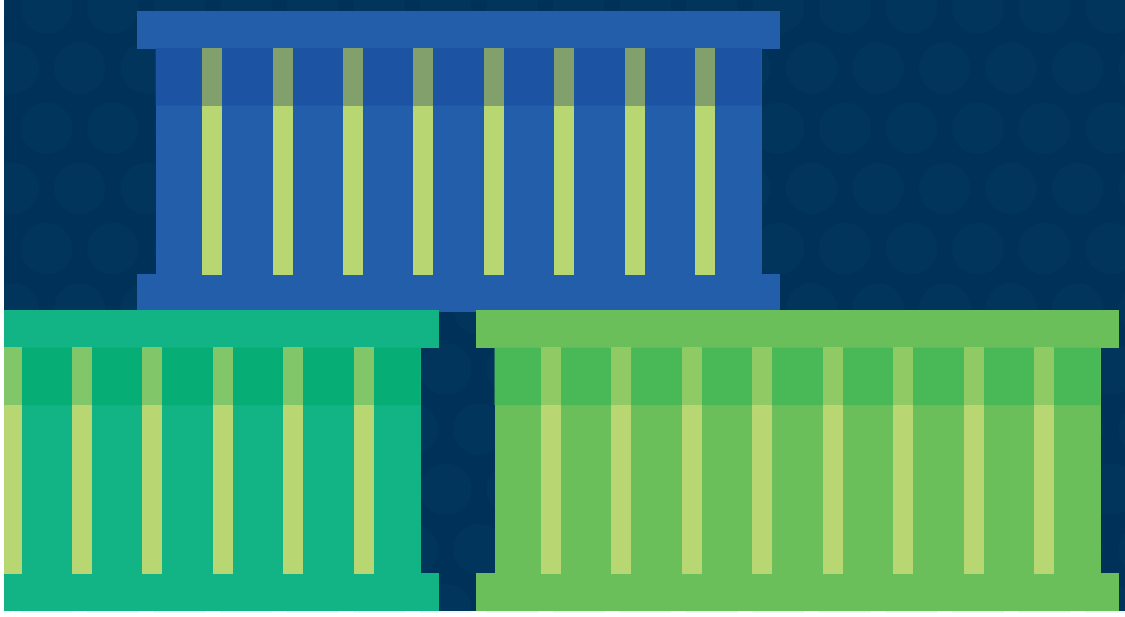
ARTICLE

# How Application Containers Can Support DevOps



suse.com

SUSE®

In order for enterprises to stay relevant beyond expanding and innovating, they need to adapt and meet even more flexible and agile requirements while controlling costs and efforts. DevOps tools, processes and culture provide a key framework for meeting these needs.

Containers are not a new technology. They have been around for years with system containers in Linux. In the past few years, however, we have seen a lot of new development and adoption of application containers and simplified tooling around them. Application-focused containers facilitate DevOps, CI/CD and micro-services based applications, thus enabling businesses to improve their processes, quality and time to market.

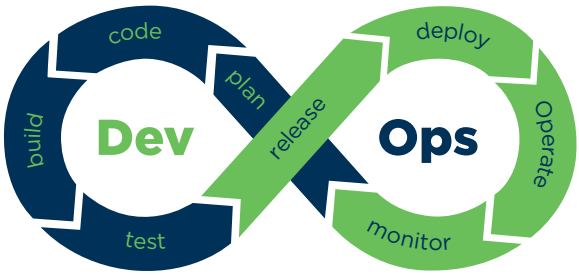
Containerization is a great tool for bringing developers and IT operators closer together as a shared resource in both continuous integration and delivery. This is certainly true for cloud-native applications that meet all or a portion of the 12 factors. In addition, containerization can be applied to legacy applications and to application migration strategies.

## Building and Sandboxing

Think of application containers as non-interacting, independent, iPhone-like applications. They are completely isolated and sandboxed, so they avoid interacting with or impacting each other or the underlying operating system.

Containers are also tightly coupled with the application architecture itself. They frequently drive, or at least support, the division of traditional monolithic applications into à la carte applications or microservices-based applications.

Application containers facilitate the traditional build step of the DevOps flow. From the definition and format of container images, to how to store and fetch them from a public or private secured registry, containers provide a lot of fast-tracked and automated steps.




## Shipping and Deploying

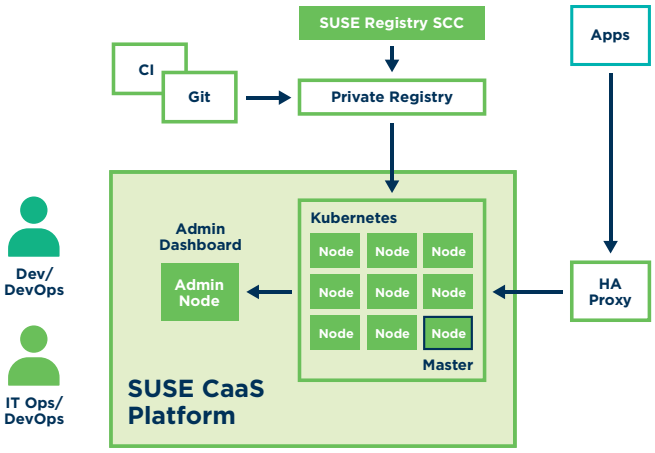
The container analogy is also valid for how containers are shipped and how they travel. On the digital ocean, they follow the DevOps streams: from staging to test and production environments, and from on-premises infrastructure and data centers to heterogeneous private and public clouds. And once they reach their destination, they ensure that the container and its contents behave exactly the same as they did at their origin. The process also ensures that, assuming a proper container image packaging, containers will run similarly in the different harbors where they are located.

## Running and Maintaining

Using containers in this DevOps phase provides many benefits. By design, containers are easy to scale; thus, they facilitate how applications can grow and shrink based on business needs, the expansion and growth of new business applications, and so on. When running live, containers also allow for more dynamic, multi-cloud strategies. For example, workloads can be offloaded to the cloud during peak periods and then brought back on premises when there is less demand and internal resources are available.

## Looping

The last step of the DevOps flow is to cycle back to development from the running phase, providing analytics and details on the performance of the application, so that further improvements can be made to the application. This step also includes other inputs such as feature requests, patches and fixes that need to make it into the next iteration of the application (these can be added during other steps in the flow as well). During this step, containers add an additional level of analysis on top of the application itself, such as scale-out/scale-up scenarios and monitoring the performance of the containers themselves.




## Conclusion (and Other Considerations)

Our new product SUSE Container as a Service Platform will further facilitate the use of containers for DevOps because the orchestration element helps to spin up the containers for developers, test, and so forth in their respective environment on their respective abstracted infrastructure and set of resources.

The SUSE CaaS Platform is an infrastructure platform for containers that allows you to provision, manage and scale container-based applications. It includes three components: MicroOS, based on SLES; Kubernetes for container management; and Salt-based configuration to set up the components plus the container engines, such as the Docker open source software and Linux containers (LXC). For more information, see the webinar SUSE Container as a Service Platform—An Introduction.

Storage and containers are also taken care of. Containerized data disappears with its container. Its whole purpose in life is to die easily, so to speak, to encourage stateless design, even though that can't always be achieved 100 percent of the time. As a result, persistent storage and data play an important role in ensuring consistency in the DevOps flow when using containers.

Networking is also a specific area to consider, especially in the context of multi-cloud. Solutions exist to address networking needs by bundling the network configuration requirements of a specific application together with its container description, for instance.

Last but not least, in addition to control, the attributes of overall stability, reliability and multi-tiered security for containers are absolutely critical for enterprise DevOps adoption.

At SUSE, we are well versed in DevOps. To learn more visit https://www.suse.com/solutions/devops/.

**For more information, contact your local SUSE Solutions Provider, visit us online or call SUSE at:**

1-800-796-3700 (U.S. and Canada)
1-801-861-4500 (Worldwide)

SUSE
Maxfeldstrasse 5
90409 Nuremberg
Germany

**www.suse.com**