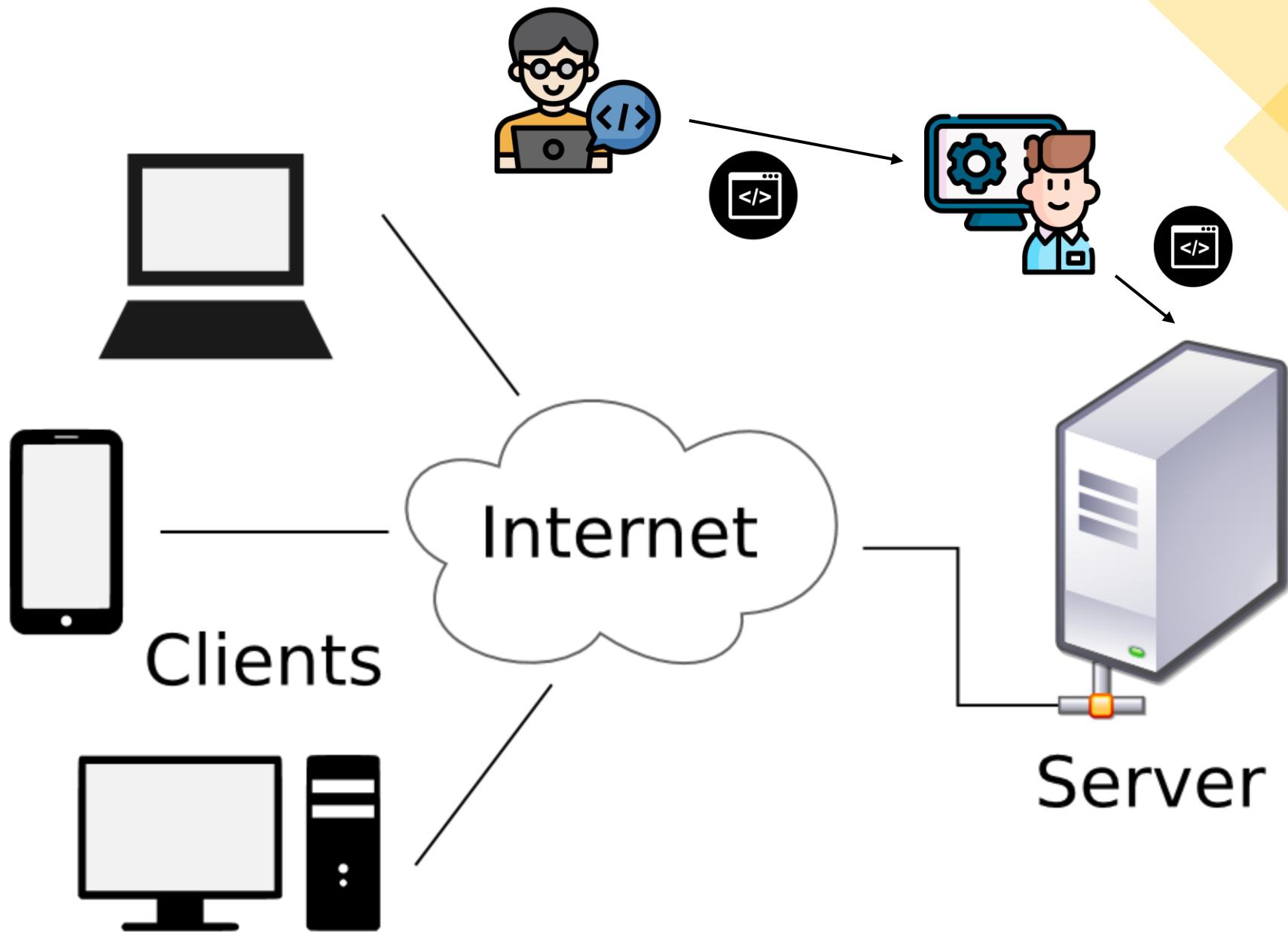


# Como ser un DevOps sin romper todo

---

(Solo unas pocas cosas)



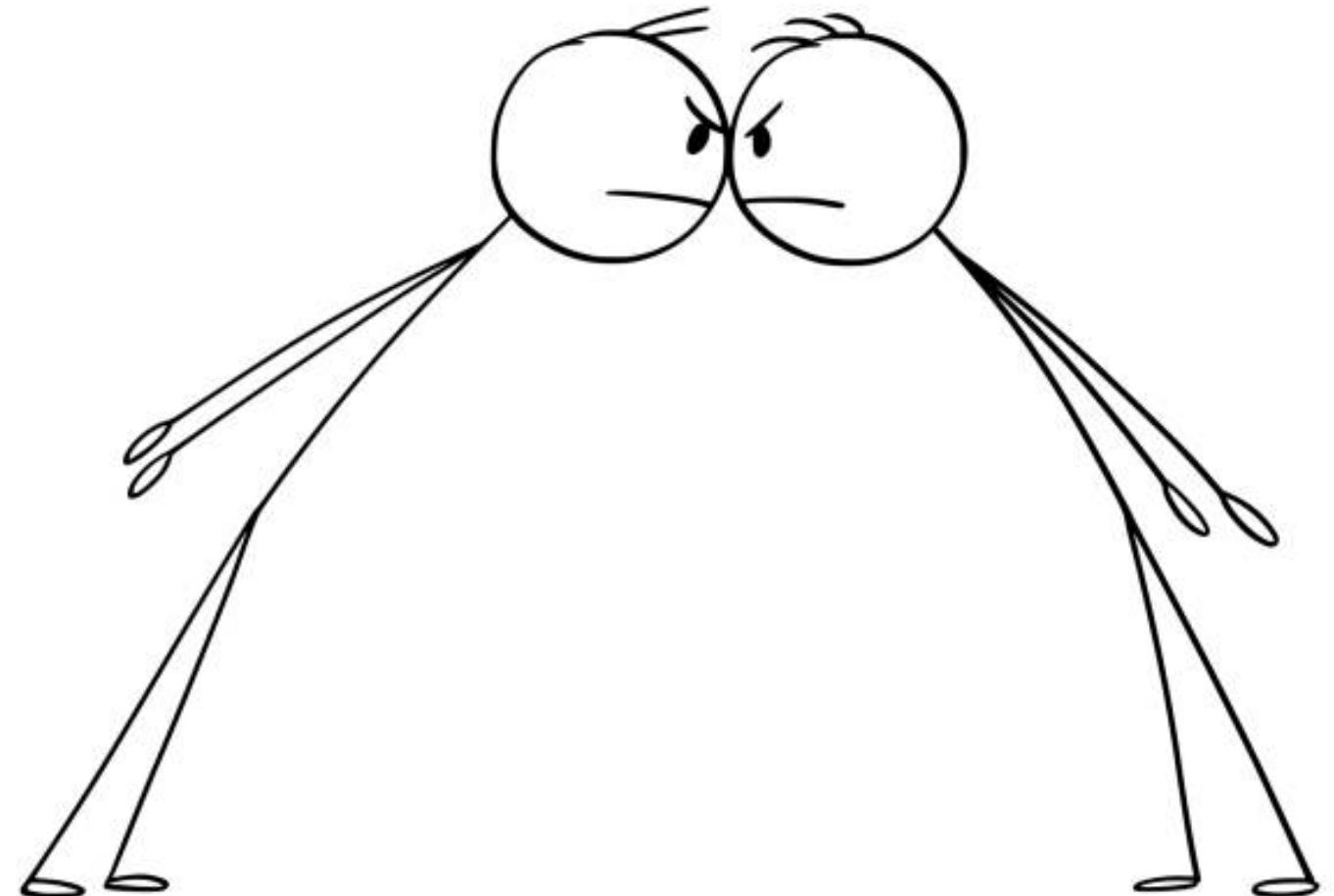
**Nuevas funcionalidades ->**

**<- Estabilidad en el servicio**

# Dev VS Ops

---

- Cumplir objetivos sin preocuparse en los demás.
- Los cambios a producción eran más tardados y más propensos a caídas en el servicio

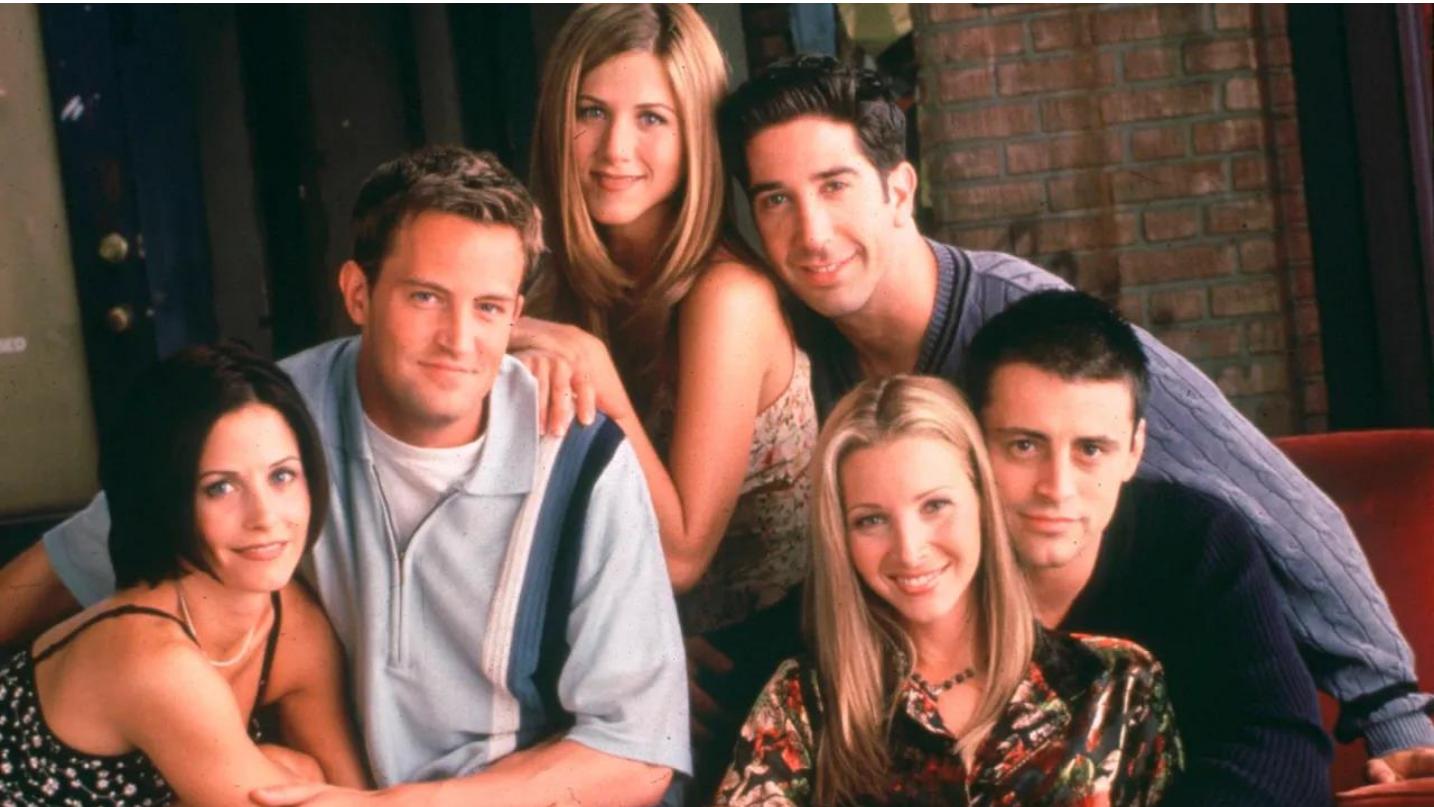


# DevOps

---

**"The organizational and cultural movement that aims to increase software delivery velocity, improve service reliability, and build shared ownership among software stakeholders"**

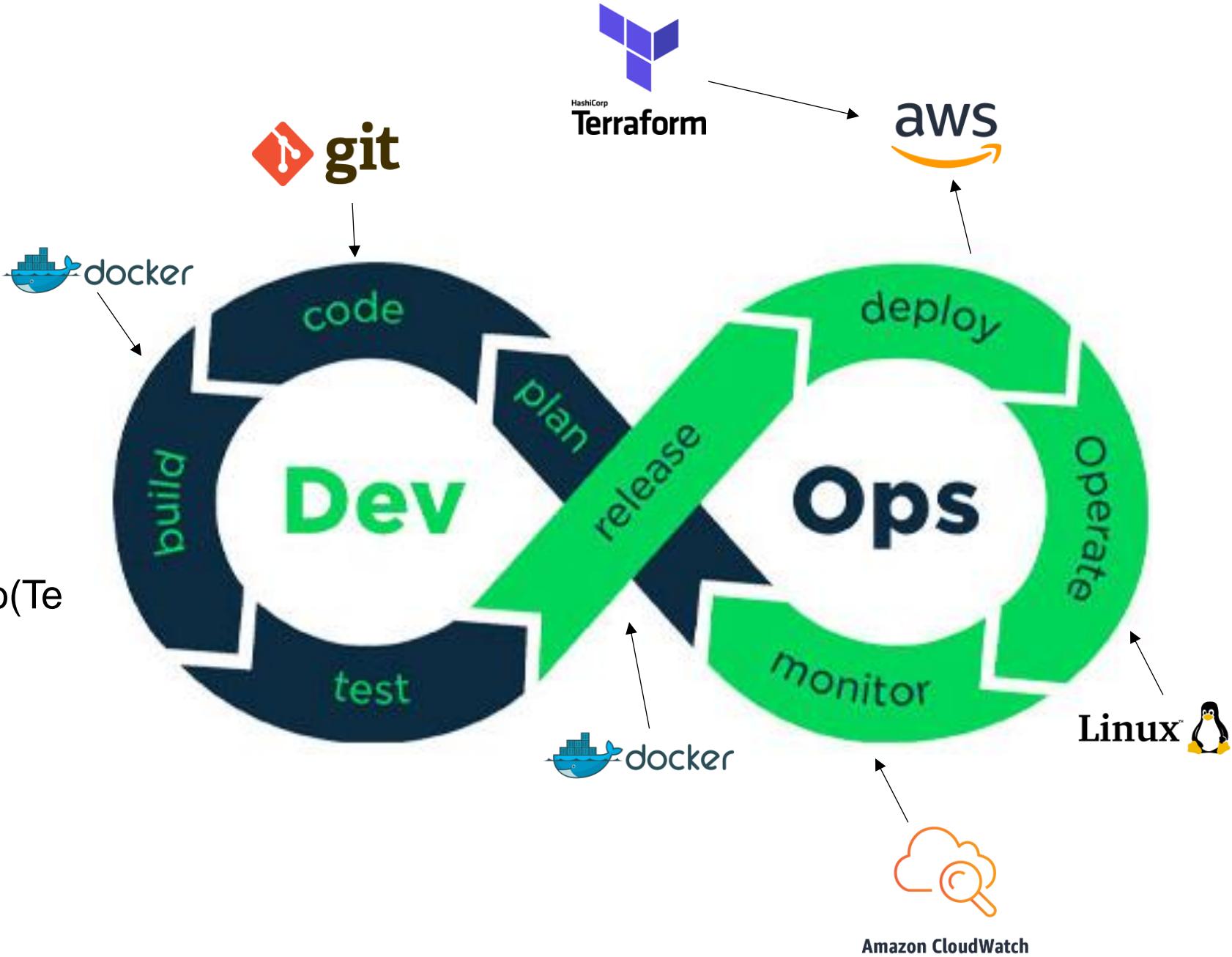
Google





GitHub Actions

- Control de versiones (Git)
- Contenedores (Docker)
- Cloud providers (AWS)
- Infraestructura como código(Terraform)
- S.O (Linux)
- Monitoreo (CloudWatch)
- CI/CD (GitHub Actions)



## ¿DevOps o SRE?

---

- Site Reliability Engineering
- Crear sistemas de software ultraescalables y fiables



DevOps  
SRE

# Diferencias entre los roles de un DevOps y un SRE

- DevOps
  - Principal enfoque: Delivery speed
  - Automatizaciones en el flujo de desarrollo
  - Infraestructura como código
  - Proporcionar un ambiente idóneo para los desarrolladores.
- SRE
  - Principal enfoque: Mayor fiabilidad en el sitio
  - Operaciones
  - Respuesta a incidentes 
  - Post Mortems
  - Monitoreo y alertas
  - Planeación y análisis

# Conocer las bases técnicas

---

- [Roadmap](#)
- Enfoque en herramientas como:
  - Version control system (Git)
  - Containers (Docker)
  - Cloud provider (AWS)
  - Infrastructure as code (Terraform)
  - CI/CD (GitHub Actions)
- Hacer uso de las herramientas constantemente (Practica)

# Desarrollar proyectos

---

- Pequeños proyectos utilizando nuevas tecnologías aprendidas
- Tratar de documentar todo
- Subir todo a GitHub
- Ayudarte de otros proyectos
- Solución completa utilizando todas las tecnologías

Aprender algo nuevo  
constantemente

## LINKS

