The value of your development toolchain

The value and costs of a healthy development toolchain for maximized developer productivity. One of our key principles at Xpirit is "people first". This means people always come first in everything we do; it doesn't imply money and shareholders are equally important or that we have to find some kind of equilibrium. It means people always come first! And this applies not only to our employees – it also includes the people that work for our clients.

Author Michael Kaufmann

Because of our people-centric culture, we deeply care about developer productivity. In more than twenty years of experience working with developers, I've seen the impact it can have if they feel productive vs. unproductive. I've seen people leaving companies because they did not feel productive, and I have seen teams thrive after implementing DevOps tooling and an engineering culture. I've seen the joy that people felt when they felt productive after a long time of unproductiveness. It was Satya Nadella, the CEO of Microsoft, who once said he would always choose developer productivity over features for end users, as developer productivity benefits everyone and increases the feature delivery long term. This shows the importance of developer productivity to companies like Microsoft.

"If any engineer has to choose between working on a feature or working on developer productivity, always choose developer productivity."
– Satya Nadella

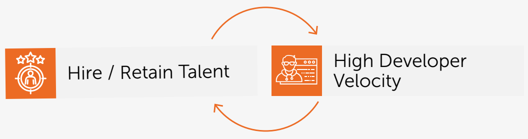

Figure 1. Developer velocity and the war of talent

The benefits of improved developer productivity

The question is: What will happen if your developers are not productive and have to work with old technologies and processes? The good developers will most likely leave. This will leave you with the ones that don't care or can't find another job easily. This will further reduce your entire productivity.

In the *war of talent*, when an engineer can find a new job very easily, a productive environment is crucial and a very important factor for people that are driven by intrinsic motivation – the talents you are looking for.

This wheel can spin in two directions: having a productive environment with a high developer velocity and satisfaction will help you attract and retain talent. This will accelerate your developer velocity and productivity and attract other good engineers. If your environment is unproductive, however, good people will leave, and the velocity will decrease further and keep other good engineers from joining your organization.

The increased productivity and increased quality in staff leads to better and more reliable software that gets shipped faster. This leads to faster feedback loops, ensuring you build what your customer wants. In the end, you achieve an increased customer satisfaction, which further helps you attract new talent.

In April 2020 McKinsey published their research study about the Developer Velocity Index (DVI) (*Srivastava S. & Trehan K. & Wagle D. & Wang J., 2020*). It's a study taken among 440 large organizations from 12 industries that considers 46 drivers across 13 capabilities. The study shows that the companies in the top quartile of the DVI

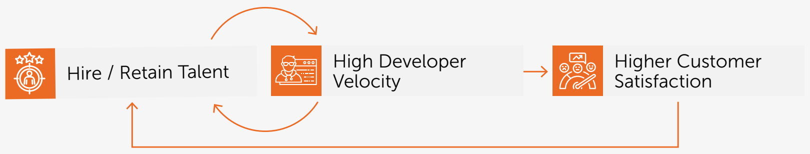

Figure 2. Acceleration and customer satisfaction

outperform other companies in their market by four to five times. Not only on overall business performance - companies in the top quartile score between 40% and 60% higher in:

> Innovation
> Customer satisfaction
> Brand perception
> Talent management

The findings align with the results from the DORA State of DevOps Report – but take them one step further by adding the business outcomes. The 2019 State of DevOps Report shows that elite performers, compared against the low performers, have:

> **Faster Value Delivery:** they have a 106 times faster lead time from commit to deploy.
> **Advanced Stability and Quality:** they recover 2,604 times faster from incidents and have a 7 times lower change failure rate.
> **Higher Throughput:** they perform 208 times more frequent code deployments.

High performance companies not only excel in throughput and stability, they also are more innovative, have a higher customer satisfaction, and a greater business performance.

With this research in mind and the priority companies like Microsoft and Google give developer productivity, it is obvious developer productivity should be a top priority for all companies that rely on software development.

The influence of the development toolchain on productivity

There are multiple factors that influence developer productivity. The most important ones, besides people, are:

> Culture
> Processes
> Toolchain

Having an engineering culture of trust and experimentation in which people are allowed to experiment and commit errors has a strong influence on developer productivity. But influencing or changing the culture is difficult. You can't just write some values on a PowerPoint slide and be done. The corporate culture is the result of the system, and you must change the entire system to change the culture. Nurturing a good culture should be an ongoing task, but it is not going to achieve results in the short term. This leaves processes and the toolchain. They go hand in hand. You can have the best tooling, but if your processes are slow and heavy, your developer productivity will be low. You can have great processes and be fully committed to DevOps principles, but if you don't have a toolchain to support the process, the productivity will still be low. Processes and the toolchain are the determining factors to increase the developer productivity in the short term.

Governance and processes

The problem with the development toolchain is its volatility. I've seen IT departments install DevOps tooling (such as build environments) and expect it to be like a mail system: Install once and just run it for years with few changes besides patches. But the amount of requests of developers to install and maintain new tools overwhelmed them. This normally leads to one of two scenarios:

> **The developer "wild west":** Developers are allowed to do everything. They have admin rights if necessary and are responsible for the tooling themselves. "The problem with this approach is often, that there is no alignment between the development teams. Each team uses the tools it wants and there is no maintained standard. This makes onboarding difficult and the allocation of teams to products inflexible: you cannot have teams work on other products or have developers switch teams without a longer onboarding period.
> **The IT castle:** The IT department provides a bare minimum of tooling and is the gate keeper to production. Requests from developers are mostly ignored or declined. This approach normally leads to shadow IT and build servers under the desks of developers.

This is, of course, an exaggeration and oversimplification. But I'm sure some readers will recognize their company in one of the two extremes.

A good development toolchain must have the right amount of governance. Give the teams the freedom to experiment – but have a common standard that is documented and provide training and guidance on that standard.

It is ok if one team wants to use Angular even if the current standard is React. There might be good reasons for it. But the decision should be explicit. Maintaining two UI frameworks and providing guidance when to use which one increases complexity and is therefore expensive.

Faster Value Delivery
Advanced Stability and Quality
Higher Throughout
Greater Performance
More Innovative
Higher Customer Satisfaction

Figure 3 - Benefits of improved developer productivity

A good governance process should look like this:

- Developers can request resources to experiment with new tooling at any time. In the cloud era this should be self-service and on demand.
- If the new tool is promising, there must be alignment with the other development teams and the team that provides the shared resources. Should the tool be added to the standard toolchain? Should it be supported? Should the tool be rejected, or allowed as a specific exception? In the end it will be a decision that affects all teams!
- If the tool is added to the standard toolchain, the documentation must be updated. There should be training for the other teams, as well as updated onboarding training.

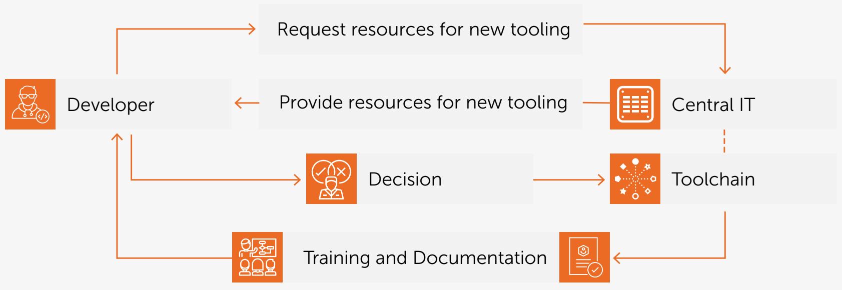
Request resources for new tooling
Developer
Provide resources for new tooling
Central IT
Decision
Toolchain
Training and Documentation

Figure 4 - Example governance process for toolchain

In-sourcing or outsourcing

Due to the high impact on developer velocity, and therefore business outcome, the first reaction normally is trying to in-source the toolchain and manage the entire process on your own. And I think that's valid, if you have the resources and knowledge to do it.

But – as I mentioned before – a development toolchain is not another "mail system" that you install and run.

If you don't have the resources to manage it in a good way, it's better to have someone do it for you that is specialized in that area.

In the end, a good development toolchain will impact the performance of your business – but it will not impact your customers directly. If you change the company that provides the service, your customers probably wouldn't realize it.

That's why we offer the complete development toolchain as a managed service. We call it the Managed DevStack. We can host it in a regional data center to allow complete data residency. The offering includes the complete development stack – including build environments – and the governance process to manage changes. I think this is a good option if you don't have the resources or experience to host it yourself.

If you prefer to host your toolchain yourself, we can help in setting up the process, documentation, and trainings.

Conclusion

The impact of a good developer toolchain is enormous. It has a direct influence on developer productivity and is responsible for increases or decreases in your development velocity. It has a massive impact on innovation, customer satisfaction, brand perception, and talent acquisition.

But building and maintaining a healthy toolchain is not easy. It takes a lot of effort, experience, and a fine-tuned governance process. It's better to seek help – or utilize a good managed service – than provide a bad toolchain on your own. </>

Furter readings

- Srivastava S., Trehan K., Wagle D. & Wang J. (April 2020). Developer Velocity: How software excellence fuels business performance
- Forsgren N., Smith D., Humble J., Frazelle J. (2019). DORA State of DevOps Report
- Brown A., Stahnke M. & Kersten N. (2020). 2020 State of DevOps Report
- Forsgren N., Humble, J., & Kim, G. (2018). *Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations* (1st ed.) [E-book]. IT Revolution Press.

Michael Kaufmann
CEO / Managing Director

xpirit.com/michael