*https://repository.uaeh.edu.mx/revistas/index.php/huejutla/issue/archive*

**Ciencia Huasteca Boletín Científico de la Escuela Superior de Huejutla**

*Publicación semestral, Vol. 10, No. 19 (2022) 35-40*

ISSN: 2007-493X

# DevOps: un vistazo rápido

# DevOps: a quick look

*Ana María Felipe Redondo [a], Felipe de Jesús Núñez Cárdenas [b]*

**Abstract:**

This essay offers the reader a quick look at DevOps, describing each of the phases that comprise it, the practices it promotes, the continuous focus terminology that reflects its philosophy, finally it also offers an overview of the multiple options of software tools that there is in the market for its implementation, thus promoting minimizing application development time and accelerating the release of new functions required by customers, thus giving life to comply with the essence of DevOp favoring constant and continuous communication between application development (Dev) and technology operations (Ops) work teams.

***Keywords:***

*DevOps, Software Development, Agile Methodologies, Software Engineering*

**Resumen:**

Este ensayo ofrece al lector una hojeada rápida sobre DevOps, describiendo cada una de las fases que lo integran, las prácticas que promueve, la terminología de enfoque continuo que refleja su filosofía, finalmente también ofrece un panorama de las múltiples opciones de herramientas de software que hay en el mercado para su implementación, promoviendo así minimizar el tiempo de desarrollo de las aplicaciones y acelerar la liberación de nuevas funciones requeridas por los clientes, dando vida con esto a cumplir con la esencia de DevOp favorecer a una comunicación constante y continúa entre los equipos de trabajo de desarrollo de aplicaciones (Dev) y de operaciones tecnológicas (Ops).

***Palabras Clave:***

*DevOps, Desarrollo de Software, Metodologías Ágiles, Ingeniería de Software*

## Introducción

"DevOps" es una combinación de los términos anglosajones “DEVelopment” y “OPerationS, usada por primera vez por Yhens Wasna y Patrick Debois en su charla sobre "Infraestructura Ágil", en la Conferencia Agile 2008 en Toronto[1].

El principal objetivo de DevOps es minimizar el tiempo de desarrollo de las aplicaciones y acelerar la liberación de nuevas funciones requeridas por los clientes. Esto a través de una comunicación constante y continúa entre los equipos de trabajo de desarrollo de aplicaciones (Dev) y de operaciones tecnológicas (Ops).

Para Gartner, “DevOps representa un cambio en la cultura de Tecnologías de la Información (TI), que se centra en la entrega rápida de servicios de TI mediante la adopción de prácticas ágiles y ajustadas en el contexto de un enfoque orientado al sistema. DevOps hace hincapié en las personas (y la cultura) y busca mejorar la colaboración entre los equipos de operaciones y desarrollo. Las implementaciones de DevOps utilizan tecnología, especialmente herramientas de automatización que pueden aprovechar una infraestructura cada vez más

[a] Ana María Felipe Redondo, Autora de Correspondencia, Universidad Tecnológica de la Huasteca Hidalguense, https://orcid.org/0000-0002-8579-6532, Email: ana.felipe@uthh.edu.mx

[b] Felipe de Jesús Núñez Cárdenas, Universidad Autónoma del Estado de Hidalgo, https://orcid.org/0000-0002-2462-3654, Email: felipe_nunez@uaeh.edu.mx

*Fecha de recepción: 17/10/2021, Fecha de aceptación: 25/10/2021, Fecha de publicación: 05/07/2021*

programable y dinámica desde la perspectiva del ciclo de vida "[2].

Una vez puesta en práctica, DevOps se convierte también en una cultura, en un cambio de mentalidad, cuyo único objetivo es acortar el ciclo de vida del desarrollo de software. Se constituye de fases y herramientas que promueven su apropiación.

## Un poco de historia

La industria del software tiene sus primeras menciones en 1936, cuando Alan Turing publica Sobre números calculables, con una aplicación al Entscheidungs problem (problema de decisión) en este artículo, Turing menciona una máquina abstracta, conocida ahora como "máquina de Turing", que cambia de un estado a otro usando un conjunto finito preciso de reglas y dependiendo de un solo símbolo que lee de una cinta, estos fueron los primeros albores un algoritmo[3]. Sin embargo, no es hasta finales de la década de los cincuentas cuando John W. Tukey usa en 1958 por primera vez el término software [4].

Una década después, surge el concepto de ingeniería del software en el ciclo de conferencias organizado por la OTAN en Garmisch, Alemania que plantea una crisis del software y sugiere establecer metodologías [5].

Desde sus orígenes hasta la fecha, la ingeniería de software ha seguido un proceso de tesis, antítesis y síntesis que explicaría las diferentes propuestas y contrapropuestas que se han sucedido a lo largo de estas décadas, tal como lo señala Barry Boehm [6].

Es en esa misma conferencia donde Boehm, resalta la evolución de la Ingeniería de Software a través de las décadas, misma de la que se presenta una breve descripción.

En los años **setentas**, es aquí donde se especifican métodos formales para el desarrollo de software, identificando: requerimientos, análisis, diseño, codificación y pruebas.

Para la década de los **ochentas**, Boehm introduce el modelo de madurez de capacidades de procesos (SW-CMM) y los primeros estándares.

Es en la década de los **noventas**, donde se percibe un salto significativo en la manera de producir software. Se define un lenguaje de modelado (UML, por sus siglas en inglés) y se genera el primer proceso comercial de desarrollo basado en la metodología de Proceso Unificado de Desarrollo (RUP, por sus siglas en inglés)" (RUP), se tiene un enfoque hacia las experiencias de usuario (UX, por sus siglas en inglés) a través de patrones de diseño y de arquitectura. Se define el Modelo Espiral para el desarrollo basado en el análisis de riesgos y su vertiente conocida como desarrollo iterativo e incremental.

En aquella conferencia, Boehm mencionada que para el inicio del **siglo veintiuno** la ingeniería de software se enfocaría hacia la rapidez en el desarrollo y el valor para el cliente. Se redacta el Manifiesto de Agilidad en respuesta al estilo promovido por el Modelo de Madurez de Capacidades (Capability Maturity Model – CMM, por sus siglas en inglés), y desde entonces a la fecha se ha cumplido su predicción, siendo en la **primer década** del siglo que aparece la práctica DevOps [7].

Métodos de DevOps

Entre los métodos más comunes para acelerar y mejorar el desarrollo de software, se encuentran Scrum, Agile y Kanban [8].

**Scrum**: es la forma más conocida de implementar Agile. Se constituye de un conjunto de roles, responsabilidades y eventos con la finalidad de dar seguimiento a los avances, retrasos y tareas pendientes. **Promueve** el trabajo en equipo, la responsabilidad y el progreso iterativo hacia una meta bien definida.

**Agile**: es una metodología que divide un proyecto en pequeñas partes, lo que permite a los equipos de desarrollo concluir el proyecto en pocas semanas y de manera eficiente. Tiene un enfoque de desarrollo incremental e iterativo, priorizando el valor del negocio

o del cliente; se fortalece a través de equipos multifuncionales.

**Kanban**: es considerado como un marco visual para la implementación de Agile, este enfoque promueve cambios pequeños pero continuos en el sistema actual. Los principios de Kanban comprenden visualizar el flujo de trabajo, limitar el trabajo en progreso, administrar y mejorar el flujo de trabajo y la mejora continua [9].

Fases de DevOps

Antes de dar paso a las fases de DevOps es importante resaltar que algunos autores integran algunas de las fases, resumiéndolas en 6 o 5 fases, aquí se describen de forma completa, desde su propuesta inicial.

DevOps es entre tantas acepciones, un ciclo infinito de las fases que lo integran, tal como se muestra en la Figura 1. La metodología inicia en el equipo de Desarrollo (Dev) responsables de la Planificación (Plan), la Codificación (Code), la Compilación (Build) (algunos simplifican ambos procesos como el Desarrollo) y la Pruebas (Test) para dar paso al equipo de Operaciones (Ops) que comienza con la Liberación (Release), el Despliegue (Deploy), el Funcionamiento (Operate) y el Monitoreo (Monitor), para dar paso de nuevo a la etapa de Planeación.

Es importante “no caer en la tentación de tener una visón de metodología de cascada”, aquí más bien es un enfoque de flujo iterativo, por lo que diferentes procesos pueden estar comprendidos en diferentes fases de forma orgánica y superpuesta, siempre ajustándose a los conceptos fundamentales de valor y mejora continua[10].

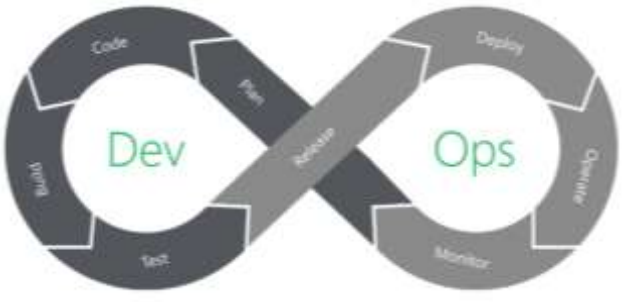


***Figura 1. El ciclo DevOps [11]***

**Planificación,** se orienta a identificar los requisitos de los clientes y favorece una comunicación continua con este. El producto de su trabajo es una hoja de ruta del producto para guiar el desarrollo futuro.

**Codificación**, esta fase implica el diseño del software y la creación del código, definiendo las herramientas, los complementos que ayuden al proceso de desarrollo, favoreciendo a un estilo de código consistente y evitar fallas de seguridad comunes y anti-patrones de código [11].

**Compilación,** es aquí donde entra en juego DevOps. Una vez que un desarrollador ha terminado una tarea, envía su código a un repositorio de código compartido (solicitud de extracción) para fusionar su nuevo código con la base de código compartida, otro desarrollador revisa y aprueba los cambios esta revisión eficaz para identificar problemas de forma prematura.

Otro aspecto relevante, es que paralelamente la solicitud de extracción activa un proceso automatizado de pruebas unitarias, de integración y de extremo a extremo para identificar cualquier regresión. Si la compilación falla, o alguna de las pruebas falla, la solicitud de extracción falla y se notifica al desarrollador para que resuelva el problema. Al verificar continuamente los cambios de código en un repositorio compartido y ejecutar compilaciones y pruebas, se minimizan los problemas de integración que surgen cuando se trabaja en una base de código compartida y resaltar los errores de ruptura al principio del ciclo de vida del desarrollo.

**Pruebas,** al concluir la compilación, automáticamente se implementa en un entorno de ensayo para realizar pruebas más profundas. Una vez que la aplicación se implementa en el entorno de prueba, se realiza una serie de pruebas manuales y automatizadas. Las pruebas manuales pueden ser las pruebas de aceptación del usuario (UAT, por sus siglas en inglés) en donde las que las personas usan la aplicación para detectar cualquier problema o mejora que deba atenderse antes de su implementación en producción. Paralelamente, las pruebas automatizadas pueden ejecutar análisis de seguridad en la aplicación, verificar cambios en la infraestructura y el cumplimiento de las mejores prácticas de refuerzo, probar el rendimiento de la aplicación o ejecutar pruebas de carga. Las pruebas que se realizan durante esta fase dependen de la organización y de lo que es relevante para la aplicación, pero esta etapa puede considerarse un banco de pruebas que le permite conectar nuevas

pruebas sin interrumpir el flujo de desarrolladores o impactar el entorno de producción.

**Liberación,** documentada también como Lanzamiento, es en esta etapa donde una compilación está lista para su implementación en el entorno de producción. Es aquí donde cada cambio de código ha pasado una serie de pruebas manuales y automatizadas, y el equipo de operaciones puede estar seguro de que los problemas de ruptura y las regresiones son poco probables. Aquí también se puede optar por implementar automáticamente cualquier compilación que llegue a esta etapa de la canalización, pero esto dependerá de la madurez de la implementación de DevOps. Esta es la forma en que las organizaciones logran implementar múltiples lanzamientos de sus productos todos los días.

**Despliegue,** una vez que la compilación esta lista, se puede lanzar a producción, pudiéndose automatizar desde la fase de lanzamiento para lograr un proceso ininterrumpido.

La misma infraestructura como código que creó el entorno de prueba se puede configurar para crear el entorno de producción. Si se detecta algún problema con la nueva compilación, simplemente se pueden dirigir las solicitudes al entorno anterior mientras se da solución.

**Funcionamiento,** la nueva versión está en funcionamiento, aquí de forma transparente el equipo de Operaciones ha estado trabajando para asegurarse de que todo funcione sin problemas. Considerando que el cliente es el mejor equipo de pruebas, es vital conocer su retroalimentación para dar forma al futuro desarrollo del producto.

**Monitoreo,** en esta fase se recopilan datos y se hace análisis sobre el comportamiento, el rendimiento, los errores del cliente en su uso en la aplicación.

Esta información llega al administrador de producto y al equipo de desarrollo para cerrar el ciclo del proceso. No es el fin del ciclo, dado que no hay un principio ni un final, solo es la evolución continua de un producto a lo largo de su vida útil, que solo termina cuando las personas avanzan o ya no lo necesitan.

**Práctica DevOps: evolución continua**

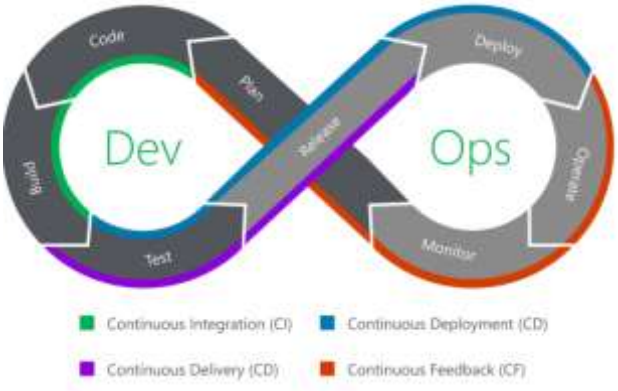


***Figura 2, Cruce entre las fases para una evolución continua.***

A la par de las fases de DevOps, también se habla frecuentemente del "todo continuo": desarrollo continuo, pruebas continuas, integración continua, entrega continua, despliegue continuo, monitoreo continuo y más. Esto se debe a que la continuidad es el núcleo de DevOps. Sin embargo, tienen un propósito.

**Desarrollo continuo**, esta práctica integra las fases de planificación y codificación del ciclo de DevOps, marcada en un borde color verde de la Figura 2.

**Pruebas continuas**, esta práctica incorpora continuas pruebas de código automatizadas y programadas 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, indicada en el borde color azul de la Figura 2.

Integración continúa, fusiona el proceso de desarrollo de muchos desarrolladores a través de un repositorio de código compartido. Aquí se alinean las fases de codificación y compilación de la canalización de DevOps, resaltada en el borde color verde de la Figura 2.

**Entrega continua,** aquí se automatiza el proceso de implementación de una nueva construcción en producción, contempla realizar pruebas automatizadas en cada nueva compilación; gestiona el aprovisionamiento y la configuración automática de los entornos de implementación, las pruebas para comprobar la estabilidad, el rendimiento y el

cumplimiento de la seguridad. Integra las fases de prueba y lanzamiento, resaltadas en el borde color azul de la Figura 2.

**Despliegue continuo**, es una versión automatizada de entrega continua, porque elimina el paso manual de aprobar nuevos lanzamientos en producción. En un modelo de implementación continua, cada compilación que pasa todos los controles y equilibrios de la canalización se implementa automáticamente en producción, indicada en el borde color morado de la Figura 2.

**Retroalimentación continua**, al ser el objetivo de DevOps lanzar nuevas funciones y correcciones lo más rápido posible, la retroalimentación de los clientes, las partes interesadas y los análisis de estos, hacen posible para tomar mejores decisiones al diseñar el próximo conjunto de cambios. Es la retroalimentación continua la que une los extremos del ciclo, retroalimentando los datos y análisis de las fases Funcionamiento y Monitoreo a la fase de Planeación para hacerlo todo de nuevo, indicada en el borde color naranja de la Figura 2.

**Infraestructura como código**, es una práctica que se puede utilizar durante varias fases de DevOps para automatizar el aprovisionamiento de la infraestructura [12].

**Cadena de Herramientas DevOps**

Contrario a lo que muchos piensan, para implementar DevOps existen de herramientas esenciales para cada uno de los miembros del equipo, es decir opera en modo multifuncional, involucra herramientas de diversos tipos y propósitos, a este conjunto se le conoce como cadenas de herramientas DevOps.

Listar la diversidad de herramientas DevOps y su uso supone un artículo aparte, tal como lo menciona Mik Kersten en su publicación “Una explosión cámbrica de Herramientas de DevOps”, [13], donde hace una analogía del periodo cámbrico de las eras geológicas con DevOps, Kersten considera que DevOps es una explosión, que trae un cambio de toda la industria del desarrollo de software, llenado los vacíos que dejó el modelo de cascada. Toda esta disrupción ha traído un sin número de soluciones, algunas se observan en el resumen gráfico del Mapa del panorama de inicio de DevOps, escrito por Jake Kaldenbaugh de

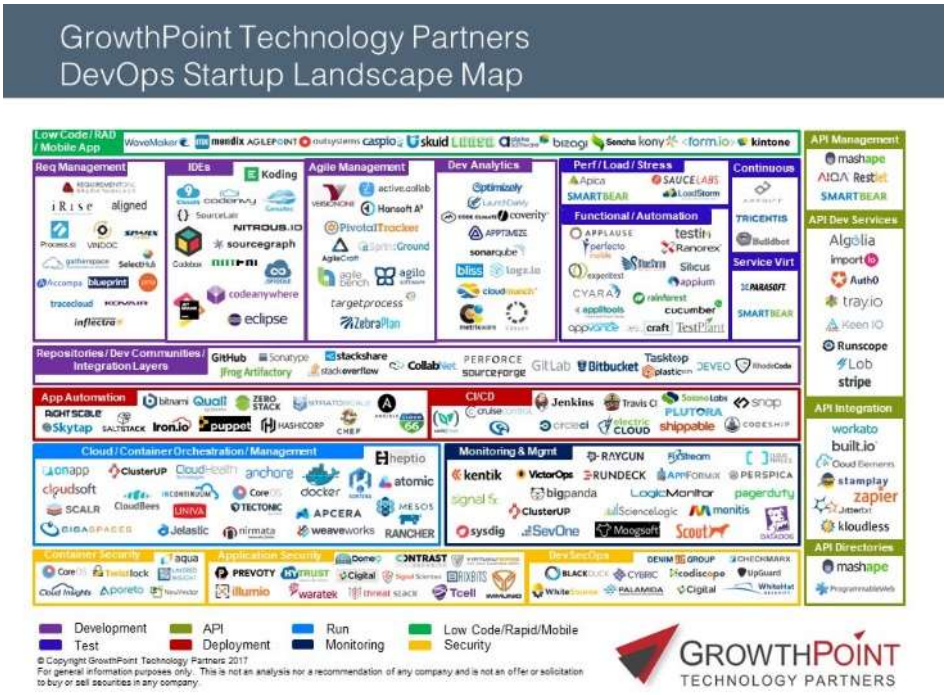


***Figura 3. Mapa del panorama de inicio de DevOps [14]***

GrowthPoint [14], que ubica de forma cuidadosa cada una de las herramientas en el espacio que les corresponde, que se puede observar en la Figura 3.

Agrupadas en herramientas para desarrollo, pruebas, propuesta de código bajo, desarrollo rápido, desarrollo móvil, administradores de requerimientos, entornos de desarrollo, administradores de metodologías agiles, analíticas de desarrollo, repositorios, administradores de APIS, administradores de servicios de desarrollo, de integración de directorios, de ejecución, de monitoreo, entre tantos, distinguir DevSecOps, integrando a la propuesta DevOps, prácticas de seguridad.

El mapa ofrece un panorama a lo largo del ciclo de vida del desarrollo, avanzando hacia las capas de infraestructura basadas en la nube y en contenedores.

Hay que resaltar que todo el mapa se soporta en las apuestas de seguridad emergentes. Del lado derecho, se ofrece un resumen de las API’s cuyo rol es

fundamental para el nuevo paradigma de desarrollo, ya que contribuyen en la integración de las aplicaciones.

## Conclusiones

DevOps ha traído a las empresas de desarrollo una alternativa tangible para acelerar los ciclos de vida de desarrollo, lo que les permite apostar para ser competitivas y responder al ritmo de las necesidades de sus clientes y el mercado.

Se pretende que este vistazo rápido a DevOps sea útil para tenerlo presente como una alternativa de solución en aquellas empresas que presenten problemas en el proceso de crear un mejor software de una forma rápida.

A más de una década de su entrada algunas empresas han reestructurado sus procesos de desarrollo de aplicaciones empresariales y la infraestructura asociada y DevOps sigue ganando terreno.

## Referencias

[1] P. Debois, «Agile 2008 Toronto: Agile Infrastructure and Operations Presentation,» 2008. [En línea]. Available: http://www.jedi.be/blog/2008/10/09/agile-2008-toronto-agile-infrastructure-and-operations-presentation/.

[2] Gartner, «Gartner Glossary,» Gartnet, [En línea]. Available: https://www.gartner.com/en/information-technology/glossary/devops.

[3] C. Prieto de Castro, «Alan Turing,» [En línea]. Available: https://paginas.matem.unam.mx/cprieto/biografias-de-matematicos-p-t/235-turing-alan.

[4] MacTutor, «John Wilder Tukey,» MatHistory, [En línea]. Available: https://mathshistory.st-andrews.ac.uk/Biographies/Tukey/.

[5] NATO Science Committee , «Software Engineering,» 1968. [En línea]. Available: https://www.scrummanager.net/files/nato1968e.pdf.

[6] International Conference on Software Engineering (ICSE'06), «A View of 20th and 21st Century Software Engineering,» 2006. [En línea]. Available: http://isr.uci.edu/icse-06/program/keynotes/boehm.html.

[7] H. Oktaba, «Historia y futuro de la Ingeniería de Software. Visión de Barry Boehm.,» Software Gurú, 2006. [En línea]. Available: https://sg.com.mx/revista/9/historia-y-futuro-la-ingenier-software-vision-barry-boehm.

[8] E. Thorpe , DevOps: Guía Completa para Principiantes Aprende DevOps paso a paso, Independently Published, 2019.

[9] Tecnologías Información, «Introducción a Agile y Scrum: Principios y enfoques,» 2018. [En línea]. Available: https://www.tecnologias-informacion.com/agile-scrum.html.

[10] GenBeta, «El ciclo de DevOps, una guía para iniciarse en las fases que lo componen,» [En línea]. Available: https://www.genbeta.com/desarrollo/el-ciclo-de-devops-una-guia-para-iniciarse-en-las-fases-que-lo-componen.

[11] «The Eight Phases of a DevOps Pipeline,» Taptu, 2019. [En línea]. Available: https://medium.com/taptuit/the-eight-phases-of-a-devops-pipeline-fda53ec9bba.

[12] NetApp, «¿Qué es DevOps?,» [En línea]. Available: https://www.netapp.com/es/devops-solutions/what-is-devops/.

[13] M. Kersten , «A Cambrian Explosion of DevOps Tools,» *IEEE Software*, vol. 35, nº 2, pp. 14-17, 2018.

[14] J. Kaldenbaugh, «Announcing GrowthPoint's DevOps Startup Landscape Map,» GROWTHPOINT, 2016. [En línea]. Available: https://growthpoint.com/growthpoint/announcing-growthpoints-devops-startup-landscape-map/.