Universidad Intercontinental de la Empresa

Asignatura

Arquitectura del Software 

Profesor

Yago Fontenla Seco

{*yago.fontenla1@uie.edu*}

¿Qué es DevOps?

DevOps es un enfoque cultural, organizacional y técnico que une Development (Dev) y Operations (Ops) con el objetivo de entregar software de forma más rápida, confiable y continua.

No es una herramienta o un rol, sino una forma de trabajar.

Promueve la **colaboración, automatización y mejora** continua en todo el ciclo de vida del software.

DevOps y Arquitectura

Conjunto de **principios y prácticas** que **operativizan la arquitectura**, permitiendo **entrega continua de valor**.

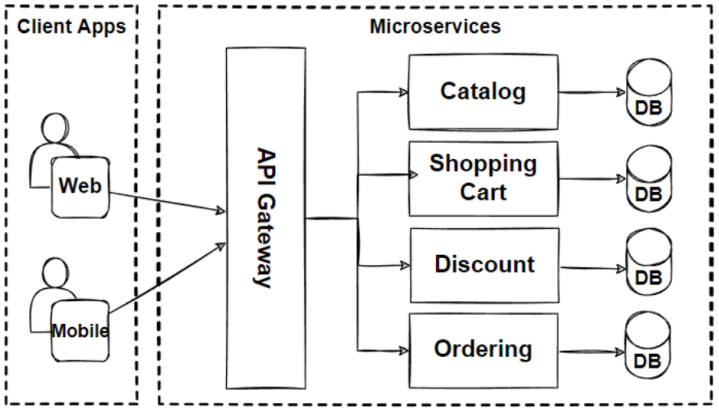
La arquitectura del software define **la estructura técnica, los componentes y sus interacciones**. Pero una buena arquitectura no solo busca **funcionar bien hoy**, sino **evolucionar rápidamente sin romperse**.

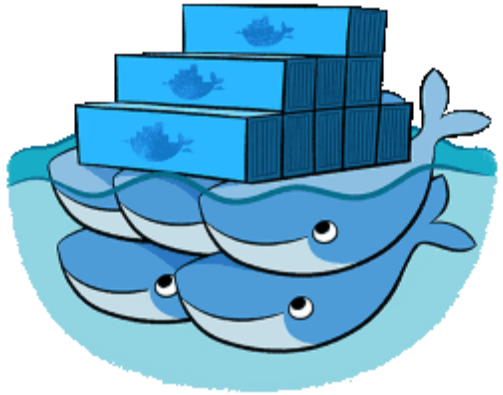
DevOps es **la cara operativa de la arquitectura ágil**. Las decisiones arquitectónicas deben permitir:

- **Automatización de despliegues.**
- **Monitoreo en tiempo real.**
- **Pruebas y entregas continuas.**

DevOps

Las arquitecturas basadas en **microservicios** y **contenedores** (Docker, Kubernetes) son habilitadores naturales del enfoque DevOps. *Una arquitectura que no permite automatizar su entrega o monitoreo, no está preparada para DevOps.*





¿Por qué DevOps?

El mundo digital actual exige velocidad, calidad y estabilidad simultáneamente. De forma similar a las metodologías ágiles, DevOps surge para **cerrar la brecha entre desarrollo y operación**.

Antes (modelo tradicional o en cascada):

- Equipos de desarrollo y operaciones separados.
- Entrega tardía: meses o años entre versiones.
- Problemas típicos: “En mi máquina sí funciona”, “Los de operaciones rompieron mi app”.
- Comunicación mínima → culpa mutua.

Ahora (modelo continuo):

- Integración desde el inicio entre desarrollo, QA y operaciones.
- Despliegues frecuentes, incluso varias veces al día.
- Feedback inmediato de usuarios y métricas de operación.
- Automatización total del pipeline (build, test, deploy).

DevOps

Características principales:

- **Colaboración interdisciplinaria:** rompe barreras entre desarrollo, QA, seguridad y operaciones.
- **Automatización del ciclo de entrega:** pipelines que integran, prueban y despliegan de forma automática.
- **Feedback constante:** los resultados operativos retroalimentan el diseño y el desarrollo.
- **Medición y mejora continua:** decisiones basadas en métricas (MTTR, tasa de fallos, tiempo de despliegue).

Cada uno de estos pilares exige **decisiones arquitectónicas conscientes**: logs estructurados, APIs observables, servicios desacoplados, infraestructura codificada.

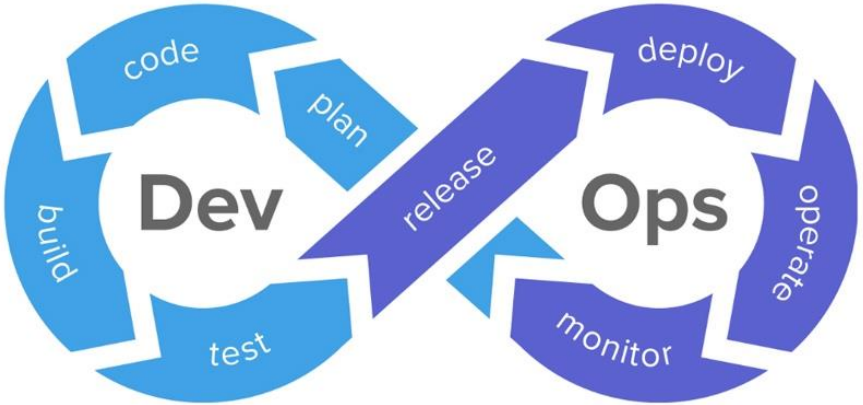
Ciclo DevOps

1. **Plan:** Definir requisitos, arquitectura y backlog.

2. **Code:** Escribir código y versionarlo (Git).

3. **Build:** Compilar, empaquetar y generar artefactos reproducibles.

4. **Test:** Ejecutar pruebas automáticas unitarias, de integración y seguridad.



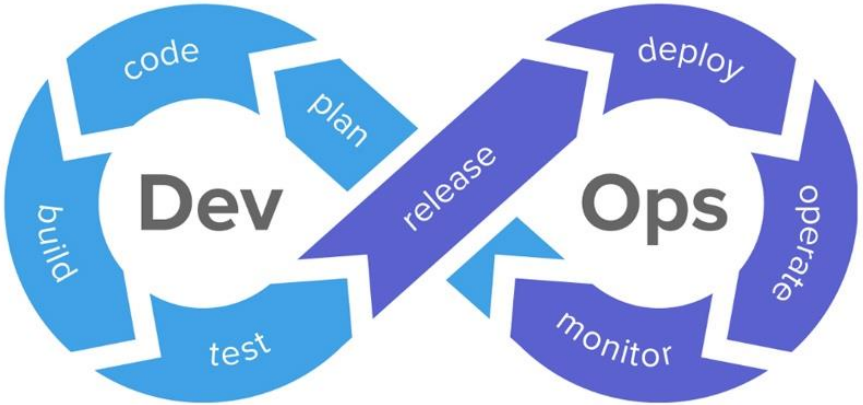
Ciclo DevOps

1. **Release:** Validar versiones listas para despliegue.

2. **Deploy:** Publicar automáticamente en entornos (staging/producción).

3. **Operate:** Ejecutar el sistema, monitorear disponibilidad y rendimiento.

4. **Monitor:** Recolectar métricas y feedback del uso real.



Beneficios de DevOps

Empresas como Netflix o Amazon despliegan **miles de veces al día** con mínima fricción, gracias a pipelines totalmente automatizados y arquitecturas adaptadas al cambio.

Entregas más frecuentes y seguras

- Reducción del “time to market”.
- Mejor capacidad de respuesta al negocio.

Reducción de errores

- Automatización = menos fallos humanos.
- Pruebas continuas = detección temprana de defectos.

Entornos reproducibles

- Infraestructura como Código (IaC): garantiza entornos idénticos.
- Facilita rollback y escalabilidad.

Beneficios de DevOps

Empresas como Netflix o Amazon despliegan **miles de veces al día** con mínima fricción, gracias a pipelines totalmente automatizados y arquitecturas adaptadas al cambio.

Mayor alineación entre equipos

- Comunicación fluida → menos conflictos y más agilidad.
- Cultura de responsabilidad compartida.

Mejora de la calidad arquitectónica

- Retroalimentación continua que impulsa refactorización y optimización.

¿Cómo implementamos DevOps?

Una serie de **herramientas, tecnologías y técnicas** que soportan los principios de DevOps y que permiten su correcta implementación en un proyecto software.

Control de versiones (Git, GitHub, GitLab): Es la **base del trabajo colaborativo y automatizado.** Permite gestionar cambios.

Cloud Computing: DevOps se potencia en **infraestructuras elásticas y escalables.** Los entornos cloud permiten **automatizar despliegues** y crear entornos efímeros bajo demanda. **Infraestructura como Código (IaC)** con Terraform, AWS.

Contenerización (Docker, Kubernetes) Empaqueta aplicaciones con todas sus dependencias → **"funciona igual en todos lados."** Facilita la **portabilidad** entre entornos de desarrollo, prueba y producción.

CI/CD (Integración y Entrega Continua) El corazón operativo de DevOps.

- **CI (Continuous Integration):** Compila y prueba automáticamente cada commit.
- **CD (Continuous Delivery/Deployment):** Automatiza el despliegue, liberación y monitoreo.

Universidad Intercontinental de la Empresa

Asignatura

Arquitectura del Software 

Profesor

Yago Fontenla Seco

{*yago.fontenla1@uie.edu*}