**WHAT ARE DevOps?**

**DevOps** is a set of practices that combines [software development](https://en.wikipedia.org/wiki/Software_development) (*Dev*) and [information-technology operations](https://en.wikipedia.org/wiki/Information_technology_operations) (*Ops*) which aims to shorten the [systems development life cycle](https://en.wikipedia.org/wiki/Systems_development_life_cycle) and provide [continuous delivery](https://en.wikipedia.org/wiki/Continuous_delivery) with high [software quality](https://en.wikipedia.org/wiki/Software_quality).

Academics and practitioners have not developed a unique definition for the term "DevOps."

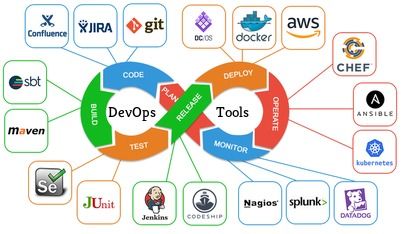
From an academic perspective, [Len Bass](https://en.wikipedia.org/wiki/Len_Bass), [Ingo Weber](https://en.wikipedia.org/w/index.php?title=Ingo_Weber&action=edit&redlink=1), and [Liming Zhu](https://en.wikipedia.org/w/index.php?title=Liming_Zhu&action=edit&redlink=1) — three computer science researchers from the [CSIRO](https://en.wikipedia.org/wiki/CSIRO) and the [Software Engineering Institute](https://en.wikipedia.org/wiki/Software_Engineering_Institute) — suggested defining DevOps as "a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality".

The term DevOps, however, has been used in multiple contexts.

**HISTORY**

In 2009, the first conference named devopsdays was held in [Ghent](https://en.wikipedia.org/wiki/Ghent), [Belgium](https://en.wikipedia.org/wiki/Belgium). The conference was founded by Belgian consultant, project manager and agile practitioner Patrick Debois. The conference has now spread to other countries.

In 2012, the State of DevOps report was conceived and launched by Alanna Brown at Puppet. As of 2014, the annual State of DevOps report was published by Nicole Forsgren, Gene Kim, Jez Humble and others. In 2014, they found that DevOps adoption was accelerating. Also in 2014, Lisa Crispin and Janet Gregory wrote the book More Agile Testing, containing a chapter on testing and DevOps.



**TOOLCHAINS**

As DevOps is intended to be a cross-functional mode of working, those who practice the methodology use different sets of tools—referred to as "[toolchains](https://en.wikipedia.org/wiki/DevOps_toolchain)"—rather than a single one. These toolchains are expected to fit into one or more of the following categories, reflective of key aspects of the development and delivery process:

1. Coding – code development and review, [source code management](https://en.wikipedia.org/wiki/Version_control) tools, code merging
2. Building – [continuous integration](https://en.wikipedia.org/wiki/Continuous_integration) tools, build status
3. Testing – [continuous testing](https://en.wikipedia.org/wiki/Continuous_testing) tools that provide quick and timely feedback on business risks
4. Packaging – [artifact repository](https://en.wikipedia.org/wiki/Binary_repository_manager), application pre-deployment staging
5. Releasing – change management, release approvals, [release automation](https://en.wikipedia.org/wiki/Application_release_automation)
6. Configuring – infrastructure configuration and management, [infrastructure as code](https://en.wikipedia.org/wiki/Infrastructure_as_code) tools
7. Monitoring – [applications performance monitoring](https://en.wikipedia.org/wiki/Application_performance_management), end-user experience

Some categories are more essential in a DevOps toolchain than others; especially continuous integration (e.g. [Jenkins](https://en.wikipedia.org/wiki/Jenkins_(software)), [Gitlab](https://en.wikipedia.org/wiki/Gitlab), [Bitbucket](https://en.wikipedia.org/wiki/Bitbucket) pipelines) and infrastructure as code (e.g., [Terraform](https://en.wikipedia.org/wiki/Terraform_(software)), [Ansible](https://en.wikipedia.org/wiki/Ansible_(software)), [Puppet](https://en.wikipedia.org/wiki/Puppet_(software))).

Forsgren *et al.* found that IT performance is strongly correlated with DevOps practices like [source code management](https://en.wikipedia.org/wiki/Version_control) and [continuous delivery](https://en.wikipedia.org/wiki/Continuous_delivery).

**RELATIONSHIP WITH OTHER APPROACHES**

**Agile**

*Main article:*[*Agile software development*](https://en.wikipedia.org/wiki/Agile_software_development)

Agile and DevOps serve complementary roles: several standard DevOps practices such as automated build and test, [continuous integration](https://en.wikipedia.org/wiki/Continuous_integration), and [continuous delivery](https://en.wikipedia.org/wiki/Continuous_delivery) originated in the Agile world, which dates (informally) to the 1990s, and formally to 2001. Agile can be viewed as addressing communication gaps between customers and developers, while DevOps addresses gaps between developers and IT operations / infrastructure. Also, DevOps has focus on the deployment of developed software, whether it is developed via Agile or other methodologies.

**ArchOps**

ArchOps presents an extension for DevOps practice, starting from [software architecture](https://en.wikipedia.org/wiki/Software_architecture) artifacts, instead of source code, for operation deployment. ArchOps states that architectural models are first-class entities in software development, deployment, and operations.

**TestOps**

TestOps is to hardware development what DevOps is to software development. The idea is a toolchain that links design and *test* *operations* together. In the case of hardware, design means [EDA](https://en.wikipedia.org/wiki/Electronic_design_automation) tools and the [CAD](https://en.wikipedia.org/wiki/CAD) department, and test means electronic measurement equipment like oscilloscopes and so on.

**Continuous delivery**

*Main article:*[*Continuous delivery*](https://en.wikipedia.org/wiki/Continuous_delivery)

Continuous delivery and DevOps have common goals and are often used in conjunction, but there are subtle differences.

While continuous delivery is focused on automating the processes in [software delivery](https://en.wikipedia.org/wiki/Software_delivery), DevOps also focuses on the organizational change to support great collaboration between the many functions involved.

DevOps and continuous delivery share a common background in [agile methods](https://en.wikipedia.org/wiki/Agile_software_development) and [lean thinking](https://en.wikipedia.org/wiki/Lean_thinking): small and frequent changes with focused value to the end customer. [Lean management](https://en.wikipedia.org/wiki/Lean_manufacturing) and continuous delivery are fundamental to delivering value faster, in a sustainable way. Continuous delivery focuses on making sure the software is always in a releasable state throughout its lifecycle. This makes it cheaper and less risky to deliver the software.

Improved collaboration and communication both between and within organizational teams can help achieve faster [time to market](https://en.wikipedia.org/wiki/Time_to_market), with reduced risks

**DATAOPS**

The application of continuous delivery and DevOps to data analytics has been termed DataOps. DataOps seeks to integrate data engineering, data integration, data quality, data security, and data privacy with operations. It applies principles from DevOps, [Agile Development](https://en.wikipedia.org/wiki/Agile_software_development) and the [statistical process control](https://en.wikipedia.org/wiki/Statistical_process_control), used in [lean manufacturing](https://en.wikipedia.org/wiki/Lean_manufacturing), to improve the cycle time of extracting value from data analytics.

**SITE-RELIABILITY ENGINEERING**

In 2003, [Google](https://en.wikipedia.org/wiki/Google) developed [site reliability engineering](https://en.wikipedia.org/wiki/Site_reliability_engineering) (SRE), an approach for releasing new features continuously into large-scale high-availability systems while maintaining high-quality end-user experience. While SRE predates the development of DevOps, they are generally viewed as being related to each other. […]

**GOALS**

IT performance can be measured in terms of throughput and stability.  Throughput can be measured by deployment frequency and lead time for changes; stability can be measured by mean time to recover. The State of DevOps Reports found that investing in practices that increase these throughput and stability measures increase IT performance.

The goals of DevOps span the entire delivery pipeline. They include:

* Improved deployment frequency;
* Faster [time to market](https://en.wikipedia.org/wiki/Time_to_market);
* Lower failure rate of new releases;
* Shortened lead time between fixes;
* Faster mean time to recovery (in the event of a new release crashing or otherwise disabling the current system).

Simple processes become increasingly programmable and dynamic, using a DevOps approach. DevOps aims to maximize the predictability, efficiency, security, and maintainability of operational processes. Very often, automation supports this objective.

DevOps integration targets [product delivery](https://en.wikipedia.org/wiki/Software_delivery), [continuous testing](https://en.wikipedia.org/wiki/Continuous_testing), [quality testing](https://en.wikipedia.org/wiki/Software_testing#Software_quality_assurance_.28SQA.29), feature development, and [maintenance releases](https://en.wikipedia.org/wiki/Maintenance_release) in order to improve reliability and security and provide faster [development](https://en.wikipedia.org/wiki/Development_cycle) and [deployment](https://en.wikipedia.org/wiki/Software_deployment) cycles. Many of the ideas (and people) involved in DevOps came from the [enterprise systems management](https://en.wikipedia.org/wiki/Enterprise_systems_management) and [agile software development](https://en.wikipedia.org/wiki/Agile_software_development) movements.

Practices that correlate with deployment frequency are:

* Continuous delivery
* Using version control for all production artifacts

Practices that correlate with a lead time for change are:

* Using version control for all production artifacts
* Automated testing

Practices that correlate with a mean time to recovery for change are:[[13]](https://en.wikipedia.org/wiki/DevOps#cite_note-2014_State_of_DevOps_Report-17)

* Using version control for all production artifacts
* Monitoring system and application health

Companies that practice DevOpshave reported significant benefits, including: significantly shorter [time to market](https://en.wikipedia.org/wiki/Time_to_market), improved customer satisfaction, better product quality, more reliable releases, improved productivity and efficiency, and the increased ability to build the right product by fast experimentation.

The 2014 State of DevOps Report found that "IT performance strongly correlates with well-known DevOps practices such as the use of version control and continuous delivery."[…]

(From <<<https://en.wikipedia.org/wiki/DevOps>>> with adaptations)