200 XP

SRE in context

10 minutes

Before we explore some of the practices associated with SRE, it would be good to place some of the ideas we just learned in the previous unit into context. In this short unit, we'll learn some of the history behind SRE and how it relates to other operations practices you may be familiar with. This will set us up for greater success later because those practices will make more sense in context. Also, when your friends ask "How is SRE different from ..." you'll have a ready answer.

History

A highly condensed history of SRE begins with its origins at Google in 2003. Ben Treynor, now Treynor Sloss, took over leadership of Google's "Production Team" (then only seven software engineers) and created the idea he famously described as "what happens when you ask a software engineer to design an operations function." The history here is useful to understand because it helps explain why SRE can feel very "software engineering" to operations people who meet it for the first time. It has adopted values and tools from that field like the importance of coding and source control systems as a fundamental tool. The initial and current implementation of Google SRE is well documented in their two books published by O'Reilly (see the Getting Started unit).

As people from Google left the company (and people in the company talked more about their practices publicly), SRE started to spread to more organizations in the industry. As SRE spread to new organizations, those organizations adopted and adapted the SRE principles and practices to fit their local culture. This expansion process has yielded a number of different implementations of SRE in the field.

DevOps and SRE

The broader industry faced the same challenges around scaling, development velocity vs. operational stability, and other software delivery issues that spawned the site reliability engineering movement. Parallel efforts to address them outside of Google (and a few larger companies at that time) yielded DevOps.

For lots of good information about DevOps, see https://docs.microsoft.com/azure/devops/learn/.

> ⓘ **Note**
>
> It is important to note that DevOps and SRE are two different parallel attempts to address the same challenges. SRE is not the next evolutionary step after DevOps. SRE was not created to be "the future of DevOps."

How SRE and DevOps differ is a topic still under considerable discussion in the field. There are some broadly agreed upon differences, including:

- SRE is an engineering discipline that focuses on reliability, DevOps is a cultural movement that emerged from the urge to break down the silos typically associated with separate Development and Operations organizations.
- SRE can be the name of a role as in "I'm a site reliability engineer (SRE)", DevOps can't. No one, strictly speaking, is a "DevOps" for a living.
- SRE tends to be more prescriptive, DevOps is intentionally not so. Nearly universal adoption of continuous integration/continuous delivery and Agile principles are the closest it comes in this regard.

The two operations practices, DevOps and SRE, share a mutual love of monitoring/observability and automation (perhaps for different reasons). This is one reason why it can often be easier to import SRE practices and principles into an organization with an existing DevOps practice. That process has to be done with care and intent. It also can and should be implemented incrementally, one does not have to make a sudden switch.

> ⚠ **Warning**
>
> Just swapping titles for people in the organization is an implementation strategy that almost never succeeds. It will not yield the benefits SRE has to offer. See the Getting Started section of this unit for some better suggestions.

Conclusion

This short unit has sought to place a small amount of context around SRE and DevOps. SRE and DevOps are adjacent operations schools of thought and are best thought of in those terms.

Now that we've briefly looked at some of the background behind SRE, let's move right to some of its core principles.

Check your knowledge

1. Given the origin of SRE, which discipline has had the most impact on it?

- ○ Customer service
- ○ Mainframe operations
- ◉ Software engineering ✓

  SRE was started with software engineering mindset.
- ○ Architecture

2. Which came first, DevOps or SRE?

- ○ DevOps
- ○ SRE
- ◉ Neither, they developed roughly in parallel. ✓

3. Is SRE the next evolutionary step from DevOps?

- ○ Yes, SRE is the future of DevOps. DevOps personnel should strive to become SREs
- ◉ No, SRE and DevOps are separate operations practices ✓

  DevOps and SRE are indeed separate even though they attempt to address similar challenges.

4. What are two best practices that are central to both DevOps and SRE?

- ◉ Automation and monitoring/observability ✓
- ○ Coding and ticket systems
- ○ Help desk automation and bug tracking

Next unit: Key SRE principles and practices: virtuous cycles

Continue >