Fachhochschule Nordwestschweiz
Hochschule für Technik

Application Performance Management

FS 2021

Performance Messen

Michael Faes

Inhalt

Rückblick: Clustering

Was ist Performance?

Perspektiven

Metriken

Performance messen

Software-Test-Dokumentation

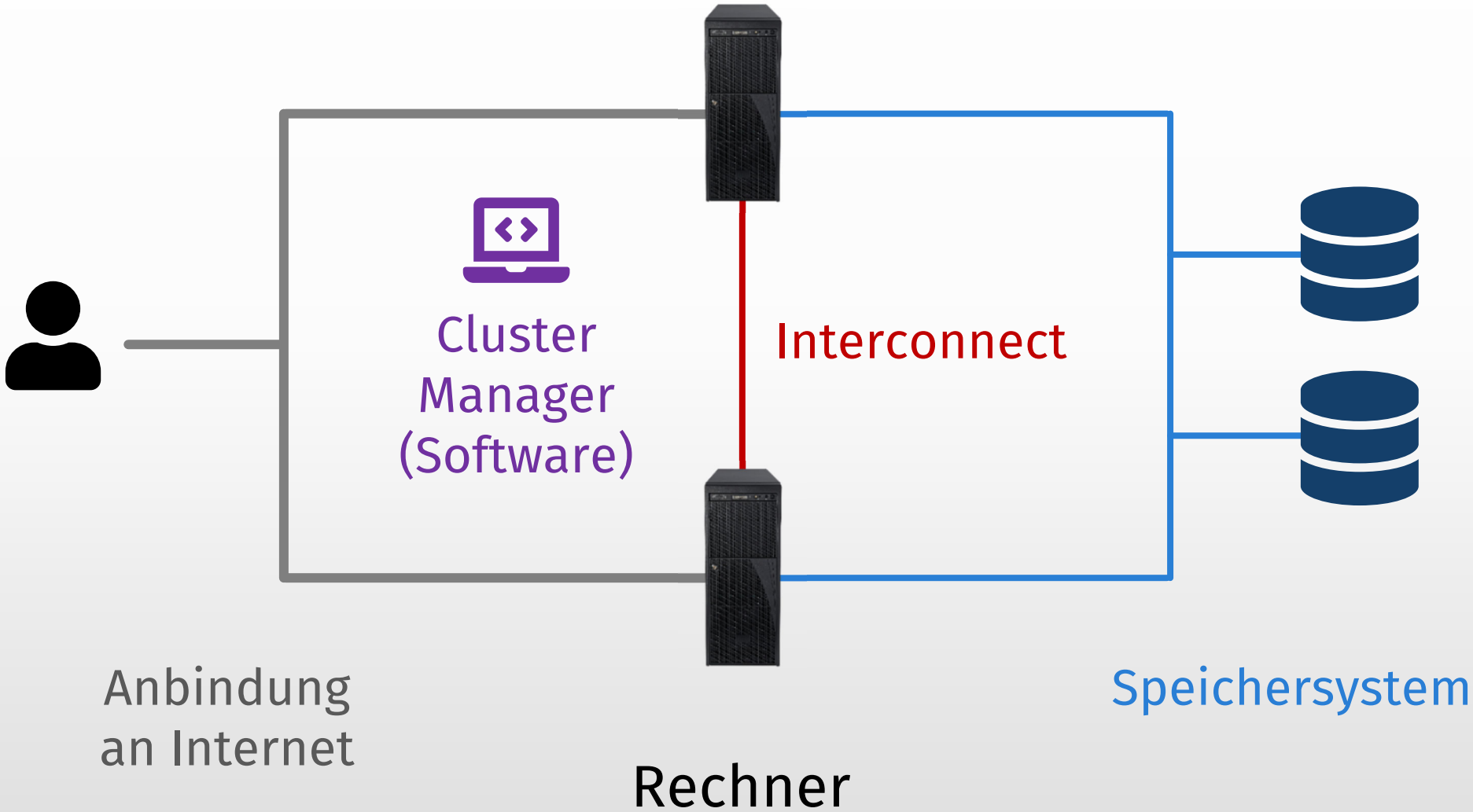
Aufbau eines Testplans

Apache JMeter

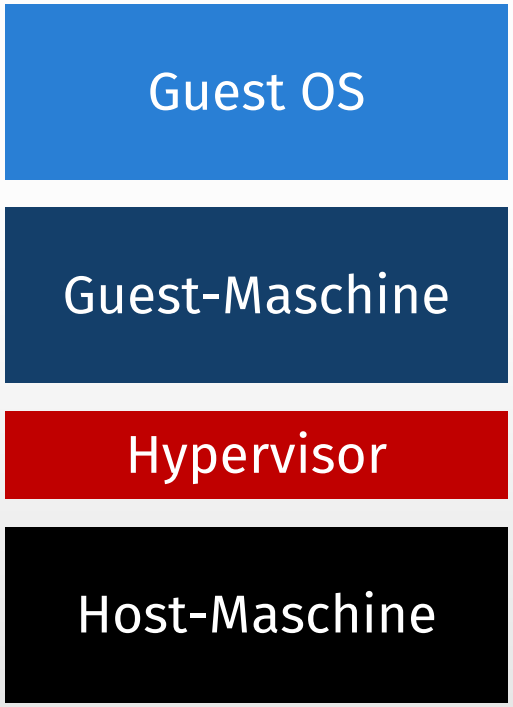
Übung

Rückblick: Clustering

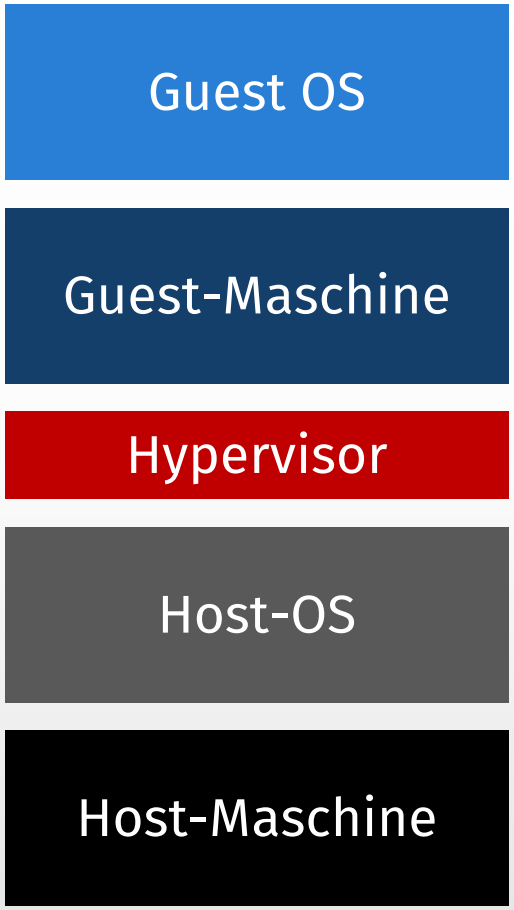
Komponenten eines HA-Clusters



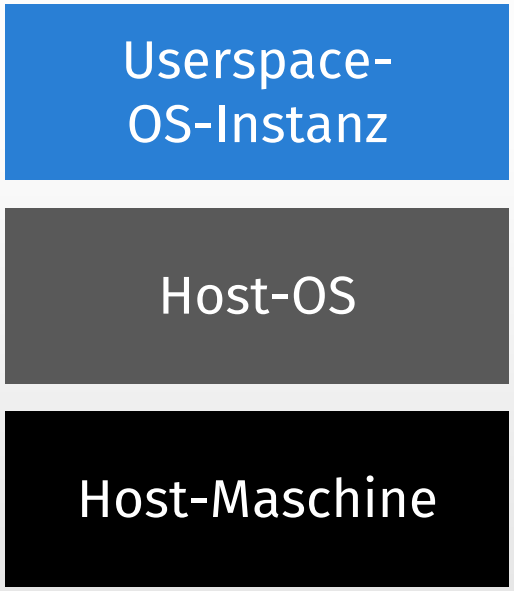
Virtualisierung: Hypervisors



“Typ-1”
(Bare-Metal) Hypervisor

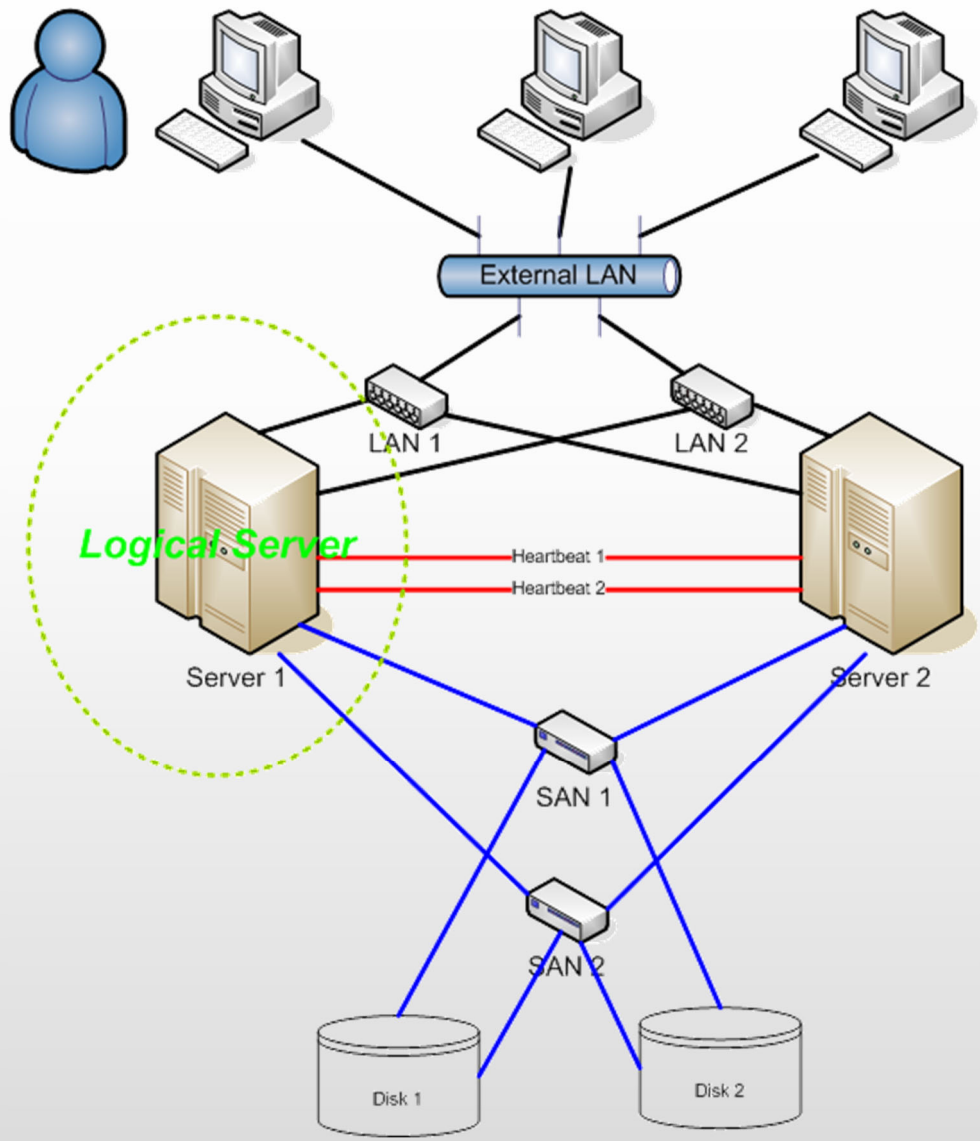


“Typ-2”
Hypervisor



Container-
Virtualisierung

Echte Redundanz



© George William Herbert
(Wikipedia)

Was ist Performance?

Warum Performance optimieren?

Für Software-Verkäufer

Schlechte Performance → unzufriedene Kunden → weniger Kunden → Verlust

Für interne Applikationen

Schlechte Performance → Produktivitätseinbusse → Verlust

Was ist Performance?

Entwickler

- Algorithmen: Asymptotisches Verh. (Big-O) aber auch konstante Faktoren
- **Ressourcenverbrauch**

Administratoren

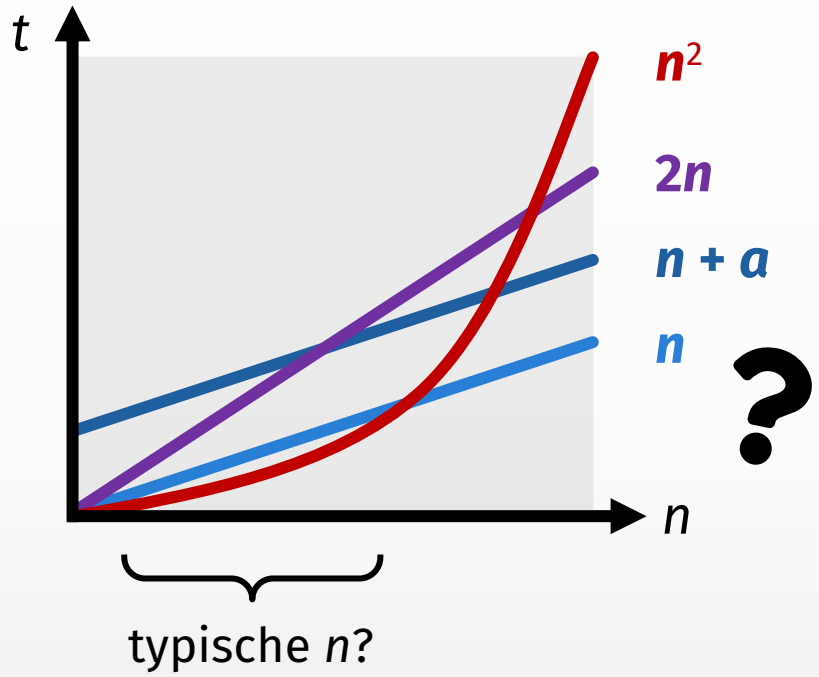
- Skalierbarkeit: Bei Bedarf mehr Rechner

Auftraggeber

- Hohe Zuverlässigkeit und **Verfügbarkeit**

Endbenutzer

- Flüssige Bedienung, kurze **Wartezeiten**



Performance in Zahlen

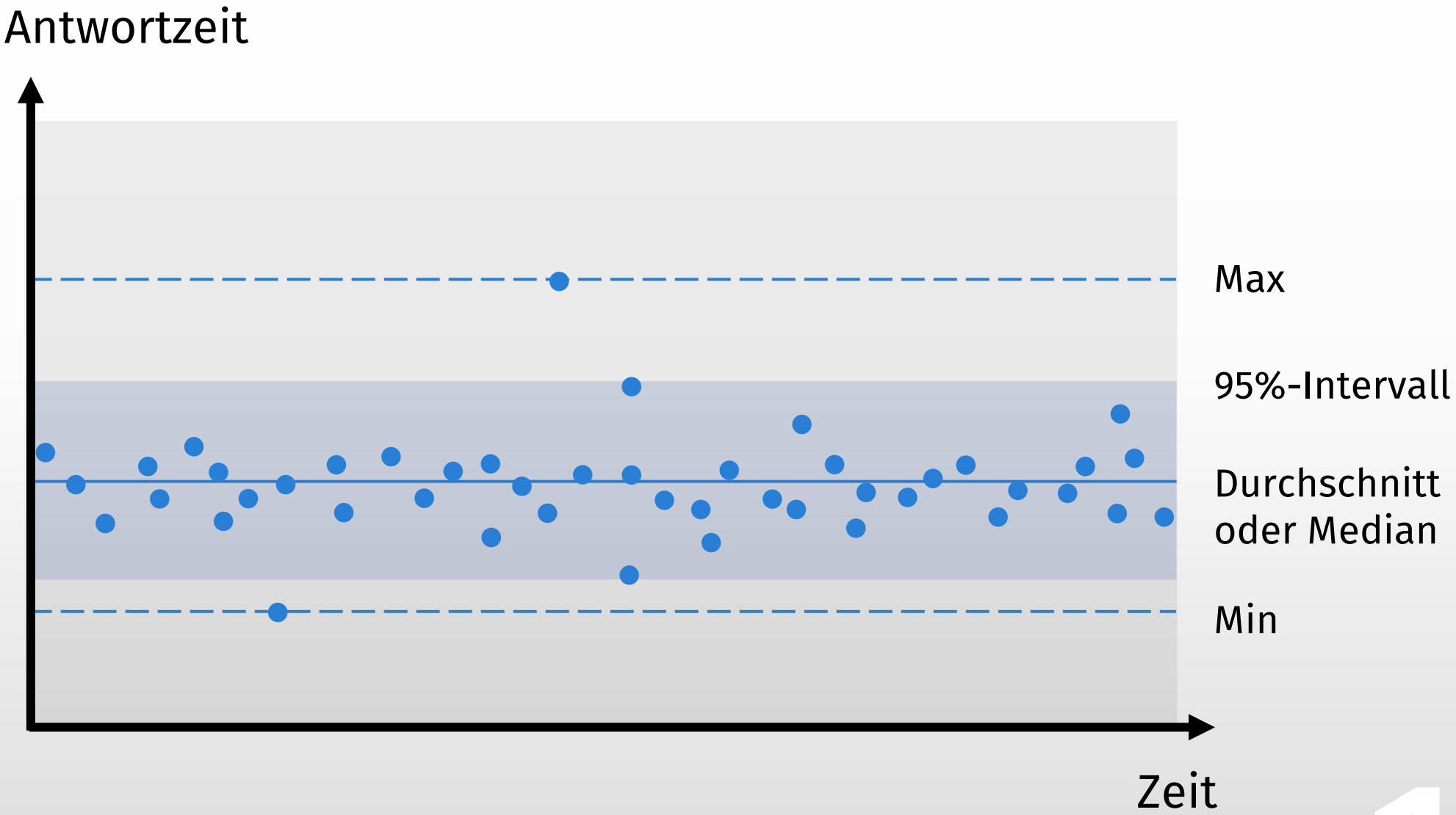
Subjektivität mit objektiven *Metriken* entschärfen

1. Welche Metriken?

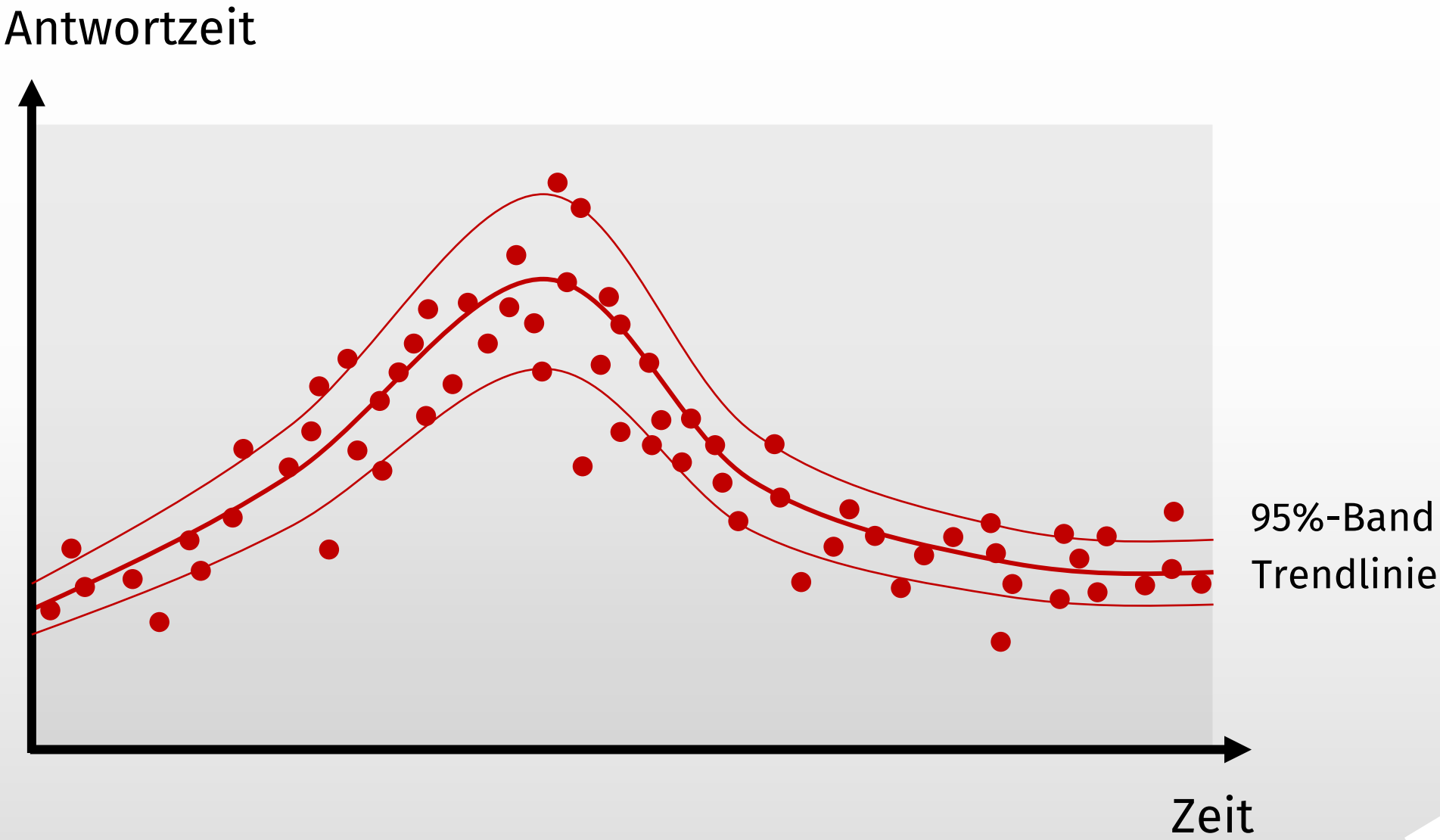
- Ressourcenverbrauch
- Durchsatz (throughput)
- Verfügbarkeit
- Endbenutzer-Antwortzeit (response time)

2. **Achtung**: Jeweils nicht *eine Zahl*, sondern Verteilung, oder variabel im Verlauf der Zeit

Beispiel: Antwortzeit



Beispiel: Antwortzeit



Ressourcenverbrauch

Auf System-Ebene

- CPU-Auslastung (in %)
- CPU load (run queue length)
  siehe “load average” von `top`
  oder ähnlichen Tools
- Speicherverbrauch
  Virt. vs realer Speicher (Swap!)
- Input/Output
  Speichermedien, Netzwerk

Auf Applikations-Ebene

- Thread-Pools
- Datenbank-Verbindungen
  (JDBC connection pool)
- Caches
  Optim. Grösse, Invalidierung
- Messaging-Queues
  Synchron und asynchron,
  z.B. RMI und JMS

Überall: Ressourcen sind begrenzt; deren Aufteilung auf Komponenten beeinflusst Gesamt-Performance einer App.

Endbenutzer-Antwortzeit

Kann objektiv gemessen werden, aber wird subjektiv wahrgenommen:

> *Die empfundene durchschnittliche Antwortzeit entspricht nicht dem Durchschnitt, sondern dem 0.9-Quantil, d.h. dem Wert, der grösser ist als 90% aller beobachteten Antwortzeiten.*

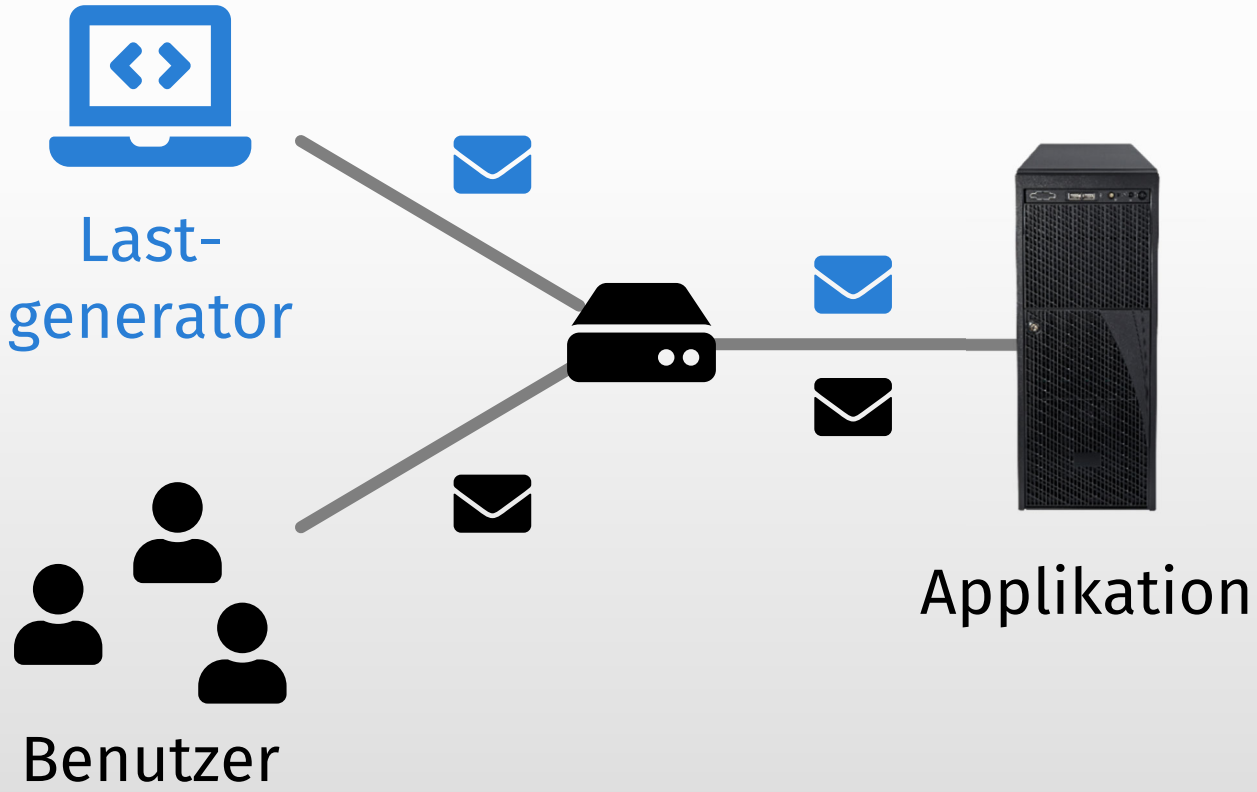
Arnold O. Allen, Introduction to Computer Performance Analysis with Mathematica (Academic Press, San Diego, CA, 1994)

Es gibt zwei Arten, Antwort-Zeit zu messen, bzw. abzuschätzen:

- Aktives Monitoring
- Passives Monitoring

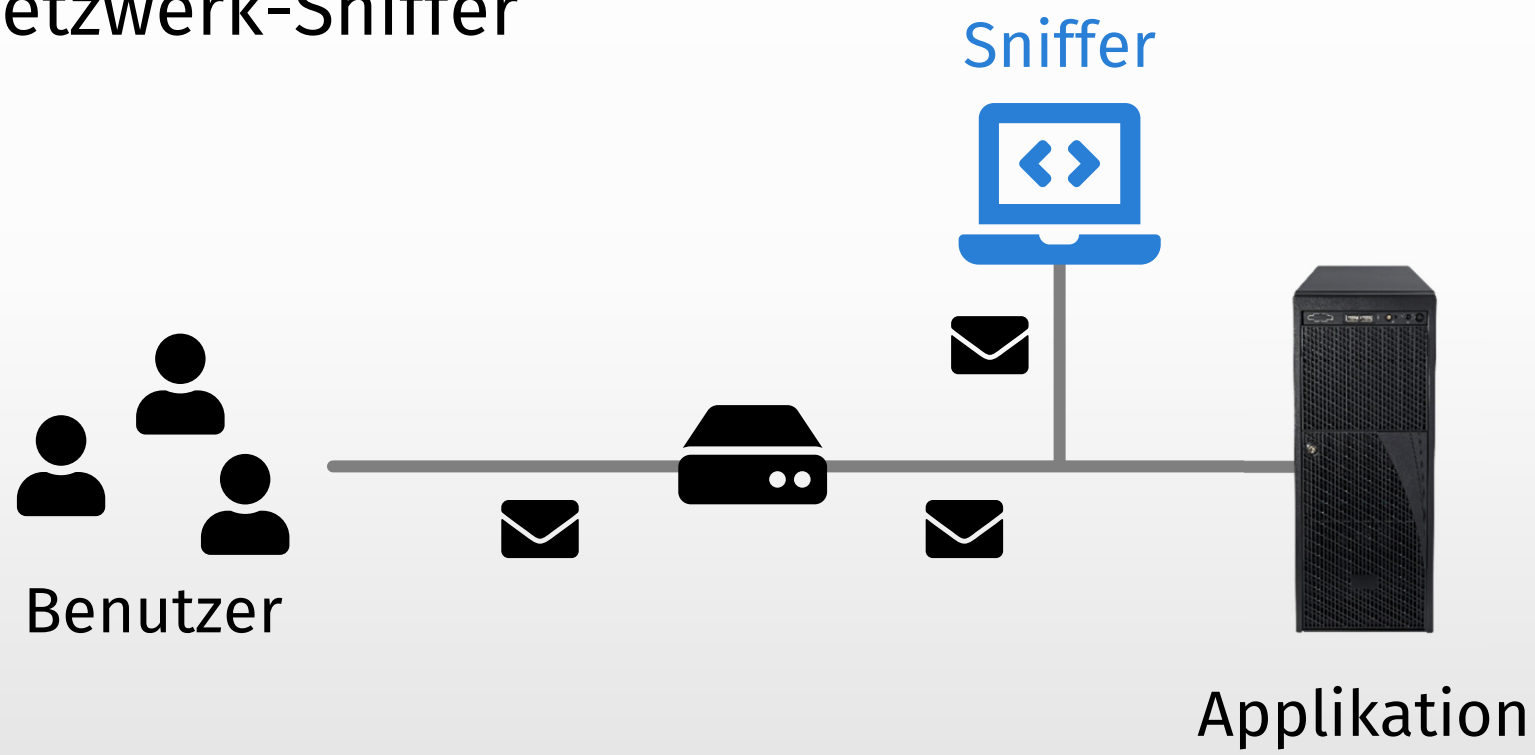
Aktives Monitoring

Synthetische Requests, die “typische” Arbeitseinheiten repräsentieren



Passives Monitoring

Beobachten echter Benutzer-Requests, z.B. mit Netzwerk-Sniffer



Vor- Nachteile?

Durchsatz

Definition: Anzahl durchgeführte Operationen/verarbeitete Requests pro Zeit (Einheit z.B. s^{-1})

Hängt grundsätzlich mit Antwortzeit zusammen, aber nicht unbedingt

- Je mehr Requests verarbeitet werden müssen, desto weniger Ressourcen stehen für jeden Request zur Verfügung → desto länger dauert das Verarbeiten
- Aber: Falls Requests viel (nicht-parallelisierbare) Arbeit benötigen und System nicht ausgelastet ist, beinflussen sie sich nicht

Reflektiert Effizienz der Applikation als Ganzes

Verfügbarkeit

Wie lange muss eine Applikation während eines bestimmten Zeitintervalls zur Verfügung stehen?

- Z.B. innerhalb eines Jahres während **99.9%** der Zeit

Wird in Service Level Agreement (SLA) festgehalten

Zusätzliche Bedingungen können bestimmt werden:

- Welcher Durchsatz muss während der Zeit garantiert sein?
- Wie hoch darf die E.U.-Antwortzeit sein?

SLA-Beispiel: Amazon S3

aws

Contact Sales Support English My Account Sign Up

Products Solutions Pricing Documentation Learn Partner Network AWS Marketplace Explore More

For all requests not otherwise specified below:

Monthly Uptime Percentage	Service Credit Percentage
Less than 99.9% but greater than or equal to 99.0%	10%
Less than 99.0% but greater than or equal to 95.0%	25%
Less than 95.0%	100%

For requests to S3 Intelligent-Tiering, S3 Standard-Infrequent Access, and S3 One Zone-Infrequent Access:

Monthly Uptime Percentage	Service Credit Percentage
Less than 99.9% but greater than or equal to 98.0%	10%
Less than 98.0% but greater than or equal to 95.0%	25%
Less than 95.0%	100%

Performance messen

Performance-Messung als SW-Test

Performance-Messung ist eine Art Software-Test!

Industrie hat Standards für Software-Testing entwickelt

Beispiel: IEEE 829, *Standard for Software and System Test Documentation* (https://ieeexplore.ieee.org/document/4578383)

Definiert u.a. folgende Dokumente:

Master Test Plan	Level Test Report
Level Test Plan	Master Test Report
Level Test Log	...

Aufbau eines Testplans

Test Plan Identifier

Introduction

Test Items

Features to be tested

Features not to be tested

Approach

Item pass/fail criteria

Suspension criteria & resumption requirements

Test deliverables

Testing tasks

Environmental needs

Responsibilities

Staffing and training needs

Schedule

Approvals

Aufbau eines Testplans

Introduction

Zielsetzung, Umfang, konkrete Ziele, evtl. Einschränkungen, Budget

Test Items

Auflistung der Komponenten, die getestet werden

Features to be tested

Auflistung aller Eigenschaften und Funktionalität, die getestet werden, inkl. Referenzen zu Spezifikationen o.ä.

Aufbau eines Testplans

Approach

Wie werden die Features getestet? Detailierte Informationen über Test-Szenarios, Testfälle, Testdaten, Techniken, Prioritäten

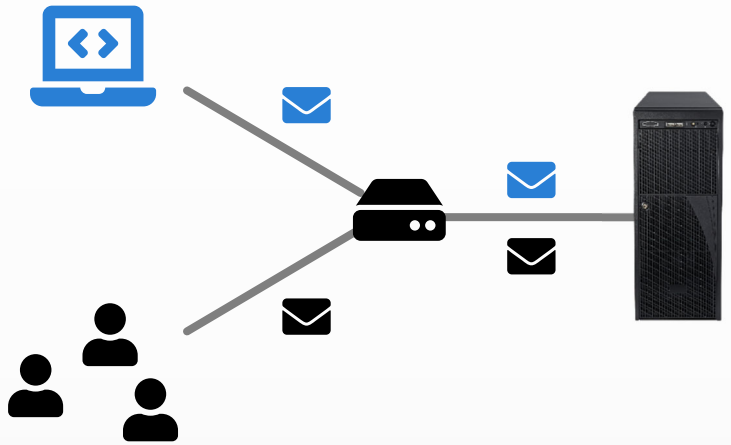
Environmental needs

Voraussetzungen an Testumgebung, z.B. Hardware, Software, andere Voraussetzungen.

Für Performance-Tests zwei Seiten: Zu testende Infrastruktur und Testing-Seite

Messen mit Apache JMeter

Software zum Messen von Durchsatz und Antwortzeit mittels aktivem Monitoring (Lastgenerator)

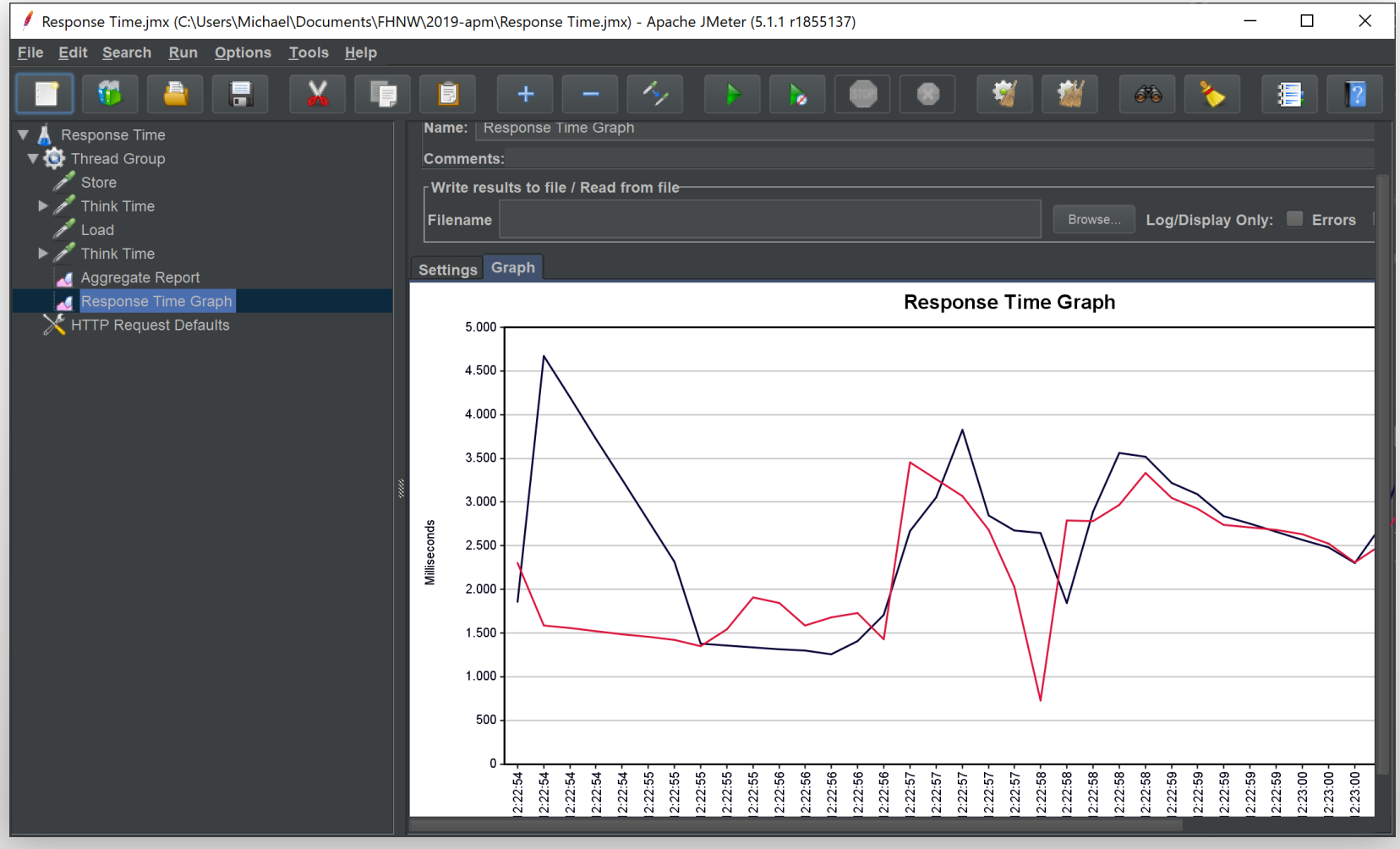


Unterstützung für Web-Apps, Web-Dienste, Datenbanken, Verbindungen

- Protokolle: HTTP(S), REST, SOAP, FTP, LDAP, JDBC, JMS, SMTP, IMAP, TCP, ...

Konfiguration über GUI, via Scripting oder via Browser-Recording

Verwendung von JMeter (Live)



Übung: Performance messen mit Apache JMeter