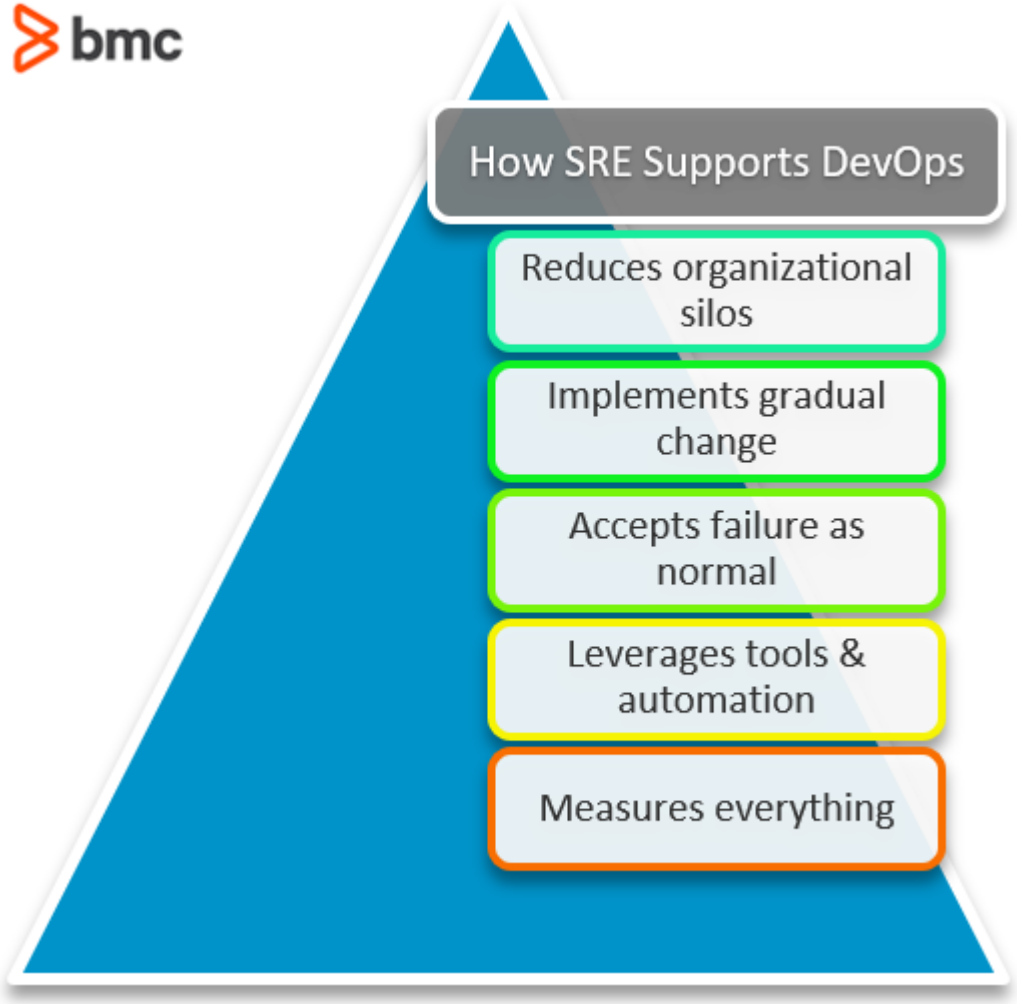
SRE vs DevOps: What's The Difference?

With the growing complexity of application development, organizations are increasingly adopting methodologies that enable reliable, scalable software.

DevOps and site reliability engineering (SRE) are two approaches that enhance the product release cycle through enhanced collaboration, automation, and monitoring. Both approaches utilize automation and collaboration to help teams build resilient and reliable software—but there are fundamental differences in what these approaches offer and how they operate.

So, this article delves into the purpose of DevOps and SRE. We'll look at both approaches, including benefits, differences, and key elements.

(This article is part of our DevOps Guide. Use the right-hand menu to navigate.)



DevOps basics

DevOps is an overarching concept and culture aimed at ensuring the rapid release of stable, secure software. DevOps exists at the intersection of Agile development and Enterprise Systems Management (ESM) practices.

Early development methodologies involved development and operations teams working in silos, which led to slower development and unstable deployment environments. To solve this, the DevOps methodology integrates all stakeholders in the application into one efficient workflow which enables the quick delivery of high quality software.

By allowing communication and collaboration between cross-functional teams, DevOps also enables:

- Reliable service delivery
- Improved customer satisfaction

(Explore our multi-part DevOps Guide.)

DevOps practices & methods

DevOps practices are based on continuous, incremental improvements bolstered by automation. While a full-fledged automation is rarely possible, for a comprehensive automation, a DevOps methodology focuses on the following elements:

Continuous delivery & integration (CI/CD)

DevOps aims to deliver applications and updates to customers rapidly and frequently. By using CI/CD pipelines to seamlessly connect processes and practices, DevOps automates updating and releasing code into production.

CI/CD also involves continuous monitoring and deployment to ensure code consistency across various software versions and deployment environments.

(Set up your own CI/CD pipeline.)

Infrastructure as code

DevOps emphasizes the abstraction of IT infrastructure so that it can be managed using software engineering methods and provisioned automatically. This results in an efficient system that allows your team to efficiently:

- Track changes
- Monitor infrastructure configurations
- Roll back changes that have undesired/unintended effects

Automated testing

Code is automatically and continuously tested while it is

being written or updated. By eliminating the bottlenecks associated with pre-release testing, the continuous mechanism speeds up the deployment.

DevOps works with…

Apart from the elements that help DevOps practices enable comprehensive automation, DevOps also relies on various methods that inherently enable faster delivery, efficient automation, and enhanced collaboration. Some methodologies that DevOps uses or otherwise pairs well with include:

- **Scrum.** This framework describes the composition and roles of teams collaborating to accelerate quality assurance and code development. The scrum framework defines designated roles in the project and key workflows within all phases of a software development lifecycle (SDLC).
- **Kanban.** A key workflow management mechanism that enables teams to define, manage, and improve on services that deliver business value.
- **Agile.** The Agile framework defines processes that improve software teams' responsiveness to changing market needs by enabling rapid, frequent, and iterative updates. Agile enables shorter development cycles which allow for a clearer understanding of business and development goals for improved customer satisfaction.

(Compare Scrum, Kanban & Agile.)

Benefits of DevOps

DevOps reduces the complexity of managing software engineering projects through collaboration and automation. Some benefits of adopting DevOps include:

- Ensure quicker and frequent delivery of application features that improve customer satisfaction

- Create a balanced approach to managing an SDLC for enhanced productivity of software teams
- Innovate faster by automating repetitive tasks
- Remediate problems quicker and more efficiently
- Minimize production costs by cutting down errors in maintenance and infrastructure management

Site reliability engineering (SRE) basics

SRE provides a unique approach to application lifecycle and service management by incorporating various aspects of software development into IT operations.

SRE was first developed in 2003 to create IT infrastructure architecture that meets the needs of enterprise-scale systems. With SRE, IT infrastructure is broken down into basic, abstract components that can be provisioned with software development best practices. This enables teams to use automation to solve most problems associated with managing applications in production.

SRE uses three Service Level Commitments to measure how well a system performs:

- Service level agreements (SLAs) define the required reliability, performance, and latency of the system as desired by end users.
- Service level objectives (SLOs) target values and goals set by SRE teams that should be met to satisfy SLAs.
- Service level indicators (SLIs) measure specific metrics and aspects that show how much a system conforms to the SLOs. Typical SLIs include request latency, system throughput, lead time, development frequency, mean time to restore (MTTR), and availability error rate.

Key principles of SRE include:

(Learn more about SRE concepts.)

The Site Reliability Engineer role

SRE essentially creates a new role: the site reliability engineer. An SRE is tasked with ensuring seamless collaboration between IT operations and development teams through the enhancement and automation of routine processes. Some core responsibilities of an SRE include:

- Developing, configuring, and deploying software to be used by operations teams
- Handling support escalation issues
- Conducting and reporting on incident reviews

- Developing system documentation
- Change management
- Determining and validating new features and updates

SRE tools

SRE teams rely on the automation of routine processes using tools and techniques that standardize operations across the software's lifecycle. Some tools and technologies that support Site Reliability Engineering include:

- **Containers** package applications in a unified environment across multiple deployment platforms, enabling cloud-native development.
- **Kubernetes** is a popular container orchestrator that can effectively manage containerized applications running on multiple environments.
- **Cloud platforms** allow you to provision scalable, flexible, and reliable applications in highly distributed environments. Popular platforms include Microsoft Azure, Amazon AWS, and Google Cloud.
- **Project planning & management tools** allow you to manage IT operations across distributed teams. Some popular tools include JIRA and Pivotal Tracker.
- **Source control** tools such as Subversion and GitHub erase boundaries between developers and operators, allowing for seamless collaboration and release of application delivery. Source control tools include Subversion and GitHub.

SRE vs DevOps

Both methodologies enforce minimal separation between Development and Operations teams. But we can sum up the key difference as this: DevOps focuses more on a cultural and philosophical shift, and SRE is more pragmatic and practical.

This highlights various differences in how the concepts

operate, including:

- **Essence.** SRE was developed with a narrow focus: to create a set of practices and metrics that allow for improved collaboration and service delivery. DevOps, on the other hand, is the collection of philosophies that enable the mindset of culture and collaboration between siloed teams.
- **Goal.** Both SRE and DevOps aim to bridge the gap between development and operations, though SRE involves prescriptive ways of achieving reliability, while DevOps works as a template that guides collaboration.
- **Focus.** Site reliability engineering mainly focuses on enhancing system availability and reliability while DevOps focuses on speed of development and delivery while enforcing continuity.
- **Team structure.** An SRE team is composed of site reliability engineers who have a background in both operations and development. DevOps teams include a variety of roles, including QA experts, developers, engineers, SREs and many others.

(Explore DevOps team structure.)

How SRE supports DevOps principles & philosophies

SRE and DevOps are not competing methodologies. That's because SRE provides a practical approach to solving most DevOps concerns.

In this section, let's explore how teams use SRE to implement the principles and philosophies of DevOps:

Reducing organizational silos

DevOps works to ensure that different departments/software

teams are not isolated from each other, ensuring they all work towards a common goal.

SRE enables this by enforcing the ownership of projects between teams. With SRE, every team uses the same tools, techniques, and codebase to support:

- Uniformity
- Seamless collaboration

Implementing gradual change

DevOps embraces slow, gradual change to enable constant improvements. SRE supports this by allowing teams to perform small, frequent updates that reduce the impact of changes on application availability and stability.

Additionally, SRE teams use CI/CD tools to perform change management and continuous testing to ensure the successful deployment of code alterations.

Accepting failure as normal

Both SRE and DevOps concepts treat errors and failure as an inevitable occurrence. While DevOps aims to handle runtime errors and allow teams to learn from them, SRE enforces error management through Service Level Commitments (SLx) to ensure all failures are handled.

SRE also allows for a risk budget that allows teams to test the limits of failure for reevaluation and innovation.

Leveraging tools & automation

Both DevOps and SRE use automation to improve workflows and service delivery. SRE enables teams to use the same tools and services through flexible application programming interfaces (APIs). While DevOps promotes the adoption of automation tools, SRE ensures every team member can access the updated

automation tools and technologies.

Measure everything

Since both DevOps and SRE support automation, you'll need to continuously monitor the developed systems to ensure every process runs as planned.

DevOps gathers metrics through a feedback loop. On the other hand, SRE enforces measurement by providing SLIs, SLOs, and SLAs to perform measurements. Since Ops are software-defined, SRE monitors toil and reliability, ensuring consistent service delivery.

Summing up DevOps & SRE

SRE and DevOps are often referred as two sides of the same coin, with SRE tooling and techniques complementing DevOps philosophies and practices. SRE involves the application of software engineering principles to automate and enhance ITOps functions such as:

- Disaster response
- Capacity planning
- Monitoring

On the other hand, a DevOps model enables the rapid delivery of software products through collaboration between development and operations teams.

Over the years, out of all organizations that already have taken advantage of DevOps, 50% of companies have already adopted SRE for enhanced reliability. One reason for this is that SRE principles enable enhanced observability and control of dynamic applications that rely on automation.

At the end, the goal of both the methodologies is to enhance the end-to end cycle of an IT ecosystem—the application lifecycle through DevOps and operations lifecycle management

through SRE.

“Gain insight to the capabilities necessary to attract top SRE talent and make them successful in your organization with artificial intelligence for operations (AIOps) and artificial intelligence for service management (AISM) capabilities.”

Related reading

- BMC DevOps Blog
- The Complete DevOps Certifications Guide
- The State of DevOps: A Report Roundup
- SRE vs ITOps: Are SRE & IT Operations The Same Thing?
- The State of SRE Today
- Implementing GitOps To Deliver Cloud NativeApplications