eficode

GUIDE

DevOps trends 2023

Prepare your organization for the major shifts in software development

Contents

Welcome to 2023! 3

Platform engineering is booming 5

Companies move beyond automation toward orchestration 9

Platforms are baking in security and compliance 12

Developer Experience becomes key in attracting developer talent 15

Organizations look for waste and inefficiencies in their value streams 19

Conclusion 24

Take the next step 25

Welcome to 2023!

DevOps has become a must-have for successful businesses, delivering on the promises of agility, alignment, and effective value creation.

As leading practitioners for more than 17 years, our team here at Eficode has seen trends come and go. We pride ourselves on our ability to spot trends early on and help companies adjust accordingly. Each year we present our views on the most important changes in software development. Some are here—some are waiting around the corner. All these trends and principles are tested and proven across a range of industries, from finance to manufacturing, gaming to telecommunications.

This is our way of giving you a competitive advantage.

Read on and ensure you and your organization are better able to:

1. **Quickly adapt to any situation**
   Teams that can deliver quickly can also react quickly to varying market and customer needs.
2. **Attract top talent**
   Elite developers join companies with the tools, platforms, and cultures that let them grow and consistently deliver.
3. **Build customer loyalty**
   Customers have more choices than ever before, and can sense when a company can quickly respond to their needs.

Have a pleasant read.

Marc Dillon, Marko Klementti and Dan Glavind

The research is in

High-performing organizations not only have high adoption-rates of technical enablers such as cloud platforms, but also tend to have high-trust and low-blame cultures.

Google 2022 State of DevOps Report

Platform engineering is booming

Platform engineering is booming

In the total lifespan of DevOps, Continuous Delivery has been a hot topic for 12 years now. More and more operations have moved to cloud native environments (such as Kubernetes) with their automation and orchestration.

Platform engineering is a natural continuation of this. Organizations realize that, to organize themselves and their toolchains so that developers work towards a product in the production environment, they need to **take the operation side into account.**

You need a high-level vision to successfully:

- create your environments
- implement the tooling
- execute migrations
- give coaching
- support the development teams
- handle maintenance
- develop the tooling going forward

We are seeing some tools getting a foothold in both emerging and larger, more mature, organizations. These companies have started taking platform engineering seriously, both in terms of organization and tooling.

Does that mean DevOps is dead?

Although it is popular in certain circles to exclaim that 'DevOps is dead', it is certainly not, and is not likely to happen for a long time.

There is still a huge need for development platforms to be skillfully crafted, operated, maintained, and kept up-to-date with technology.

There may still be some struggles here in the coming year as high-performers take advantage of platform engineering while lower performing and legacy organizations fail to transform.

The dawn of the platform teams

In the book *Team Topologies* by Matthew Skelton and Manuel Pais, Platform Teams are one of the four topologies for organizations (along with Stream-aligned, Enabling, and Complicated Subsystem teams). Combining that topology with your existing tooling is a natural continuation of DevOps or Site Reliability Engineering. This forms what we today call platform engineering:

> *Platform Engineering = building the platform, tailoring a ready-made solution or building something in-house to optimize flow and value creation by developers.*

Platform teams will need resources allocated for the engineering itself, but also people who know how to train others. Competence development becomes a key factor as traditionally, when we look at the complicated subsystem teams, these consist of engineers that are super-skilled but not necessarily outward-facing.

Thus, to build a functioning platform team and platform engineering organization, you need people who can do the training and share their knowledge.

Consider this for 2023

1. Do you treat your internal platform as a product and developers as customers?
2. Do you have platform owners in place?
3. What resources does the team have?
4. How are they trained, and are they capable of training others?
5. What responsibilities do they have?

Ask your software engineers

How long do you have to wait to set up a new project, experiment, or proof of value, including the environments?

If the answer is anything other than “I can do it in minutes, myself, whenever I want,” DevOps will present huge gains for your organization.

Companies move beyond automation toward orchestration

Companies move beyond automation toward orchestration

Simply put, orchestration is standardizing the end-to-end workflow that development teams need as an entire, event-driven system. Think in terms of flow and being results-driven instead of scripts for automating individual tasks. There is a natural synergy between platform engineering and orchestration.

We recently conducted a survey, asking how both platform engineers and application engineers see:

- their platform
- the platform teams
- their responsibilities

We wanted to investigate if the majority of organizations looking to build an internal developer platform, will build it on top of Kubernetes. We asked if Kubernetes was part of their platform engineering.

The answers included a healthy blend of concepts, tools and techniques built on top of Kubernetes, such as Crossplane, Backstage, Talos Linux, the Argo suite of tools, and Flux.

We also see more people attending cloud native events. Perhaps not so much because of Kubernetes itself, but because of the landscape that builds on top of Kubernetes, and the orchestration it both provides and requires.

Added responsibilities for developers

The problem is that developers increasingly need to understand what happens in the lower layers— 'under the hood'— with increasing cognitive load. And related to this, with 'you build it, you run it' platform engineering, developers are also to some extent responsible for defining the operations environment their applications run in.

Message queues, database systems, different search server configurations, or Kubernetes-like container configurations, are also increasing the cognitive load.

The solution is platform engineering. At Eficode, our customers today use tools such as Rancher, Crossplane, and Pulumi. These kinds of tools that manage other tools have traditionally been for developers, but are now becoming parts of orchestrated platforms — not only for the DevOps engineers, SRE, or operations.

This exciting shift changes the way applications are created today, and how the different parts of the DevOps pipeline work together.

Consider this for 2023

1. Do you create a project and then automate it, or do you start from a template that already has everything built in?
2. Do you have project templates for all of your major technologies?
3. Does your automation include orchestration by default?
4. Do your developers have to think where and how does their application run?

Platforms are baking in security and compliance

Platforms are baking in security and compliance

According to the Google State of DevOps 2022 report, teams with low levels of security practices have 1.4x greater odds of having high levels of burnout than teams with high levels of security. And teams that focus on establishing security practices are significantly more likely to recommend their team to someone else.

Continuous Delivery has long been a “trend”. It has established itself as the backbone for developers and organizations to build working software, and a necessity for software supply chain security. This is because developer platforms are both the execution and audit environment for many supply chain security practices.

With the consolidation of tooling, that’s also happened a lot this year. GitHub, for example. have delivered advanced security tooling, and GitLab also offers various security tooling.

GitLab conducted their 2022 Global DevSecOps Survey and looked into how much security checking is put into pipelines. You can see that most organizations already have some sort of security compliance automated tooling in their development pipelines. However, organizations with more requirements, or bigger organizations with departments for security and compliance, are not necessarily well connected to the actual security and compliance tooling already in the automated pipeline.

In 2023, and even more so in 2024, tooling and culture where automated information is taken and communicated, together with the security and compliance, will speed up also traditional organizations’ delivery.

Even though we are talking about 2023 trends, a lot relates back to the first Agile Manifesto value of people and interactions. Our modern, web-based tooling brings people together to solve problems, only now we have the best tools in the world to do this.

A great way to stay compliant beyond the audit

There is a big difference between being compliant and passing an audit. It's much easier to pass an audit than to actually be compliant. Today's tools give you a huge advantage, ensuring compliance while day-to-day changes are ongoing.

Consider this for 2023

1. Are you employing the available security tools in all of your projects' automation?
2. Do you have a common vocabulary between developers and the security and compliance organizations?
3. Do you employ information in an understandable format (pushed) or do you have to train your developers to understand the output of various tools?

Always consider this security and compliance tooling for your pipelines:

- License compliance
- Vulnerability scanning at the code level
- Infrastructure As Code scanning
- Dependency scanning
- Container scanning
- Secret detection

Many security and compliance organizations don't necessarily understand how these tools work, or what results they produce.

However, most security tooling today starts producing audit reports, audit events, or some sort of compliance management that you can put in place, which starts technical discussions between departments.

Many organizations benefit from putting in the processes and tooling because it initiates the discussions and the cultural change.

Developer Experience becomes key in attracting developer talent

Developer Experience becomes key in attracting developer talent

Developer Experience may not be a “new” trend, but it is more important than ever.

The experience a developer gets when starting to work on a project. The ease of use and perceptions they have when using the tools and frameworks, and ultimately the satisfaction they feel when delivering their work to production.

Fierce competition for developer talent forces management, the IT organization, and the R&D DevOps organization to consider developers as customers of the organization’s platforms and processes.

This has intensified during the pandemic, the ‘Great Resignation’, and the rise of remote work. But in 2023, the battle for talent will be harder than ever. So let’s look at some important focus areas for winning this battle for top talent.

Platform engineering

Platform engineering is a big part of successful Developer Experience. How do you onboard into a new project?

- The development environment and tooling
- Meeting your teammates and understanding what you have to do
- What kind of programming language you use
- How you test and deliver applications

All of that is Developer Experience.

Psychological safety

Psychological safety has become essential in a thriving, loyal workforce. You need to build an organization that feels safe for new and old development, organization, people.

You also have to make sure you are not making the developers feel too responsible for all of the tools.

Managing cognitive load by moving from high to low context

Working within complex development environments unfortunately adds to the cognitive load of having to learn new tools.

The way this is approached in the modern DevOps is by moving from **high to low context**. What this means is that the developers do not necessarily need to understand the output of each tool, but instead the information is provided in an understandable format from each of the tools within the current context.

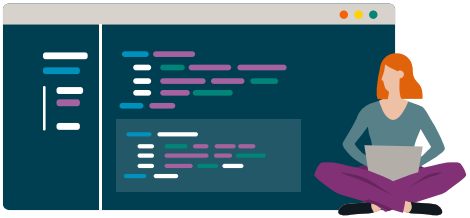
Modern organizations employ proactive warnings and dashboards collected from multiple systems. These tools have then been agreed together with the developers and the security and compliance teams, to ensure that the information and actions are correct for the delivery acceptance and throughout the pipeline.

> *Example: A feature created 6 months ago cannot be refactored properly if the report of the problem comes 2 days before the release deadline...*

We recommend that you from time to time ask your developers:

- What do you want to do that's being done by someone else or by a machine?
- What do you want someone else, or a machine, to do for you?

Some developers want to go deep under the hood — some just want to write their code and feel safe doing it.



Developer vs. customer experience: Striking the balance

There are a lot of similarities between Developer Experience and Customer Experience. Developers are essentially users, just not end users, and therefore UX and DevOps have a lot to do with each other.

The best-performing IT organizations today consider developers as their customers, and then build the Developer Experience that same way they build the customer experience into their products.

So if both DX and CX are crucial, which one is the most important?

It depends a bit on what sort of organization you have.

One thing is for sure: without end customers, your organization cannot function or even exist. But then it depends on if you are, for example:

- an established organization with an income stream
- a startup building something new you don't yet know if it functions

From an organizational point of view, the customers or end users, and the developers, today have similar demands. For startups, the developers are in the very center, or they will easily be the most important. You need developers from day one, simply to gain an understanding of the customers.

So, it is a fine balance to strike between the two types of experience. It is easy to focus too much on one. Traditionally, we have often seen that the Developer Experience, and indeed developers, have been a little forgotten in comparison. But in the last year we have seen a shift, which may well continue in 2023.

Consider this for 2023

1. Are your developer attrition levels better than industry norms?
2. How long does it take for a new developer to get their first change into production?
3. How do you enjoy the organization?
4. Can you speak freely with your teammates?
5. Can you speak with any stakeholder involved in the development?

Organizations look for waste and inefficiencies in their value streams

Organizations look for waste and inefficiencies in their value streams

How many people in your organization actually know all the steps involved when you create value for your customers?

As we, at Eficode, hold more and more value stream mapping workshops with our customers, we consistently get to witness organizations immediately realize that some wasteful steps are either unnecessary or relatively easy to automate.

This is an incredibly healthy trend.

You find “obvious” steps to automate that you wouldn’t have found otherwise:

- Handovers and transitions that are otherwise bottlenecks
- Activities that have ‘fallen between the chairs’, that people think are in a pipeline but aren’t

In short, you align the silos and reduce waste. It is part of value stream management: *continuously improving the flow of information and value across your operations and development.*

Developers’ platforms and platform engineering produce lots of automation that traditional value streams just start implementing into. With a value stream map, you can already start moving tickets down to the platform engineering and your developer platforms.

The trend is that companies will invest a lot in understanding their processes that go into value creation.

Know and act faster

More and more companies realize that, because Quality Assurance comes so far “to the right” in the process, they need to rework more because the feedback comes too late.

Value stream mapping brings you the input early on. You look at the whole picture right now, and can scope the value stream you really want (a more linked one, hopefully).

Of course you also need metrics to measure if your value stream actually achieves your goals. And that’s an exciting area that is likely to evolve further in the near future.

A likely next step: metrics

Understanding what the value stream mapping tells you is a continuous process. One of the trends we are likely to see on top of this in the future, is being able to produce valid metrics that define success.

Whether you create new products, or change existing ones, your requirements come from somewhere.

- From current customers
- From innovation that someone else has done
- From in-house innovation

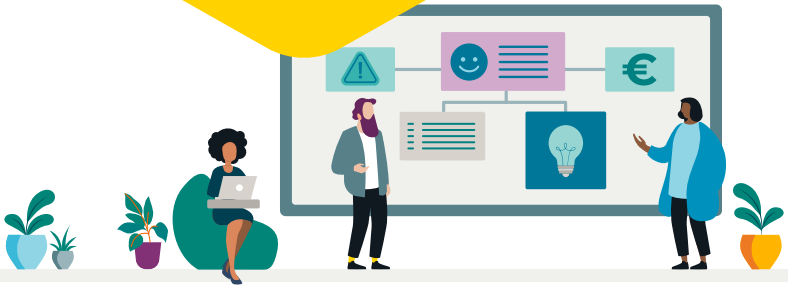
Regardless where a requirement comes from, it’s often not clear what effect that specific change had when it was finally released.

But when you have the operations environments, the Continuous Delivery pipelines, and the actual developer platforms and platform engineering in place, you have a unique opportunity.

You can start connecting the original requirements and ideas with the outcomes from the pipeline. You can better understand what kind of things have a real impact on customers or users, or the product you’re developing.

Consider this for 2023

1. How your users approach your product or service
2. How they are onboarding to the product or service
3. How they receive value from that
4. How you receive feedback about every step in the journey



Pro tip: Quickly find and reduce lean waste

If flow is interrupted in your pipeline because humans are waiting for something, you may ask why it makes sense to optimize something that's further down the line.

The simple answer is: because many find the act of value stream mapping itself will quietly reduce waste, as you uncover and resolve obvious pathologies.

1. Are we waiting for things that we didn't need to wait for?
2. Are we optimizing things that might not be the most critical bottleneck in the pipeline?
3. Are there things that are automated, but are moving too fast for what happens down the line?

We occasionally find throttling something back might provide a more manageable workload for someone in the middle of a value stream.

A value stream mapping example

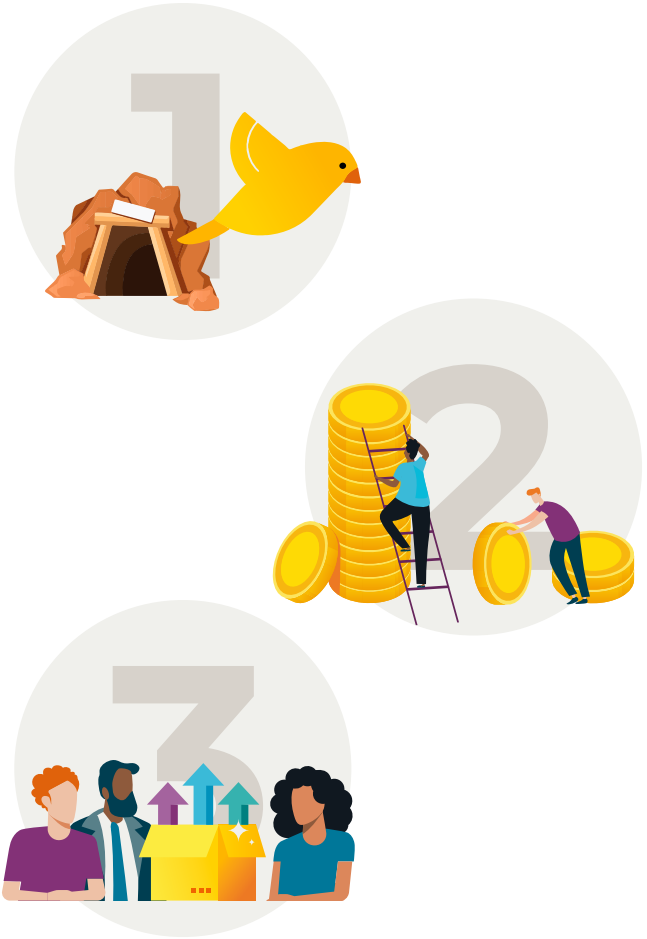
One of our customers from the gaming industry has been able to map the revenue stream coming from their users of a game into the releases that they do. This is their process:

1. Do a canary release (with their current production, user environments) of a new feature to a limited pilot group of users that they have mapped will represent the bigger mass.
2. Look at the revenue streams from the current feature, and then the smaller percentage of their new piloting feature.
3. If the revenue streams grow higher, even for a fraction, they know this feature actually helps the revenue stream, and thus they can deploy to more users.

This may be an extreme example, but often you can actually see similar effects from more traditional products.

You can look at user retention or time spent on a product or similar, and then attach that metric to the features you're developing. And with modern developer platforms and automated operation environments, you can scale out to a smaller pilot group first, and then compare the results.

With value stream mapping, we often think in terms of DevOps and the development value stream, but the operational value stream is where the money is.



Conclusion

Our DevOps trends for 2023 are about making a psychologically safe organization that thrives . Making an orchestrated platform that allows developers to focus on value creation, having security and compliance built into your pipelines, and clearly understanding all of the streams of value in your organization. These are not merely trends — today they are essential to stay competitive.

1. Platform engineering
2. Orchestration over automation
3. Security and compliance
4. Developer Experience
5. Value Stream Management

Take the next step

Get a clear view of where you are now and how you can reach your strategic goals. Assessing the current state of your software delivery process, and how satisfied you are with it, is a stepping stone towards becoming a high-performance software organization. **Check out our assessments.**

Or just **contact us** to talk about your DevOps journey.

eficode