Scalability and resilience in practice: current trends and opportunities

Julien Ponge
Red Hat
Lyon, France
jponge@redhat.com

Mark Little
Red Hat
Newcastle upon Tyne, United Kingdom
mlittle@redhat.com

***Abstract*—In this presentation we report on the current trends for implementing distributed services that are both scalable and resilient based on our experiences across the Red Hat Middleware portfolio. We tackle the topics of reactive event-driven services and distributed consensus at cloud scale.**

***Keywords*-distributed computing; middleware**

I. Introduction

Modern distributed applications are composed of multiple, loosely-coupled services that interact over the network. ”Cloud native”, ”function-as-a-service”, or ”microservices” are all terms that relate to this type of approach were an application is not formed of one deliverable package (also called a ”monolith”), but rather results of the composition and interactions between independent networked services.

Each service typically focuses on exactly one functional concern such as doing identity management, analytics, exposing an API for third-party services, processing images, or feeding some machine-learning system with data from various sources. Services may not all belong to the same entity, so applications are typically a mix of in-house and third-party services scattered across on-premise, private and public clouds. There are operational benefits in having multiple independent services reused by independent applications, including the ability to scale them in a fine-grained fashion to adapt to fluctuating workloads, faster turnarounds for upgrades, and the agility to experiment with new functionalities by introducing new services.

The flip side of the coin of having applications made of networked services over the network is that modern distributed systems need to embrace the network and all the problems that come with it [1]. This includes latency in service communications, a lower bandwidth compared to components assembled within the same application process and lost messages. It is not just possible to ignore distributed systems and their challenges anymore. Middleware have a long history of providing abstractions to tame the challenges of distributed computing [2], and the rise of new ”serverless” application architecture is no short of providing renewed challenges for middleware research and practice.

In this presentation we report on the current trends, challenges and research opportunities for implementing modern distributed systems, based on our experiences across the Red Hat Middleware portfolio. We will focus on reactive event-driven services (for workload processing) and distributed consensus at cloud scale (for consistent distributed state). Reactive systems are both scalable as workloads grow, and resilient as failures arise, keeping latency under control. This requires taming the challenges of asynchronous programming [3], [4] and dealing with errors as being the norm rather than the exception [5]. Distributed consensus is more relevant than ever as scalability drives the development of new research and techniques to cope with current architecture styles and workloads, both to provide practical algorithms [6] and improve throughput [7].

References

[1] P. Bailis and K. Kingsbury, “The Network is Reliable,” *Queue*, vol. 12, no. 7, pp. 20:20–20:32, Jul. 2014. [Online]. Available: https://queue.acm.org/detail.cfm?id=2655736

[2] W. Emmerich, M. Aoyama, and J. Sventek, “The Impact of Research on the Development of Middleware Technology,” *ACM Transactions on Software Engineering and Methodology*, vol. 17, no. 4, pp. 19:1–19:48, Aug. 2008.

[3] B. Liskov and L. Shrira, “Promises: Linguistic Support for Efficient Asynchronous Procedure Calls in Distributed Systems,” in *Proceedings of the ACM SIGPLAN 1988 Conference on Programming Language Design and Implementation*, ser. PLDI ’88. New York, NY, USA: ACM, 1988, pp. 260–267.

[4] E. Meijer, “Your Mouse is a Database,” *Queue*, vol. 10, no. 3, pp. 20:20–20:33, Mar. 2012. [Online]. Available: http://queue.acm.org/detail.cfm?id=2169076

[5] J. Armstrong, “Making reliable distributed systems in the presence of software errors,” Ph.D. dissertation, The Royal Institute of Technology, Stockholm, Sweden, December 2003.

[6] D. Ongaro and J. Ousterhout, “In Search of an Understandable Consensus Algorithm,” in *Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference*, ser. USENIX ATC’14. Berkeley, CA, USA: USENIX Association, 2014, pp. 305–320.

[7] H. Howard, D. Malkhi, and A. Spiegelman, “Flexible Paxos: Quorum Intersection Revisited,” in *20th International Conference on Principles of Distributed Systems (OPODIS 2016)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 70. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017, pp. 25:1–25:14.