

# Requirements

In this lesson, we'll discuss the requirements a technology for implementing microservices has to fulfill.

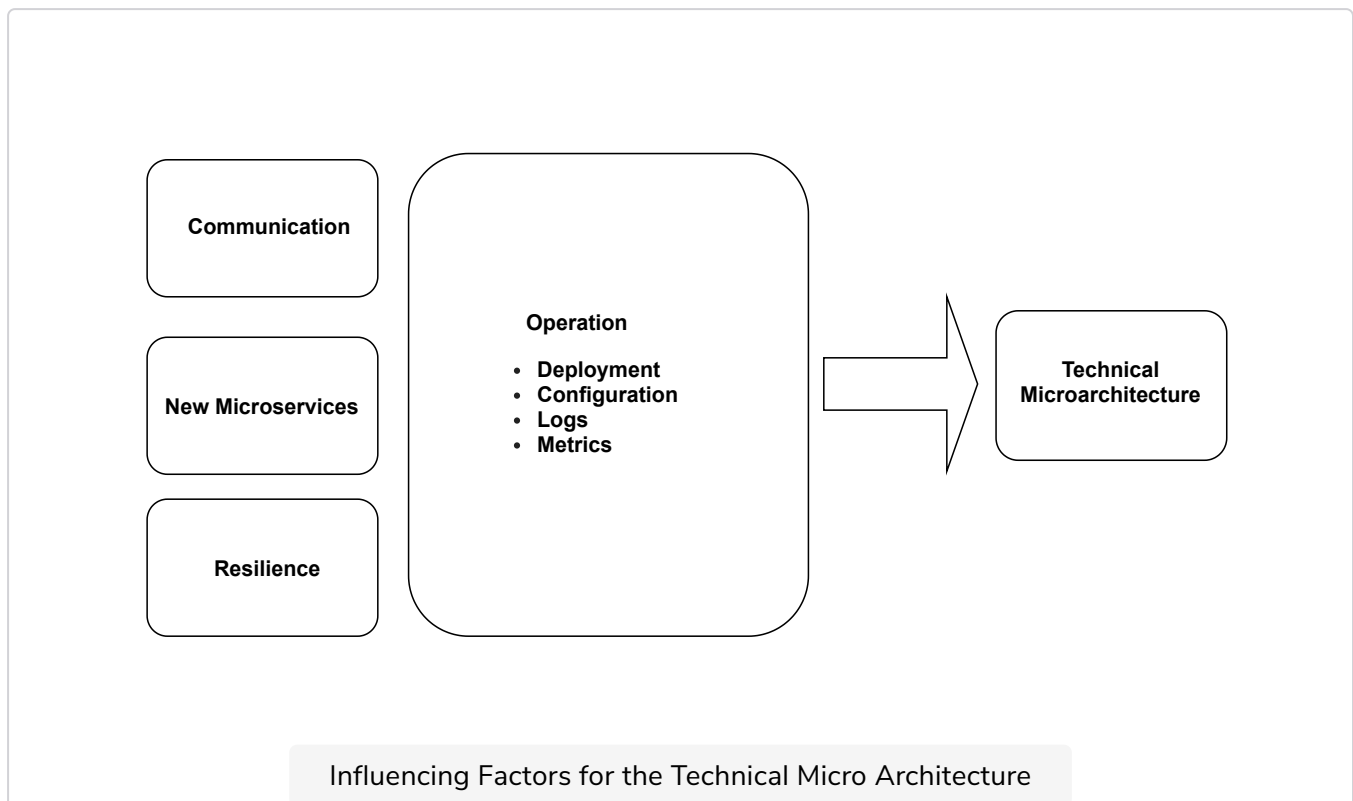
## WE'LL COVER THE FOLLOWING



- Communication
- Operation
  - Configuration
  - Deployment
  - Logs
  - Metrics
- New microservices
  - Option 1: Microservices increase in size
  - Option 2: Constant size microservices increase in number
- Resilience

A technology for implementing microservices has to fulfill different requirements. The figure below gives us a birds-eye view of what these are.

We'll be discussing each of these in detail below.



## Communication #

Microservices have to **communicate** with *other microservices*. This requires **UI integration** in the **web UI** or **protocols** such as **REST** or **messaging**.

It is a *macro architecture decision* which communication protocol is used (see [Architecture Decisions](#)).

However, the microservices have to support the chosen communication mechanism. Therefore,

The macro architecture decision influences the micro architecture.

The technology choices at the micro architecture level have to ensure that the communication protocol defined by the macro architecture can really be implemented in each microservice.

In principle, every modern programming technology can support the typical communication protocols. Therefore, this requirement does not represent a real restriction.

# Operation #

Operating the microservices should be as easy as possible.

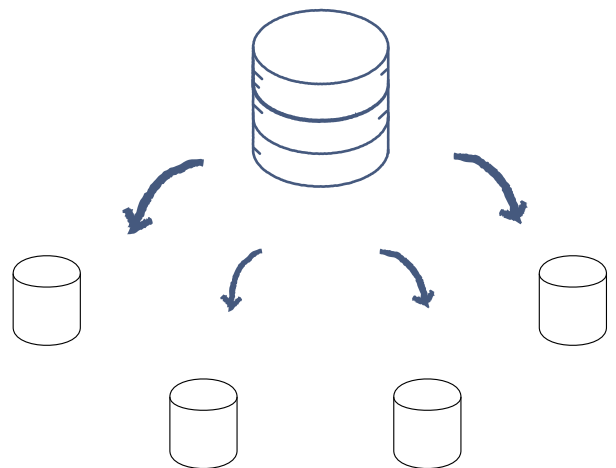
Topics in this area are:

- Deployment
- Configuration
- Logs
- Metrics

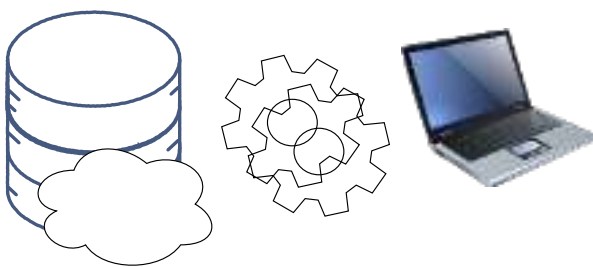
Let's cover these one by one:

## Configuration #

- The microservice has to be adapted to different scenarios. It is possible to use custom code for reading the configuration. However, an existing library can facilitate this task and promote a uniform application configuration.



## Deployment #



- The microservice has to be installed in an environment and has to run in this environment.

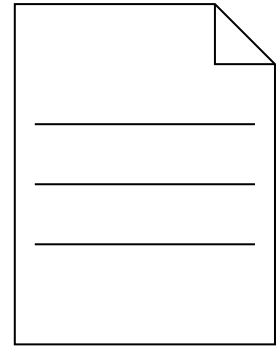
## Logs #

Writing log files is easy. However, the format should

be uniform for all microservices.

In addition, a simple log file is not enough when a server has to collect the logs from all microservices and provide them for analysis.

Therefore, technologies have to be in place for formatting the log outputs and for sending them to the server where all logs are stored and analyzed.



## Metrics #

Metrics have to be delivered to the central monitoring infrastructure.

This requires appropriate frameworks and libraries. In principle, different libraries can be used for implementing a macro architecture rule for which instance predefines a log format and a log server.

In this case, the micro architecture has to choose a library for the microservice. Macro architecture rules can also determine the library.

However, this limits the technological freedom of the microservices to those programming languages which can use the chosen library.

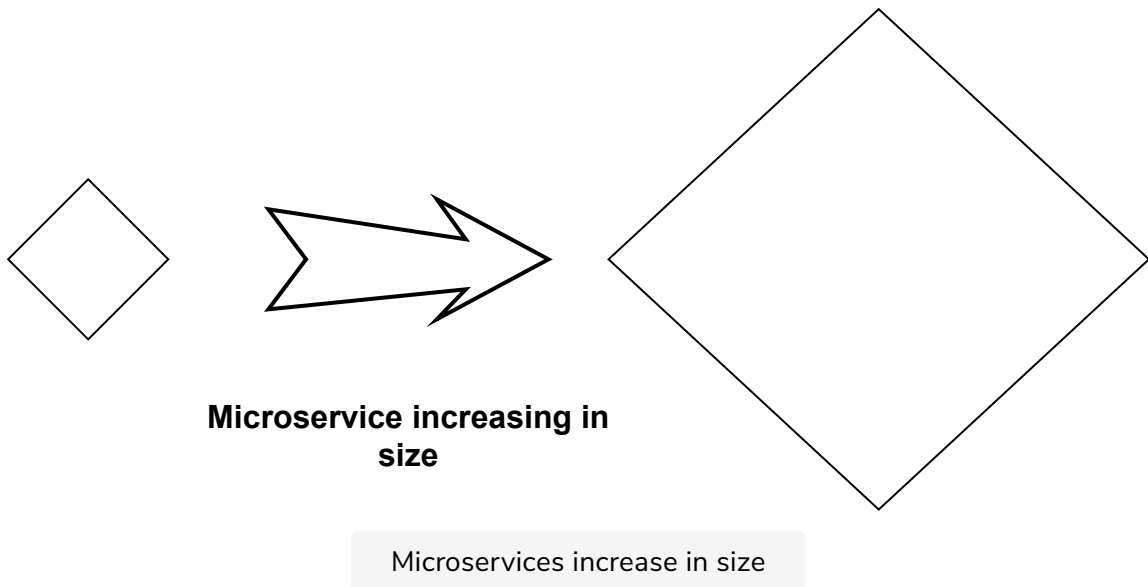
## New microservices #

It should be easy to create new microservices. When a project over time accumulates more and more code, there are two options:

1. The microservices become larger.
2. The number of microservices of constant size increases.

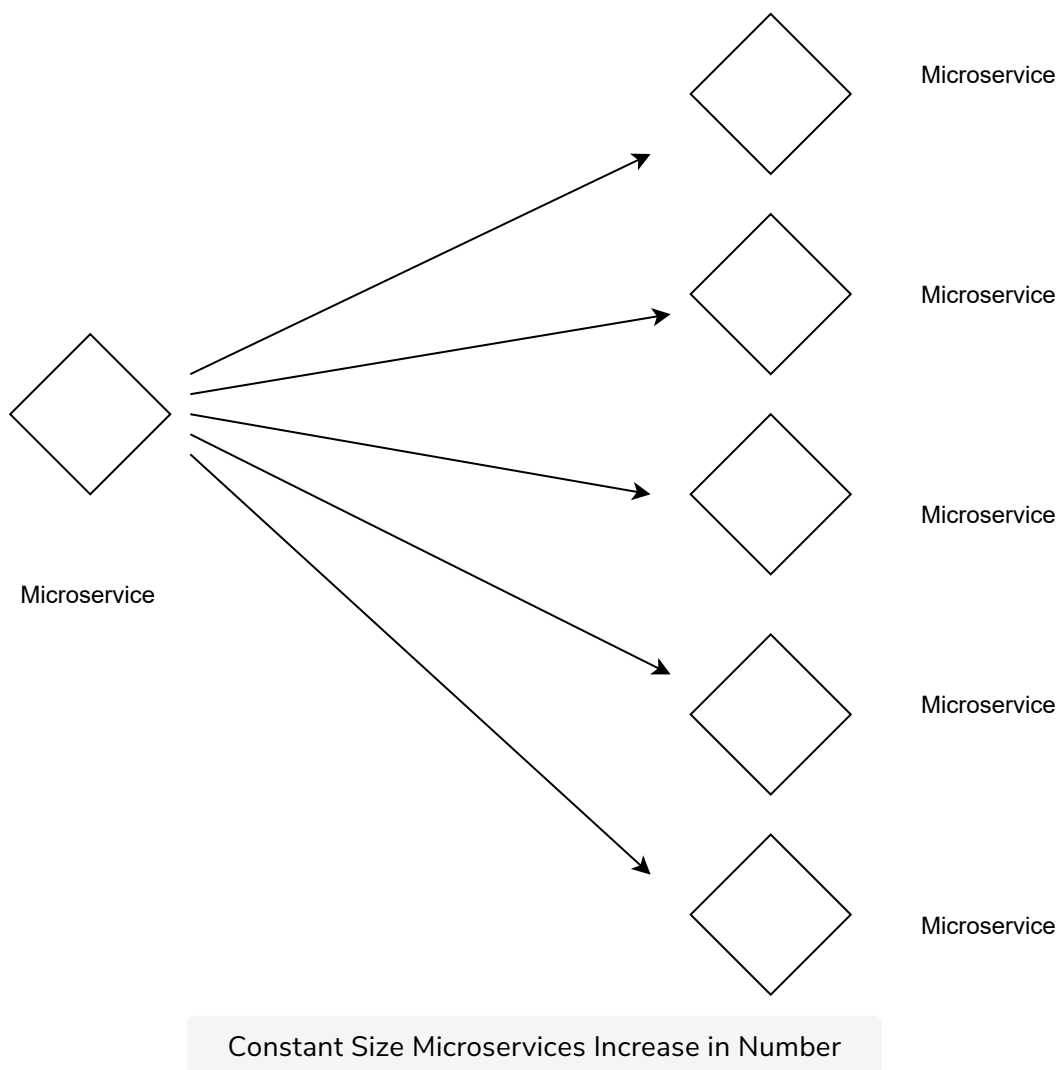
### Option 1: Microservices increase in size #

If the microservices increase in size, at some point they will not deserve the name microservice anymore.



## Option 2: Constant size microservices increase in number #

To avoid the increase in the size of microservices, it is easy to generate new microservices to keep the size of the individual microservices constant over time.



# Resilience

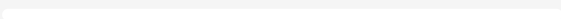
Each microservice has to be able to deal with the failure of other microservices. This has to be ensured when microservices are implemented.

## QUIZ

1

What are the factors that influence the technical Micro Architecture?

COMPLETED 0%



1 of 5



---

In the *next lesson*, we'll discuss Reactive Programming.