Chapter Conclusion

We'll conclude this chapter with a quick summary.

WE'LL COVER THE FOLLOWING

Summary

The Netflix stack provides a variety of projects to build microservice architectures. The stack solves the typical challenges of synchronous microservices as follows:

Service discovery

Service discovery is offered by Eureka.

Eureka focuses on Java with the Java client, but also offers a REST API and libraries for other languages and can therefore be used with other languages.

Resilience

For **resilience**, Hystrix is the de facto standard for Java and covers this area very well.

- Non-Java applications can use Hystrix via a sidecar at most.
- There are ports of Hystrix for other languages like Go.

- Hystrix is independent of the other technologies and can be used on its own.

Load balancing

Ribbon implements client-side **load balancing**, which has many advantages.

However, since Ribbon is a Java library, other technologies are difficult to use with Ribbon, especially in the area of load balancing. There are numerous load balancers that provide alternatives.

Ribbon relies on **Eureka** for service discovery but **can also use Consul**.

Routing

Routing is solved by Zuul. Zuul's dynamic filters are very flexible, but many developers are familiar with reverse proxies based on web servers like Apache httpd or nginx.

In this case, a reverse proxy might be the safer option. Additional features like SSL termination or request throttling may also be required, which Zuul does not offer.

Chapter 11 shows how Apache httpd can be used with Consul to provide routing. Zuul requires Eureka to find the microservices.

The servers in the Netflix stack are written in Java so that the servers from the Netflix stack and the microservices can be packed into JAR files.

Thanks to Spring Cloud, they are uniformly configurable.

The handling of metrics and logs is identical and this uniformity can be an advantage for operation.

Advantages

- Eureka and client-side caching is very fast and resilient.
- Zuul is very flexible because of its filters.
- Client-side load balancing avoids single points of failure or bottlenecks.

- Hystrix is very mature and the de facto standard for Java.

Challenges

- The Netflix stack implements many solved problems (e.g. reverse proxy, service discovery) again.
- The technologies focus on Java and therefore limit technology freedom.
- The code depends on the Netflix stack, Ribbon, Hystrix, and also Eureka due to `@EnableDiscoveryClient` and the client API.

In the next chapter, we'll discuss a new recipe: REST with Consul and Apache httpd.