Microservices in the German Industry: Insights into Technologies, Characteristics, and Software Quality *

Jonas Fritzsch[12], Justus Bogner[21], Stefan Wagner[1], and Alfred Zimmermann[2]

[1] University of Stuttgart, Stuttgart, Germany
{jonas.fritzsch, stefan.wagner}@iste.uni-stuttgart.de
[2] University of Applied Sciences Reutlingen, Reutlingen, Germany
{justus.bogner, alfred.zimmermann}@reutlingen-university.de

Abstract

While Microservices initially were a topic mainly driven by practitioners, they currently are heavily researched in academia. In this paper, we contribute a qualitative study with the goal to provide insights into industry adoption and implementation of Microservices. During 17 in-depth interviews with software professionals from 10 companies we analyzed 14 service-based systems. The interviews focused on applied technologies, adherence to Microservice characteristics, and the perceived influence on software quality. We summarize the most important take-aways along with insights into common practices of companies across different domains and sizes. Researchers may build upon our findings and take them into account when designing industry-focused methods or techniques. This extended abstract includes some figures and excerpts from a forthcoming publication of the authors [1].

1 Introduction and Research Design

In todays' industry practice, the Microservices architectural style [2, 5] is an important foundation for enterprise applications with strong requirements for e.g. modifiability or scalability. Since most scientific publications focus on characteristics from a few early adopters, academia needs to be careful that there is no significant gap between the scientific notion of Microservices and their implementation in industry. Existing empirical studies on industry adoption [4, 7] reveal a high degree of diversity in service-oriented industry practice. Schermann et al. [7] even report that several academic assumptions may be incorrect for typical service-based systems, e.g. the assumed size of service ecosystems or the usage of dynamic service discovery. While the majority of empirical industry research is quantitative and survey-based, important insights into the rationale for these differences could be gained via qualitative and interview-based methods. Our research objective is therefore to provide insights into industry adoption and implementation of Microservices as well as into software professionals' reflections and rationales in this area. This objective is framed by the following three research questions:

1. Which technologies do companies use for the implementation and operation of Microservices and with what rationale?
2. Which characteristics of Microservices do companies respect, which do they neglect, and for which reasons?
3. How do companies perceive the influence of Microservice architectures on software quality?

*based on [1]: Bogner, J., Fritzsch, J., Wagner, S., Zimmermann, A.: Microservices in Industry: Insights into Technologies, Characteristics, and Software Quality. In: 2019 IEEE International Conference on Software Architecture Companion (ICSA-C). IEEE Computer Society, Hamburg, Germany (2019).

To pursue the goal of providing additional empirical and industry-focused research in the area of Microservices, we conducted 17 in-depth interviews with software professionals based in Germany from 10 different companies. As a top-level guidance, we followed the five-step case study research process as described by Runeson and Höst [6]. Since qualitative results are very rich and informative, they enable us to focus on participants' rationales. As a concrete method, we chose semi-structured interviews [3, 8], because they provide a basic agenda, but also allow for dynamic adaptation based on the responses.

All interview artifacts and results are available in our online repository.[1]. It entails an *interview preamble* [6] outlining the interview process and topics, as well as an *interview guide* [8] containing the most important questions. During the interviews of ∼45 to ∼75 minutes, we loosely followed this interview guide based on the participant's reactions. Lastly, the repository contains a *case characterization matrix* [8] with most relevant participant attributes and study concepts. This matrix served as a basis for our subsequent detailed analysis of each individual case transcript and was finally used during *cross-case analysis* [8] to identify important generalizations and summaries.

Our interviewees were acting in a technical role (e.g. developer or architect) with significant professional experience (mean of 14.7 years) and had solid knowledge of service orientation. Recent participation in the development of a system based on service orientation (ideally Microservices) was also required. All our participants were based in Germany, even though some of the larger companies were active in several European countries or even globally. Table 1 provides more details about the demographics. Half of the companies were software & IT service providers that develop systems for external clients. For the companies from other domains, system ownership was always internal. All individual case descriptions can be found in our online repository mentioned above.

Table 1: Company and Participant Demographics

Company	Company Domain	Employees	Participant	Participant Role	Experience	System
C1	Financial Services	1 - 25	P1	Developer	6	S1
C2	Software & IT Services	>100,000	P2	Lead Architect	30	S2
			P3	Architect	24	S3
			P4	Architect	30	S4
C3	Software & IT Services	26 - 100	P5	Architect	20	S5
			P6	Lead Developer	8	
C4	Software & IT Services	101 - 1,000	P7	Architect	9	S6
			P8	Architect	17	S7
C5	Software & IT Services	>100,000	P9	Lead Developer	7	S8
C6	Tourism & Travel	1,001 - 5,000	P10	Developer	9	S9
			P11	Data Engineer	7	
			P12	Architect	12	S10
C7	Logistics & Public Transport	101 - 1,000	P13	Architect	17	S11a
			P14	DevOps Engineer	5	S11
C8	Retail	5,001 - 10,000	P15	Lead Architect	9	S12
C9	Software & IT Services	101 - 1,000	P16	Architect	18	S13
C10	Retail	1,001 - 5,000	P17	Architect	22	S14

[1] https://github.com/xJREB/research-microservices-interviews

Table 2: System Characteristics

ID	Purpose	# of People	# of Services	Communication	Languages	Artifacts
S1	Derivatives management system (banking)	7	9	REST, AMQP	Java, Kotlin	JAR
S2	Freeway toll management system	10	10	Oracle Advanced Queuing, REST	Java, C++ for algorithm	Docker
S3	Automotive problem management system	50	10	REST, Kafka	Java	Docker
S4	Public transport sales system	∼300	∼100	REST, Kafka, AmazonMQ	Java, Node.js	Docker
S5	Business analytics & data integration system	7	6	REST, Kafka	Java, Scala	Docker
S6	Automotive configuration management system	20	60	REST, Kafka, MQTT, JMS	Java, Node.js	Docker
S7	Retail online shop	∼200	∼250	REST (JSON or OData), SOAP (legacy)	Java, Node.js, Go, Kotlin	JAR/WAR
S8	IT service monitoring platform	15	9	REST, AMQP	Java, Node.js	Docker
S9	Hotel search engine	∼50	∼10	gRPC, Kafka, REST, Unix domain socket	Java, PHP	Docker
S10	Hotel management suite	50	20	REST, Google Pub/Sub	PHP, Java	Docker
S11	Public transport management suite (HR management part: S11a)	∼175	10 products	REST, SOAP, file transfer, RPC/RMI	Java, C++	JAR/EAR
S12	Retail online shop	∼85	∼45	REST, Kafka	Java, Kotlin, Scala, Go	Docker
S13	Automotive end-user service mgmt. system	30	7	REST, AMQP	Java	Docker
S14	Retail online shop	∼350	∼175	REST, Kafka, Amazon SQS	Java, Clojure, Node.js, Ruby, Scala, Swift	Docker, JAR/WAR, ZIP

2 Results and Implications

In the context of the 14 systems (see Table 2), we found that companies generally relied on well-established technologies for service implementation, communication, and deployment. The analysis revealed that RESTful HTTP, Docker containers, and Java were very popular with our participants. As reasons, interviewees respectively named interoperability, portability, and the availability of developers with Java knowledge. Most systems, however, did not exhibit a high degree of technological diversity as commonly expected with Microservices. Some participants even saw a high degree of heterogeneity as harmful and tried to harmonize technology usage.

Furthermore, several Microservices characteristics (e.g. decentralization or DevOps principles) were not or only partially followed, especially when the system was created for an external customer. Only very few companies strictly followed the "you build it, you run it" principle. Participants also expressed valid arguments why they generally build fewer and more coarse-grained services, which seems to blur the line between service- and Microservice-based systems even further. The number of investigated systems was too small to point out domain-specific aspects. Solely, the two systems operated by the retailers stood out from the rest with respect to following Microservices characteristics postulated in literature.

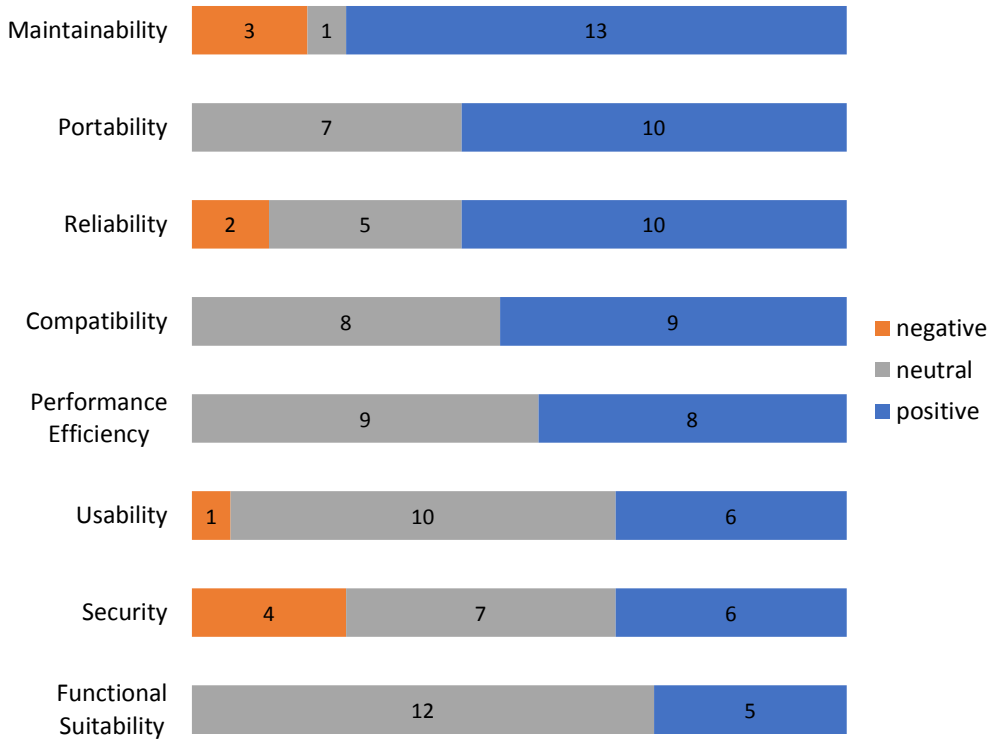

Figure 1: Perceived Impact of Microservices on Software Quality (ISO 25010)

The impact of Microservices on software quality (see Fig. 1) was rated positively by the majority of our interviewees. While maintainability received the most positive mentions, security was perceived as controversial by participants. One group appreciated the possibility to apply individual security settings to services depending on their criticality, while the other group criticised the increased attack surface of the whole system.

Researchers creating industry-focused methods should take these insights into account. Since this is a qualitative study, we must be careful to generalize distributions from the 14 cases. Instead, the rationales and relations between concepts were the focus of this study. Future research could perform deeper analysis of some of the discovered rationales and trends to make them more actionable for the creation of methods.

References

[1] Justus Bogner, Jonas Fritzsch, Stefan Wagner, and Alfred Zimmermann. Microservices in Industry: Insights into Technologies, Characteristics, and Software Quality. In *2019 IEEE International Conference on Software Architecture Companion (ICSA-C)*, Hamburg, Germany, 2019. IEEE Computer Society.

[2] Martin Fowler. Microservices Resource Guide, 2015.

[3] S.E. Hove and Bente Anda. Experiences from Conducting Semi-structured Interviews in Empirical Software Engineering Research. In *11th IEEE International Software Metrics Symposium (METRICS'05)*, number Metrics, pages 23–23. IEEE, 2005.

[4] Holger Knoche and Wilhelm Hasselbring. Drivers and Barriers for Microservice Adoption – A Survey among Professionals in Germany. *Enterprise Modelling and Information Systems Architectures (EMISAJ)*, 14(1):1–35, 2019.

[5] Sam Newman. *Building Microservices: Designing Fine-Grained Systems*. O'Reilly Media, 1st edition, 2015.

[6] Per Runeson and Martin Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131–164, apr 2009.

[7] Gerald Schermann, Jürgen Cito, and Philipp Leitner. All the Services Large and Micro: Revisiting Industrial Practice in Services Computing. In *Lecture Notes in Computer Science*, volume 9586 of *Lecture Notes in Computer Science*, pages 36–47. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[8] Carolyn B. Seaman. Qualitative Methods. In *Guide to Advanced Empirical Software Engineering*, pages 35–62. Springer London, London, 2008.