BMT23
5th International Conference
Business Meets Technology.
Valencia, 13th-15th July 2023

PLATFORM ENGINEERING FOR CLOUD-NATIVE ORGANIZATIONS

Latif, Ajouaoui[a]; Albrecht, Fritzsche[b]; Guido, Soeldner[c]; and Jens-Henrik, Soeldner[d]

[a] *Google Cloud Germany. Germany. (latifajouaoui@google.com)*
[b] *Friedrich-Alexander-University Erlangen-Nuremberg. Germany. (albrecht.fritzsche@fau.de)*
[c] *Soeldner Consult GmbH. Germany. (guido.soeldner@soeldner-consult.de)*
[d] *Ansbach University of Applied Sciences. Germany. (jens.soeldner@hs-ansbach.de)*

ABSTRACT: Cloud Computing allows companies to scale seamlessly, providing a broad range of state-of-the-art services. Another promise is to free users from the operational and administrative burdens. However, with the advent of cloud-native applications, this promise becomes questionable – especially when DevOps principles are used during development. Experience in practice shows that teams often struggle dealing with both the infrastructure, finding the right architecture, and implementing business logic. When working in decentralized teams, things are even worse, as standardization across teams cannot be assumed. To tackle those issues, automation by means of techniques such as Infrastructure-as-code help to ease to cope with infrastructural concerns. However, when working with several teams in a decentralized manner, operational overhead is still there. Organizations struggle with standardization of infrastructure code and there is no clear centralized visibility for security-related concerns within the development lifecycle. To address these issues, we propose two things: First, building up a Platform Team, which serves as an organizational structure for continuous delivery. A Platform Teams can be the size of a typical small DevOps Team and support the whole organization with standardized security-hardened modules. Second, an Ops-Platform is needed that is operated by the Platform Team to centrally provide and maintain those modules. Other Dev-Teams can then consume those modules. In this paper, we report insights from the implementation of this approach in practice. We find out that developers are 75% less focused on operations by using such a platform and name specific success factors.

KEY WORDS: *Cloud Native Applications; DevOps; Infrastructure-as-Code; Platform Engineering; Platform Team.*

1. MOTIVATION AND INTRODUCTION

Within the last years, enterprises have already migrated large portions of their workload to the cloud – whether it is a private, public or hybrid cloud. However, many companies still fail to grasp all the benefits of cloud computing. According to the State of DevOps report

How to cite: Latif, A., Albrecht, F., Guido, S., and Jens-Henrik, S. 2023. Platform Engineering for cloud-native organizations. In Proc.: *5th International Conference Business Meets Technology*. Valencia, 13th-15th July 2023. 77-82. https://doi.org/10.4995/BMT2023.2023.16741

released by Puppet and DORA (DevOps Research & Assessment) (Puppet Labs, 2023), that highlights a DevOps maturity model, the overwhelming majority of companies are struggling with to reach the highest level of DevOps majority. This results in a set of problems to be solved: First, long lead times can be observed. Secondly, processes are often carried out manually, usually initiated by means of tickets created by developers. Third, the all overall complexity of the tools and the way how to integrate them is usually high leading to the fact that developers are overwhelmed. Last, waiting times slow done developers as many companies lack an internal self-service developer platform.

To tackle those issues, highly mature DevOps organizations tend to have their organization having structured in both application teams that are stream-aligned and platform teams (López-Fernández et al., 2021). Basically, stream-aligned teams are development teams that are building code and putting them into production. On the other hand, platform teams support that teams and might provide among other things (Leite et al., 2020):

- Platform servicing such as CI/CD and infrastructure provisioning by using industry-wide best practices such as Infrastructure-as-Code (IaC).
- Evangelization and mentoring DevOps practices for promoting cultural values, such as communication, transparency and knowledge sharing.
- Rotary human resources, which means that platform teams might help and provide product teams with human resources when these teams lack of specific skills to accomplish their work.

In this paper, the research question is how such a platform team can be formed to support enterprises with cloud-native technologies in an efficient manner. We will show how a platform team integrates with product teams within an organization and depict their responsibilities. In particular, we focus on the technical requirements for efficient infrastructure requirements based on Infrastructure-as-Code and GitOps.

The paper is structured as follows: First, we show related work. Second, we give an overview about the technical background. In the following, we will depict the requirements and the processes for an efficient platform team. Last, we will discuss considerations for future work.

2. RELATED WORK

Platform Teams and Platform Engineering is a relative new trend. First publications on platform teams date back to 2020. In their paper, Leite et al. (2020) describe platform teams as an organizational structure for continuous delivery and provide information about their role in an organization and how they interact with product teams.

Srivastava (2023) argues that Platform and Site Reliability Engineering is just as crucial for startups as it is for bigger organization. Done right, it enables companies to efficiently deliver high-quality, secure, compliant, robust, and reliable products to their customers.

Seremet & Rakic (2022) describe how Platform Engineering and Site Reliability Engineering go hand in hand to unlock the productivity of application development teams. In detail they describe how both roles share a common set of principles and how they differ from each other.

In their State of DevOps Report: Platform Engineering Edition Puppet (2023) describes what makes platform engineering the key to DevOps success at scale, why platform engineering is on the rise and how platform engineering benefits the entire organizations when it is done well.

3. BACKGROUND

Platform Teams for cloud-native environments don't come from nowhere – instead they use state of the art software engineering principles to accelerate software delivery. For infrastructure provisioning Infrastructure-as-Code (IaC) (Morris, 2020). has become a de-facto standard. Basically, the idea behind IaC is to declaratively the describe the target-state of the cloud environment within a source control system. IaC tools then apply all necessary changes to rollout cloud resources in the target environment. IaC principles help to create consistency, speed up deployment, improve accountability, and increase efficiency while also being able to help mitigate unnecessary costs.

GitOps (Beetz & Harrer, 2021) extends the idea of IaC for all kind of deployments – not only infrastructure provisioning. Git remains the single source of truth and deployment information is described in a declarative way as well. In addition, an application deployment tool is to be used and there is a monitoring tool in place, which continuously reconciles the states and carries out changes in the target system if necessary.

Site Reliability Engineering – also called SRE – describes a set of best practices for automating IT infrastructure tasks such as system management and application monitoring. The focus of SRE is on improving the reliability of scalable software systems amidst frequent updates from the development teams. A key component of SRE is observability: By defining Service Level Objectives (SLOs) and measuring actual Service Level Indicators (SLI), teams get a better understanding of reliability and have ways to define system availability goals. In addition, SRE puts its focus on a cultural change in such a way that every failure is seen as a failure in the reliability system.

4. PLATFORM TEAM ENGINEERING FOR CLOUD-NATIVE-ENVIRONMENT

In many organizations, DevOps teams are solely responsible for all the aspects within a software product. However, when dealing in cloud-environment and possible with Kubernetes, many application teams struggle with efficient processes for infrastructure provisioning and security issues. Platform teams help to address those issues by abstracting infrastructural issues from application teams.

Platform team and application teams can be part of an organization. However, companies might not have the size or the required skills to build up a platform team. In that case, platform engineering can be offered as a service across organizations.

For such offerings, standardization is a key requirement as it increases efficiency for supporting application teams. In addition, tooling is needed that provides support for consuming provided IaC modules as well as having self-service capabilities. For GitOps support and facilitating security best practices, we propose to set up a well-defined structure in a Git repository that enables to go infrastructure and application deployments hand in hand.

Figure 1 depicts the responsibilities and interaction of the Platform Engineering team. As described, the Platform Teams provides an abstraction layer above the underlying infrastructure – usually a public cloud hyperscaler or a private cloud. It then provides the following services to the product teams:

- Most important, a DevOps tool with pipeline support in order to run workflows is provided to run workflows.
- The Platform Team provides standardized modules that are optimized in terms of programming and security best practices and are compliant for the organization.
- Application Teams need a way to interact with the Platform Team. In most use cases, using Git is sufficient in order to support IaC pipelines and GitOps workflows. For advanced use cases, a self-service catalog or an internal developer platform can be used.
- Besides infrastructure deployment support, a CI/CD platform should also be provided. Functionality should include at least support for container images builds and app delivery.
- In order to support SRE, observability should be integrated in the platform tooling and all provisioned resources should have observability support included.
- Platform tooling should support product teams in the complete DevOps lifecycle – from building, testing to operating.
- If Application Teams need support, the Platform Team might provide consultancy and human resources to the respective teams.

Figure 1 depicts the relationship between the Platform Tooling and the Applications that are built on top of it.

The platform engineering team serves as a central point of contact for various aspects of the infrastructure and platform. As the team is responsible for the development, deployment and management of the cloud platform, it can act as the main point of contact for other teams within the organization.

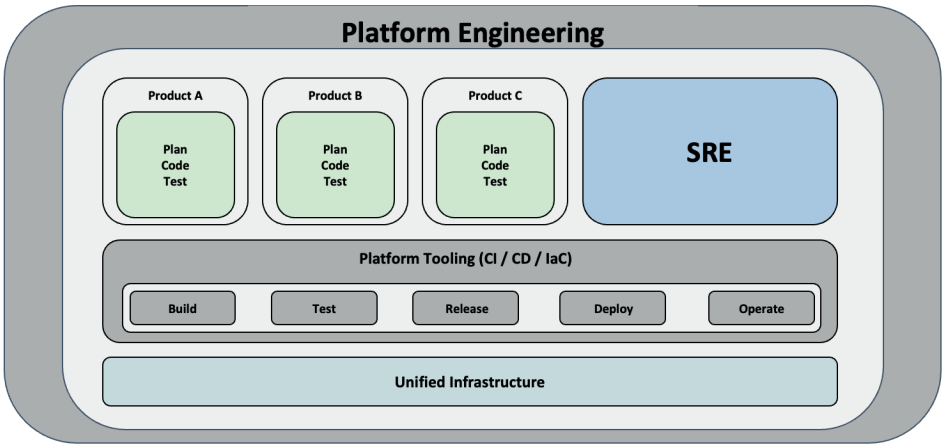

Figure 1. Overview of Platform Engineering.

5. CONCLUSION AND FUTURE WORK

In this paper, we have conceptually shown how Platform Engineering can be achieved and what kind of interactions between the Platform Team and Product Team exist. We described the service offerings and the tooling provided by the Platform Team. In the future, we want to publish dedicated business cases on how Platform Engineering works in practice. We are working on a maturity model for Platform Engineering as well as describe the technical capabilities that are needed in the Platform Tooling.

REFERENCES

Beetz, F., & Harrer, S. (2021). GitOps: The Evolution of DevOps?. IEEE Software, 39(4), 70-75. https://doi.org/10.1109/MS.2021.3119106

Gartner *Report (2022). Software Engineering Leader's* Guide to Improving Developer Experience.

Leite, L., Kon, F., Pinto, G., & Meirelles, P. (2020). Platform teams: An organizational structure for continuous delivery. In Proceedings of th*e IEEE/ACM 42nd International Conference on Software Engineering Workshops, June 2020 (pp. 505*-511). https://doi.org/10.1145/3387940.3391455

López-Fernández, D., Díaz, J., García, J., *Pérez, J., & González-Prieto, Á. (2021). DevOps team structures: Characterization a*nd implications. IEEE transactions *on software engineering, 48(10), 3716-3736. https://doi.org/10.1109/TSE.2021.310*2982

Morris, K. (2020). Infrastructure as code. O'Reilly Media.

Puppet Labs (2023). State of DevOps report 2017. Available from: https://www.puppet.com/resources/state-of-platform-engineering. Last accessed: 29-05-2023

Ser*emet, Z., & Rakic, K (*2022). Platform Engineering and Site Reliability Engineering: The Path to DevOps Suc*cess. https://doi*.org/10.2507/daaam.scibook.2022.13

Srivastava, Y.S. (2023). Start Small, Sc*ale Big: Building and Scaling Platforms and {SRE} Culture at Startups*.