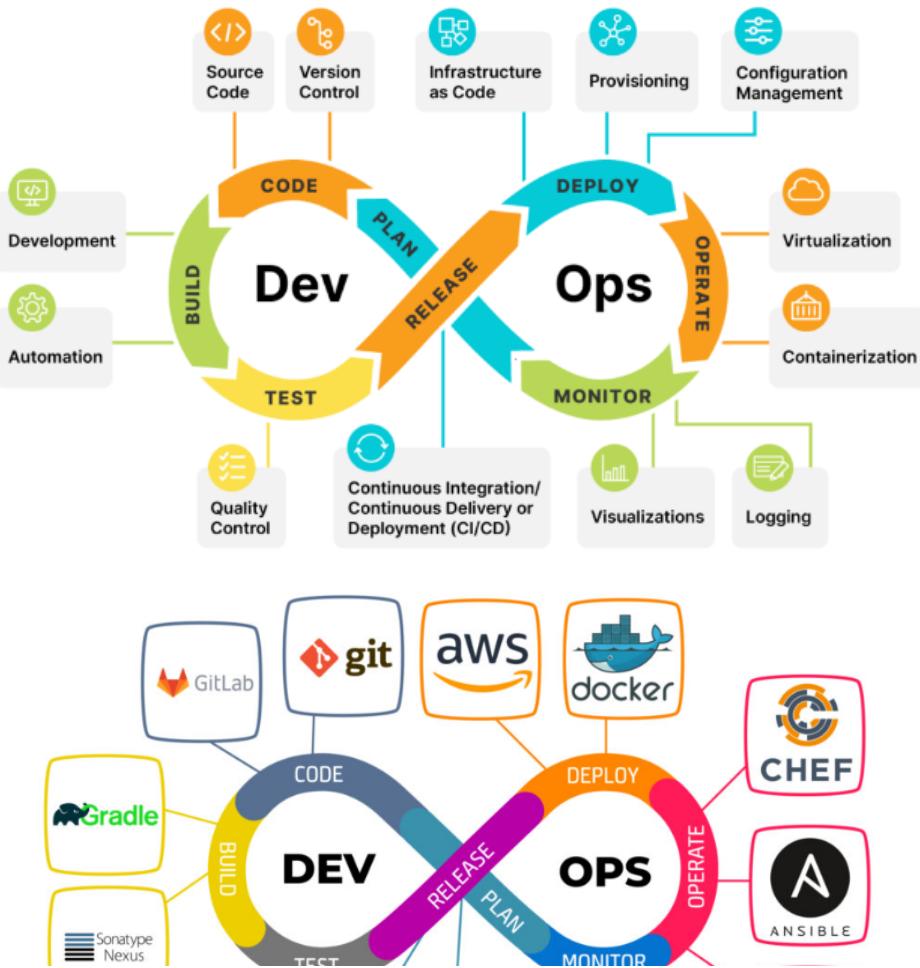
Cadena de HERRAMIENTAS de DevOps

*Promueven postulados principales de DevOps -> **automatización, la colaboración y la integración entre los equipos de desarrollo y operaciones.**

- **Planificación.** Define los *requisitos y valores empresariales*. (Jira o Git para hacer un seguimiento de los problemas conocidos y llevar a cabo la gestión de los proyectos).
- **Codificación.** Diseño del software y la creación del código.
- **Compilación.** Gestión de las *versiones y las compilaciones* del software, utilizando herramientas automatizadas que ayudan a compilar y crear paquetes de código para publicarlos después para la producción.

«empaquetación» -> Docker, Ansible, Puppet, Chef, Gradle, Maven o JFrog Artifactory.

- **Prueba.** Realización de pruebas continuas (manuales o automatizadas) para garantizar la calidad de la programación. Ejm JUnit, Codeception, Selenium, Vagrant, TestNG o BlazeMeter.
- **Puesta en marcha.** Uso de herramientas que ayudan a *gestionar, coordinar, programar y automatizar las tareas de producción de las versiones de productos*. Ejm Jenkins, Kubernetes, OpenShift, OpenStack, Docker o Jira.
- **Funcionamiento.** Gestión del software durante su producción.
- **Supervisión.** Se *identifica y recopila* información sobre problemas que surgen en una versión de software específica que se encuentra en producción. Ejm herraientas: New Relic, Datadog, Grafana, Wireshark, Splunk, Nagios o Slack.





Prácticas de DevOps

*Basadas en una o en varias fases del ciclo de desarrollo.

- **Desarrollo continuo.** Abarca las *fases de planificación y codificación* del ciclo de DevOps. Puede incluir también mecanismos de control de versiones.
- **Realización de pruebas continuas.** Incorpora continuas *pruebas de código automatizadas y programadas* con antelación que se realizan a medida que el código de aplicación se está creando o actualizando. Gracias a estas pruebas, el código pasa antes a la fase de producción.
- **Integración continua (CI).** En esta práctica se combinan herramientas de gestión de configuración (CM) con otras herramientas de pruebas y desarrollo para saber qué *cantidad del código que se está creando* está *listo para pasar a producción*. Para ello, debe existir un **intercambio** fluido de información entre las **fases de prueba y de desarrollo** que permita identificar y resolver con rapidez problemas en el código.
- **Entrega continua.** Esta práctica automatiza la *introducción de cambios* en el código para pasar a un entorno de *reproducción o de almacenamiento provisional* tras la fase de pruebas. Un miembro de la plantilla podría entonces decidir si es conveniente promover estos cambios de código a la fase de producción.
- **Puesta en marcha continua (CD).** Automatiza el *lanzamiento de código* nuevo o modificado a la fase de producción. Tecnologías de contenedor, como Docker y Kubernetes, hacen posible esta fase de puesta en marcha continua al ayudar a mantener la coherencia del código entre los diferentes entornos y plataformas de puesta en marcha.
- **Supervisión continua.** Implica la supervisión continua del código en la fase de producción y la infraestructura subyacente que la sustenta. A través de un bucle de retroalimentación en el que se notifican errores o problemas, este podría volver a la fase de desarrollo.
- Infraestructura como código. Se puede utilizar durante varias fases de DevOps para automatizar el aprovisionamiento de la infraestructura que se necesita para publicar el software. Los desarrolladores añaden "código" de infraestructura procedente de las herramientas de desarrollo actuales.

Ventajas

Mejoran el rendimiento y crean productos de más calidad en menos tiempo, lo que aumenta la satisfacción de los clientes.

- Reducción del tiempo de comercialización
- Adaptación al mercado y a la competencia
- Mantenimiento de la estabilidad y la confiabilidad del sistema

- Mejora del tiempo medio de recuperación

- **Proyecto SpringBatch**

En qué consiste el proceso SpringBatch -

- **Proyecto SpringBootRestJPA con manejo de excepciones - usuario & Compras**

Cliente que se conecta a un servicio - JSON.

“templateRest: Se conecta para dejarlo como un pojo

*Manejo de excepciones en mundo REST

Fuentes:

¿Qué es DevOps? Explicación de DevOps | Microsoft Azure. (s. f.). <https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-devops>