How DevOps Can Support Business Agility for All Companies to Stay Business-Relevant

By Dr. Thomas Di Giacomo
Chief Technology Officer
SUSE

August 2016

Article

www.suse.com

Article Reprint
Cloud Computing

SUSE
We adapt. You succeed.

Stay Business-Relevant and Achieve Business Agility with DevOps

In our modern, fast-paced, digital-first world, responding quickly to internal and external changes without losing vision is absolutely key for all companies who want to survive, thrive and ultimately surpass the competition.

Introduction

Today, most companies' success relies on software and applications, directly or indirectly, but in all cases with a significant impact on their overall performance. From that perspective, having the right culture and the right processes and tools for software and application development, as well as their delivery and maintenance, is not only necessary but essential for companies to differentiate themselves and succeed in every market.

DevOps for Business Agility

While achieving business agility requires more than software and IT (for instance, the sales and marketing approach and the service and business models must be considered), applications are necessary to all businesses' success. And today, DevOps is a dominant way to achieve business agility from a software and application perspective—from ideas to delivery to the market (and looping back indefinitely). When talking about DevOps, the first aspect to acknowledge is that this is a balanced combination of an adapted culture, appropriate tools and delivery/management processes. If one of these doesn't match, the flow isn't performing as it should, if it's performing at all.

In this article, we focus on the appropriate tools for DevOps in the context of business agility, say Enterprise DevOps. And although related processes can be generalized somewhat, together with culture they are more company/situation specific than the tools themselves, thus our focus on tools here. We would, however, be happy to discuss culture and processes with you directly.

From its creation, SUSE has been a truly open, open source company. Deeply rooted in software development, we have applied DevOps principles for years, even before the term itself existed. We have learned many lessons, and continue the open-ended quest to keep learning and improving, while building tools to support the DevOps process. In the spirit and tradition of open source, these tools are available to all (including you) and jointly developed and used by various communities. You can see in the figure below how these tools and others can facilitate the DevOps phases.

The Phases and the Tools

Before digging into the various phases of a DevOps flow (since this is in the context of business agility and enterprise needs), it is obviously important to consider security,

interoperability and reliability as pre-requisites to all the steps involved. Automation, from unit tasks to high-level tasks, is also a foundational element of the DevOps approach, where the different phases should actually blur as much as possible to reduce friction and speed up the whole flow.

Let's go through the various phases of a typical DevOps flow. While there are slightly different ways to represent it and slightly different ways to break it out in phases, the infinity symbol-based representation illustrates a generic DevOps loop. Because it is a closed loop, the order of the phases is not particularly relevant, but let's start from "plan" for the sake of listing them. Keep in mind that there should be as little hard line as possible between the phases, meaning that most of the tools overlap on different phases on purpose (so splitting them by phases is not an exact science).

Plan

There are a lot of available tools (including of course those in the open-source arena) that can be used for the planning phase: from feature, idea, and project management to issue, bug and general collaborative tracking. This broad category includes tools such as Trello, Taiga, Jira, Redmine, Mantis, Request Tracker and Bugzilla.

Code

Obviously developers need programming languages supported by the underlying operating system or platform where their applications are expected to run. With DevOps that means running similarly on the development and production environments (that is where/why, for instance, a discussion about containers should occur).

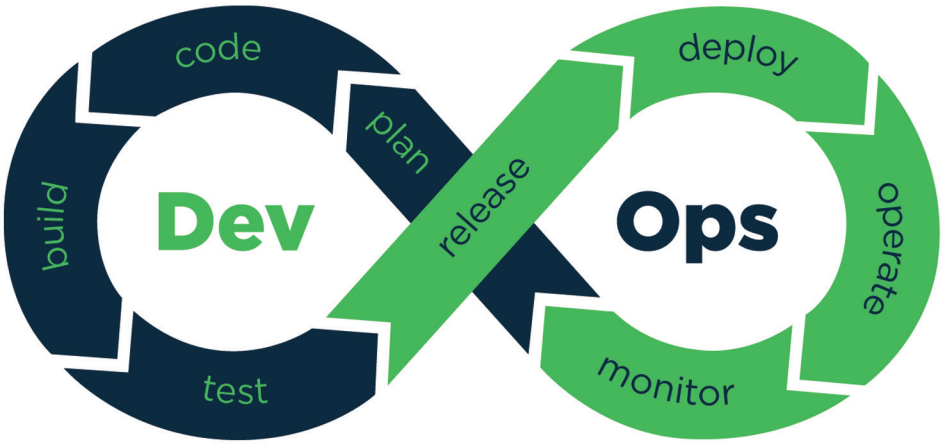
The applications need some sort of Integrated Development Environments ("or not" would argue some of the developers, but we will save that discussion for another time), especially Source Control Management (SCM) for collaborative continuous development.

In terms of OS, VM, public cloud or container host, Linux is obviously by far the best choice for any coder. One can use developers' programs from enterprise Linux distros (such as the one SUSE provides) or free community-based distros. openSUSE[1] for instance, with both Leap and its Tumbleweed rolling release, is sharing the same code base as SUSE® Linux Enterprise, facilitating the move back and forth and benefitting the whole DevOps approach. It is also important for developers to pre-check whether their dev environment allows them to build their code for their target architecture(x86, Aarch64, z, Power or others). We could also mention minimal/lean/micro-OS as particularly relevant for serving as hosts. SUSE Linux Enterprise JeOS[2] is one example of this.

Regarding SCM, some of you might remember CVS or be familiar with Subversion. But the source control tool you are probably more familiar with is Git (possibly together with web-based related hosting service Github or Gitlab, including issue tracking capabilities).

Build

When coding is complete, it's time to build the application(s)/package(s)/image(s). This is an area where companies like SUSE have been very active, for themselves as well as for whole open-source community. SUSE has put a lot of effort into Open Build Service[3], a generic system to build and distribute packages from sources consistently and on a wide range of operating systems and hardware architectures. To



1 **www.opensuse.org/**
2 **www.suse.com/products/server/jeos**
3 **https://doc.opensuse.org/projects/kiwi/doc**

create an OS/host image, Open Build Service can be complemented with Kiwi or SUSE Studio™[4], for instance, to build and deploy standalone or public and private cloud services. Open Build Service can also be used with PackageHub for integration into supported enterprise Linux distributions.

Test and Continuous Integration/ Continuous Deployment

To keep up with the benefits provided by DevOps practices, continuous testing and integration must be included in the flow. OpenQA, for example, is an automated testing framework for GUI applications as well as the bootloader and kernel. This tool complements traditional scripting tests and output checks that are difficult in those cases. One of the most commonly used platforms for CI/CD is Jenkins, but there are also alternative solutions such as Travis CI and Concourse.

Continuous Deployment and Configuration Automation

Automated deployment and configuration is another important phase of the process. Here again, there are a variety of options. From Chef (complemented with Crowbar in SUSE OpenStack Cloud[5] for instance), Puppet, Juju and Ansible, to Salt (integrated with SUSE Manager[6]), there is an appropriate tool for your use, based on your existing architecture and your technical, operational and business needs.

Operate and Monitor

Once deployed, within the DevOps philosophy, applications are operated or managed (container and resource orchestration play a key role here) and monitored to give constant input to the DevOps loop and flow to improve performance, fix issues or adapt to possible shortcomings or new requirements that pop up. For instance, together with Icinga, SUSE Manager provides insights into what is happening on the systems and SUSE Enterprise Storage™ helps to automatically adjust data placement to improve application performance. A lot of other solutions also help to provide insight, such as traditional Nagios, Zabbix, Monit, Prometheus and Magnum (via SUSE OpenStack Cloud) for containerized environments and so forth. Other solutions focus specifically on application performance, such as New Relic and Graphite, which also provide some analytics to interpret the data (Logz.io or ELK).

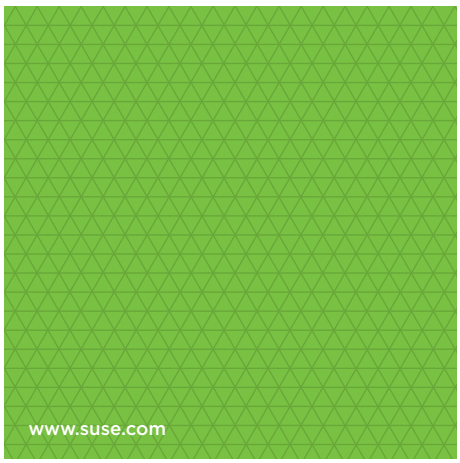
Conclusion

Whether your company is already a DevOps ninja or entirely new to the concept, business agility will continue to become more and more important and the tools to help improve agility will become more and more advanced. Thus, you need to be prepared to constantly adapt and improve the way you provide applications to your business. We recently worked with Tyro Payments[7] to facilitate adoption of DevOps practices for them to achieve shorter time to market. You can check out their story to learn how they have improved their business agility.

Much like CI and CD for applications, adopting a DevOps mentality and implementing it in real life is a constant journey with no final destination—at least as far as the industry and analysts can see for now. The next steps depend on your own situation, in terms of the business, the existing culture, and the tools and processes. We would be glad to discuss how we can best support your needs the next time we meet. In future articles I will share our container and Platfrom-as-a-Service perspectives as they relate to DevOps and improving your business agility. These are clearly useful and key elements of a DevOps strategy and I look forward to sharing my views in the coming months.

---

4 **https://susestudio.com**
5 **www.suse.com/products/suse-openstack-cloud**
6 **www.suse.com/products/suse-manager**
7 **www.suse.com/docrep/documents/kgu61iyowz/tyro_payments_case_study.pdf**



Contact your local SUSE Solutions Provider, or call SUSE at:

1 800 796 3700 U.S./Canada
1 801 861 4500 Worldwide

SUSE
Maxfeldstrasse 5
90409 Nuremberg
Germany