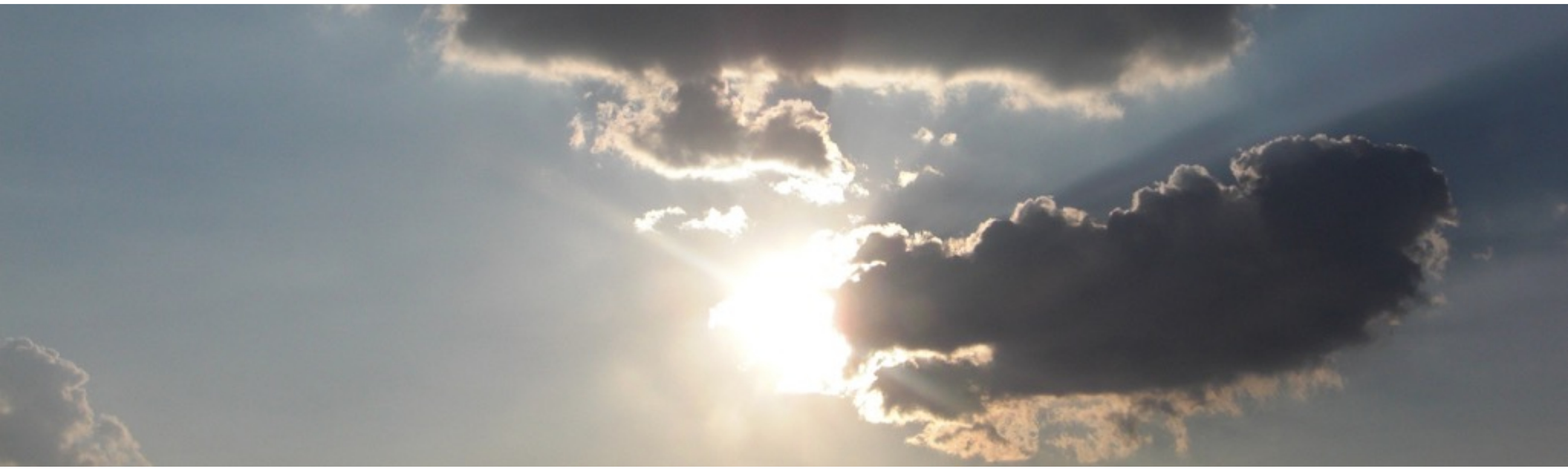
ISEngineering

Technische Universität Berlin



Experiment-Driven Evaluation of Cloud-based Distributed Systems

Markus Klems, Information Systems Engineering, TU Berlin

11th Symposium and Summer School On Service-Oriented Computing

Agenda

Technische Universität Berlin

- Introduction
- Experiments
- Experiment Automation
- Trade-offs
- Conclusion

ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Motivation

- Many modern applications integrate distributed system software that runs on cloud infrastructure.

- Cloud-based distributed systems promise to deliver on multiple desirable objectives:
  - performance,
  - scalability,
  - elasticity,
  - low cost,
  - high availability,
  - (and certain consistency guarantees).

Problem

Technische Universität Berlin

How can we find out if a specific cloud-based distributed system really delivers on its promises?

Possible approach		Weakness
Rely on the opinion of experts	→	Opinions might be biased or wrong
Use published experiment results as a basis for decision making	→	Experiment results might not be applicable to the specific use case
Simplified simulation or experiment	→	Potentially disregards important aspects of complex systems
Experimentally evaluate only a single system objective	→	Many objectives are desirable and could be conflicting

Research Question & Contributions

Technische Universität Berlin

Research Question: How can experiments be utilized to evaluate multiple objectives of cloud-based distributed systems?

Question	Contribution
How well can we reproduce related experiments?	Results of new experiments and experiment reproductions.
How can we automate experiments?	A new approach and system implementations for experiment automation in compute clouds.
How can we describe and evaluate practical trade-off problems between conflicting objectives?	An experiment-driven trade-off evaluation method with 2 instantiations of the method.

ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Agenda

Technische Universität Berlin

- Introduction
- **Experiments**
- Experiment Automation
- Trade-off Evaluation
- Conclusions

ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Related Work

Selected Related Work	**P**	**S**	**E**	**A**	**C**
Cooper, et al. (2010): Benchmarking Cloud Serving Systems with YCSB.	**X**	**X**	**X**	X	
Bodík, et al. (2010): Characterizing, Modeling, and Generating Workload Spikes for Stateful Services.	**X**				
Trushkowsky, et al. (2011): The SCADS Director: Scaling a Distributed Storage System under Stringent Performance Requirements.	**X**	**X**	**X**		
Patil, et al. (2011): YCSB++: Benchmarking and Performance Debugging Advanced Features in Scalable Table Stores.	**X**				**X**
Rabl, et al. (2012): Solving big data challenges for enterprise application performance management.	**X**	**X**			
Fior, et al. (2013): Under Pressure Benchmark for DDBMS Availability.	**X**			**X**	

Legend: **P** (Performance), **S** (Scalability), **E** (Elasticity), **A** (Availability), **C** (Consistency)

Publications

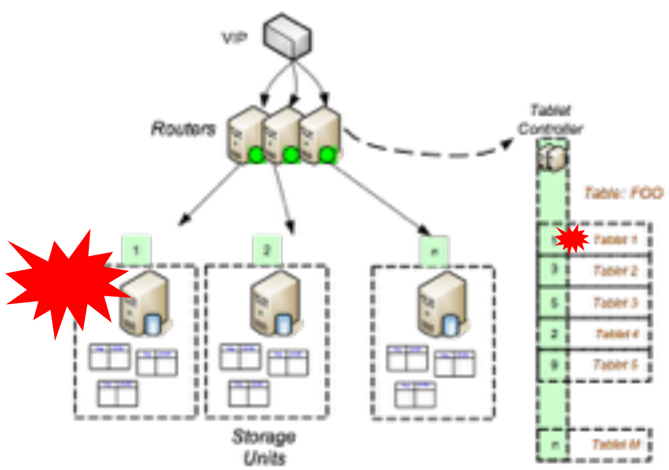
Selected Publications	P	S	E	A	C
Klems, Bermbach, and Weinert (2012): A Runtime Quality Measurement Framework for Cloud Database Service Systems.	X	X	X	X	X
Klems and Lê (2013): Position Paper: Cloud System Deployment and Performance Evaluation Tools for Distributed Databases.	X	X			
Klems, Silberstein, Chen, Mortazavi, Albert, Narayan, Tumbde, and Cooper (2012): The Yahoo! Cloud Datastore Load Balancer.	X		X		
Kuhlenkamp, Klems, and Röss (2014): Benchmarking Scalability and Elasticity of Distributed Database Systems.	X	X	X		

Legend: **P** (Performance), **S** (Scalability), **E** (Elasticity), **A** (Availability), **C** (Consistency)

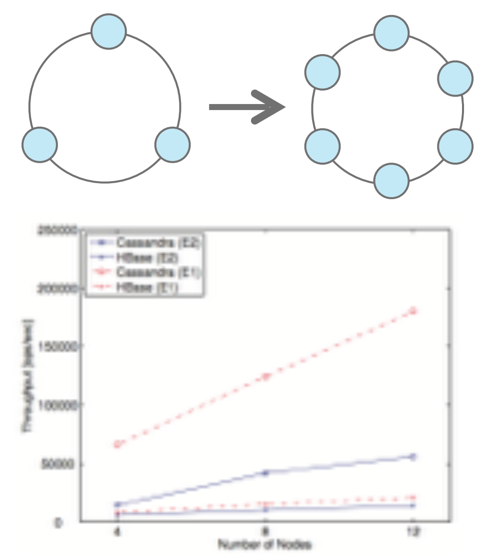
Experiments

Technische Universität Berlin

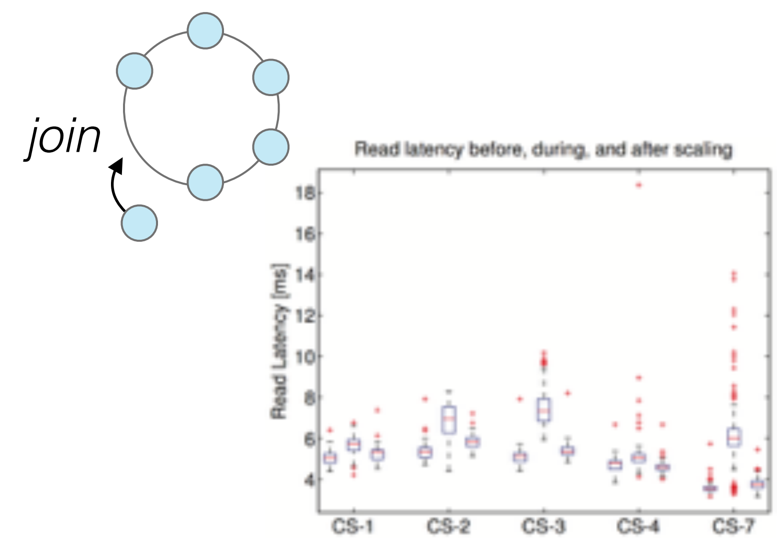
Hotspot performance



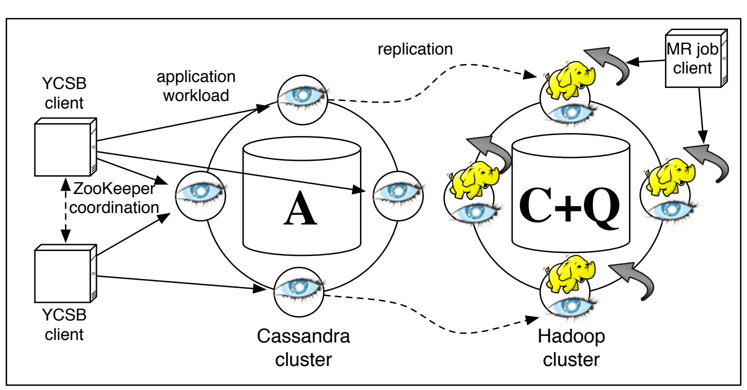
Scalability



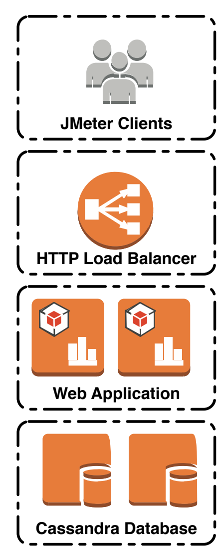
Elasticity



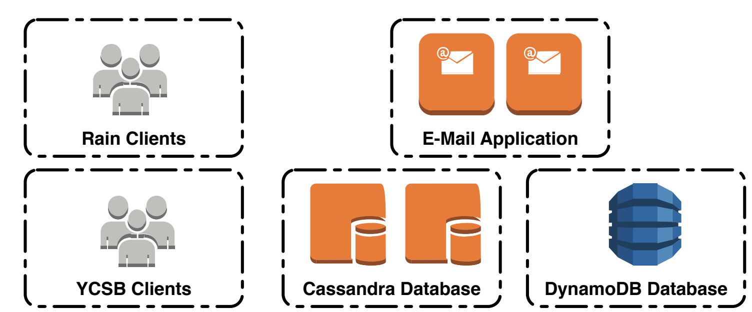
Multi-cluster setups



NoSQL-backed web app



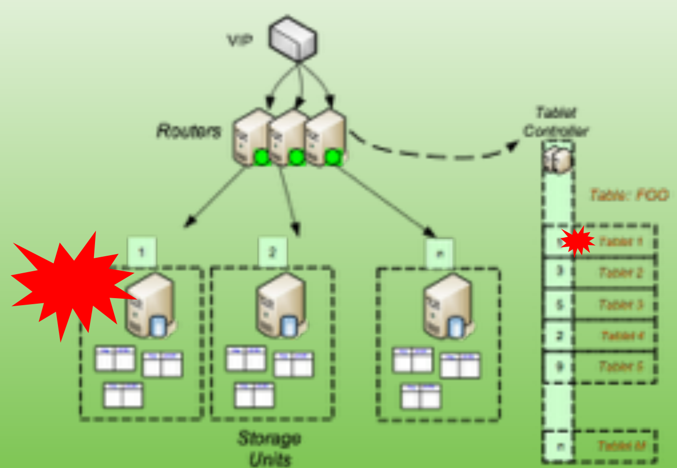
NoSQL-backed email app



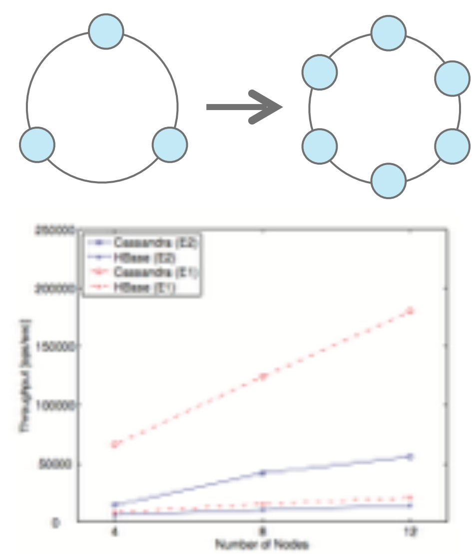
ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Experiments

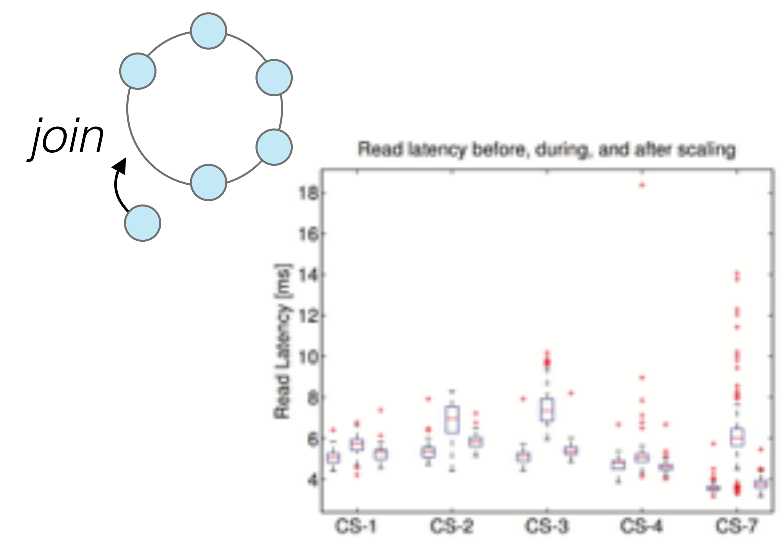
Hotspot performance



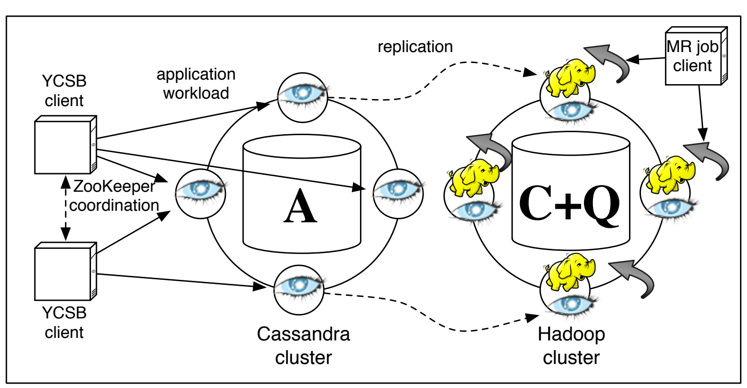
Scalability



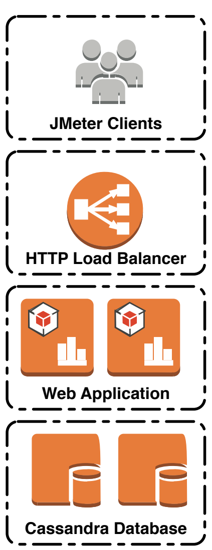
Elasticity



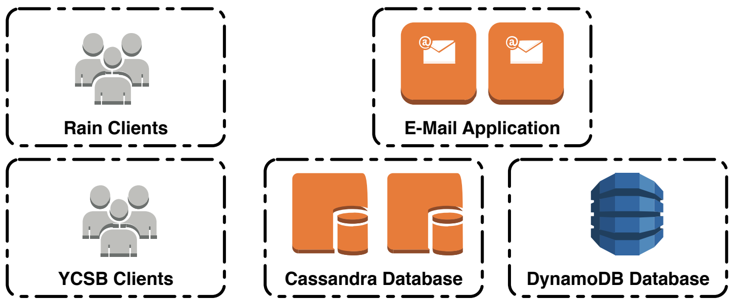
Multi-cluster setups



NoSQL-backed web app



NoSQL-backed email app

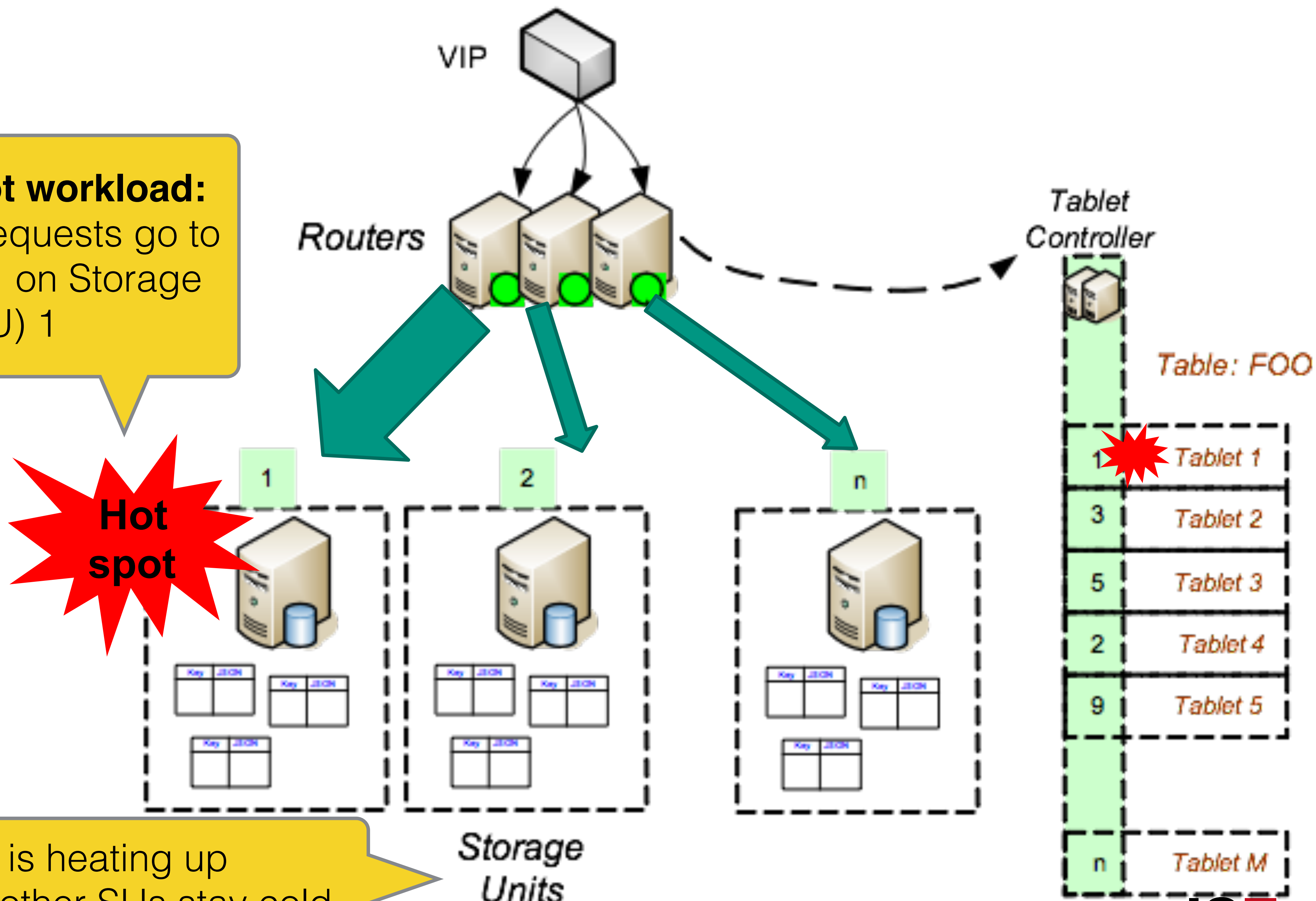


Wirtschaftsinformatik –
Information Systems Engineering

Sherpa Ordered Tables under Hotspot Workloads

Technische Universität Berlin

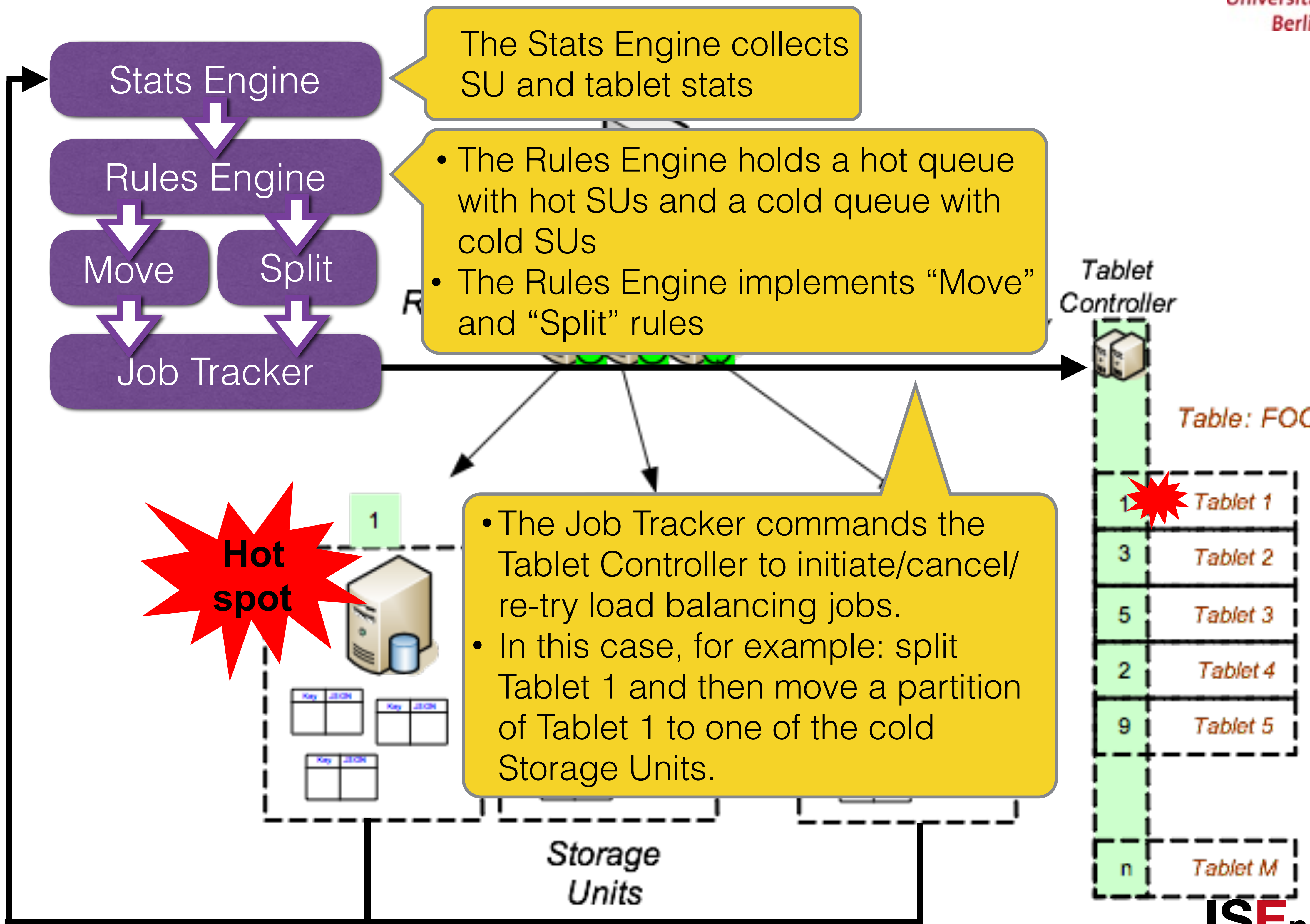
Hotspot workload: many requests go to Tablet 1 on Storage Unit (SU) 1



- SU1 is heating up
- The other SUs stay cold

ISEngineering
Wirtschaftsinformatik – Information Systems Engineering

Yak resolves Hotspots via Split & Move



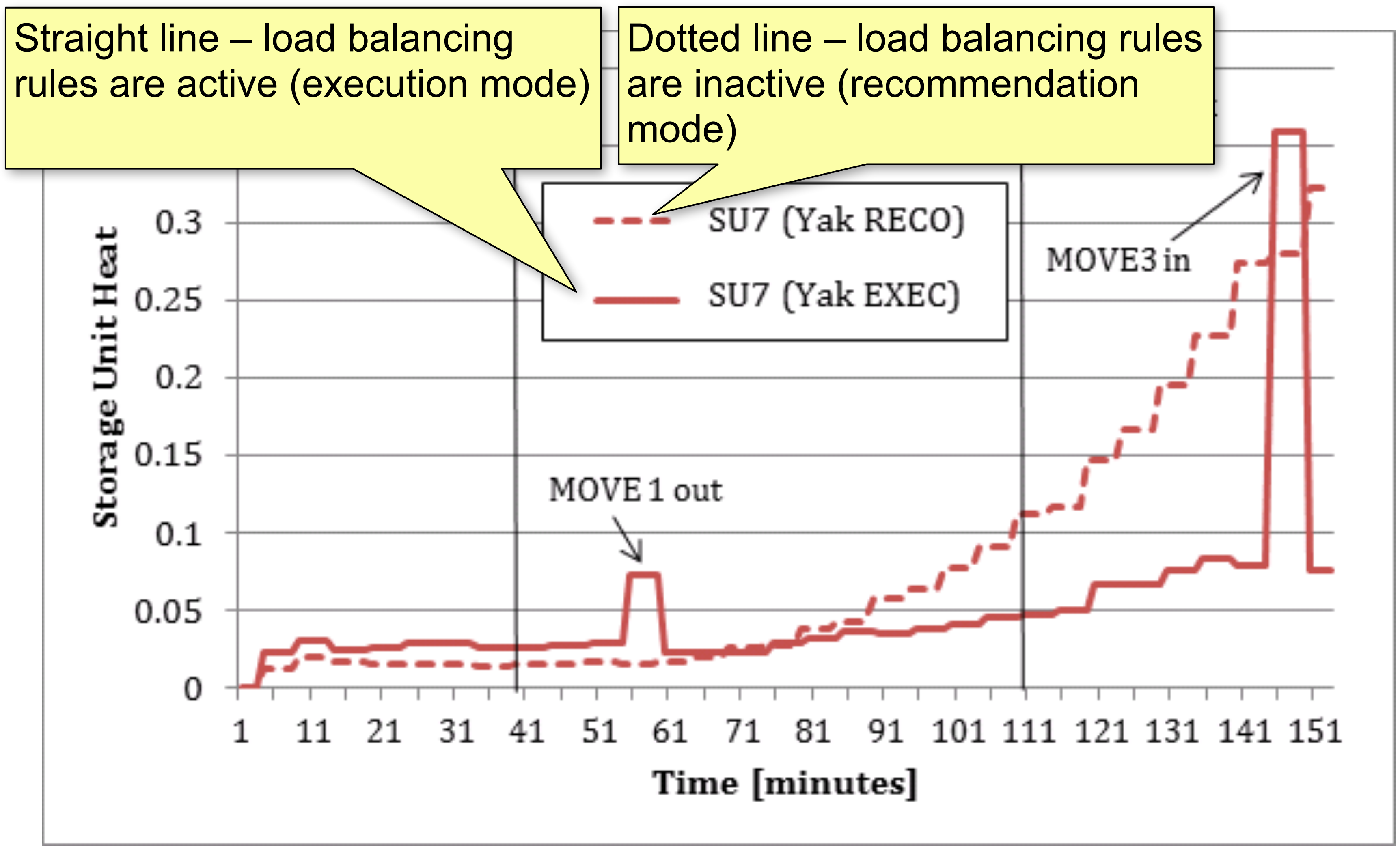
Technische Universität Berlin

ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Sherpa hotspot performance results

Technische Universität Berlin

- Response time degradation in Sherpa with Distributed Ordered Table setup under certain hotspot workloads
- Performance can be improved by online data migration at cost of server load spikes, due to data movement

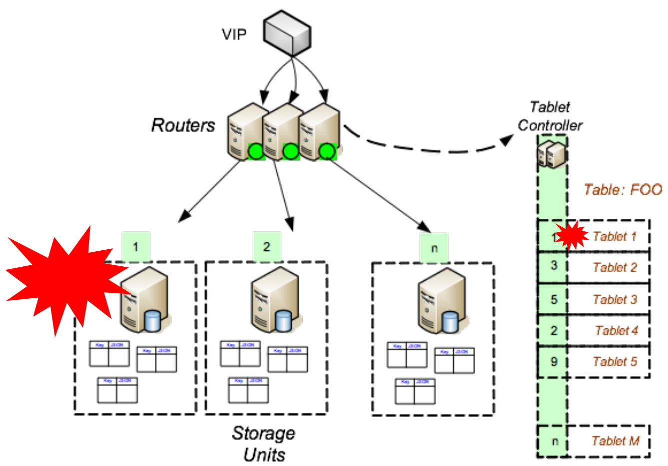


The hot Storage Unit “Sherpa7”

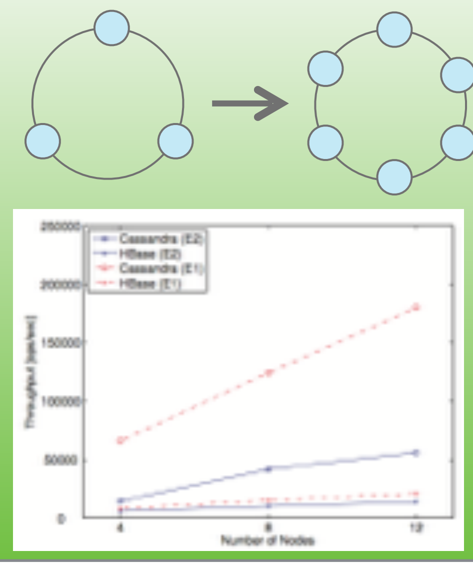
ISEngineering
Wirtschaftsinformatik – Information Systems Engineering

Experiments

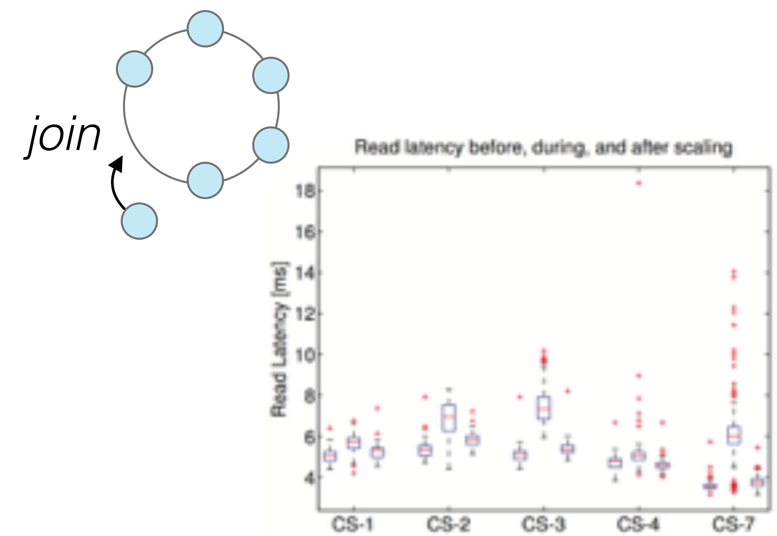
Hotspot performance



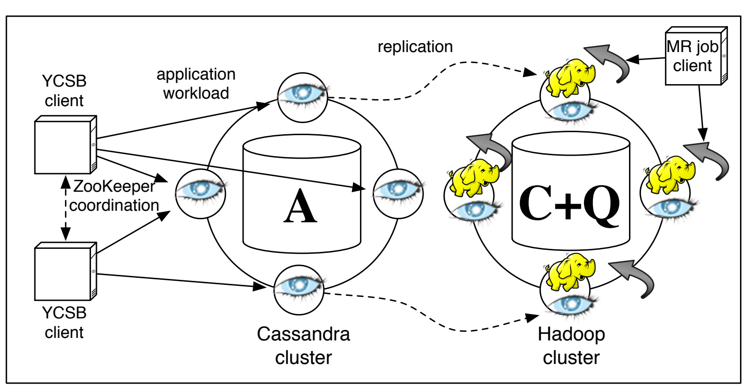
Scalability



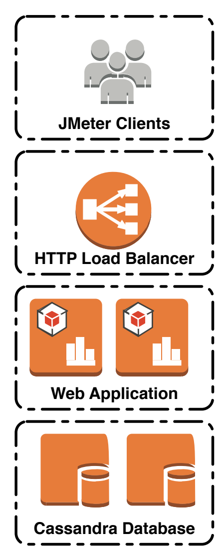
Elasticity



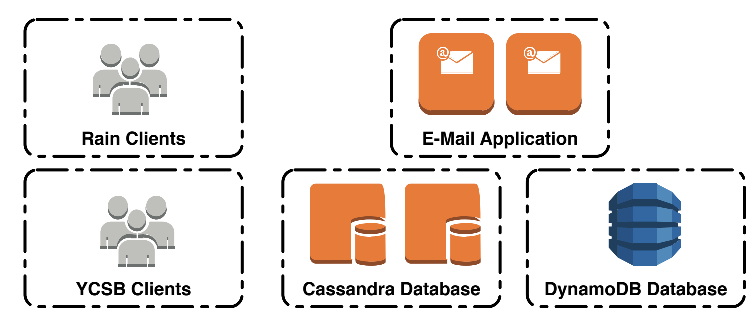
Multi-cluster setups



NoSQL-backed web app

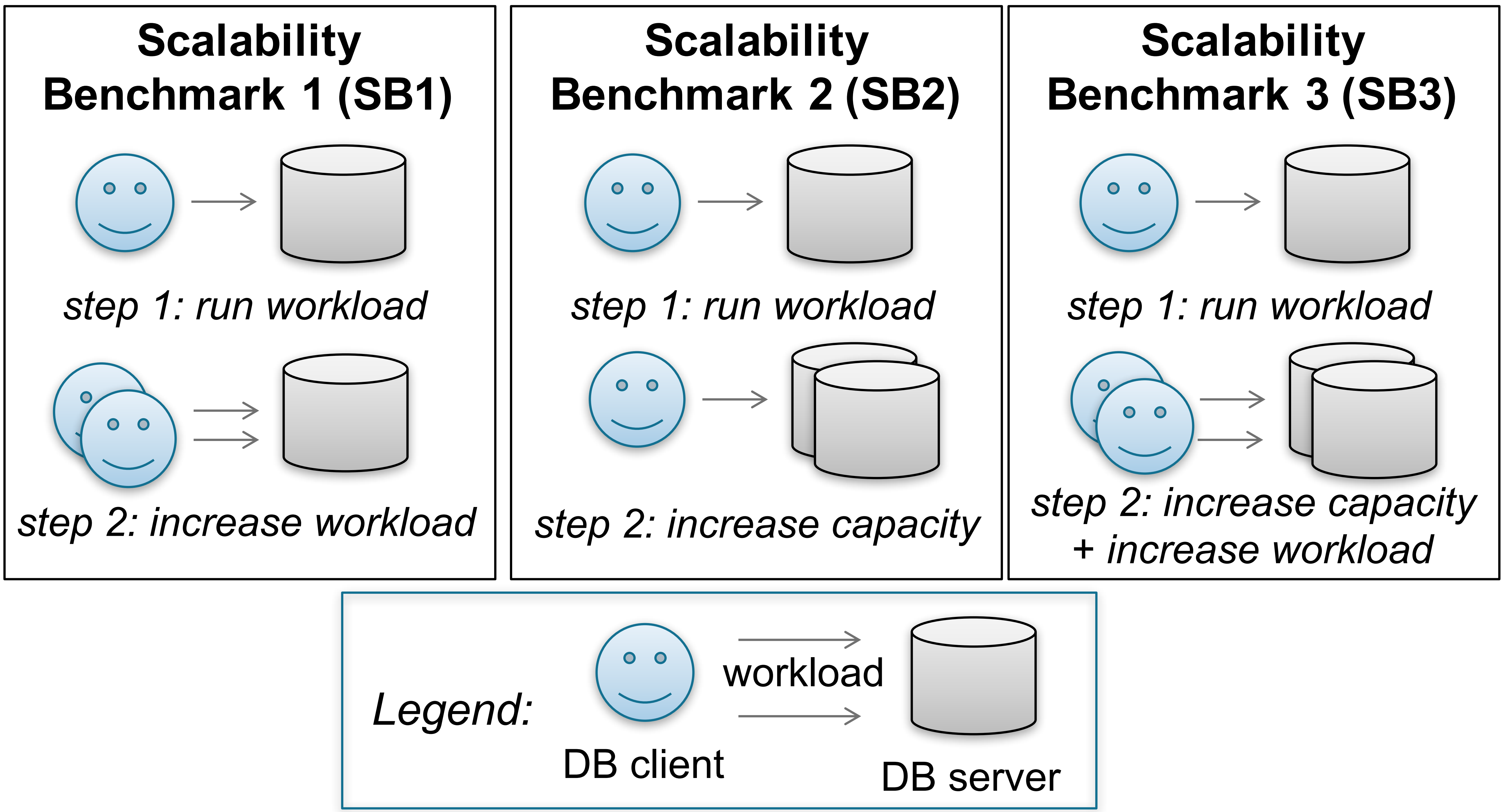


NoSQL-backed email app



Wirtschaftsinformatik –
Information Systems Engineering

Scalability Benchmarking



ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

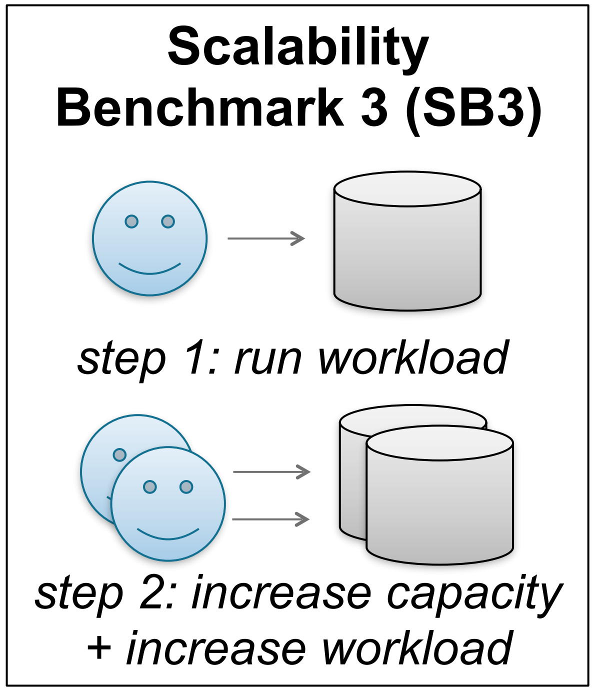
Experiment Reproduction

Reproduction of experiments by Rabl, et al.*

- Selected results of the original experiments:
  - Cassandra is the winner in terms of throughput.
  - HBase has low write latency, however, higher read latency.
  - Linear scalability of both Cassandra and HBase.

**Tilmann Rabl, Sergio Gómez-Villamor, Mohammad Sadoghi, Victor Muntés-Mulero, Hans-Arno Jacobsen, and Serge Mankovskii. 2012. Solving big data challenges for enterprise application performance management. Proc. VLDB Endow. 5, 12 (August 2012), 1724-1735.*

- Our experiment objective
  - Create an experiment plan that reproduces the original experiment setups.
  - Change system capacity and change load proportionally between subsequent workload runs (SB3).



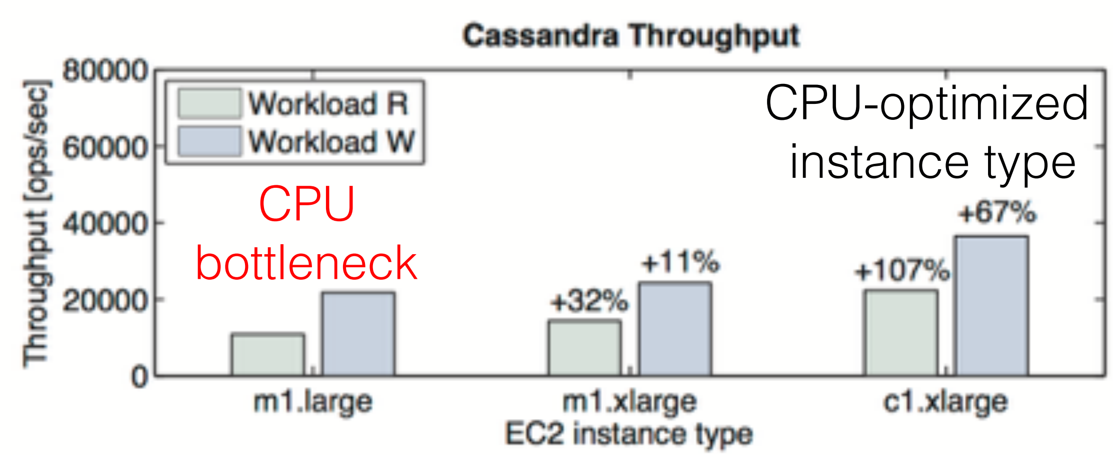
ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Cassandra and HBase Scalability Benchmark

- HBase and Cassandra scale linearly from 4-12 servers
- Cassandra: better performance for read-heavy workloads
- HBase: better performance for write-heavy workloads
- Cassandra performance is CPU-bound when using EC2 general purpose instances

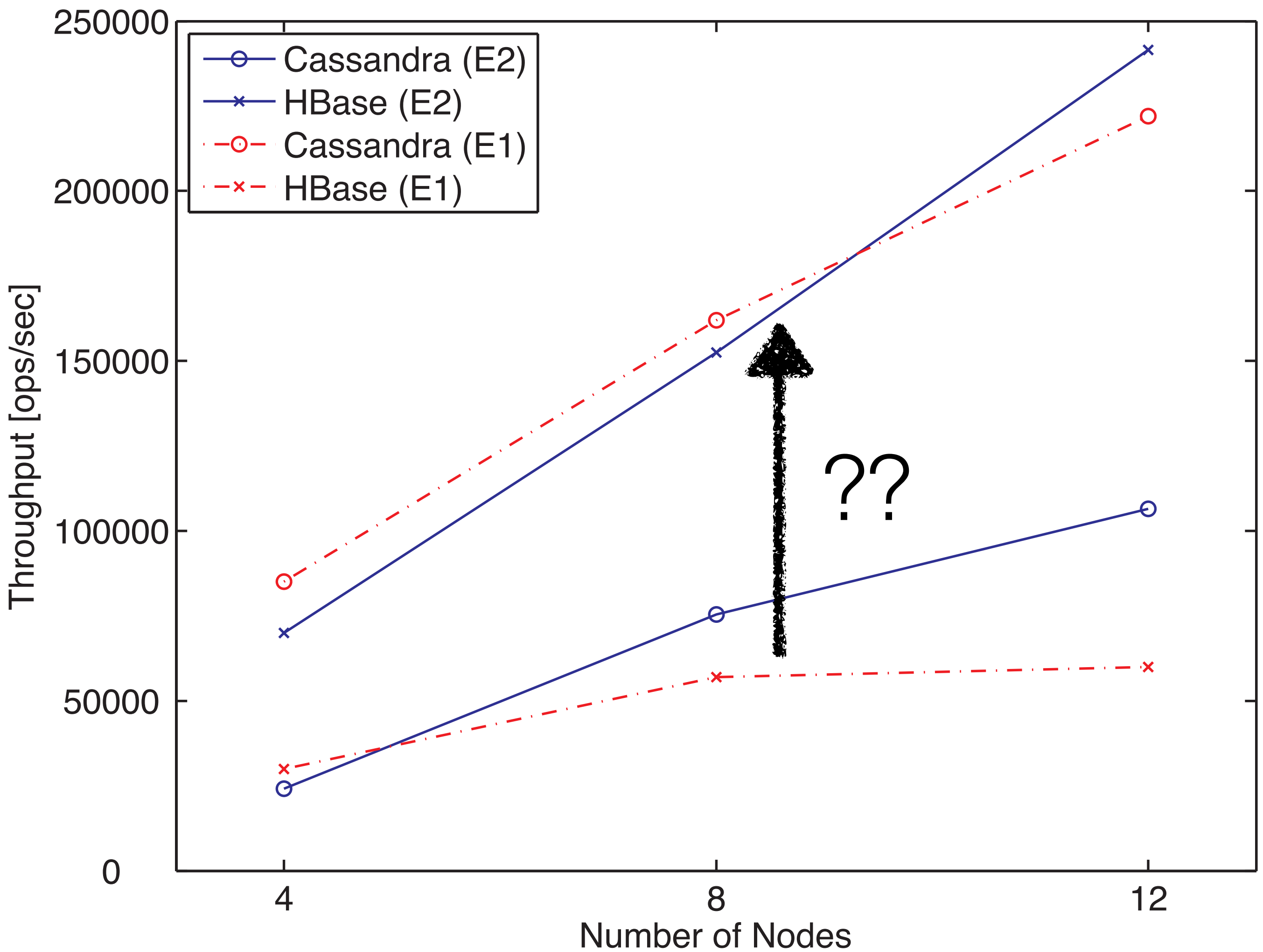
		Avg RT [ms]	
DB	Workload	Max Load	95% Load
HBase	Read-heavy	111	45
	Write-heavy	0	2
	Scan-heavy	162	49

		Avg RT [ms]	
DB	Workload	Max Load	95% Load
Cassandra	Read-heavy	26	21
	Write-heavy	13	10
	Scan-heavy	122	56



ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

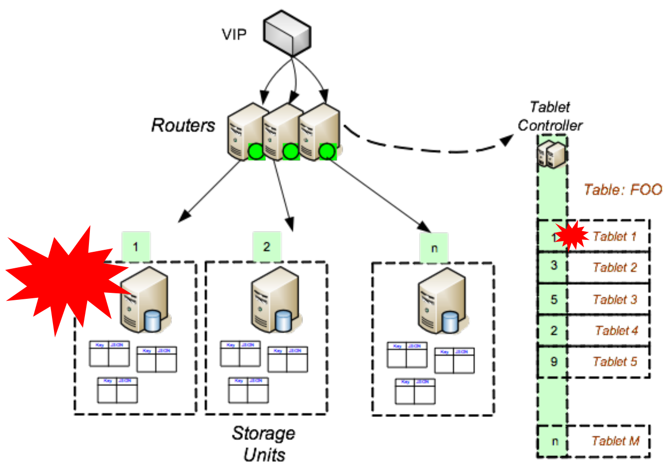
Selected Observations: Client-side Bottleneck



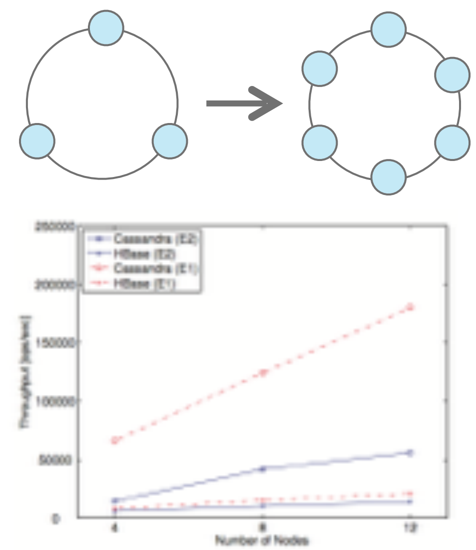
- We observed a client-side performance bottleneck when we reproduced the original experiments with HBase and a write-heavy workload.
- Increasing the number of YCSB client servers (x2) increased performance considerably, as shown in the graph.

Experiments

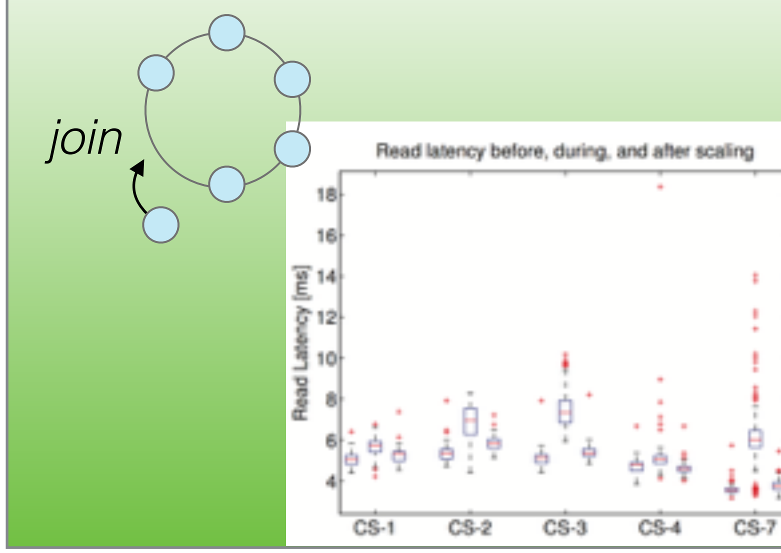
Hotspot performance



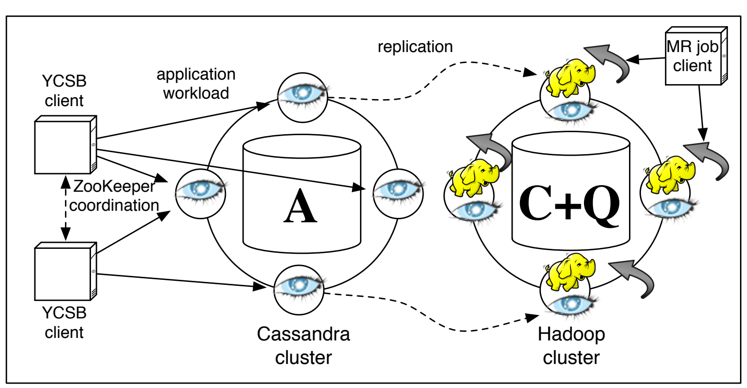
Scalability



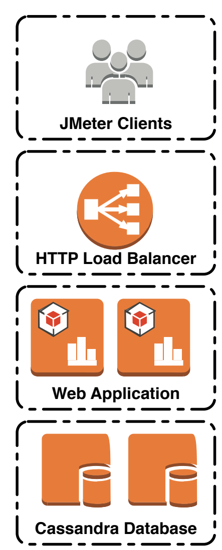
Elasticity



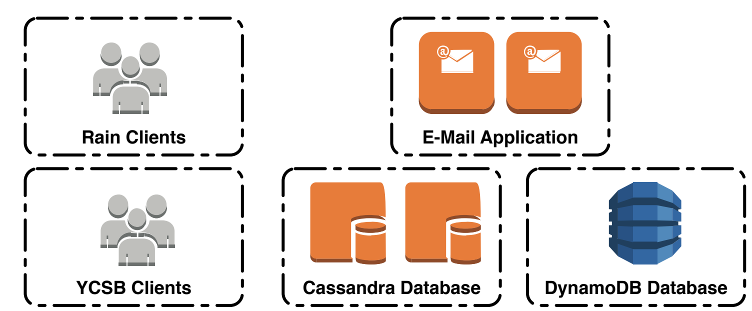
Multi-cluster setups



NoSQL-backed web app



NoSQL-backed email app



ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

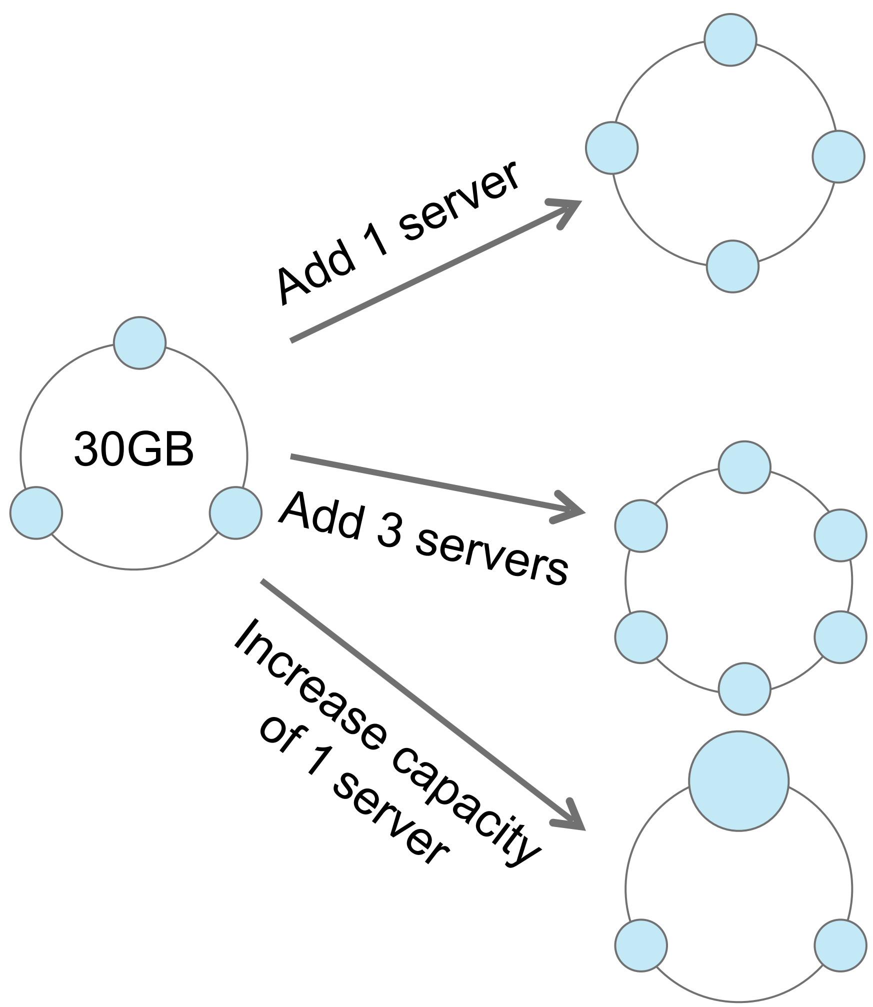
Elasticity Benchmarking

Technische Universität Berlin

- EB1:
  - Apply constant load and execute a single scaling action during the execution of a workload.
  - Scaling actions are executed at pre-specified points in time
- EB2:
  - Change load and execute one or more scaling actions during a workload run.
  - Scaling actions are executed automatically, e.g., by a rule-based control framework that uses moving averages of CPU utilization as sensor inputs.

ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Cassandra Elasticity Benchmark (EB1)



Streaming	Scaling time	Avg. read lat.
5 Mbit/s	198 min	5.7 ms
40 Mbit/s	31 min	6.9 ms
unthrottled	16 min	7.5 ms
disabled	1.3 min	5.2 ms

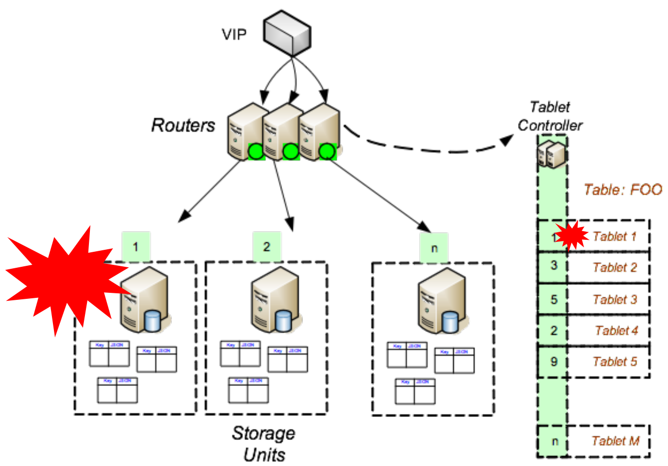
Streaming	Scaling time	Avg. read lat.
unthrottled	13 min	6.5 ms
disabled	0.8 min	5.8 ms

Streaming	Scaling time	Avg. read lat.
N/A	8 min	6.1 ms

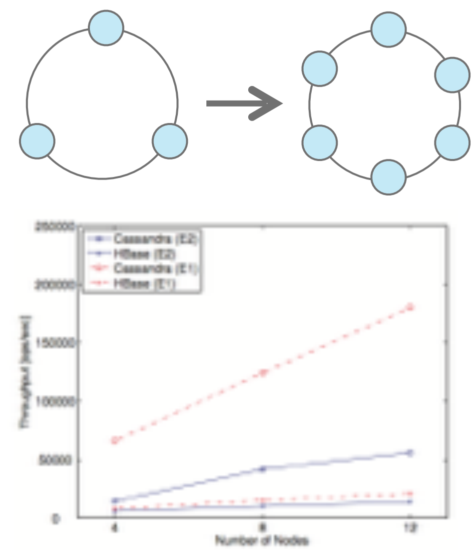
ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Experiments

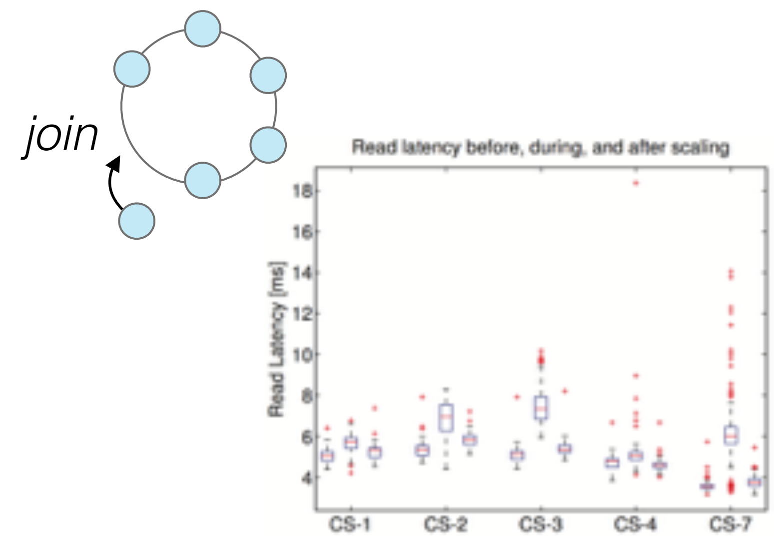
Hotspot performance



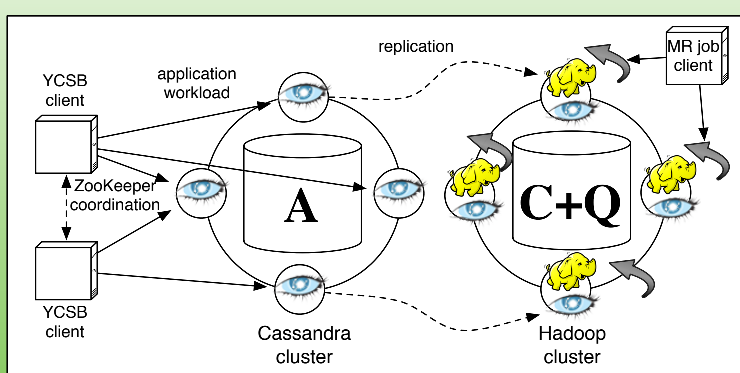
Scalability



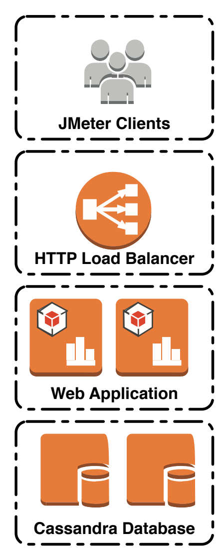
Elasticity



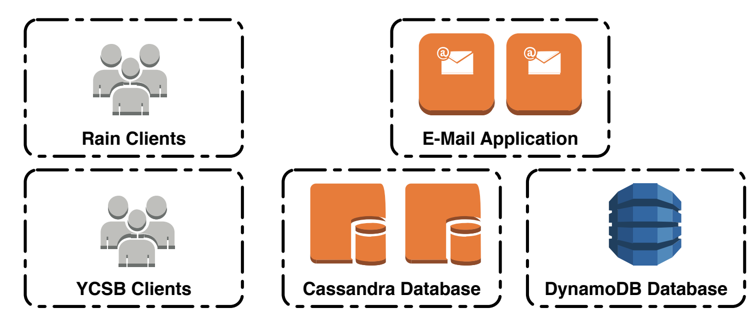
Multi-cluster setups



NoSQL-backed web app



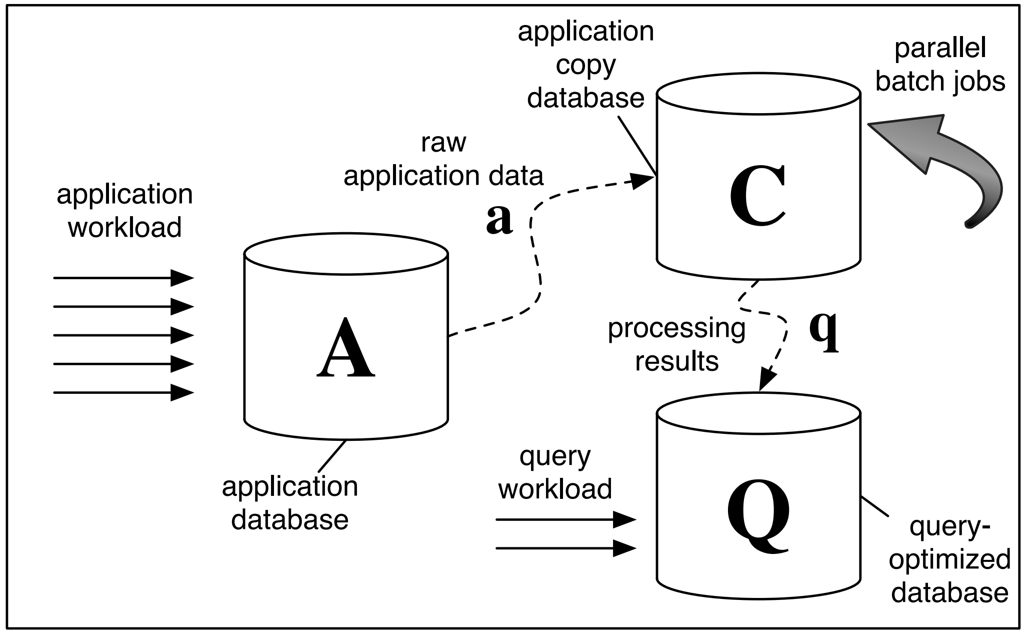
NoSQL-backed email app



Technische Universität Berlin

ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Cassoop Architecture



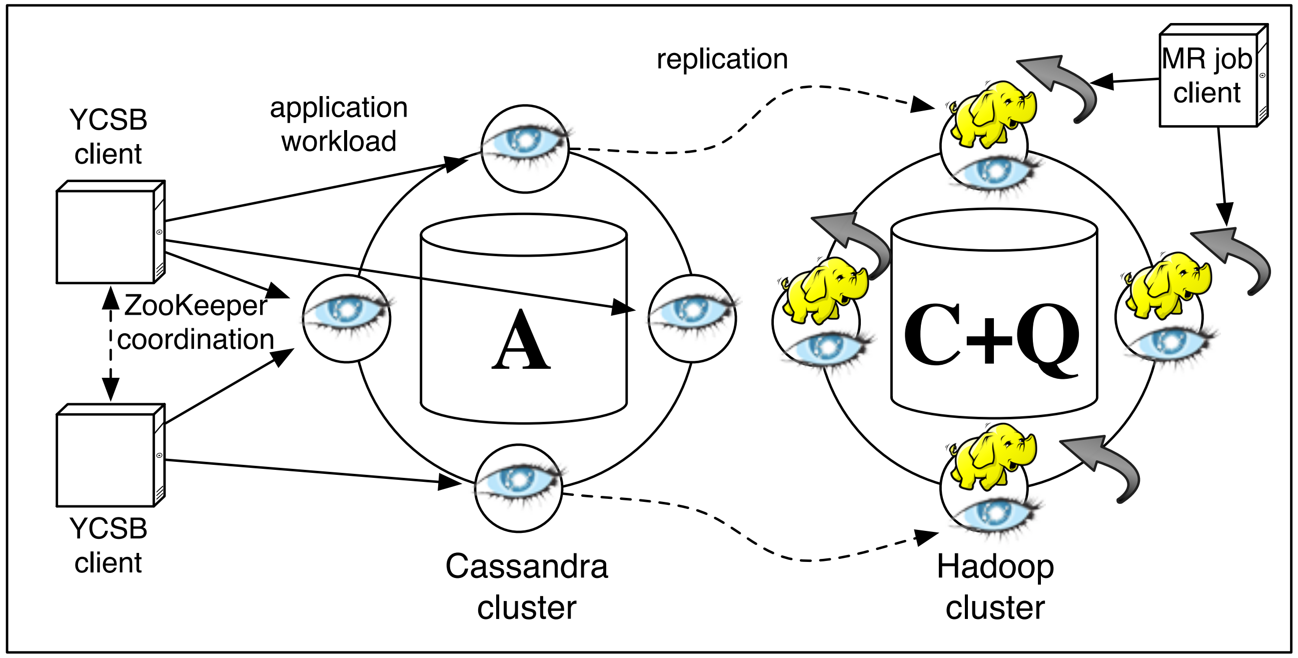
Basic System Architecture Design

- **Application database**: serves simple single-row requests, such as create, read, update, and delete, and simple multi-row requests, like row scan and column slice.
- **Parallel processing framework**: runs batch jobs that use the application data as input source, process it, and materialize query-optimized data, e.g., OLAP cubes.
- **Query engine**: serves complex analytical queries, such as "Sales of iPads in all Apple stores in New York City during the week of Thanksgiving", from a query-optimized database.

Experiment Setups

Setups:

- CH1 = single-cluster setup with 8 servers where each server has Cassandra and Hadoop installed
- CH2 = dual-cluster setup with 6 dedicated Cassandra servers and 2 dedicated Hadoop servers
- **CH3 = dual-cluster setup with 4 dedicated Cassandra servers and 4 dedicated Hadoop servers**
- CH4 = dual-cluster setup with 2 dedicated Cassandra servers and 6 dedicated Hadoop servers



Each setup CH{2,3,4} is evaluated twice with:

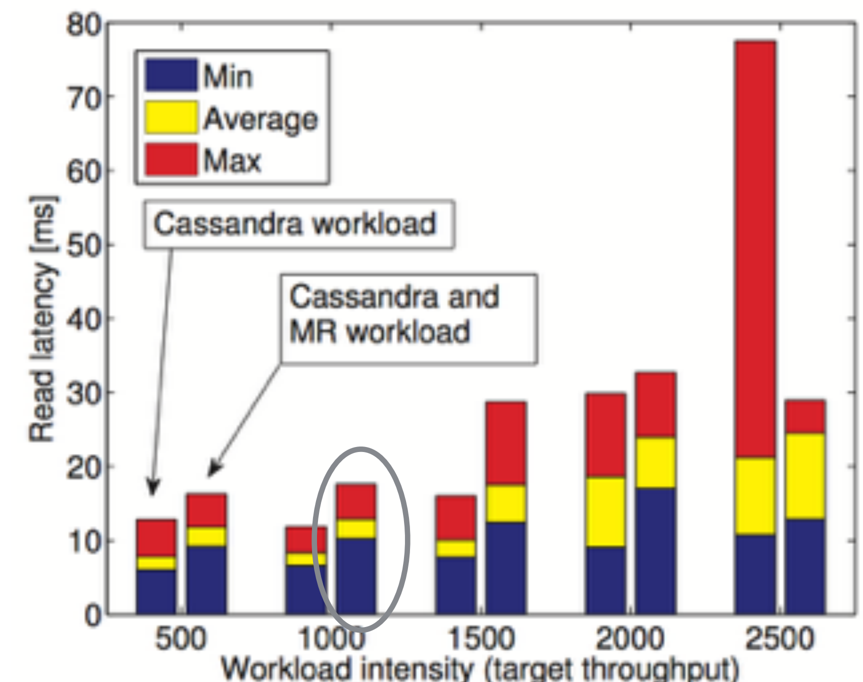
a. **Synchronous replication** between clusters (Cassandra Consistency Level "ALL")

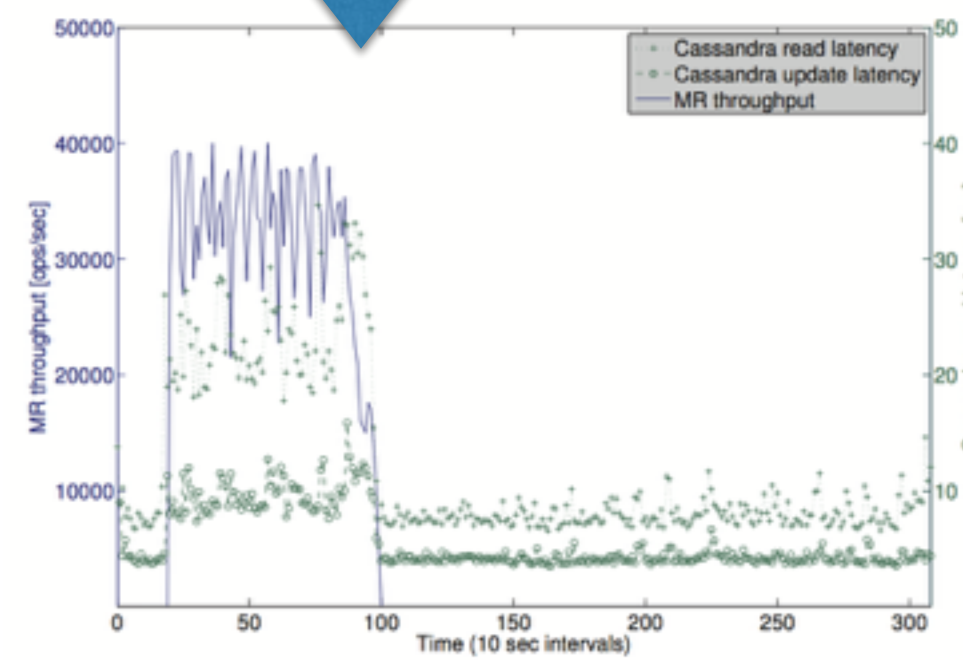
b. **Asynchronous replication** between clusters (Cassandra Consistency Level "QUORUM")

Experiment Results

The main results of our experiments are:

- We measure a **performance impact of Map Reduce jobs** on Cassandra read latency for nearly all experiments
- Dual-cluster setups: Cassandra scales nearly linearly with the number of servers
- Increasing capacity of the Hadoop cluster results in a nearly linear increase of throughput
- Interestingly, similar to the Cassandra scalability measurements, in the case of the shared Single-cluster CH1, Hadoop performance deteriorates and is slightly worse than in the CH4b dual-cluster setup.
- We calculate the **overhead of synchronous replication** for all three dual-cluster setups in terms of average read and write response time increase during the map-task. The performance impact computes to an average response time increase of approximately **35% for read requests** and approximately **50% for write requests**.





Technische Universität Berlin

ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Agenda

- Introduction
- Experiments
- **Experiment Automation**
- Trade-off Evaluation
- Conclusions

ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

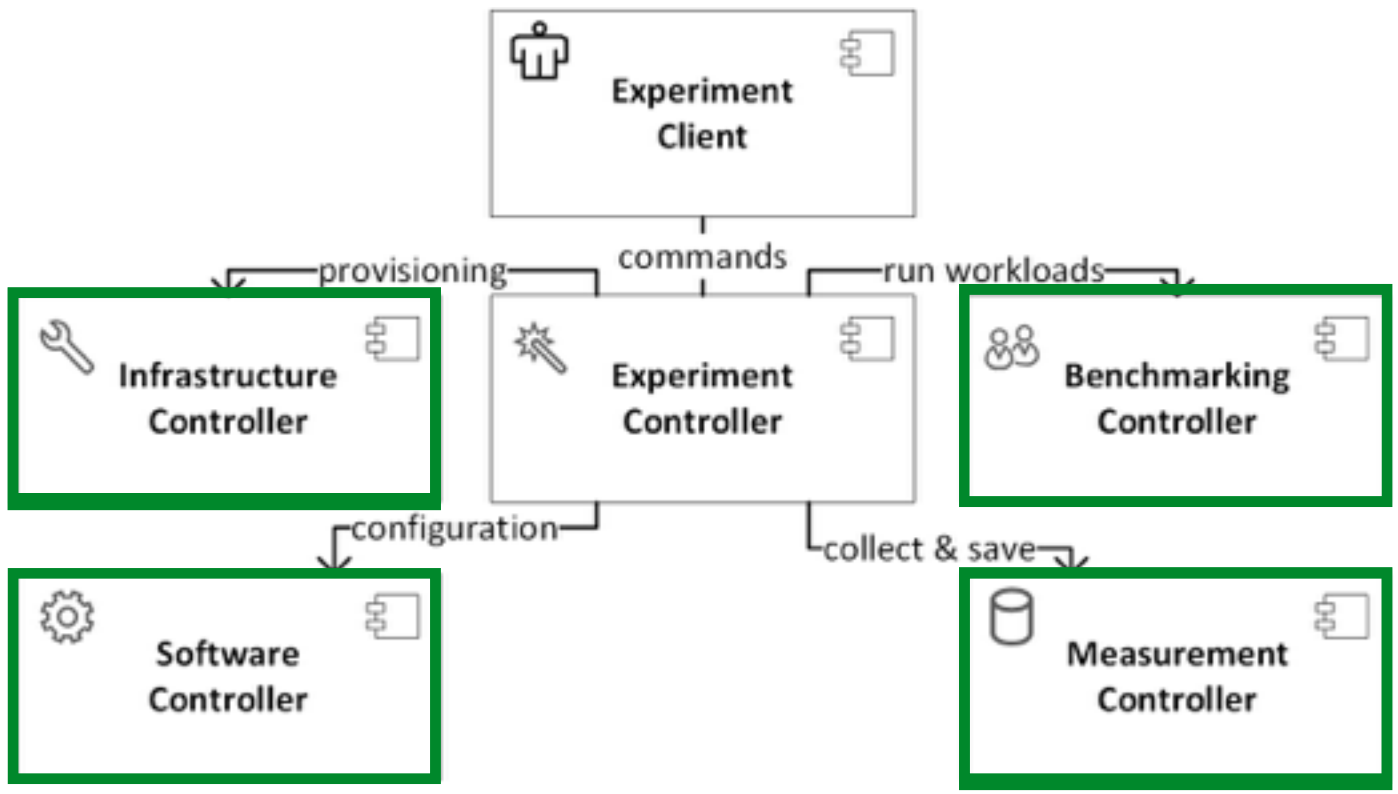
Experiment Automation

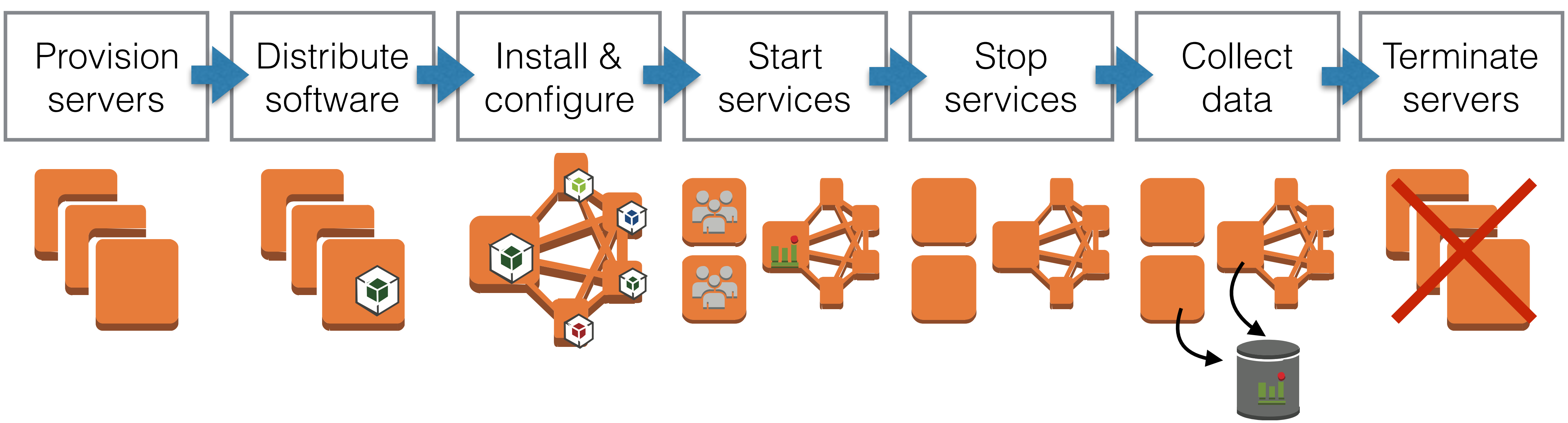
Our main challenges:

- Each experiment involves a complex, error-prone setup
- Difficulties to collaborate, repeat, and reproduce experiments
- Large numbers of experiments are needed to systematically evaluate
  - scalability and elasticity
  - infrastructure configuration alternatives: instance types, storage devices
  - system configuration alternatives: caching, replication, clustering, etc
  - different workloads

➡ Development of Elastic Lab, an automation system for experiments with distributed systems in the cloud

Design of Elastic Lab



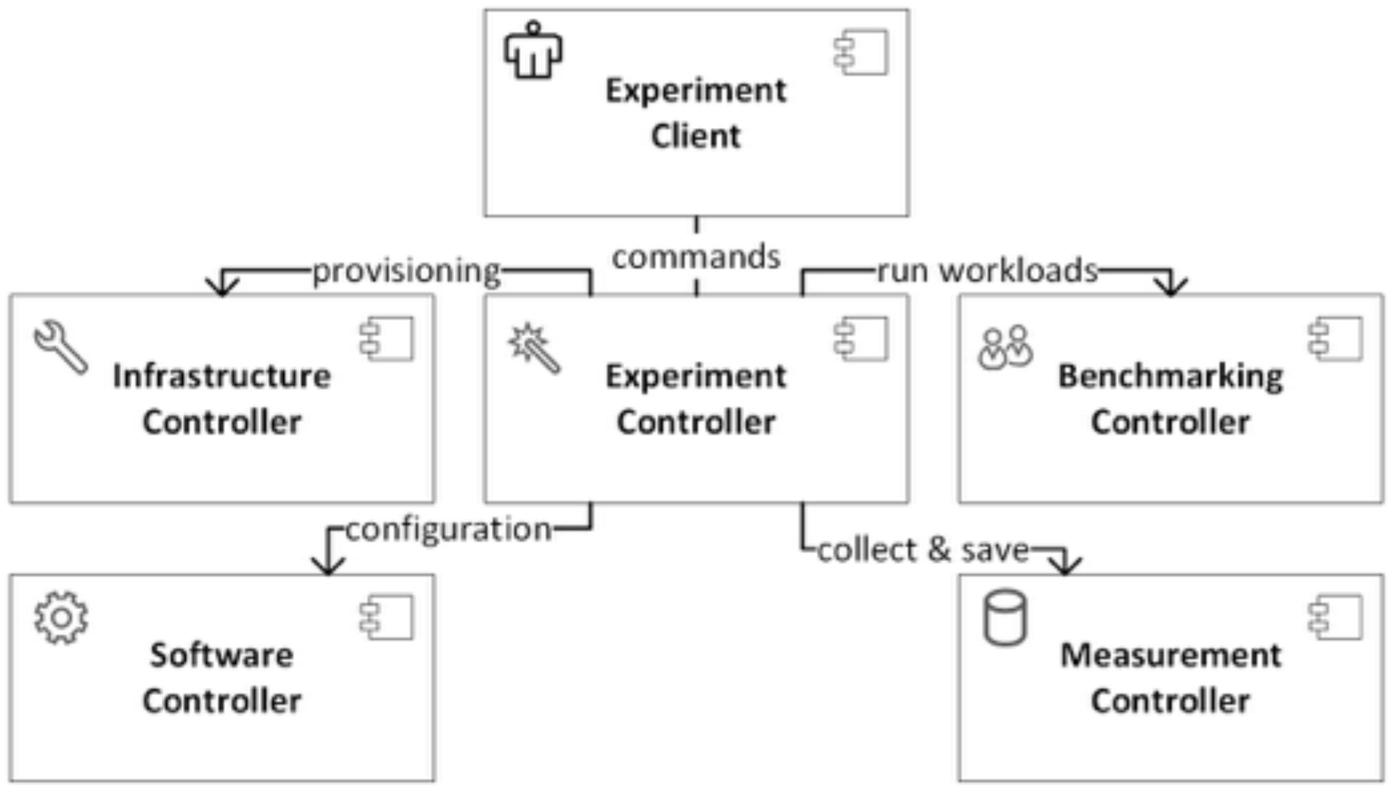


Related Experiment Automation Approaches & Systems

Related System	Main differences to Elastic Lab	Main similarities with Elastic Lab
Expo	• Grid infrastructure • Experiments with scientific simulation systems	
Weevil	• Grid infrastructure • Infrastructure provisioning not automated	• Experiments with distributed systems (Freenet, Chord)
Waif	• Experimental evaluation of file server performance with NFS workloads	• Cloud infrastructure (AWS)
CloudBench	• SUT is “hard-coded” as virtual appliance; software configurations not automated	• Experiments with distributed systems and applications on cloud infrastructure

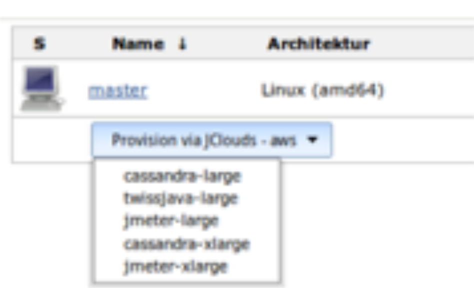
ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Design Alternatives: Experiment Client

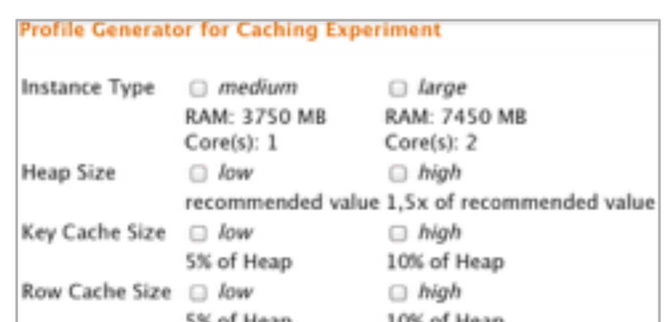


Experiment Automation Systems
Expo
Weevil
Waif
CloudBench
Elastic Lab
Elastic Lab 2.0
Elastic Lab CE

continuous



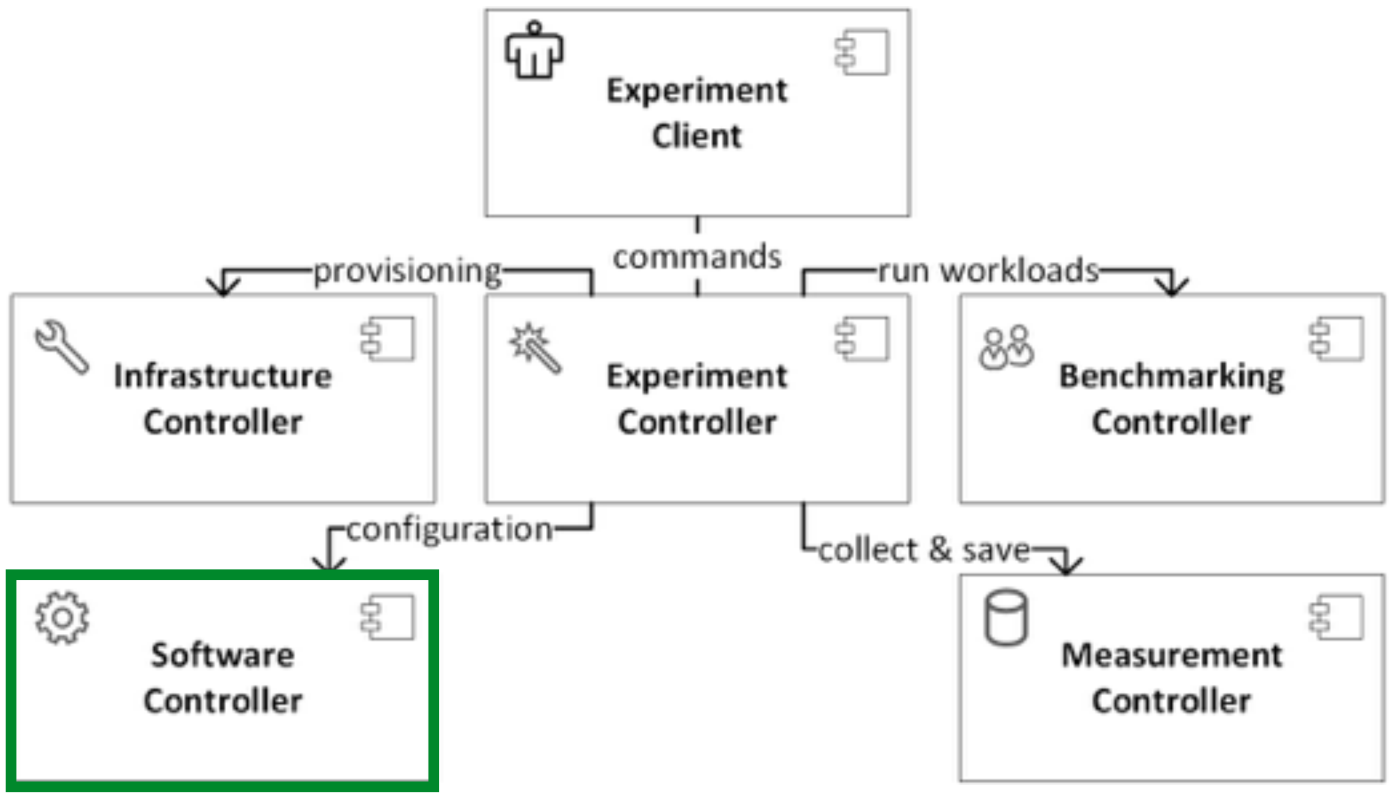
planned



interactive

ISEngineering
Wirtschaftsinformatik – Information Systems Engineering

Design Alternatives: Software Controller



Experiment Automation Systems
Expo
Weevil
Waif
CloudBench
Elastic Lab
Elastic Lab 2.0
Elastic Lab CE

script-based software automation

config. management based software automation

virtual appliance based software automation

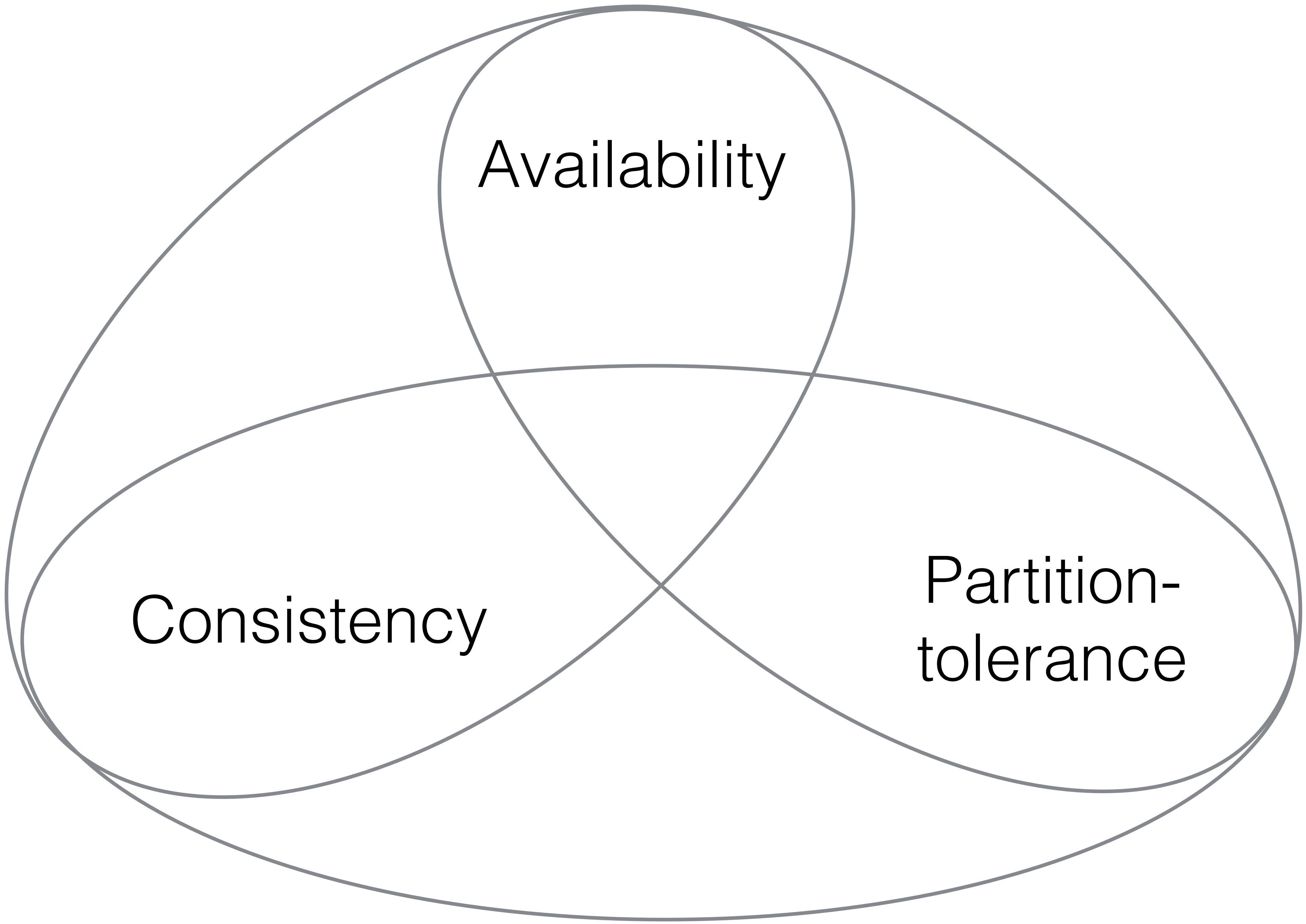
Agenda

Technische Universität Berlin

- Introduction
- Experiments
- Experiment Automation
- **Trade-off Evaluation**
- Conclusions

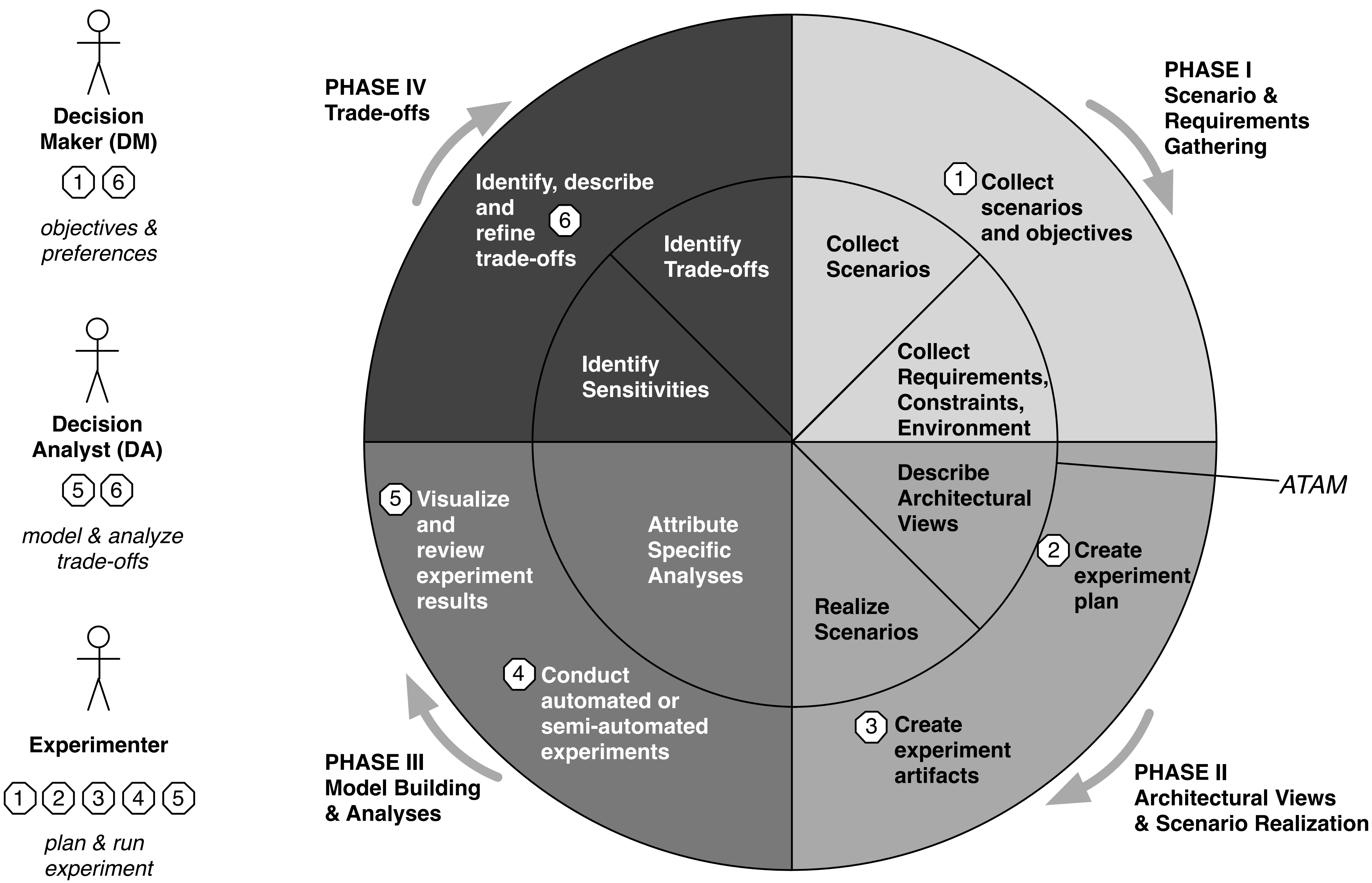
ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

CAP Trade-off



ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Trade-off Evaluation Approach



ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

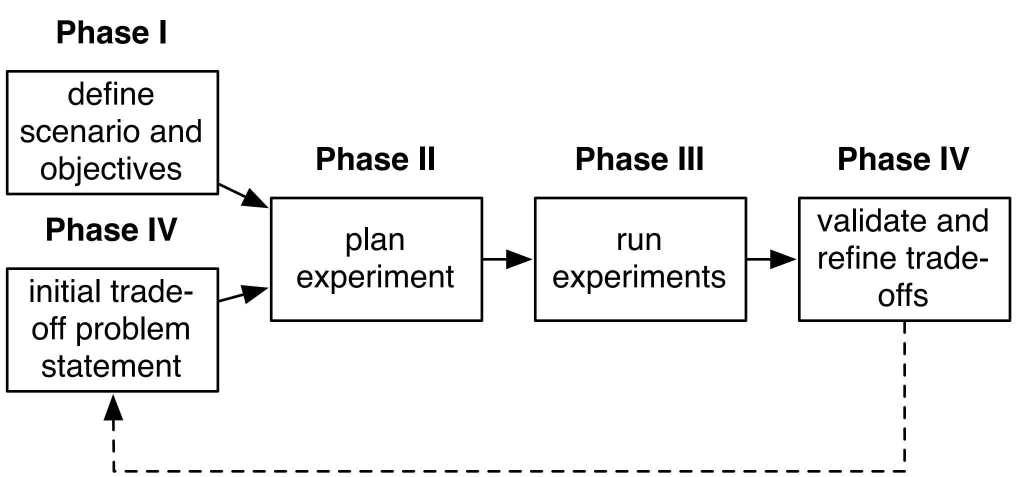
Experiment-Driven Trade-off Evaluation Method (ETEM)

Technische Universität Berlin

Two instantiations of ETEM

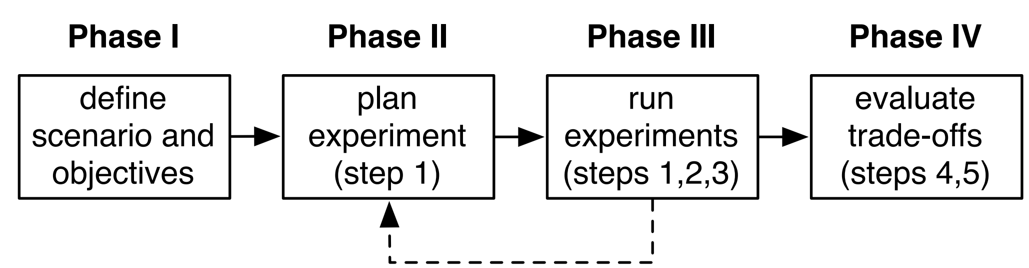
Trade-off Refinement Method

Describe a trade-off problem statement (in more detail)



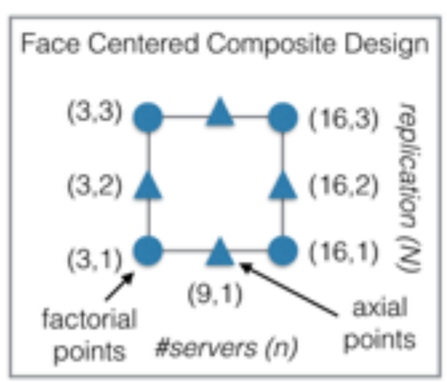
Experiment-Driven Multi-Objective Optimization Method

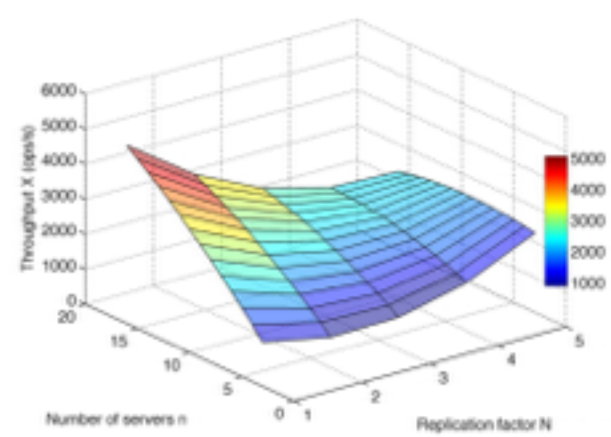
Find an optimally balanced solution between conflicting objectives, using both experimental data and subjective preferences as input



Experiment-Driven Multi-Objective Optimization Method

Phase I	Phase II	Phase III	Phase IV
define scenario and objectives	plan experiment (step 1)	run experiments (steps 1,2,3)	evaluate trade-offs (steps 4,5)





$$(\mathrm{MOP}) \begin{cases} \max_{\boldsymbol{x} \in \boldsymbol{X}} \quad \boldsymbol{f}(\boldsymbol{x}) \\ \boldsymbol{X} = \{\, \boldsymbol{x} \mid \boldsymbol{x} \in \mathbb{Z}_+^p \wedge \boldsymbol{g}(\boldsymbol{x}) \leq 0 \,\} \\ \boldsymbol{f}(\boldsymbol{x}) \in \mathbb{R}^m, \quad \boldsymbol{g}(\boldsymbol{x}) \in \mathbb{R}^h \end{cases}$$

ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Agenda

- Introduction
- Experiments
- Experiment Automation
- Trade-off Evaluation
- **Conclusions**

Conclusions

Technische Universität Berlin

1. Reproduction of related experiments
   - Successful reproduction of general performance, scalability and elasticity results
   - Absolute performance measurements are difficult to reproduce
2. Automation of experiments
   - Design alternatives of experiment automation systems determine the types of experiments that we can conduct with reasonable effort
   - Automated experiments enable us to
     - conduct a broad variety of performance, scalability, and elasticity benchmarking experiments
     - evaluate replication setups via automated parameter testing
3. Evaluation of trade-off problems
   - Experiments enable us to describe trade-off problems in more detail
   - Iterative system optimization methods enable balanced decisions by using a combination of experiment data and subjective preferences

Technische Universität Berlin

Thank you!

ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering