# Towards Simulation-Driven Optimization of Container Orchestration Mechanisms

## 14th Symposium on Software Performance (2023)
## Session 4: Quality

Timo Dittus, Martin Sträßer, Samuel Kounev – Descartes Research Group

08.11.2023

*https://se.informatik.uni-wuerzburg.de*

# Introduction

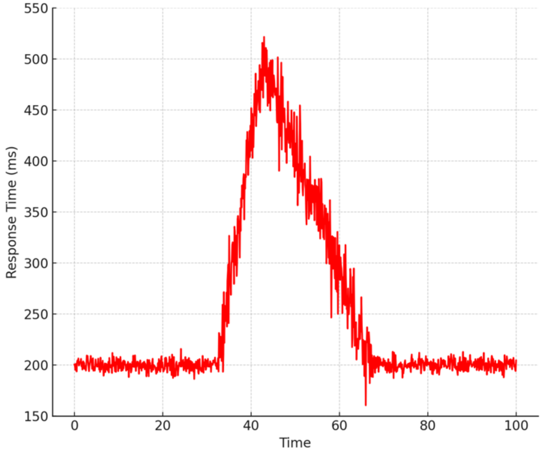
**Typical container orchestration tasks (definition by Google):**

- Provisioning and deployment
- Scaling containers up or down and load balancing
- Allocating resources between containers
- Moving containers to another host to ensure availability if there’s a shortage of resources or an unexpected outage
- Performance and health monitoring of the application
- Service discovery

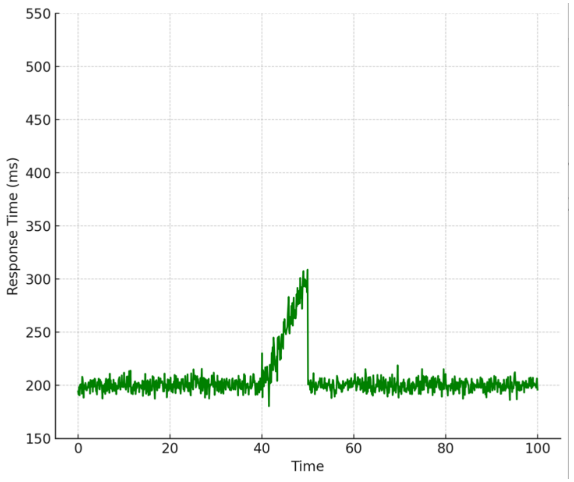
# Motivation

## Container orchestration (CO) frameworks significantly impact performance

```yaml
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: order-service-hpa
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: order-service
  minReplicas: 1
  maxReplicas: 5
  metrics:
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 70
```

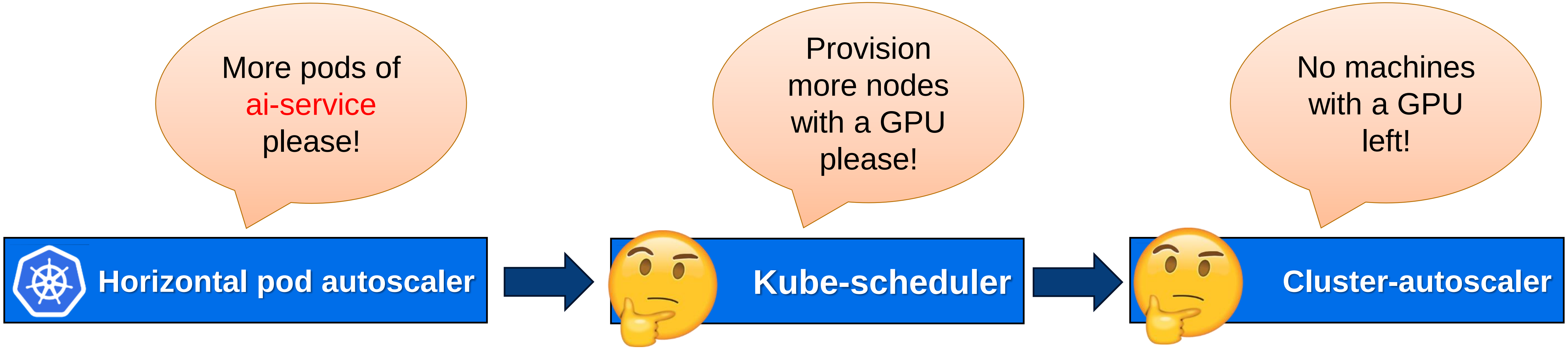



```yaml
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: order-service-hpa
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: order-service
  minReplicas: 2
  maxReplicas: 10
  metrics:
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 30
```



# Motivation

- Testing configurations for CO mechanisms is costly and time-consuming, because:
- CO mechanisms have a (very) large variety of configuration parameters
- … where one mechanism’s configuration can exert influence on another’s




**➡ Automatically optimizing configurations via a simulation strongly preferrable!**

# Recap: MiSim

## MiSim: A Simulator for Resilience Assessment of Microservice-based Architectures

Sebastian Frank[1,2], Lion Wagner[2], Alireza Hakamian[2], Martin Straesser[3], André van Hoorn[1]
[1]University of Hamburg, Hamburg, Germany
[2]University of Stuttgart, Stuttgart, Germany
[3]University of Würzburg, Würzburg, Germany
{sebastian.frank, andre.van.hoorn}@uni-hamburg.de, mir-alireza.hakamian@iste.uni-stuttgart.de,
martin.straesser@uni-wuerzburg.de

- State-of-the-Art microservice simulator
- Based on discrete event simulation, uses CPU performance model
- Focus on resilience mechanisms

# Recap: MiSim-Orchestration

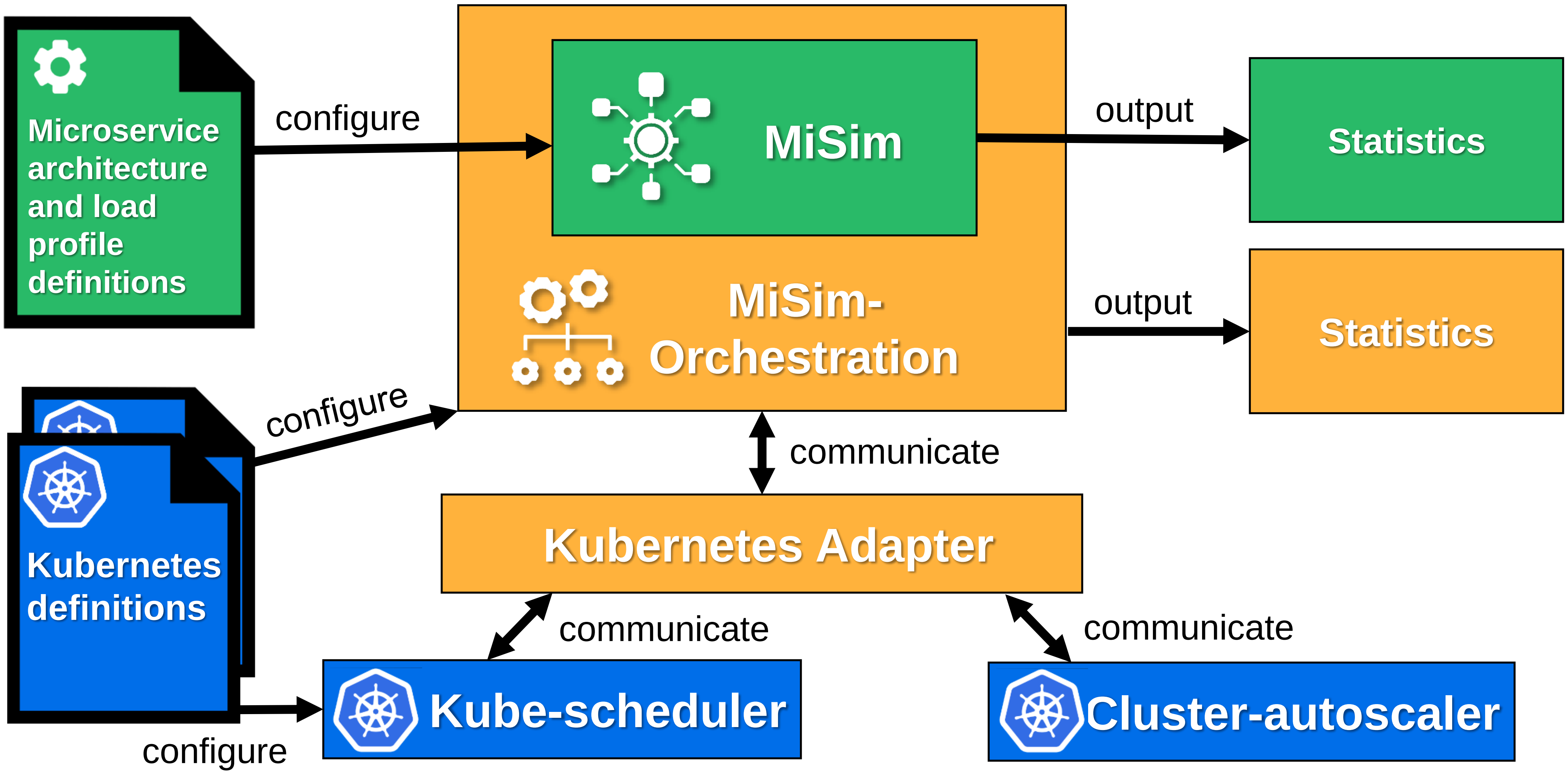
- Extend MiSim by simulating a Kubernetes environment
- Let MiSim communicate with real Kubernetes components, pretending to be in an actual cluster
- … to enable realistic behavior

**Kubernetes-in-the-Loop: Enriching Microservice Simulation Through Authentic Container Orchestration**

Martin Straesser[1], Patrick Haas[1], Sebastian Frank[2], Alireza Hakamian[3], André van Hoorn[2], and Samuel Kounev[1]

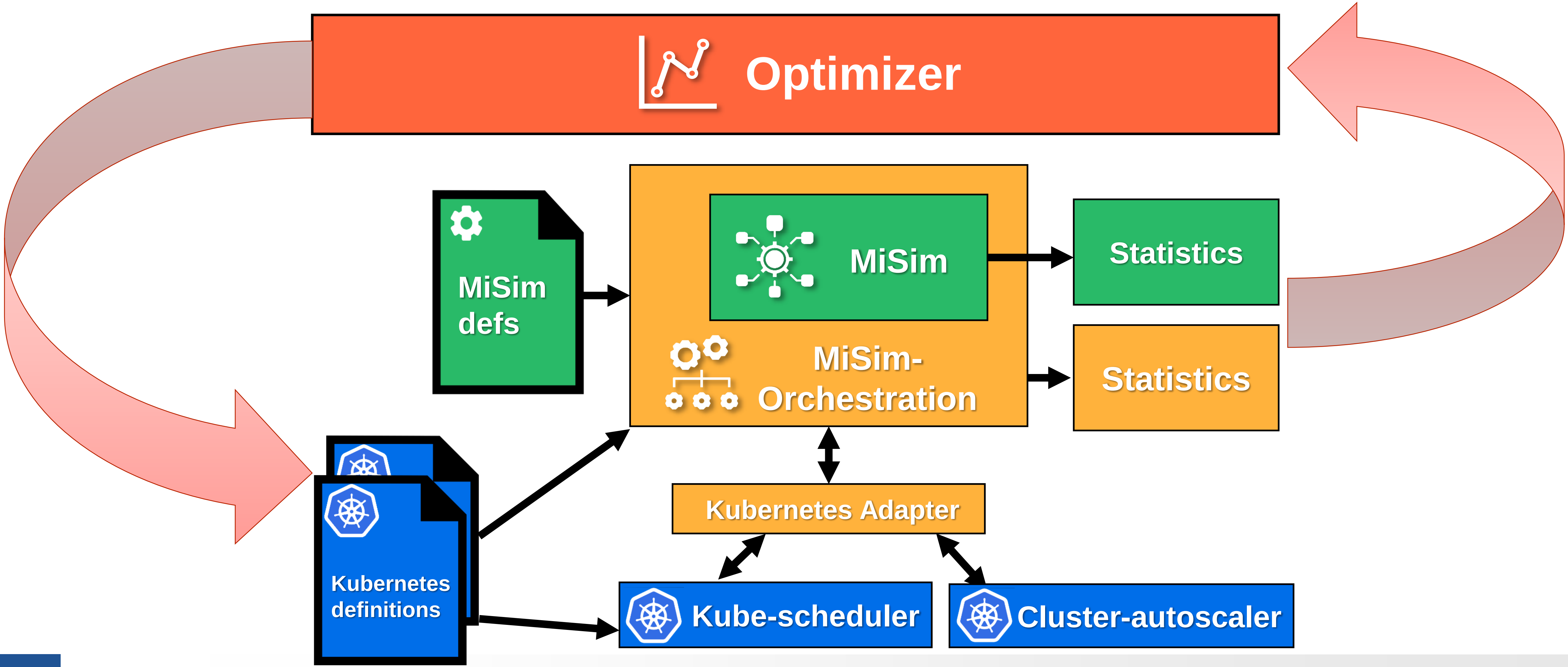
[1] University of Würzburg, Würzburg, Germany
{martin.straesser, samuel.kounev}@uni-wuerzburg.de,
patrick.haas@informatik.uni-wuerzburg.de
[2] University of Hamburg, Hamburg, Germany
{sebastian.frank, andre.van.hoorn}@uni-hamburg.de
[3] University of Stuttgart, Stuttgart, Germany
mir-alireza.hakamian@iste.uni-stuttgart.de

# Recap: MiSim and MiSim-Orchestration

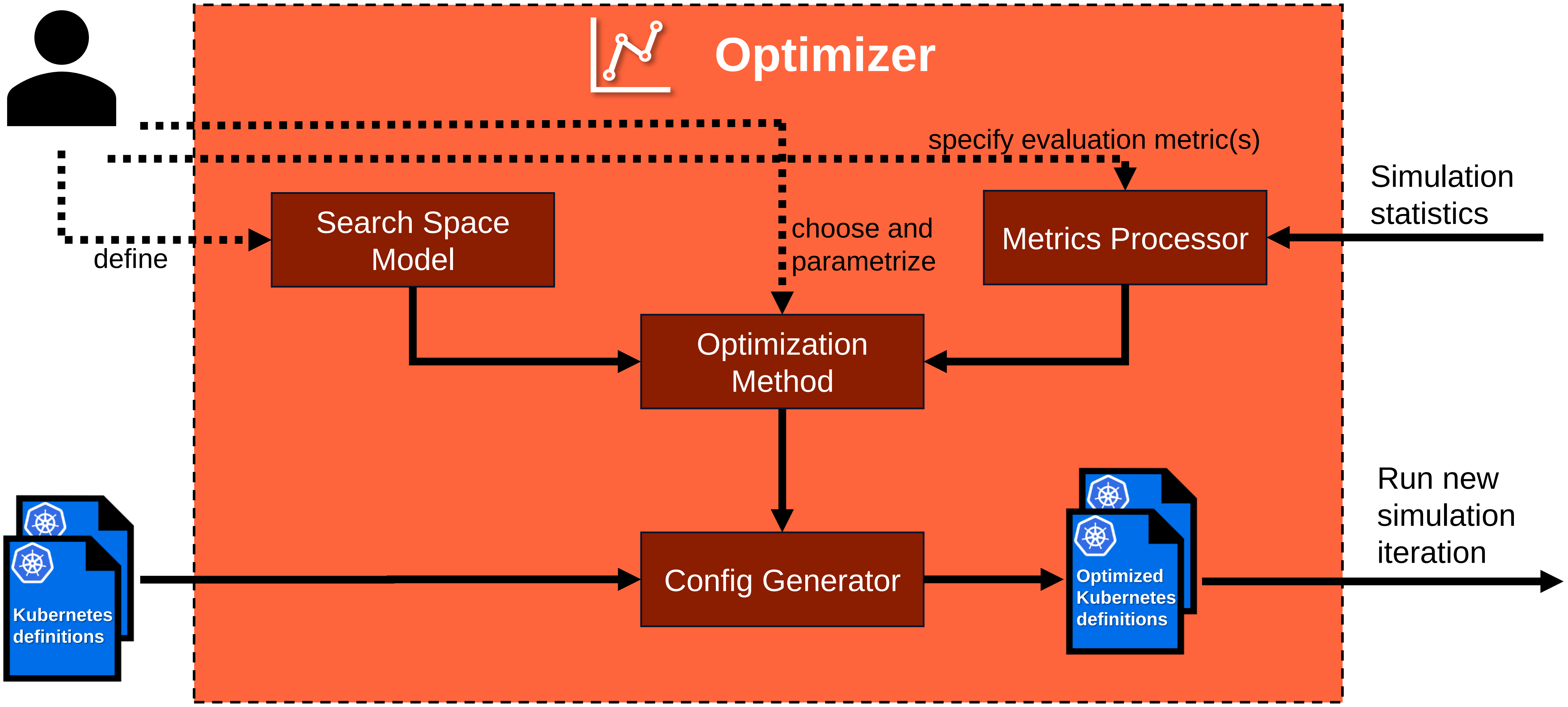


# Simulation-driven Optimization

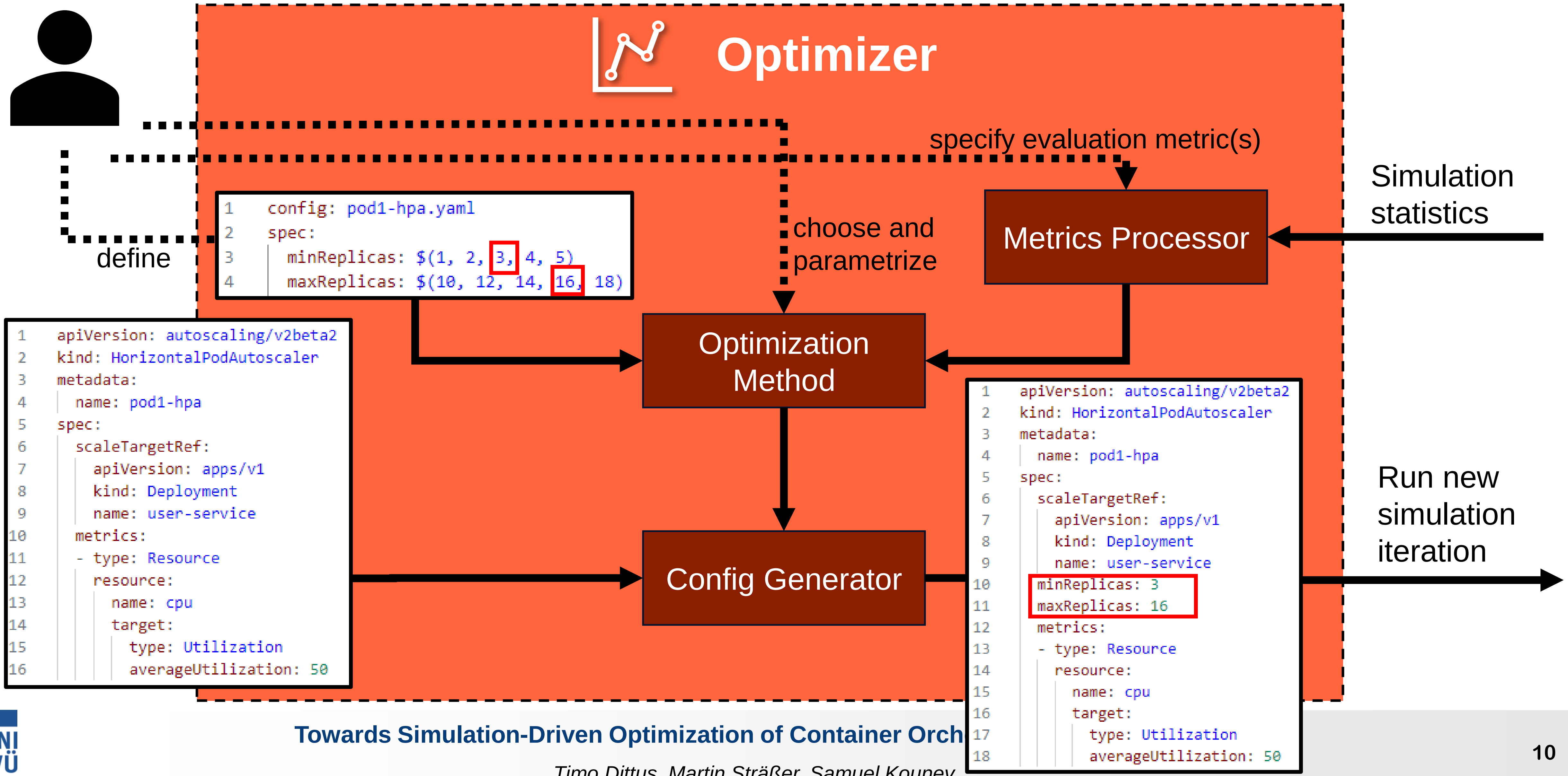
**Key concept: Run simulation iteratively in an optimization loop**



# Simulation-driven Optimization



# Simulation-driven Optimization



# Characterizing the Optimization Problem

- **Objectives can be opposing to each other** - e.g., costs vs. service quality
- **Derivative-free optimization** a.k.a. black-box optimization
- **Unfeasible to try all configurations**
  - Suppose we have **15 configuration parameters**, each restricted to **5 possible values** and an **average simulation runtime of 10s** (very optimistic!)
  - **Can test 0.0102% of all possible combinations** in a full year of running 24/7
  - Improvements in simulation time possible, however, unlikely to improve by many magnitudes
- **Therefore, a highly efficient and systematic optimization method is required**

# Potential Optimization Methods

| | Probablistic Model-based | Evolutionary Algorithms | Bandit-based Methods | Combined Techniques |
|---|---|---|---|---|
| **Description** | Predict good solutions with learned probabilistic model | Mimic mechanisms from natural selection to optimize solutions iteratively | Dynamically allocate resources, apply early stops to unpromising trials | Combine strengths of two or more techniques |
| **Strength / Weakness** | Can be bad for very high-dimensional search space | Great for very high-dimensional search space | Highly efficient, but may lead to inaccurate optimization | May also transfer weaknesses of techniques |
| **Examples** | Bayesian Optimization, Tree-structured Parzen Estimators | Genetic Algorithm, Differential Evolution | Successive Halving, Hyperband | DEHB: Differential Evolution w/ Hyperband, BOHB: Bayesian Optimization w/ Hyperband |

# Evaluation, Analysis and Interpretability

## Evaluation

- Ideally, real and fully known microservice architecture to replicate in simulation with recorded traces
- … then deploy in real cloud to measure unoptimized vs. optimized metrics
- … and see if they improve in similar magnitude to the simulated unoptimized vs. optimized metrics

## Analysis and Interpretability

- Automatically find correlations
- …maybe by simplifying with some constraints (e.g., binning values)
- Visualize results with techniques from hyperparameter tuning

| HPA: minReplicas | HPA: maxReplicas | Node affinity: required label for “clock-speed” | Node affinity: preferred label for “location” | **SLO violations** | **Operating costs** |
|---|---|---|---|---|---|
| 5 | 9 | high | eu-west | 0-20 | 233 |
| 5 | 12 | high | eu-west | 0-20 | 400 |
| 5 | 15 | high | eu-west | 0-20 | 418 |
| 9 | 24 | medium | eu-west | 0-20 | 190 |
| 5 | 12 | high | eu-west | 0-20 | 392 |
| 5 | 16 | high | eu-east | 21-40 | 456 |
| 4 | 15 | high | eu-east | 41-60 | 444 |

# Summary

**Problem** Configuration of CO environments is complex, testing is time-consuming and costly

**Idea** Automate process of selecting and testing configurations, leverage simulation

**Benefit** Effortlessly and efficiently find good configurations

**Action** Build optimizer feature for simulation, perform lots of testing and analysis

# Thank you for listening!

# Any questions?