Microservices and Containers – Architectural Patterns for Cloud and Edge

Claus Pahl[1], Pooyan Jamshidi[2], Olaf Zimmermann[3]

Abstract: Software architecture research needs to address the specific needs and constraints of specific deployment contexts. We propose an architectural style for cloud-deployed software referring to principles and patterns. Patterns map abstract principles to development and deployment platform solution templates Together, principles and patterns link common software architecture concepts, such as services, adaptivity or models at runtime, to deployment specifics such as virtualisation and controller-based feedback loops. The results of this broader framework shall be discussed in the context of recent trends such as microservices.

Keywords: Software Architecture, Architectural Style, Patterns, Cloud, Edge, Microservices, Container.

1 Architectural Principles for Cloud Software

Software architecture should support the specific needs and constraints of specific deployment contexts. An architectural style for cloud-deployed software can provide this support based on the following principles [PJZ18]: (1) service-orientation: services provided using layering, modularity, and loose coupling. (2) virtualization: services for shared resources and portable containers. (3) uncertainty: distribution, heterogeneity, and multi-user involvement cause uncertainty. (4) adaptivity: operation and management support allow for dynamic adaptation.

Patterns map principles to development and deployment platform solution templates: (1) microservices: flexible composition with independent, self-managed containers and cloud-native services. (2) models at runtime: allows aspects of uncertainty to be addressed dynamically. (3) controller-based feedback loop: allows controllers to adapt to and manage change. Together, principles and patterns link common software architecture concepts, such as services, adaptivity or models at runtime, to deployment specifics such as virtualisation and controller-based feedback loops.

The results of this broader framework shall be discussed in the context of open challenges such as self-adaptive microservices (see [Ja18, Pa17, Me19]), but also deployments not only in cloud, but also related edge and IoT settings [Le19].

[1] Software and Systems Engineering, Free University of Bozen-Bolzano, Bozen, Italy
[2] Computer Science and Engineering, University of South Carolina, Columbia, USA
[3] Institute for Software, Hochschule für Technik Rapperswil, Rapperswil, Switzerland

doi:10.18420/SE2020_34

2 Self-Adaptive Microservices

A self-adaptive system dynamically adapts its behaviour to preserve or enhance quality attributes in uncertain operating conditions. Here, the development of microservice applications as such self-adaptive systems is still a challenge. In practice, e.g., the Kubernetes container orchestration platform facilitates to deploy and manage microservice applications, natively only supports autoscaling (automatically change the number of instances of a service) and self-healing (automatically restart failed service instances. Microservice quality attributes can be improved by using planning techniques (determine the best adaptation strategy for each microservice), machine learning (learn new adaptation strategies from past adaptation results), reasoning under uncertainty (cope with noisy monitoring data), and multi-objective optimization (cater for multiple, possibly conflicting requirements). However, an important observation is that independent and frequent deployments, the need for a high degree of automation in a DevOps context, and complex run-time architectures make microservices ideal candidates to be deployed as self-adaptive systems.

3 Containerised Edge and IoT

Containers and microservices are lightweight approaches for architecting and deploying software that are particularly needed for an edge computing model, which aims to provide low-cost local clusters at the outer edge of the cloud, possibly composed of IoT devices themselves. We can demonstrate that fully containerised microservice architectures for data streaming platforms can be deployed even on small, e.g., single-board device clusters. They can serve IoT use cases for which the data volume is not too high. Benefits include flexible container deployment and (at least) sufficient performance and scalability despite device limitations. Challenges still exist in the dynamic load-driven distribution management of the containers in clusters and the adaptivity problem already discussed. Autoscaling is feasible, which we implemented for a serverless architecture using a fuzzy controller.

Literaturverzeichnis

[Ja18] Jamshidi, P.; Pahl, C.; Mendonca, N. C.; Lewis, J.; Tilkov, S.: Microservices: The Journey So Far and Challenges Ahead. IEEE Software, 35(3):24–35, 2018.

[Le19] von Leon, David; Miori, Lorenzo; Sanin, Julian; El Ioini, Nabil; Helmer, Sven; Pahl, Claus: A Lightweight Container Middleware for Edge Cloud Architectures. In: Fog and Edge Computing. John Wiley and Sons, Ltd, Kapitel 7, S. 145–170, 2019.

[Me19] Mendonca, N. C.; Jamshidi, P.; Garlan, D.; Pahl, C.: Developing Self-Adaptive Microservice Systems: Challenges and Directions. IEEE Software, 2019.

[Pa17] Pautasso, C.; Zimmermann, O.; Amundsen, M.; Lewis, J.; Josuttis, N.: Microservices in Practice, Part 1: Reality Check and Service Design. IEEE Software, 34(1):91–98, 2017.

[PJZ18] Pahl, Claus; Jamshidi, Pooyan; Zimmermann, Olaf: Architectural Principles for Cloud Software. ACM Trans. Internet Technol., 18(2):17:1–17:23, Februar 2018.