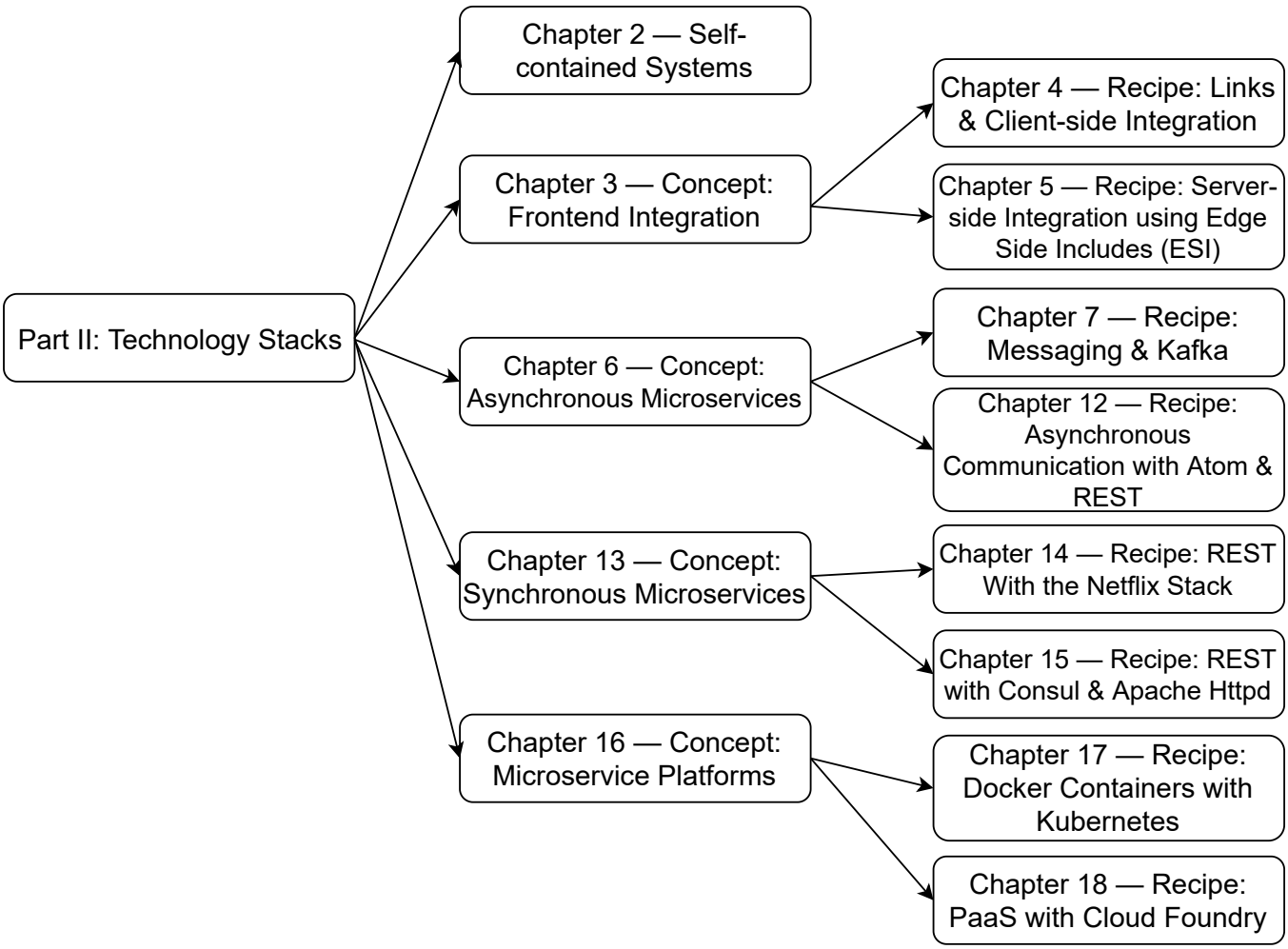
Structure of the Course

In this lesson, we'll get a quick overview of the structure of this course.

WE'LL COVER THE FOLLOWING

- Appendix
- Target Group
- Prior Knowledge
- Quick start
- Acknowledgements

This course is **part II** of a series of courses and focuses on *technology stacks*.



Overview of this course

- Chapter 2 explains **self-contained systems (SCS)** as an especially useful approach for microservices. It focuses on microservices that include a UI as well as logic.
- One possibility for integration for SCS in particular is **integrating at the web frontend** (chapter 3). Frontend integration results in a loose coupling between the microservices and a high degree of flexibility.
- The recipe for web frontend integration presented in chapter 4 capitalizes on **links and JavaScript** for dynamic content loading. This approach is easy to implement and utilizes well-established web technologies.
- On the server, integration can be achieved with **ESI (Edge Side Includes)** (chapter 5). ESI is implemented in web caches so that the system can attain high performance and reliability.
- The concept of **asynchronous communication** is the focus of chapter 6. Asynchronous communication improves reliability and decouples the systems.
- **Apache Kafka** is an example of an asynchronous technology (chapter 7) for sending messages. Kafka can save messages permanently and thereby enables a different approach to asynchronous processing.
- An alternative for asynchronous communication is **Atom** (chapter 8). Atom uses a REST infrastructure and thus is very easy to implement and operate.
- Chapter 9 illustrates how to implement **synchronous microservices**. Synchronous communication between microservices is often used in practice although this approach can pose challenges regarding response times and reliability.
- The **Netflix Stack** (chapter 10) offers Eureka for service discovery, Ribbon for load balancing, Hystrix for resilience, and Zuul for routing. The Netflix Stack is widely used in the Java community.
- **Consul** (chapter 11) is an alternative option for service discovery. Consul

contains numerous features and can be used with a broad spectrum of technologies.
- Chapter 12 explains the concept of **microservices platforms**, which support operations and communications of microservices.

- The **Kubernetes** (chapter 13) infrastructure can be used as a microservices platform and is able to execute Docker containers, as well as having solutions for service discovery and load balancing. The microservice remains independent of this infrastructure.

- **PaaS (Platform as a Service)** is another infrastructure that can be used as a microservices platform (chapter 14). Cloud Foundry is used as an example. Cloud Foundry is very flexible and can be run in your own computing center as well as in the public cloud.

Appendix

The appendix contains a local set up of coding environments.

Target Group

This course explains basic principles and technical aspects of microservices. Thus, it is interesting to different audiences for a variety of reasons.

For **developers**, this course offers a **guideline for selecting a suitable technology stack**. The example projects serve as a basis for learning the foundations of the technologies.

The microservices contained in the example projects are written in Java using the Spring Framework. However, the technologies used in the examples serve to integrate microservices. So additional microservices can be written in different languages.

This course also introduces technologies for deployment such as **Docker, Kubernetes, or Cloud Foundry** that also solve some operational challenges.

Managers are presented with technical details that will benefit them in their careers.

Prior Knowledge

The course assumes the reader to have basic **knowledge of software architecture and software development**. All practical examples are documented in such a way that they can be executed with little prior knowledge.

This course focuses on technologies that can be employed for microservices using different programming languages. However, the examples are written in Java using the Spring Boot and Spring Cloud frameworks so that **changes to the code require knowledge of Java**.

Quick start

The examples are explained in the following sections:

Concept	Recipe	Lesson
Frontend Integration	Links and Client-side Integration	Example
Frontend Integration	Edge Side Includes (ESI)	Example
Asynchronous Microservices	Kafka	Example
Asynchronous Microservices	REST and Atom	Example
Synchronous Microservices	Netflix Stack	Example
Synchronous Microservices	Consul and Apache httpd	Example

All projects are available on GitHub. The projects always contain a `HOW-TO-`

All projects are available on GitHub. The projects always contain a `HOW-TO-RUN.md` file showing step-by-step instructions on how the demos can be installed and started.

The examples are independent of each other, so you can start with any one of them.

Acknowledgements

I would like to thank everybody who discussed microservices with me, who asked me about them, or worked with me. Unfortunately, these people are far too numerous to name them all individually. The exchange of ideas is enormously helpful and also fun!

Many of the ideas and their implementation would not have been possible without my colleagues at INNOQ. I would especially like to thank Alexander Heusingfeld, Christian Stettler, Christine Koppelt, Daniel Westheide, Gerald Preissler, Hanna Prinz, Jörg Müller, Lucas Dohmen, Marc Giersch, Michael Simons, Michael Vitz, Philipp Neugebauer, Simon Kölsch, Sophie Kuna, Stefan Lauer, and Tammo van Lessen.

Also, Merten Driemeyer and Olcay Tümce who provided important feedback.

Finally, I would like to thank my friends and family, whom I often neglected while writing this course – especially my wife. She also did the translation into English.

Of course, my thanks also go out to the people who developed the technologies, which I introduce in this book and thereby created the foundation for microservices.

I also would like to thank the developers of the tools of https://www.softcover.io/ and Leanpub.

Let's get started!