**Fase 1: plan**

La primera fase del ciclo DevOps se basa en habilitar los mecanismos necesarios para dar la posibilidad a cualquier usuario de aportar de forma continua sus ideas y que estas puedan ser transformadas en requerimientos u objetivos, siendo estos priorizados e incluidos en próximas iteraciones en forma de historias de usuario.

**Fase 2: build**

DevOps ayuda a la construcción de software propiciando mecanismos que sean capaces de favorecer la creación de entornos de desarrollo repetibles, utilizando por ejemplo tecnologías como Docker.

**Fase 3: integración continua**

La integración continua (CI) es una técnica que tiene como objetivo detectar los posibles problemas que pueda tener el software de forma temprana, permitiendo así solucionarlos antes de que sea demasiado tarde o la solución implique un gran cambio en la aplicación o componente. Para esta labor es conveniente realizar de forma periódica (por ejemplo, varias veces al día) compilación del código, ejecución de tests unitarios, revisión de la calidad del código y detección de vulnerabilidades. Los resultados de las acciones anteriores se pueden exponer mediante herramientas apropiadas que permitan al desarrollador visualizarlas de una forma ágil.

**Fase 4: despliegue**

Uno de los grandes objetivos de DevOps es facilitar el paso de un software que se está desarrollando a una versión funcional. En una metodología tradicional, se requiere de una intervención manual desde la construcción de los artefactos hasta el paso de los mismos a producción. DevOps proporciona técnicas que permiten automatizar esta transición, pero también para la subsanación de posibles “catástrofes” que puedan ocurrir.

Programación de despliegues. Existen herramientas que permiten programar los despliegues, organizando las diferentes historias de usuario en versiones, lo que permite además disponer de una trazabilidad completa de qué y cuándo se despliega en cada entrega.

Compatibilidad de componentes. Aseguramiento de que todos los componentes son compatibles unos con los otros. Cuando hablamos de componentes no solo nos referimos a software, sino que también hay que poner en la ecuación a sistemas externos, hardware o requerimientos de arquitectura.

Transición entre entornos. Es muy importante mantener la integridad mientras avanza el software entre los diferentes entornos: desarrollo, testing, staging y producción. No es nada raro que el software sufra pequeños ajustes durante esta transición, por lo que es fundamental garantizar que se siga manteniendo la integridad de la versión.

Solución de desastres. Aunque las técnicas DevOps ayudan a evitar que llegue una versión inestable al entorno productivo, siempre existe una pequeña posibilidad de que se encuentre un fallo, el cual haga que sea necesario realizar un rollback al último estado estable. Dada la automatización de la que disponen las herramientas de despliegue, es posible recuperar de una forma muy rápida el correcto funcionamiento del aplicativo.

**Fase 5: operación**

Una vez una versión de software sea funcional, es fundamental realizar un seguimiento de la misma:

Monitorización del rendimiento de la aplicación y servidor. Para ello hay que realizar de forma automatizada la lectura de parámetros que permitan dar la voz de alarma en caso de fallo.

Seguimiento de problemas, incidentes y cambios. Una comunicación fluida entre las diferentes personas es fundamental para poder identificar y resolver los posibles problemas que se vayan encontrando.

**Fase 6: feedback continuo**

Es muy importante escuchar lo que todos los implicados tienen que decir acerca de los productos de software. La cultura DevOps fomenta la utilización de herramientas que facilitan exista un flujo continuo de comunicación. Medios como encuestas, herramientas de ticketing, chats o incluso redes sociales pueden ser utilizados para cumplir esta finalidad.

**En que partes del desarrollo se puede hacer uso de Git?**

En este caso puede ser muy útil en las partes de elaboración y ejecución del código de manera que el desarrollo sea mucho más agradable y sencillo de llevar a cabo mediante las facilidades que ofrece git, así como en compartir repositorios y así poder llevar un control de lo que se avanza y cada individuo realiza por su parte para ayudar en el proyecto en cuestión.