FITCH: Supporting Adaptive Replicated Services in the Cloud

Vinicius Cogo [1]
André Nogueira [1]
João Sousa [1]
Alysson Bessani [1]
Marcelo Pasin [2]
Hans P. Reiser [3]

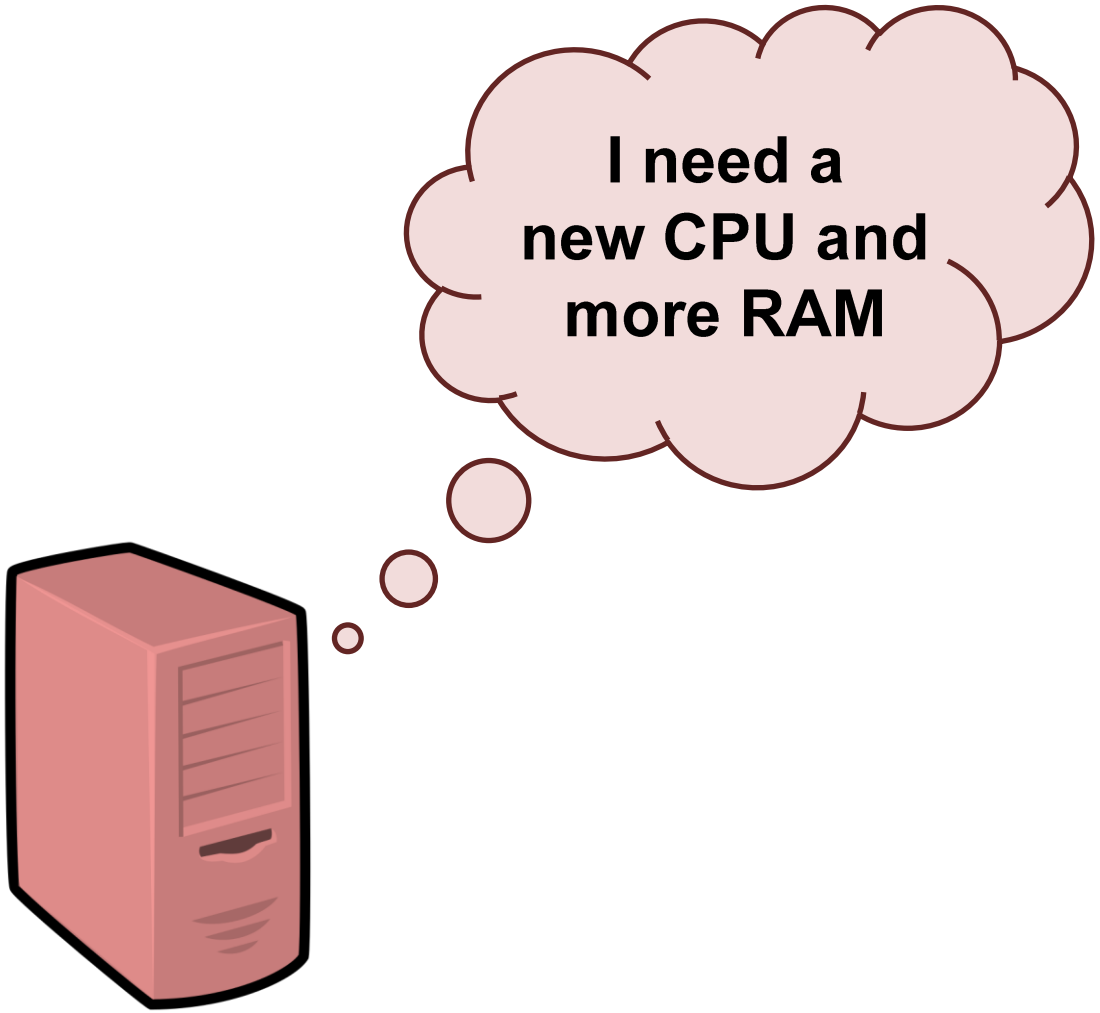
[1] University of Lisbon, Portugal

[2] University of Neuchatel, Switzerland

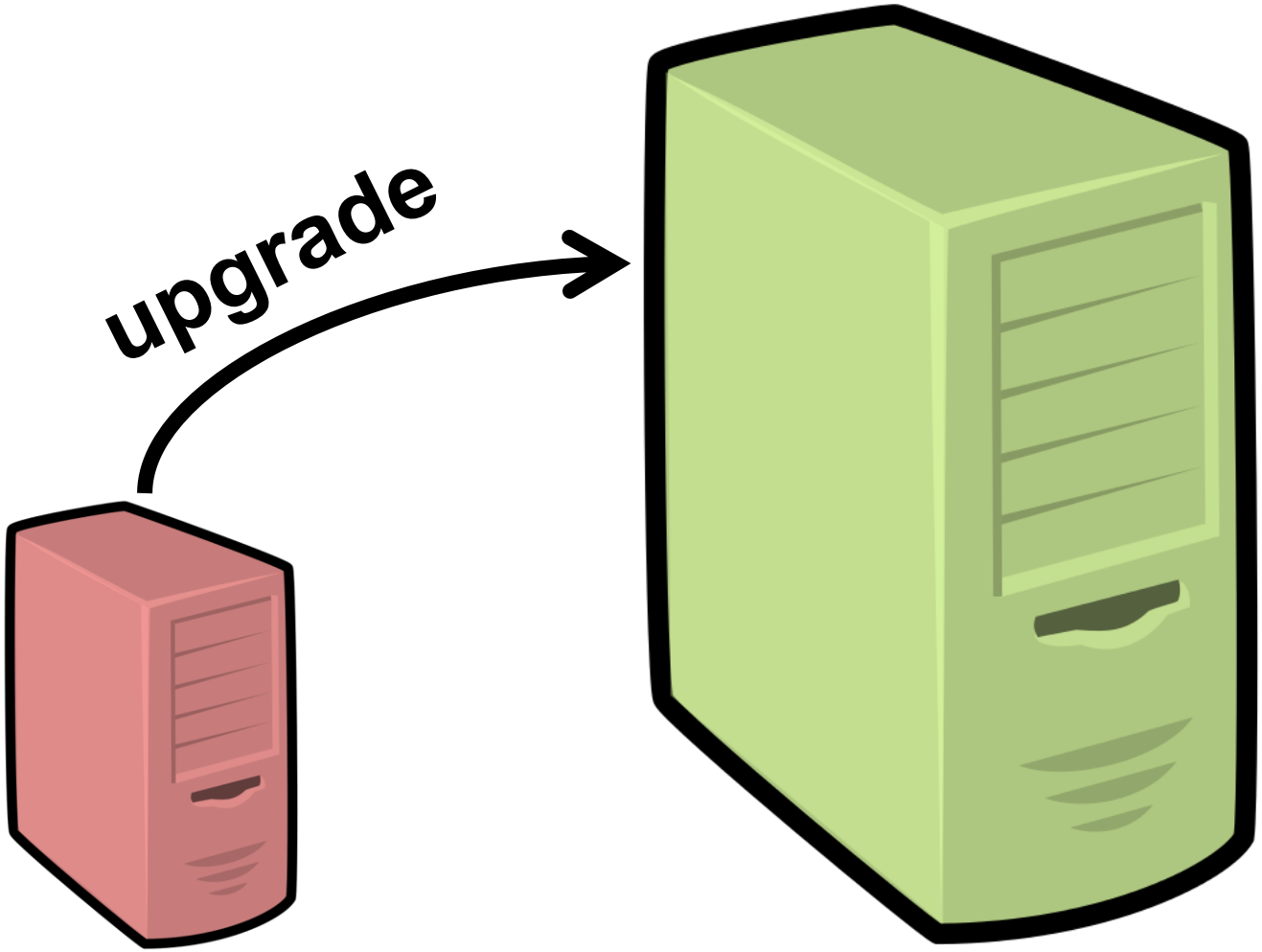
[3] University of Passau, Germany

What is an adaptation?

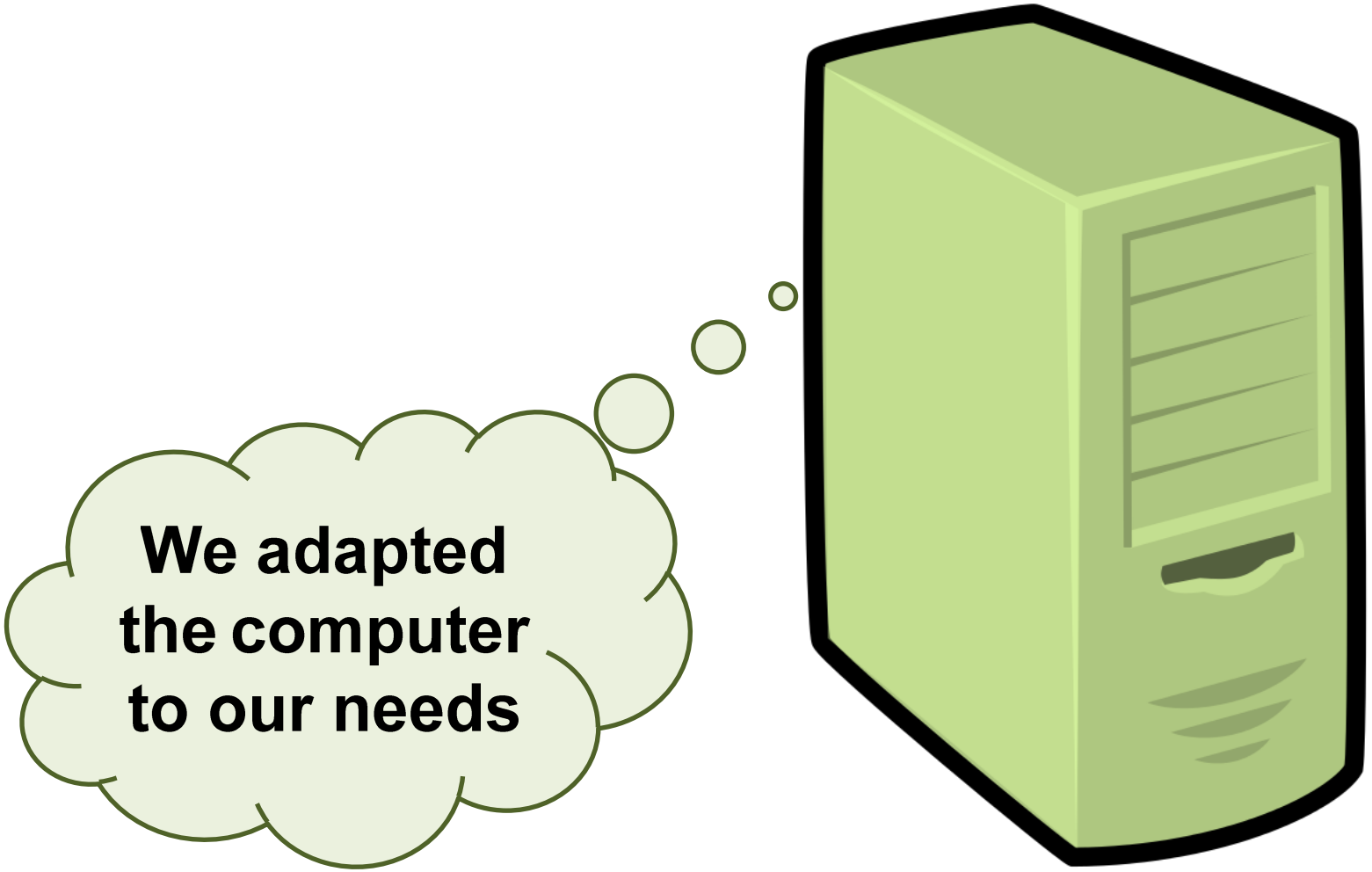
What is an adaptation?



What is an adaptation?

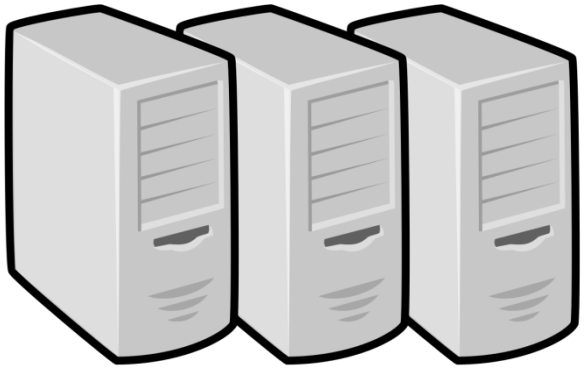


What is an adaptation?



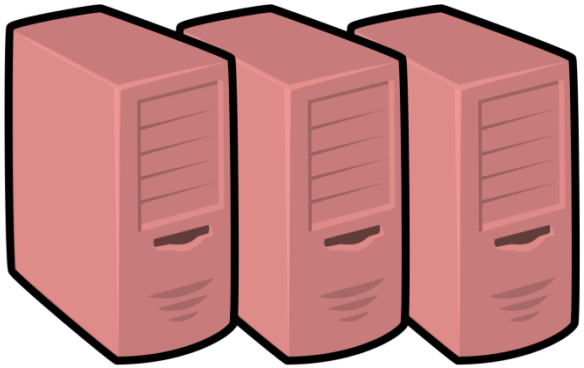
What is a service adaptation?

What is a service adaptation?



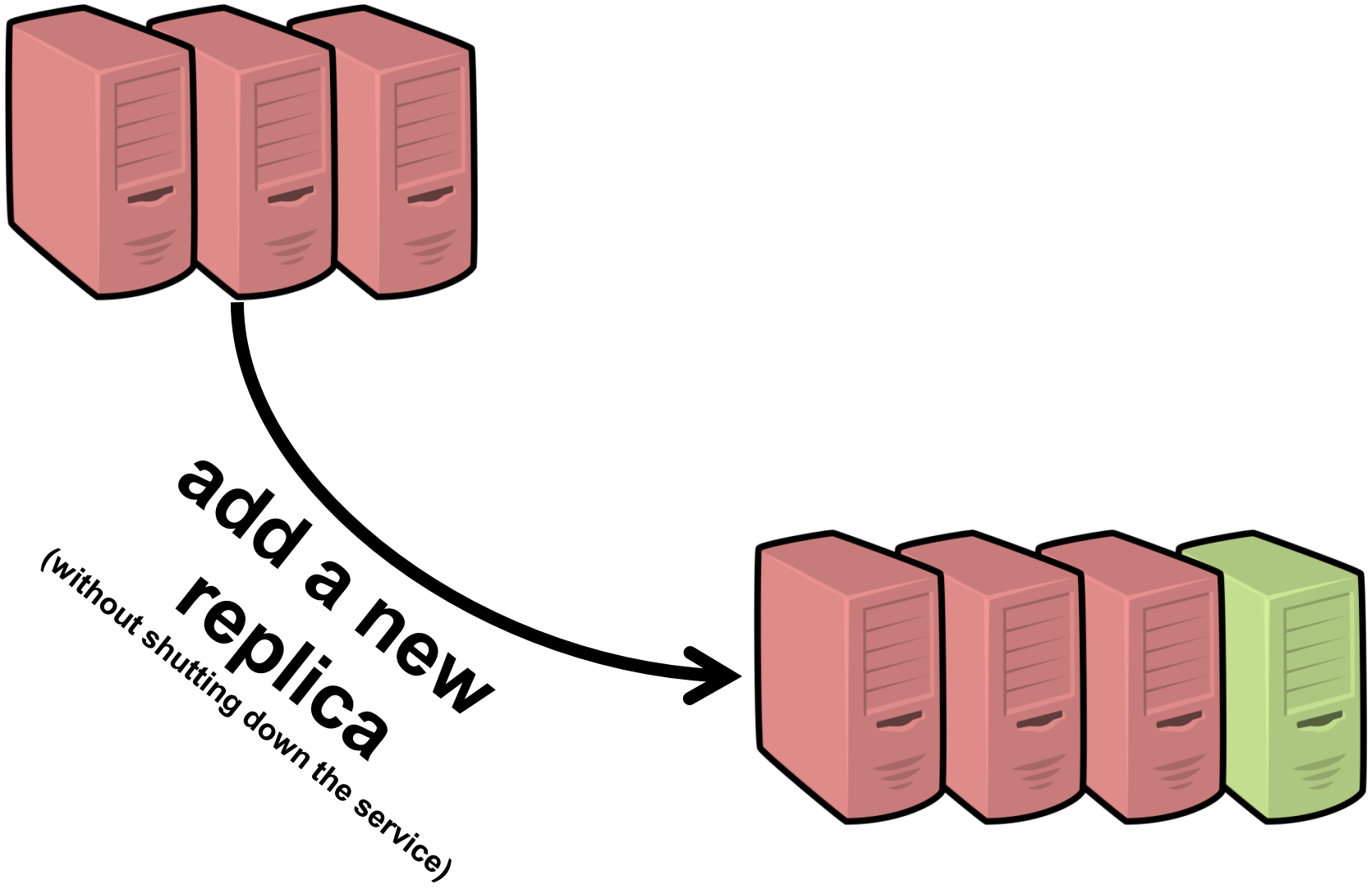
3 machines providing a service

What is a service adaptation?

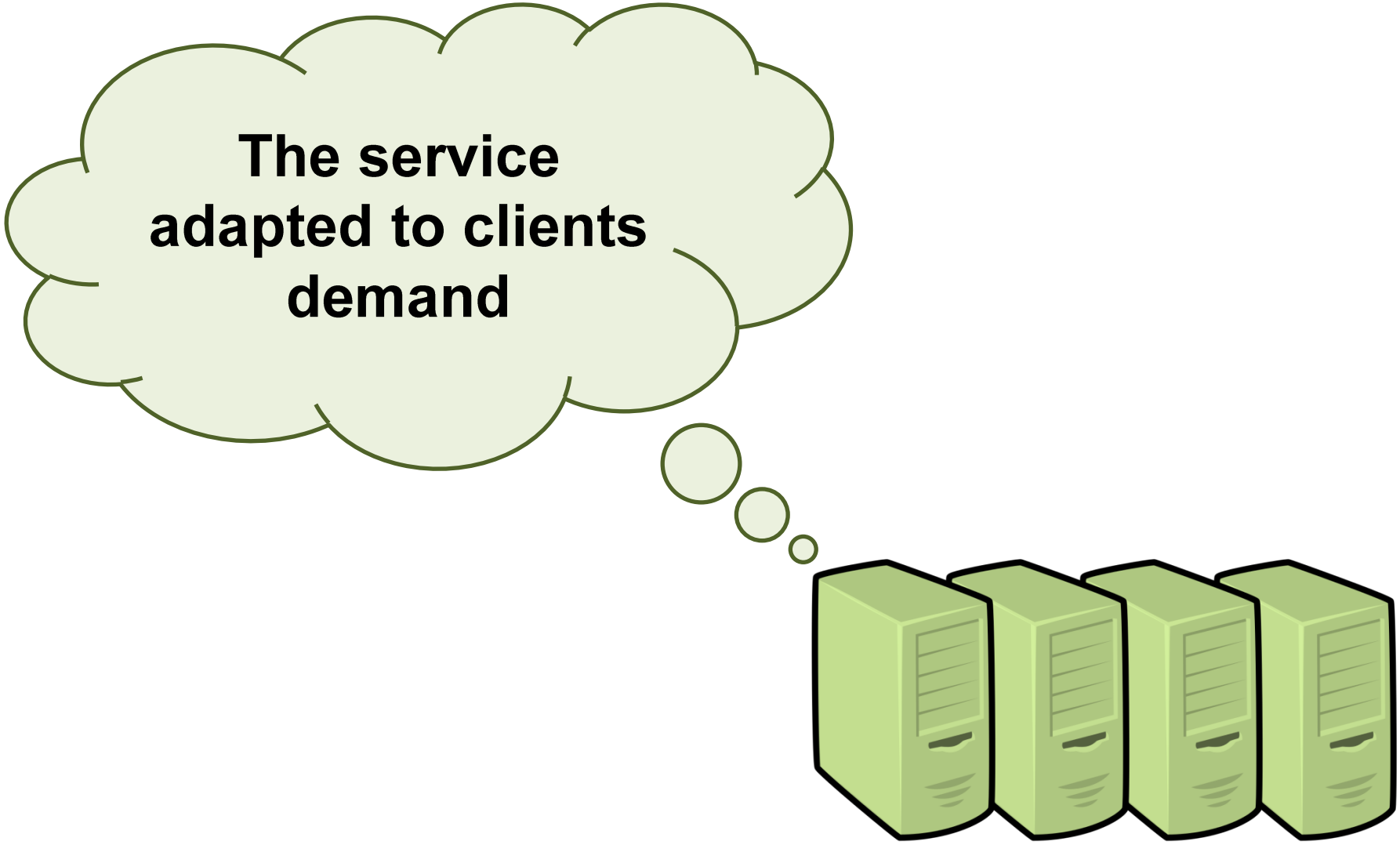


The service becomes overloaded

What is a service adaptation?



What is a service adaptation?



Motivation

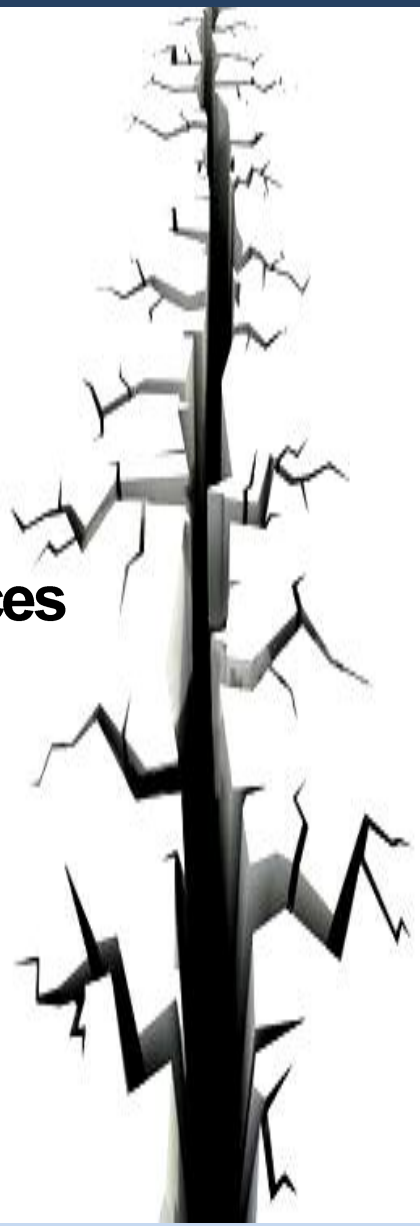
Dynamism
of
cloud computing

Dynamic
adaptation of
replicated services

Motivation

Dynamism
of
cloud computing

- "Infinite" elasticity
  - Create new instances
  - Destroy instances
- Different VM types
- Monitoring tools
- Pay-as-you-go model

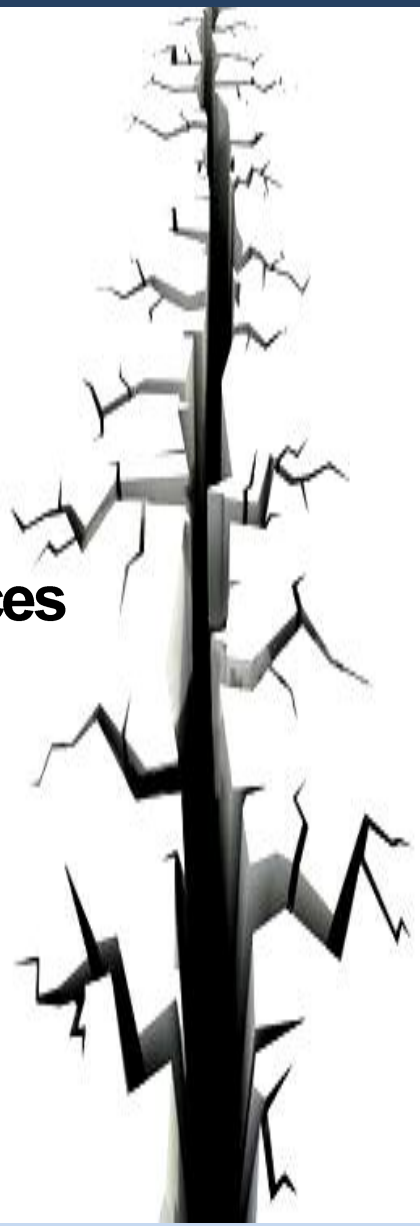


Dynamic
adaptation of
replicated services

Motivation

Dynamism
of cloud computing

- "Infinite" elasticity
  - Create new instances
  - Destroy instances
- Different VM types
- Monitoring tools
- Pay-as-you-go model

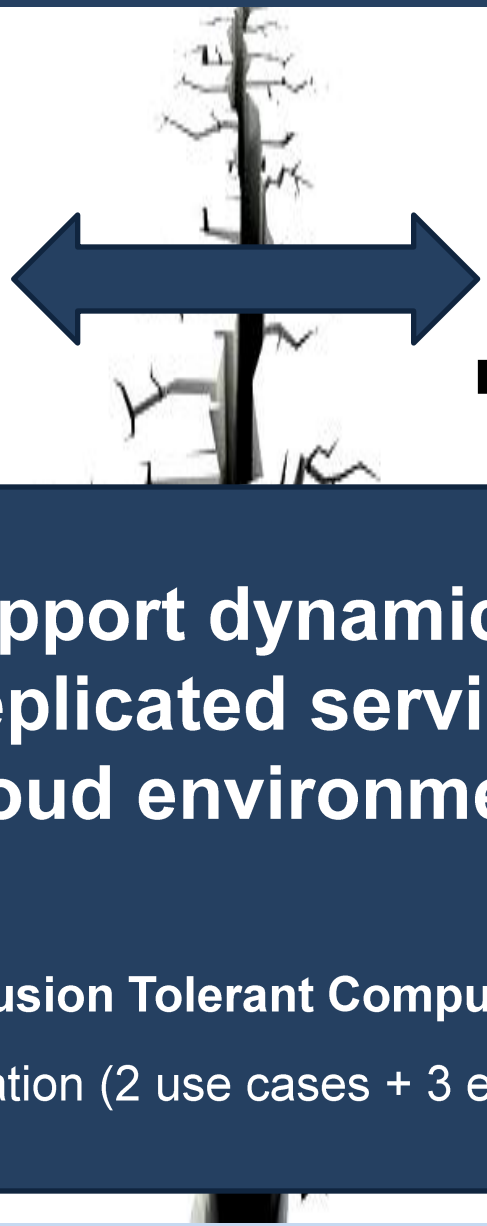


Dynamic
adaptation of replicated services

- Group membership
  - Insert new replicas
  - Remove replicas
- State transfer
- Adaptation heuristics

Motivation

Dynamism
of
cloud computing



Dynamic
adaptation of
replicated services

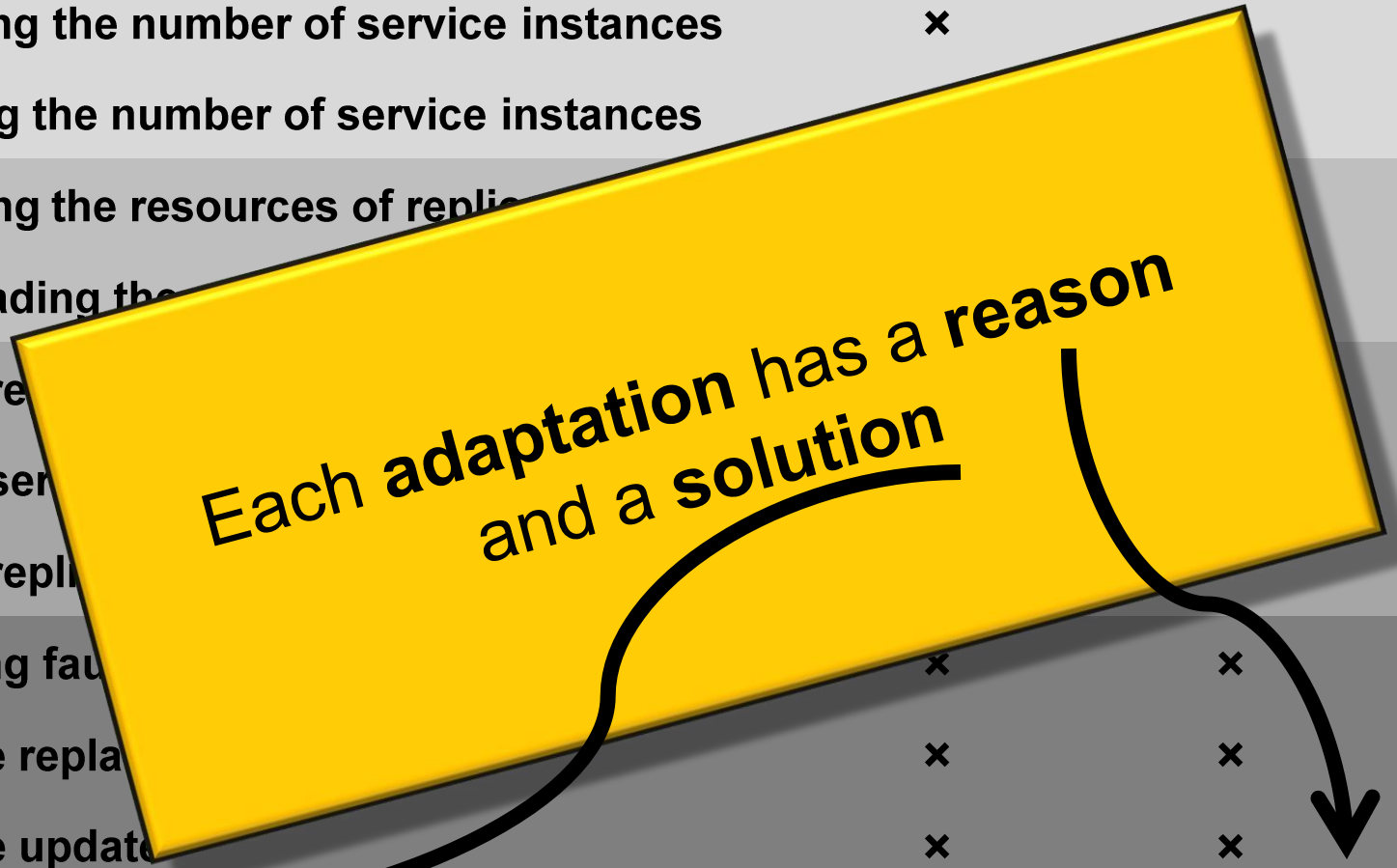
Our goal is to **support dynamic adaptation**
of **replicated services**
in **cloud environments**

Contributions:

1. **FITCH (Fault-and-Intrusion Tolerant Computing Hardpan)**
2. An experimental evaluation (2 use cases + 3 experiments)

2. Adaptations of replicated services

Adaptation solution	Performance	Economic	Security
Increasing the number of service instances	×		×
Reducing the number of service instances			
Upgrading the resources of repli			
Downgrading the			
Moving re			×
Moving ser			
Moving repli			×
Replacing fau	×	×	×
Software repla	×	×	×
Software updat	×	×	×
Replacing old service instances	×		×



2. Adaptations of replicated services

Adaptation solution	Performance	Economic	Security
Increasing the number of service instances	×		×
Reducing the number of service instances		×	
Upgrading the resources of replicas	×		
Downgrading the resources of replicas		×	
Moving replicas to different cloud providers	×	×	×
Moving service instances close to clients	×		
Moving replicas away from attackers	×		×
Replacing faulty replicas	×	×	×
Software replacement	×	×	×
Software update	×	×	×
Replacing old service instances	×		×

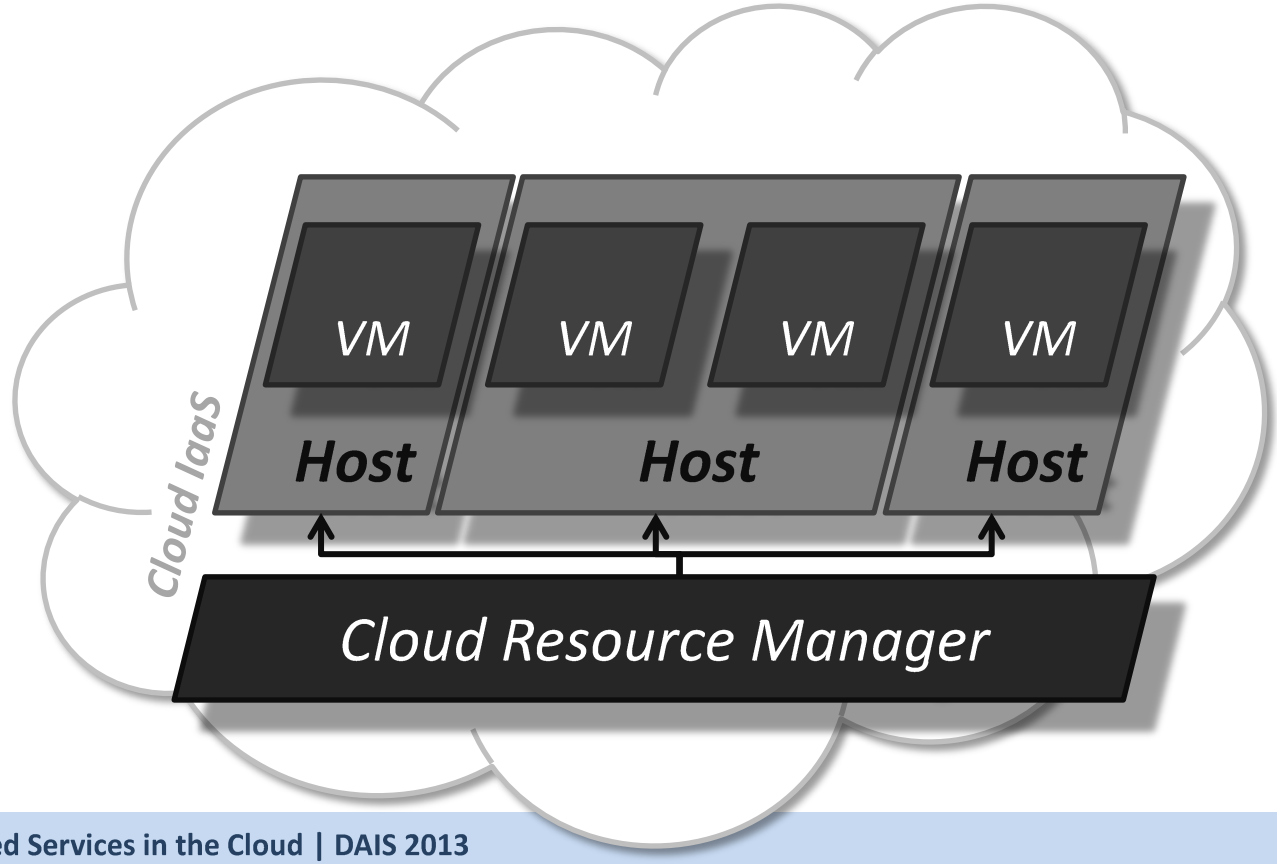
2. Adaptations of replicated services

Adaptation solution	Performance	Economic	Security
Increasing the number of service instances	×		×
Reducing the number of service instances			
Upgrading the resources of repli			
Downgrading the			
Moving re			×
Moving ser			
Moving repli			×
Replacing fau	×	×	×
Software repla	×	×	×
Software updat	×	×	×
Replacing old service instances	×		×

But **how** can we **perform** all these **adaptations** in a **cloud environment?**

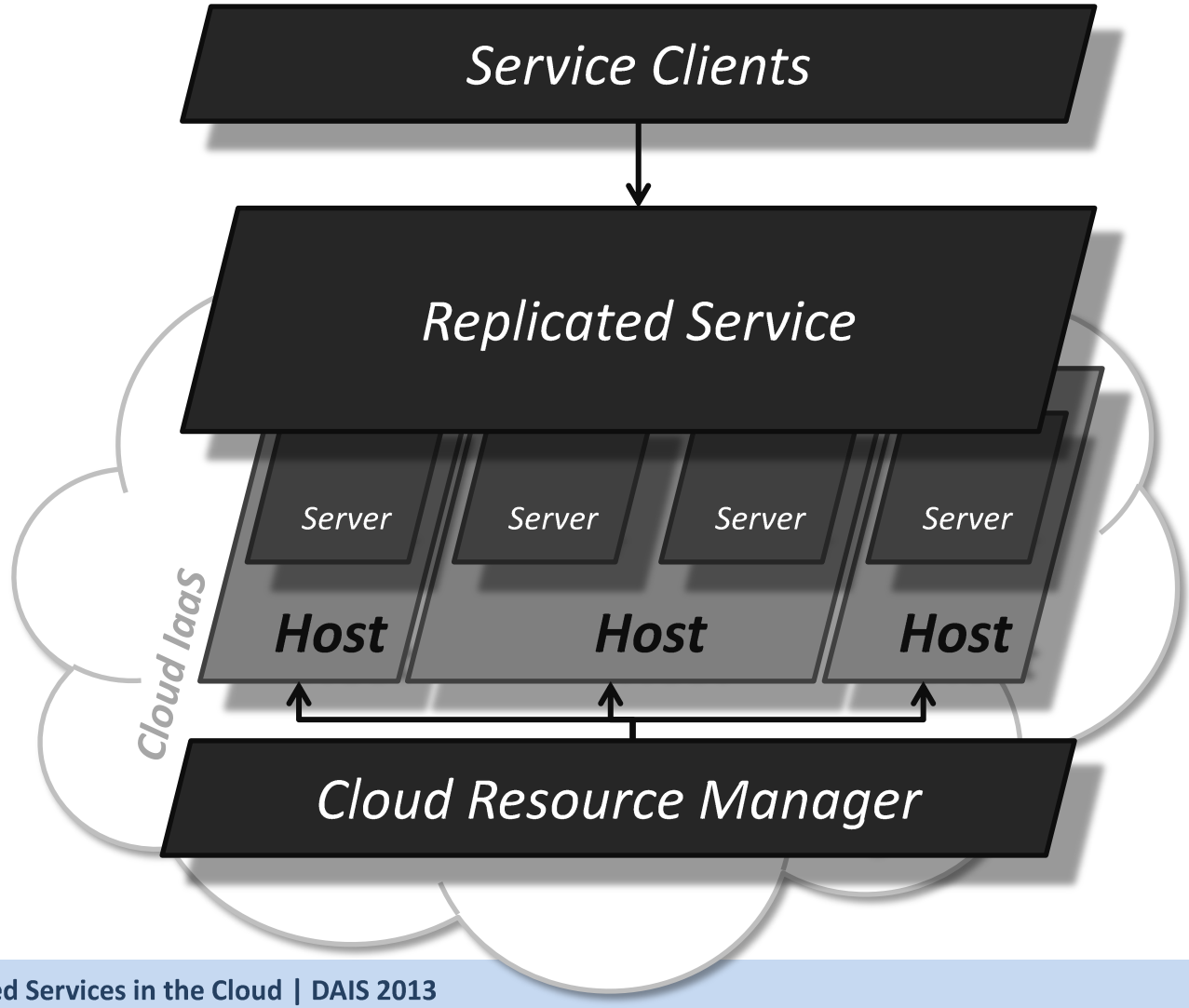
3. The FITCH architecture

Architectural overview



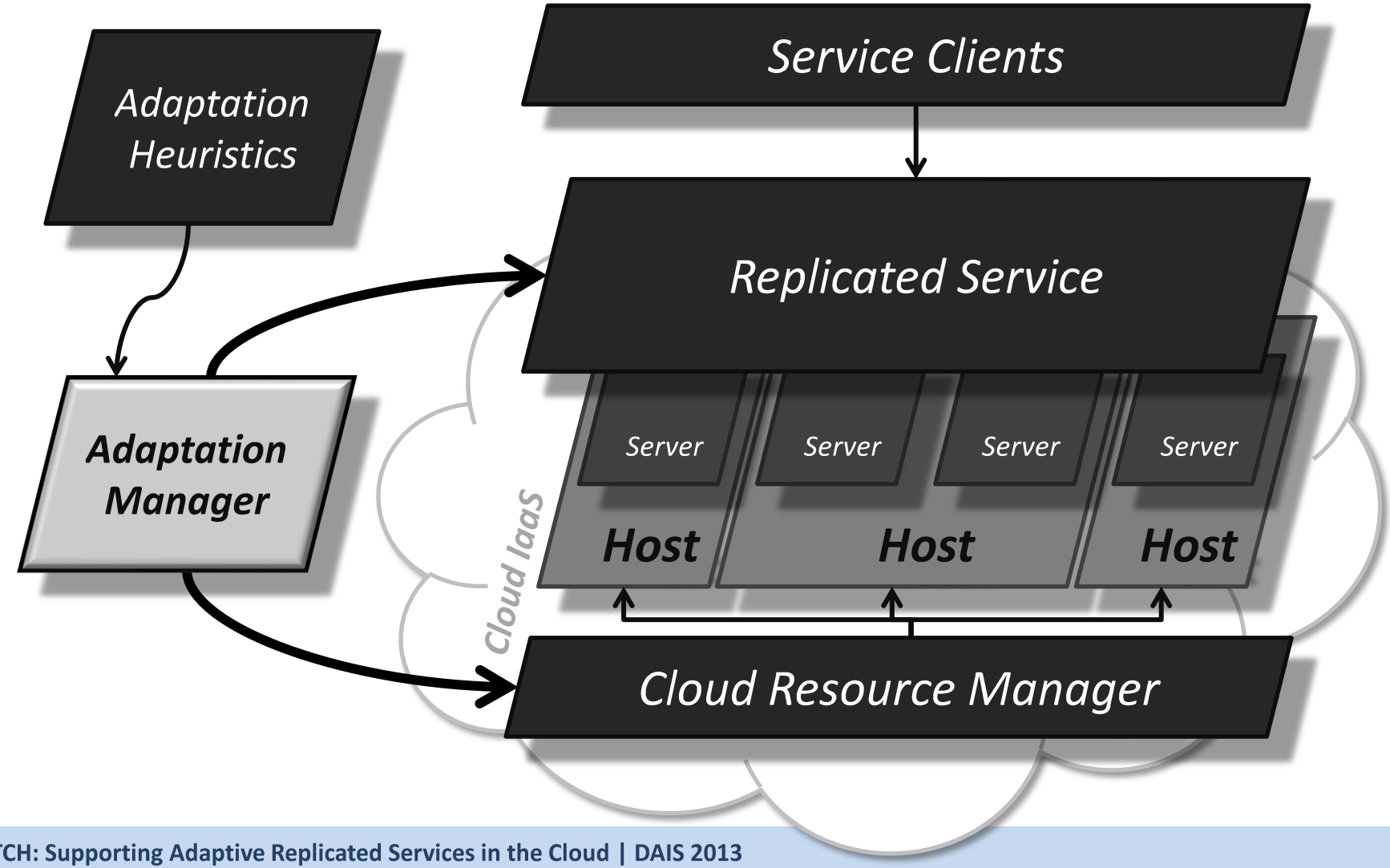
3. The FITCH architecture

Architectural overview



3. The FITCH architecture

Architectural overview



3. The FITCH architecture

System and threat models

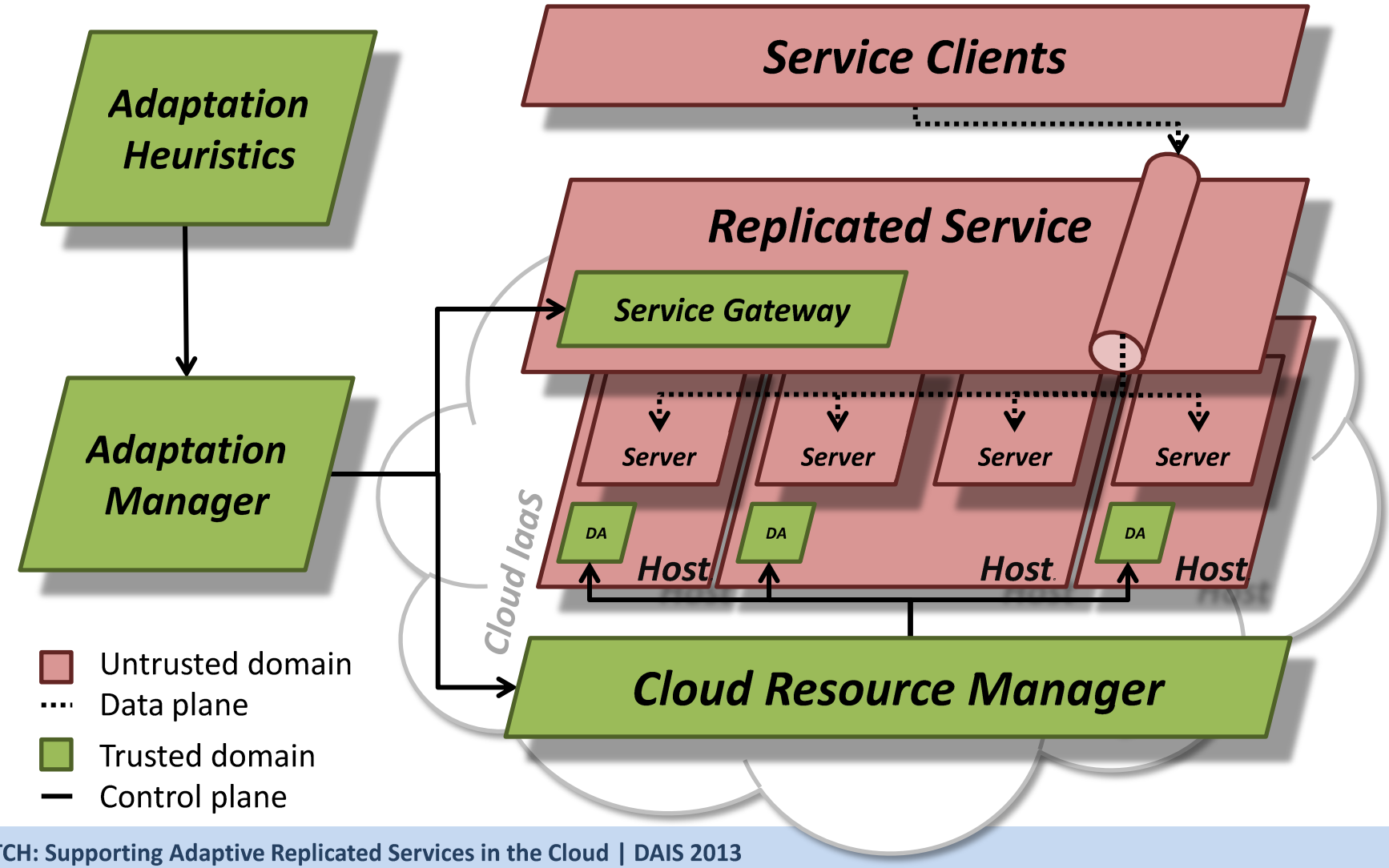
- Hybrid distributed system model

2 sub-systems

- Control plane (trusted)
  - Synchronous system model (bounded comp. and comm.)
  - Can be subject only to accidental faults (fail-stop)
- Data plane (untrusted)
  - Partially synchronous system model
  - Can be subject to Byzantine faults (arbitrary)

3. The FITCH architecture

Architectural components



3. The FITCH architecture

Service model

- **FITCH supports diverse replicated services on untrusted domain**

3. The FITCH architecture

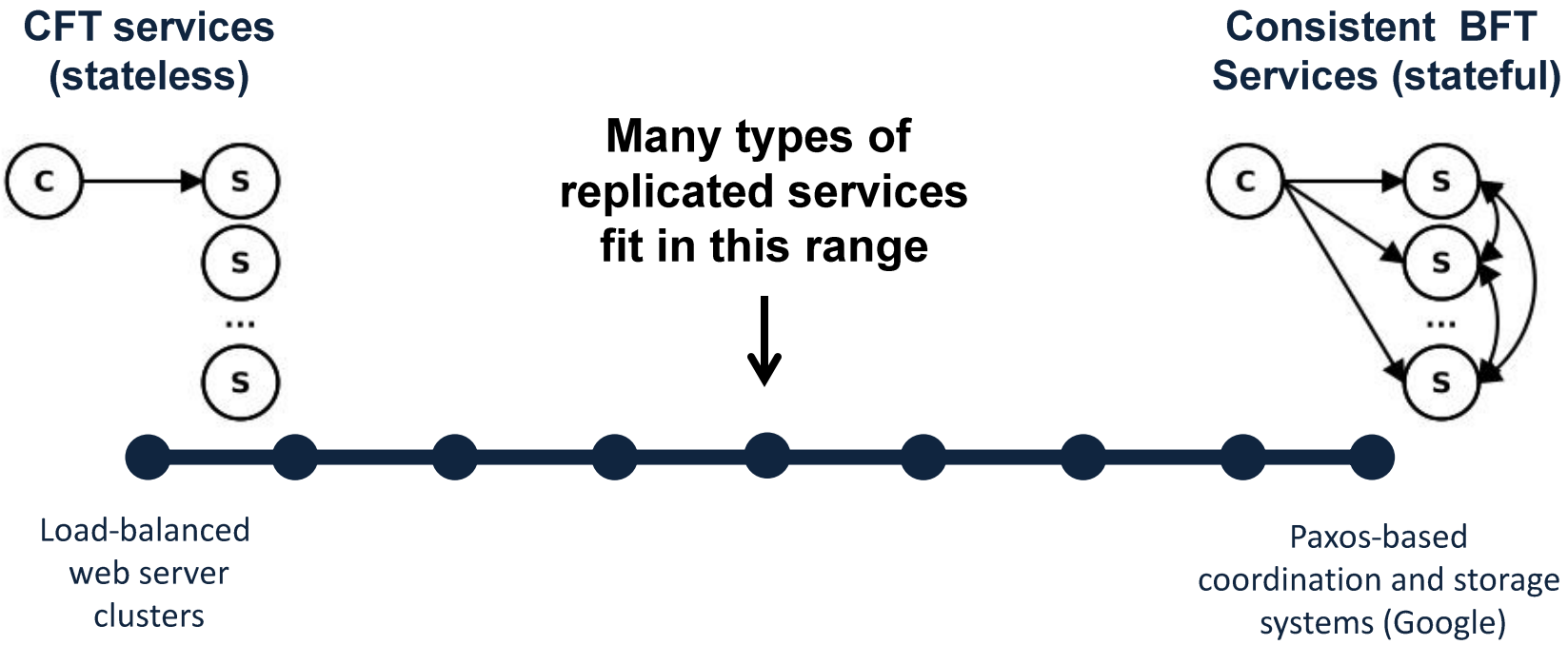
Service model

- **FITCH supports diverse replicated services on untrusted domain**

3. The FITCH architecture

Service model

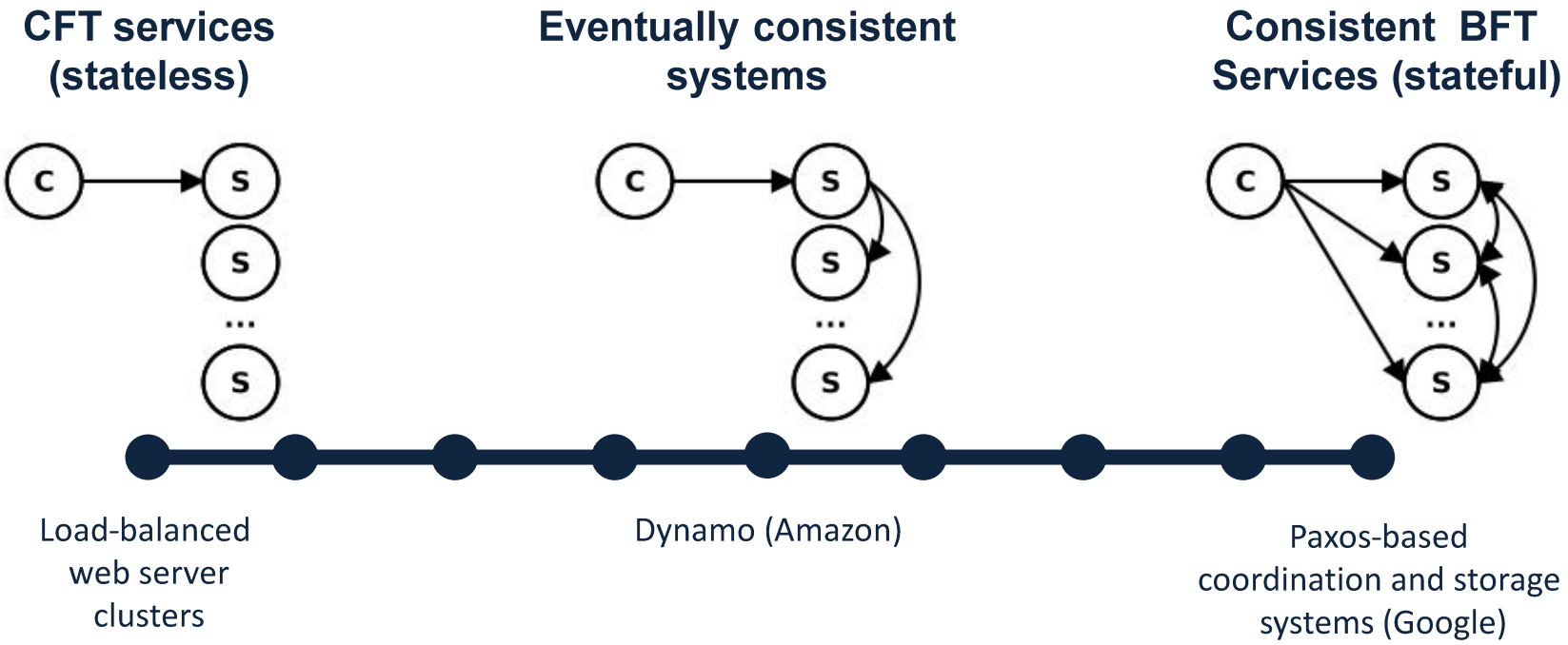
- **FITCH supports diverse replicated services on untrusted domain**



3. The FITCH architecture

Service model

- **FITCH supports diverse replicated services on untrusted domain**



3. The FITCH architecture

Service model

- **FITCH supports diverse replicated services on untrusted domain**

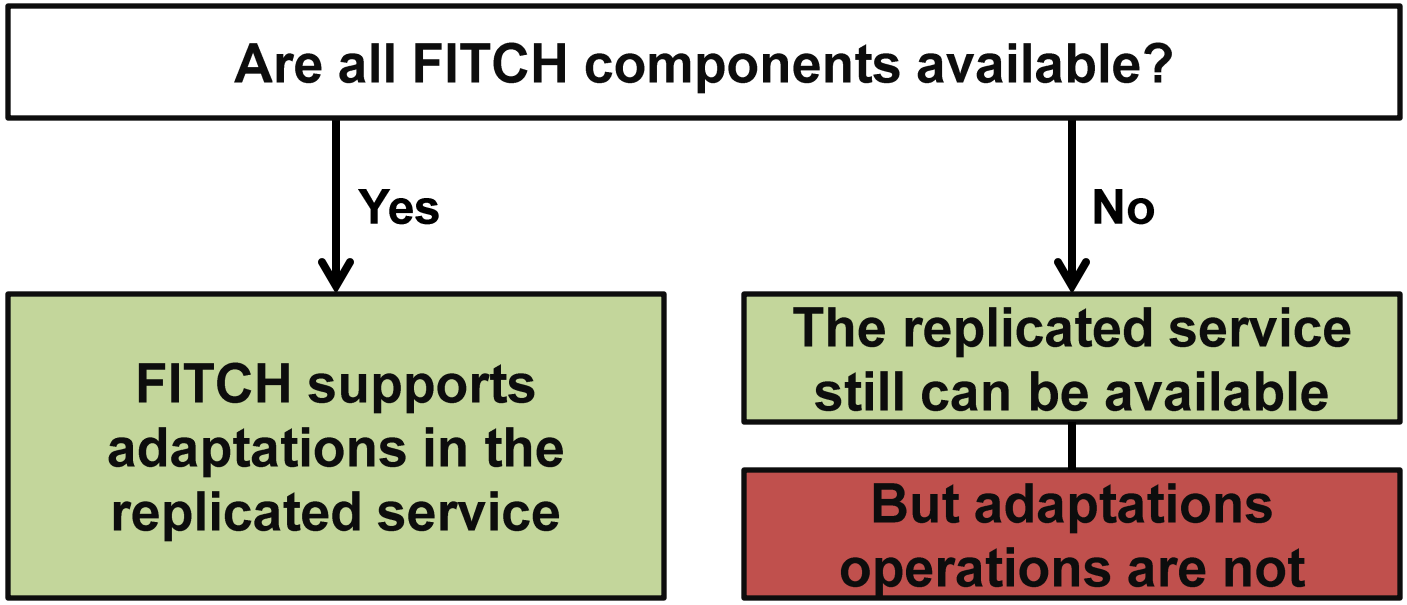
3. The FITCH architecture

Service adaptation

- 3 basic operations in service group membership

Add replica	Remove replica	Replace replica

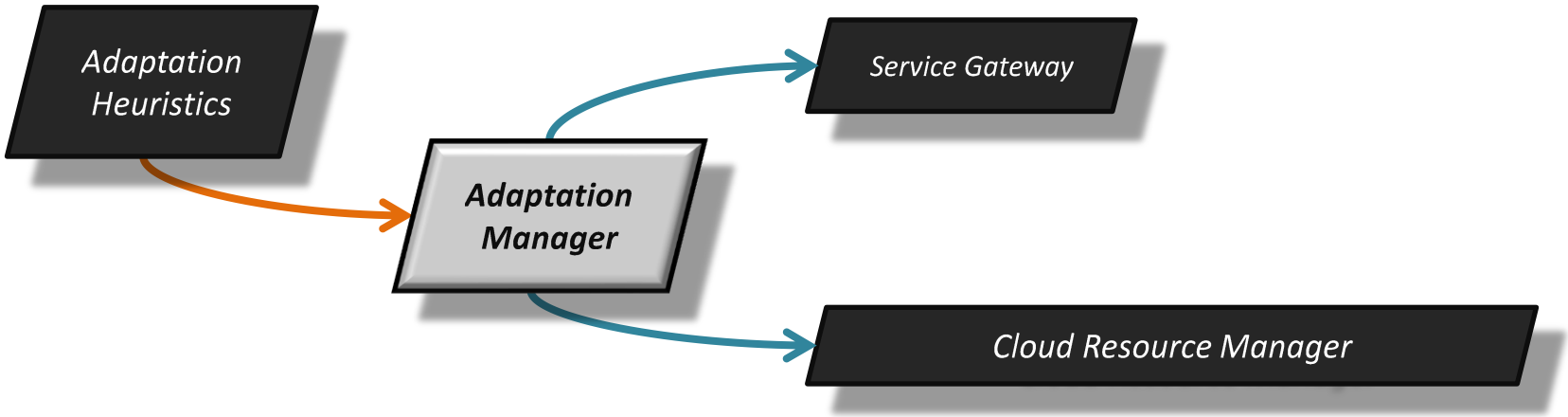
- And what about FITCH unavailability?



3. The FITCH architecture

Service adaptation algorithm

If replacing
- If adding
  1. Create a new VM R1
  2. Add R1 to the service group
- If removing
  3. Remove a replica R2 from the service group
  4. Destroy the VM of R2



4. Implementation

2 use cases

- **A CFT web service (stateless)**
  - Tolerates only **crash faults**
  - Service implemented by WS-Test using Java and Apache Tomcat
  - Clients access through a **LVS (Linux Virtual Server) load balancer**
  - **WS-Test benchmark**

- **A consistent BFT key-value store (stateful)**
  - Tolerates **Byzantine faults (arbitrary)**
  - Service implemented with **BFT-SMaRt** **(https://code.google.com/p/bft-smart/)**
  - Clients access through a **service lookup**
  - **YCSB benchmark**

5. Experimental evaluation

Experimental environment

Component	Qty.	Software	Hardware
Adaptation Manager	1	Java	**Dell PowerEdge 850**
Client (stateless)	5	WS-Test	1 core, 2.8 GHz
Cloud RM	3	OpenNebula	2 GB RAM
Client (stateful)	1	YCSB	**Dell PowerEdge R410**
Service Gateway	1	LVS (stateless) Tomcat (stateful)	8 cores, 2.27 GHz
Physical Cloud Hosts	6	Xen	32 GB RAM

5. Experimental evaluation

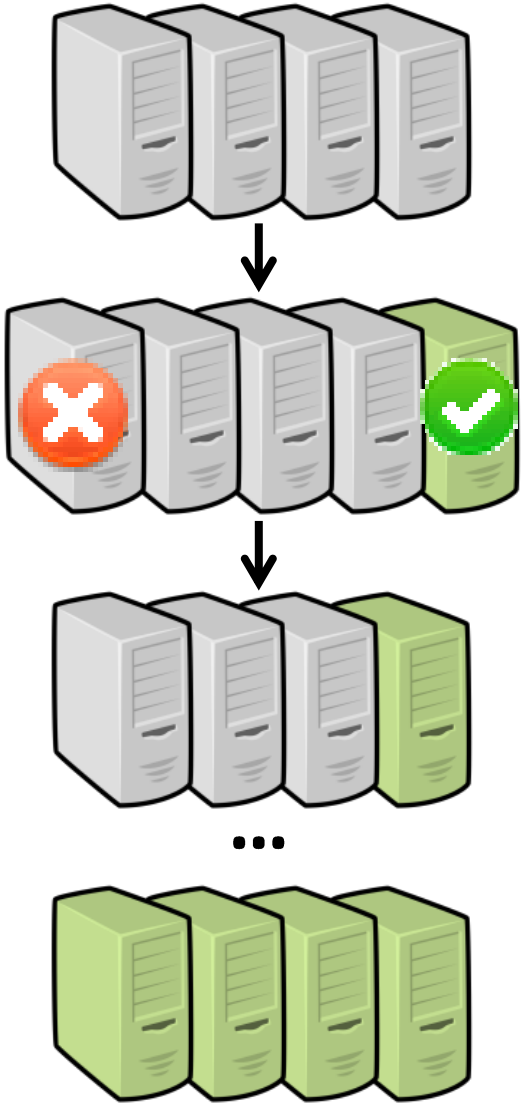
3 experiments

- Proactive recovery
- Scale-out and scale-in
- Scale-up and scale-down

5. Experimental evaluation

1st experiment: Proactive recovery

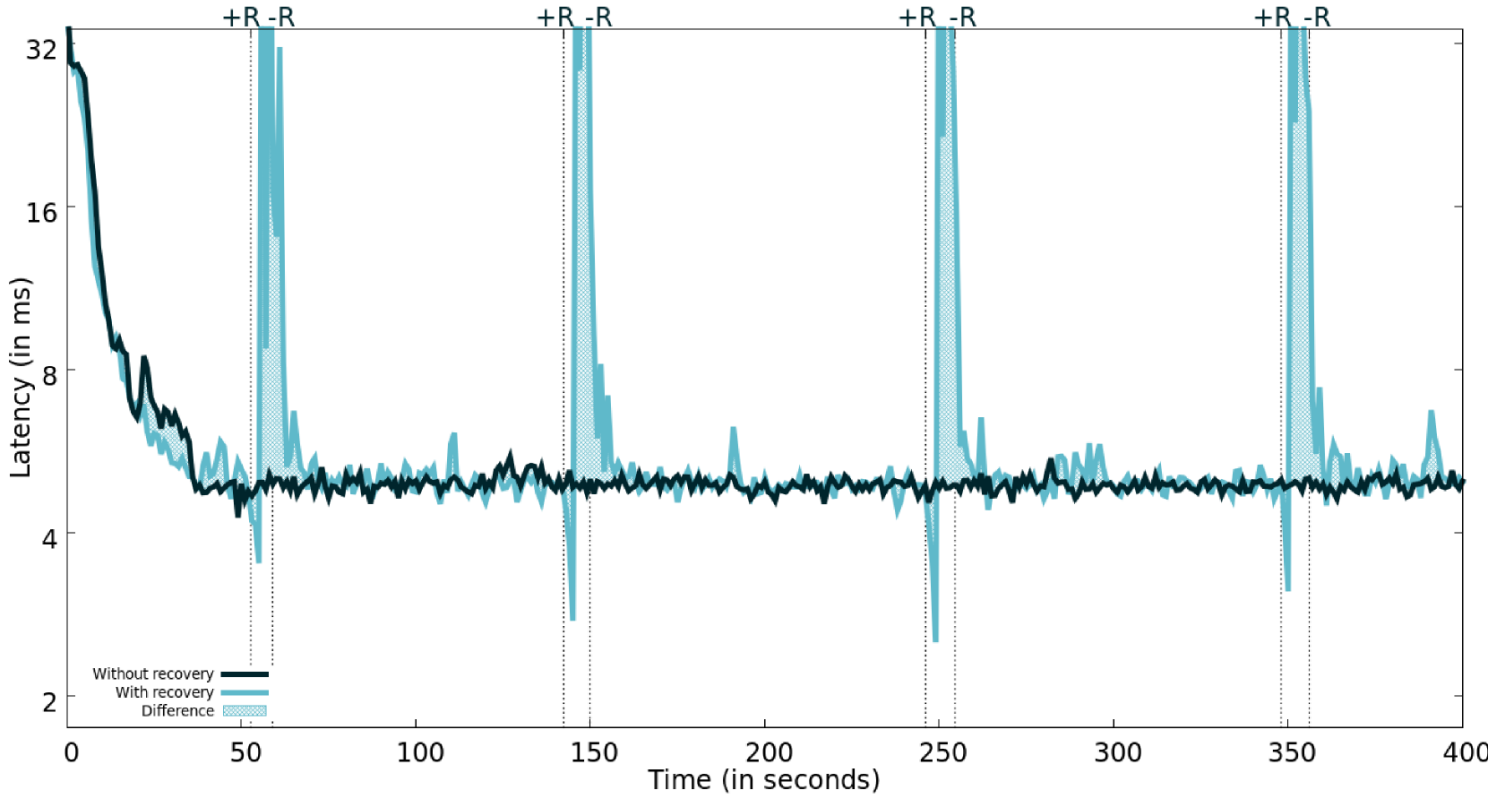
- **Recover** the entire **group** of replicas **proactively** (**one replica at a time**)
- **Recover** the entire service repeatedly **as soon as possible** (worst scenario)



5. Experimental evaluation

Proactive recovery

Impact on a CFT web service (stateless)



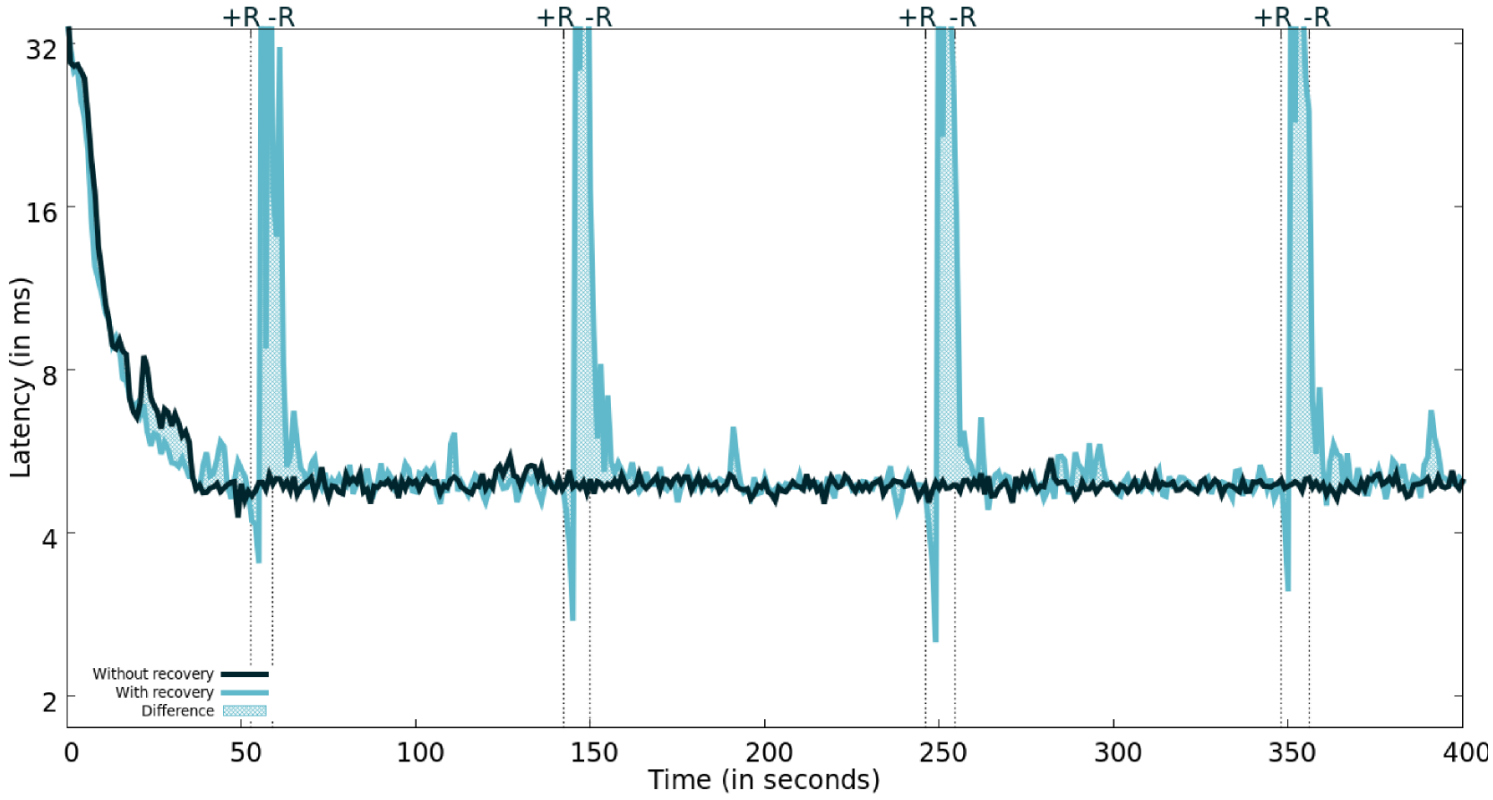
5. Experimental evaluation

4
replicas

4
adaptations

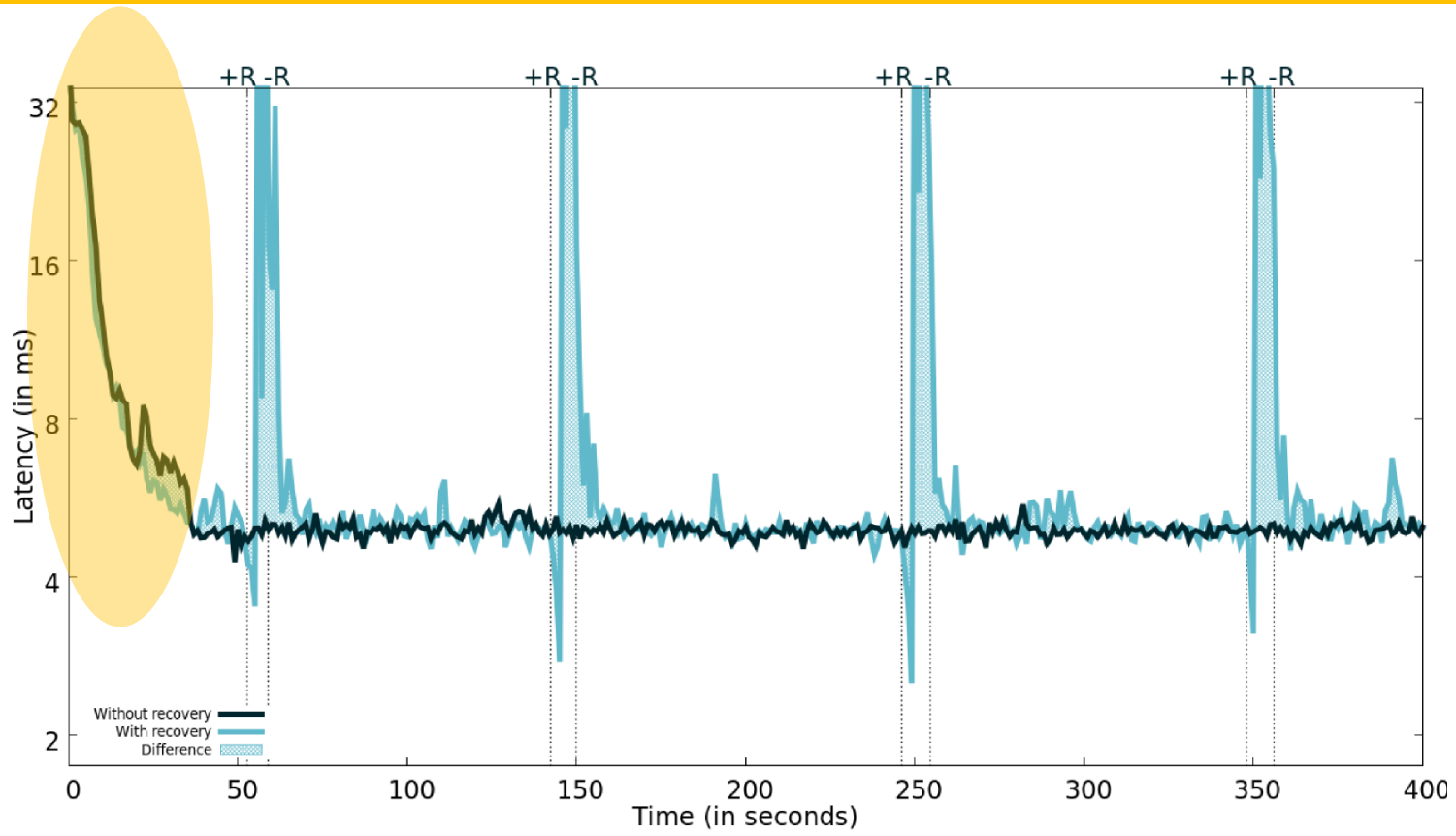
6 min
experiment time

90 s
each adaptation time



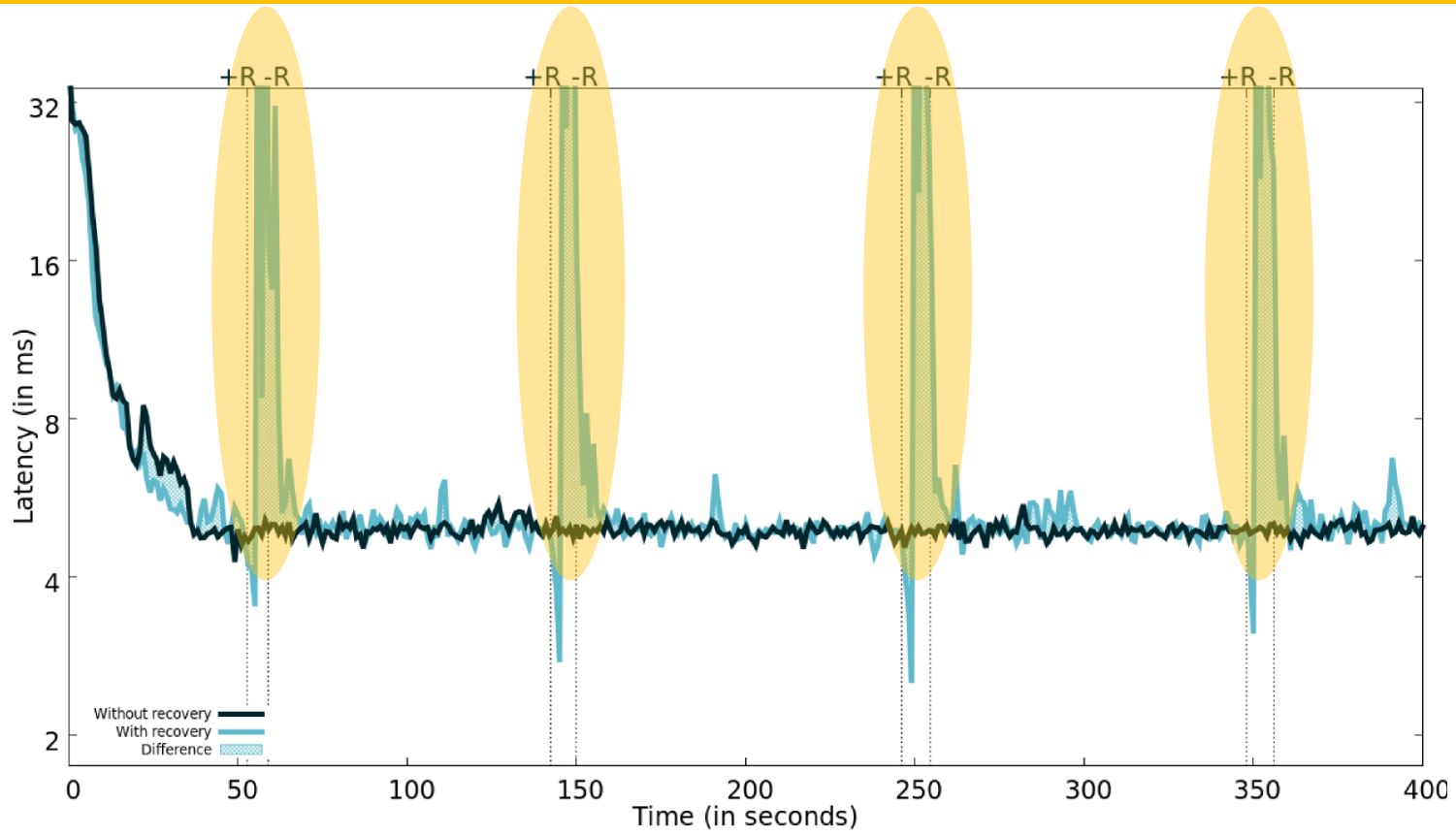
Warm-up (30 s)

(Apache Tomcat loading servlet + JIT)



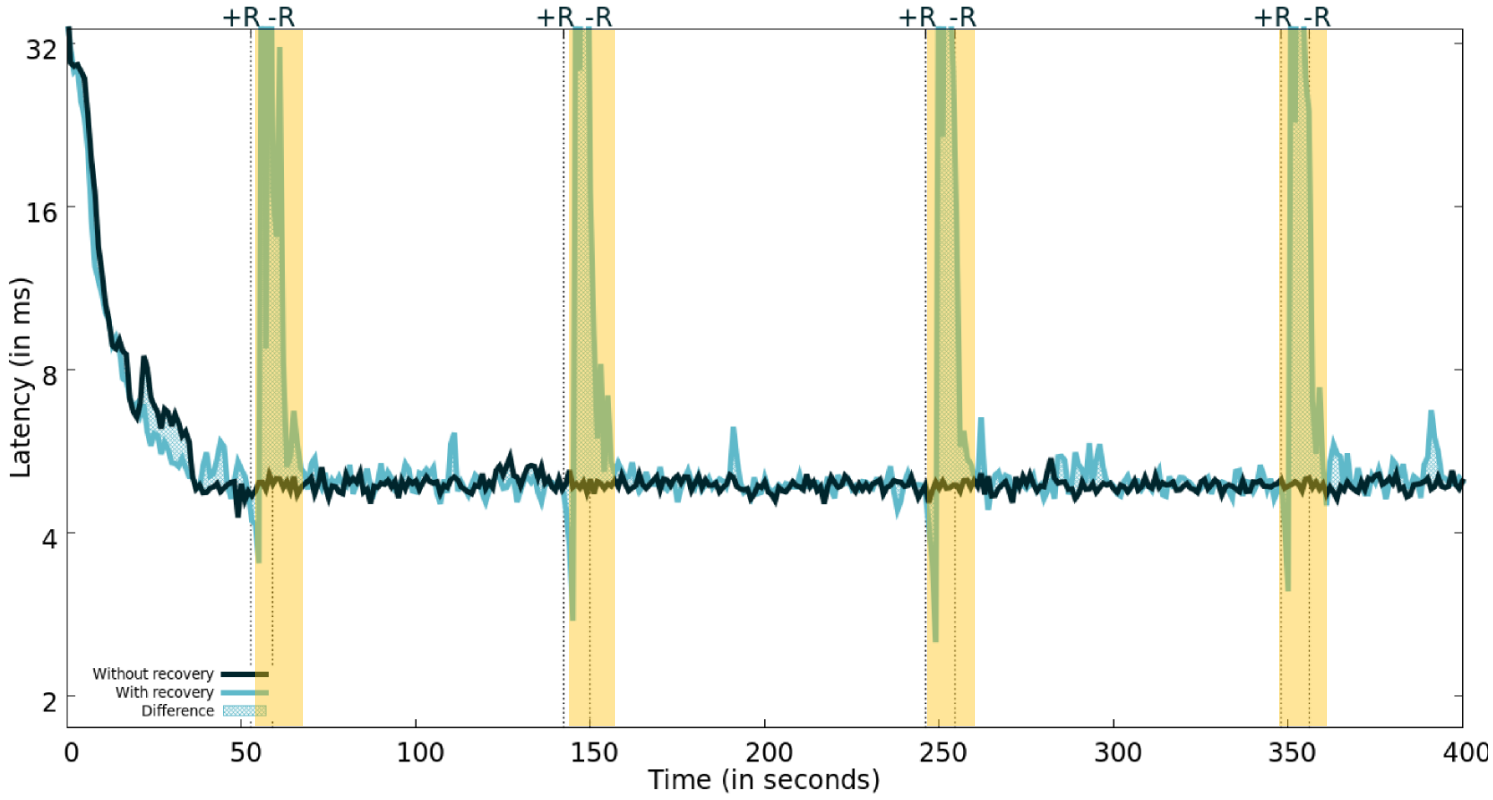
5. Experimental evaluation

Recoveries increase latency **20-** to **30-fold**
(Load balancer reconfig. (~2 s) + warm-up (~7 s))



5. Experimental evaluation

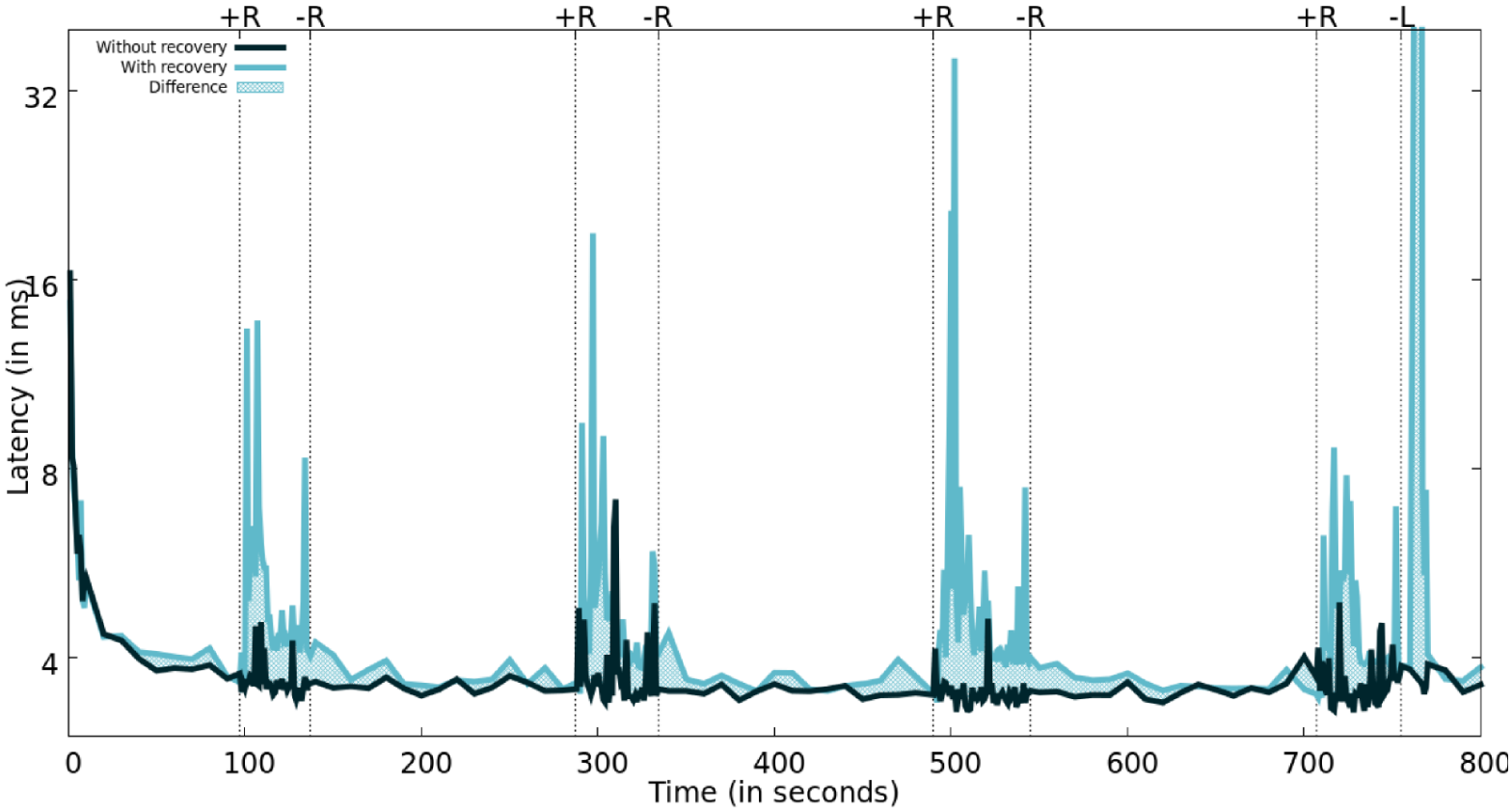
Difference of **60%**,
concentrated in **7.6%** of the time



5. Experimental evaluation

Proactive recovery

Impact on a BFT key-value store (stateful)



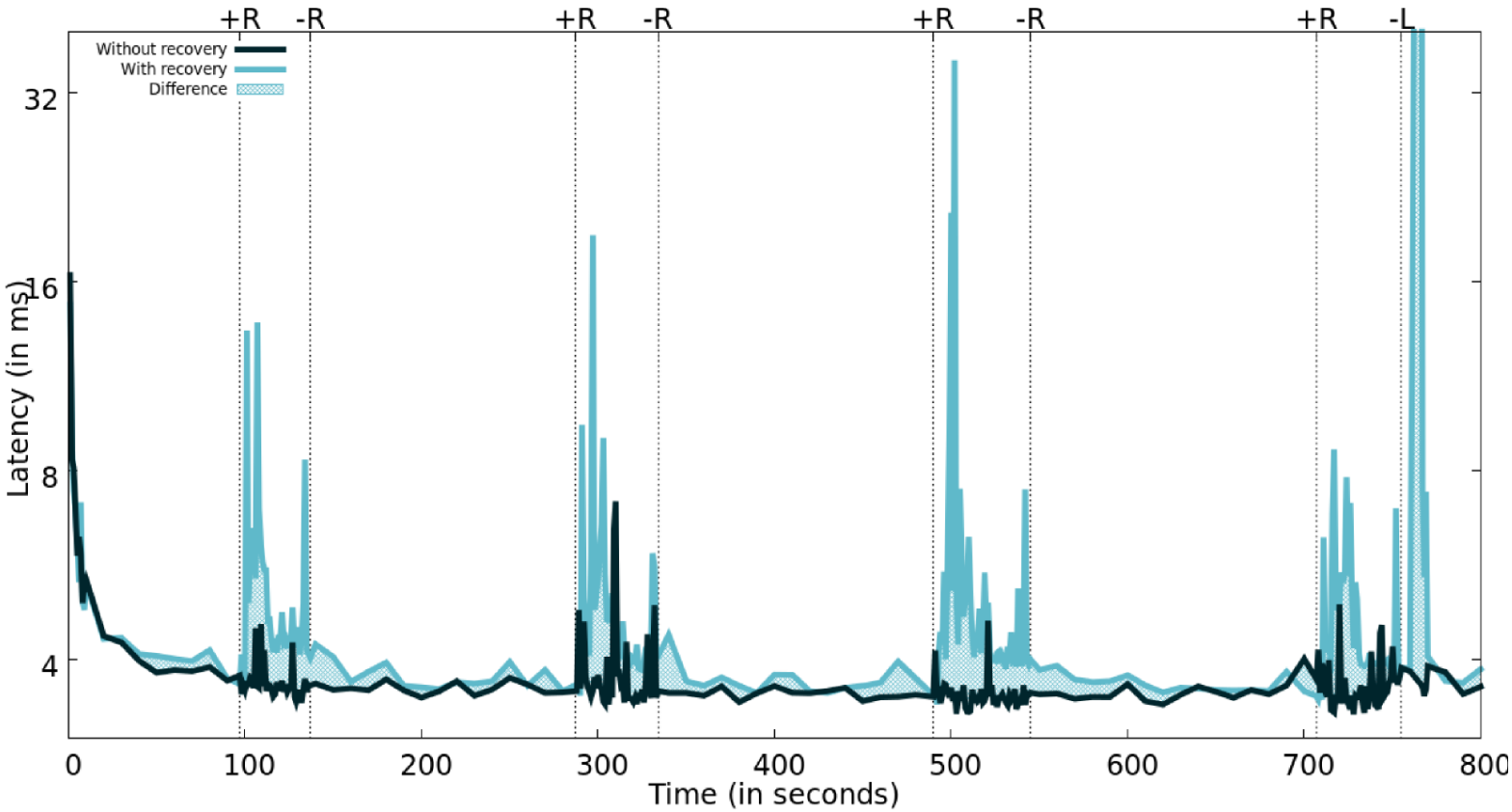
5. Experimental evaluation

4 replicas

4 adaptations

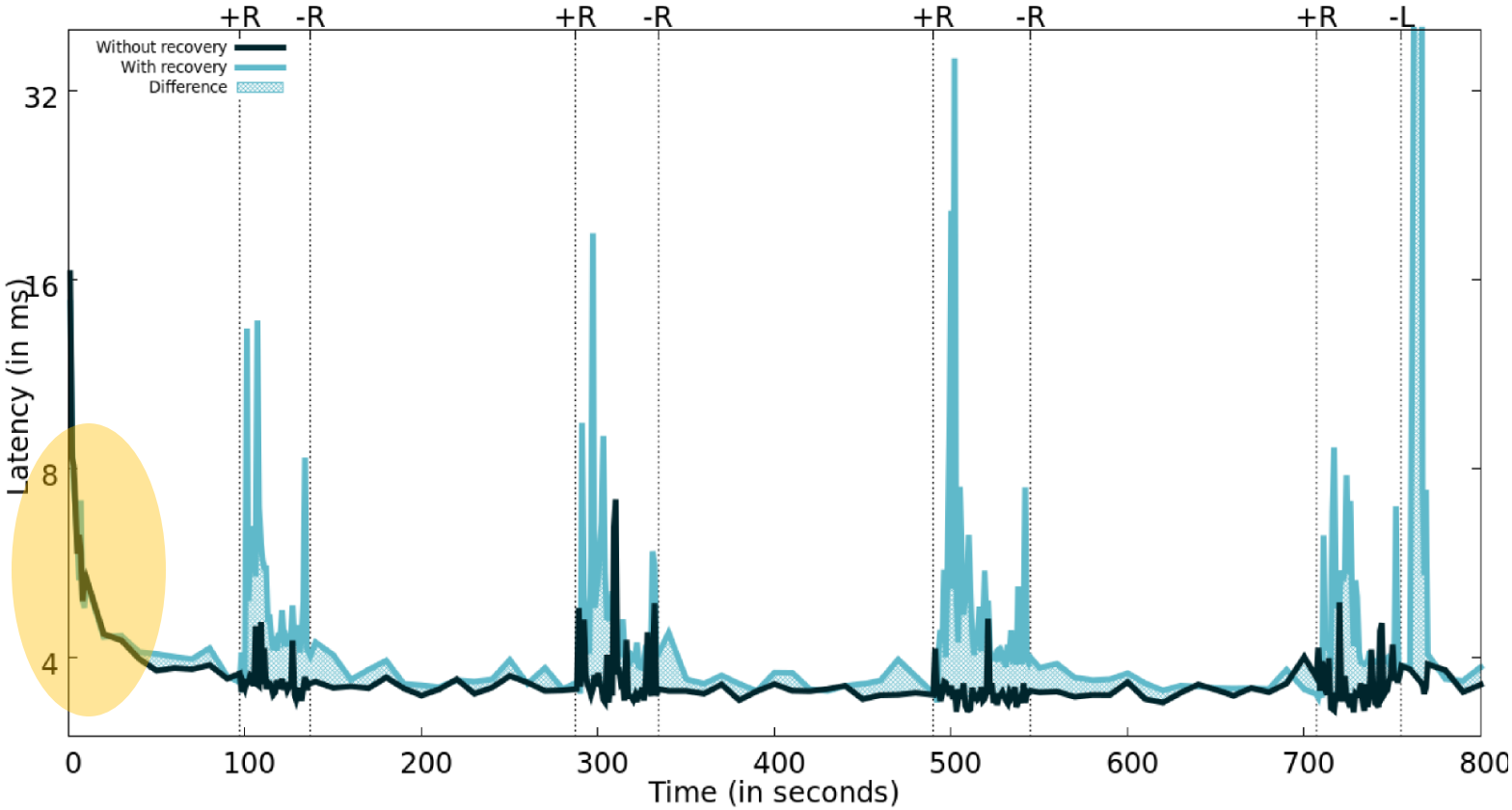
13 min experiment time

3 min each adaptation time



5. Experimental evaluation

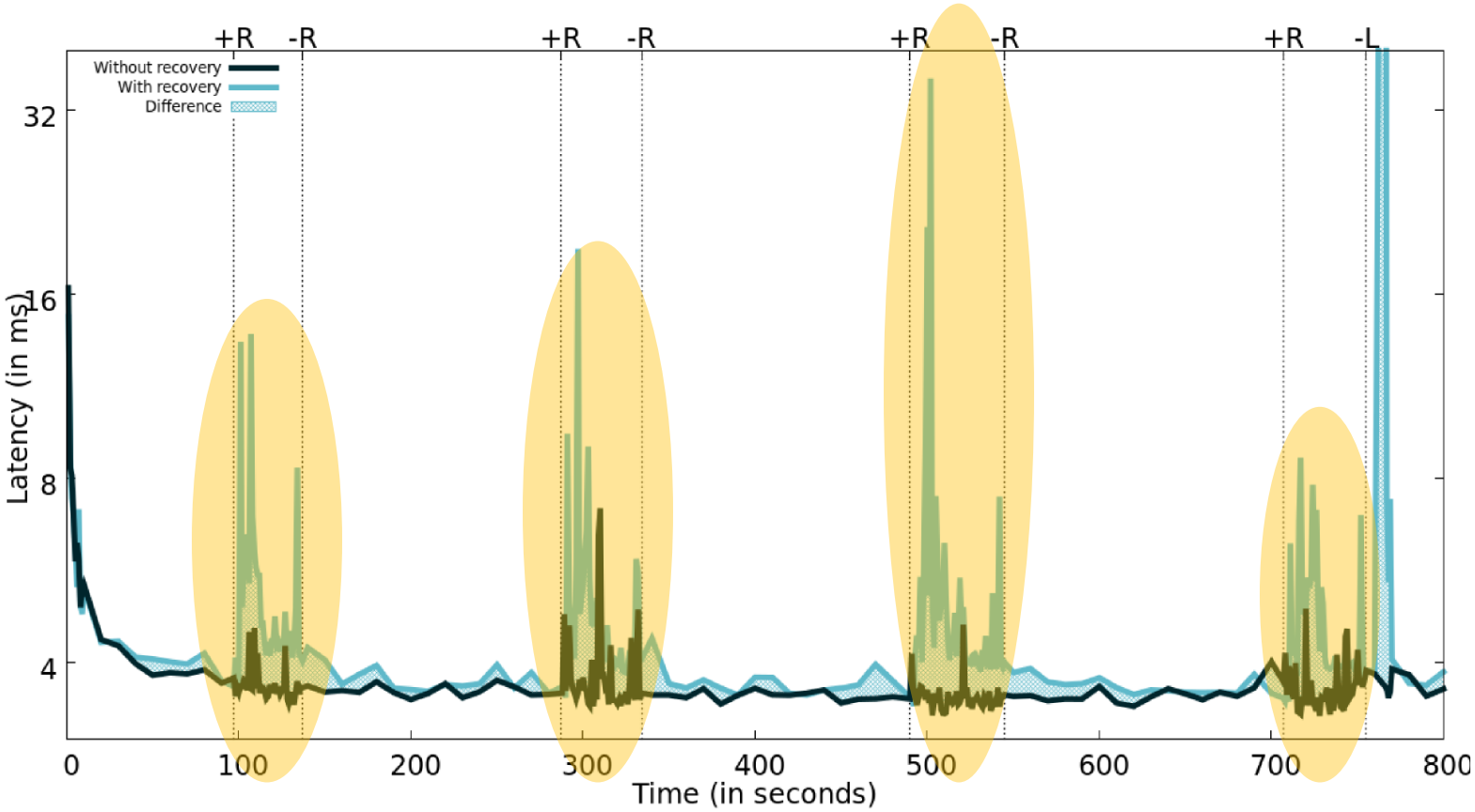
Warm-up (45 s)
(JIT)



5. Experimental evaluation

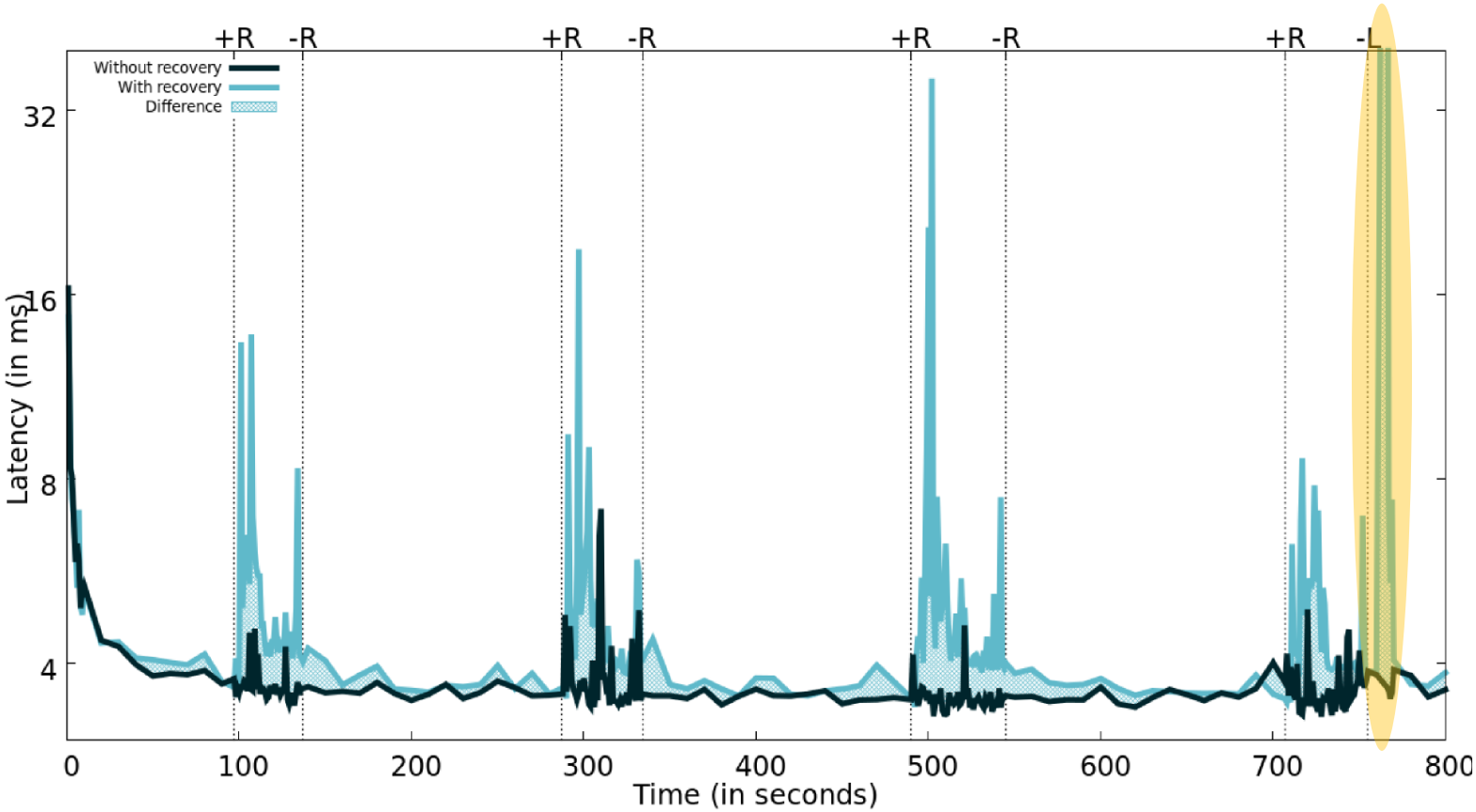
Recoveries increase latency **3-** to **10-fold**

(Reconfiguration (w/ JIT) + state transfer (~35 s) + warm-up (~12 s))



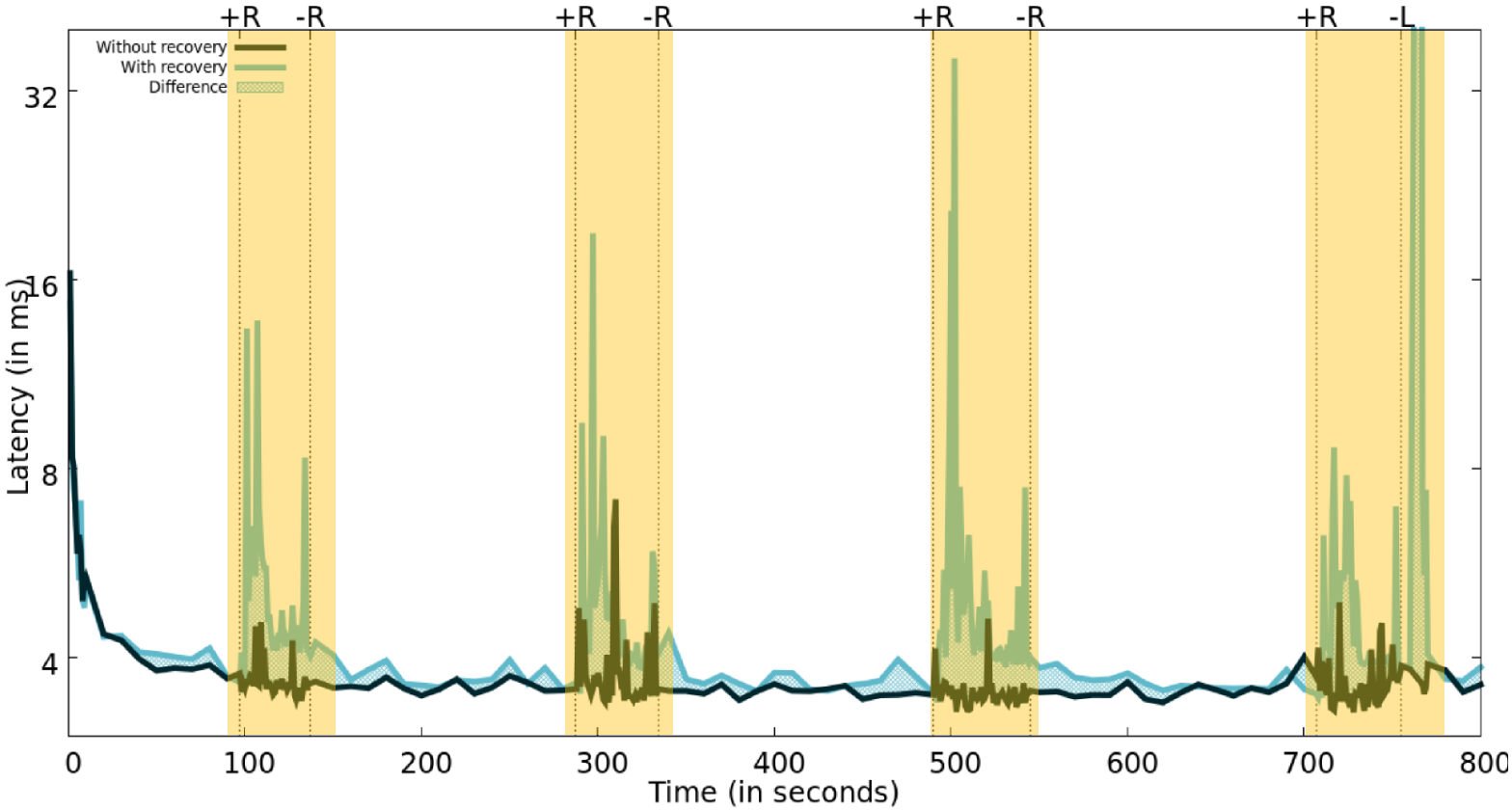
5. Experimental evaluation

Service stops (PUT) during **3 s**
(Leader election protocol)



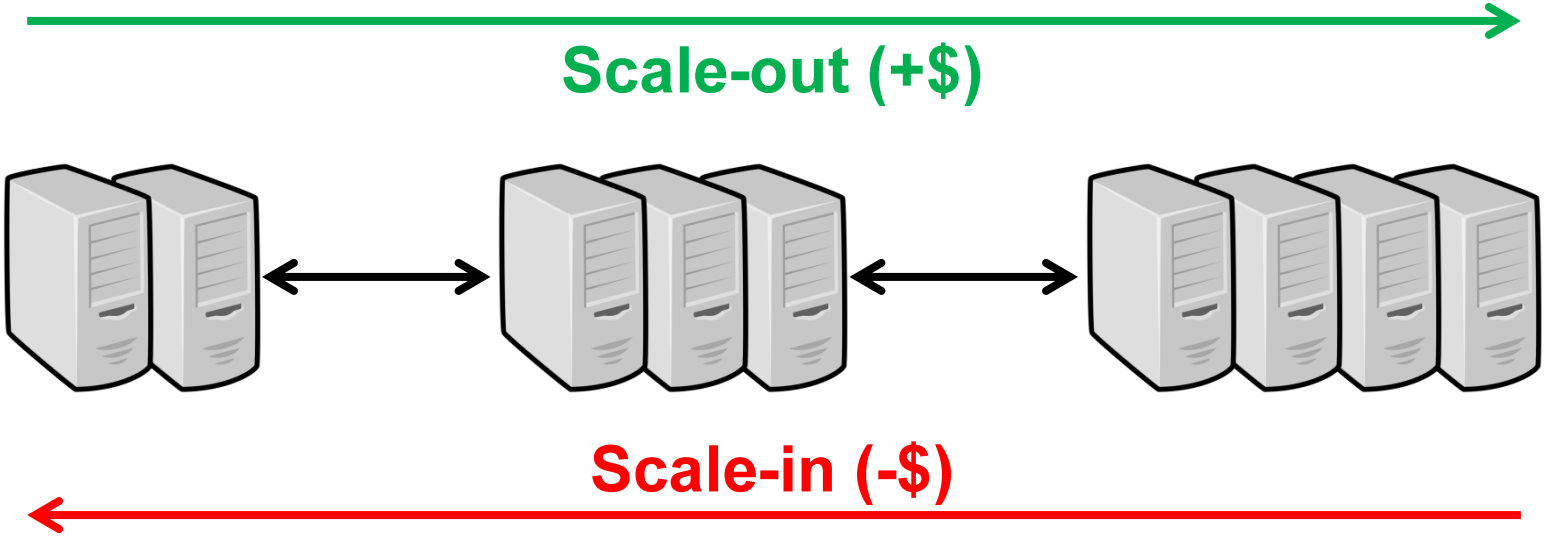
5. Experimental evaluation

Difference of **12%**,
concentrated in **29%** of the time



5. Experimental evaluation

2nd experiment: Scale-out and scale-in

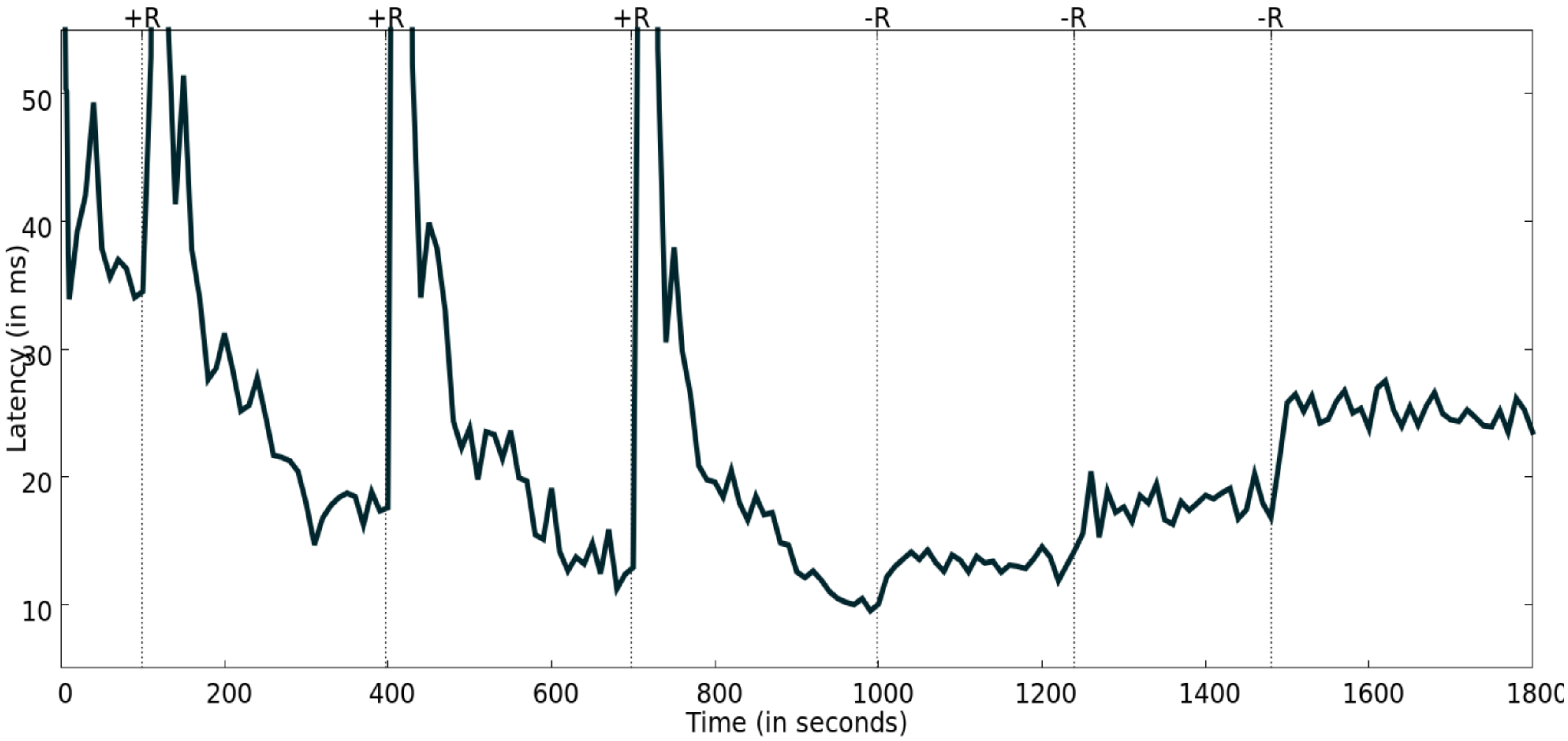


- Changes only the **number of replicas**
- Applied only in the **CFT service**

5. Experimental evaluation

Scale-out and scale-in

A CFT web service (stateless)



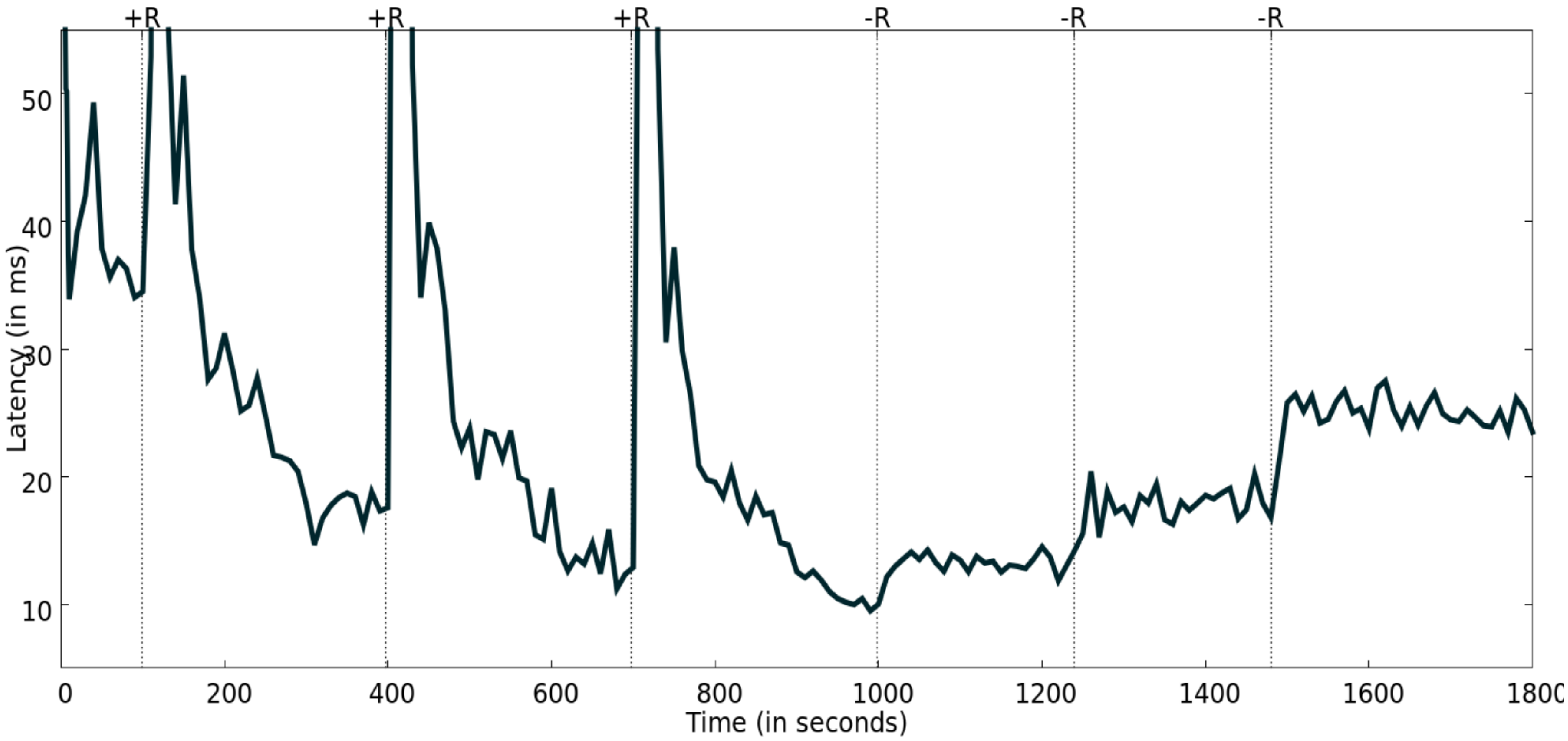
5. Experimental evaluation

2
replicas

6
adaptations

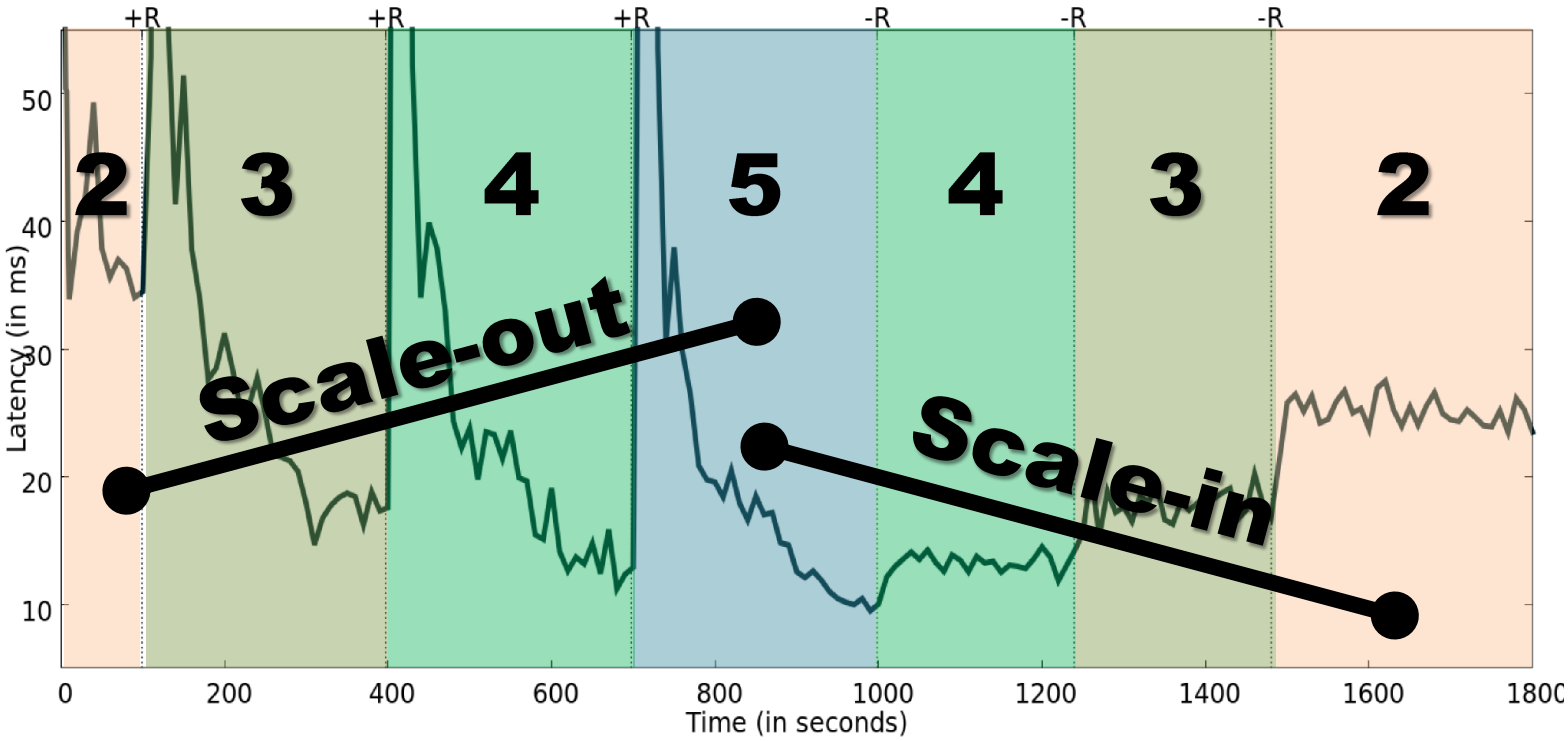
30 min
experiment time

5 min
each adaptation time



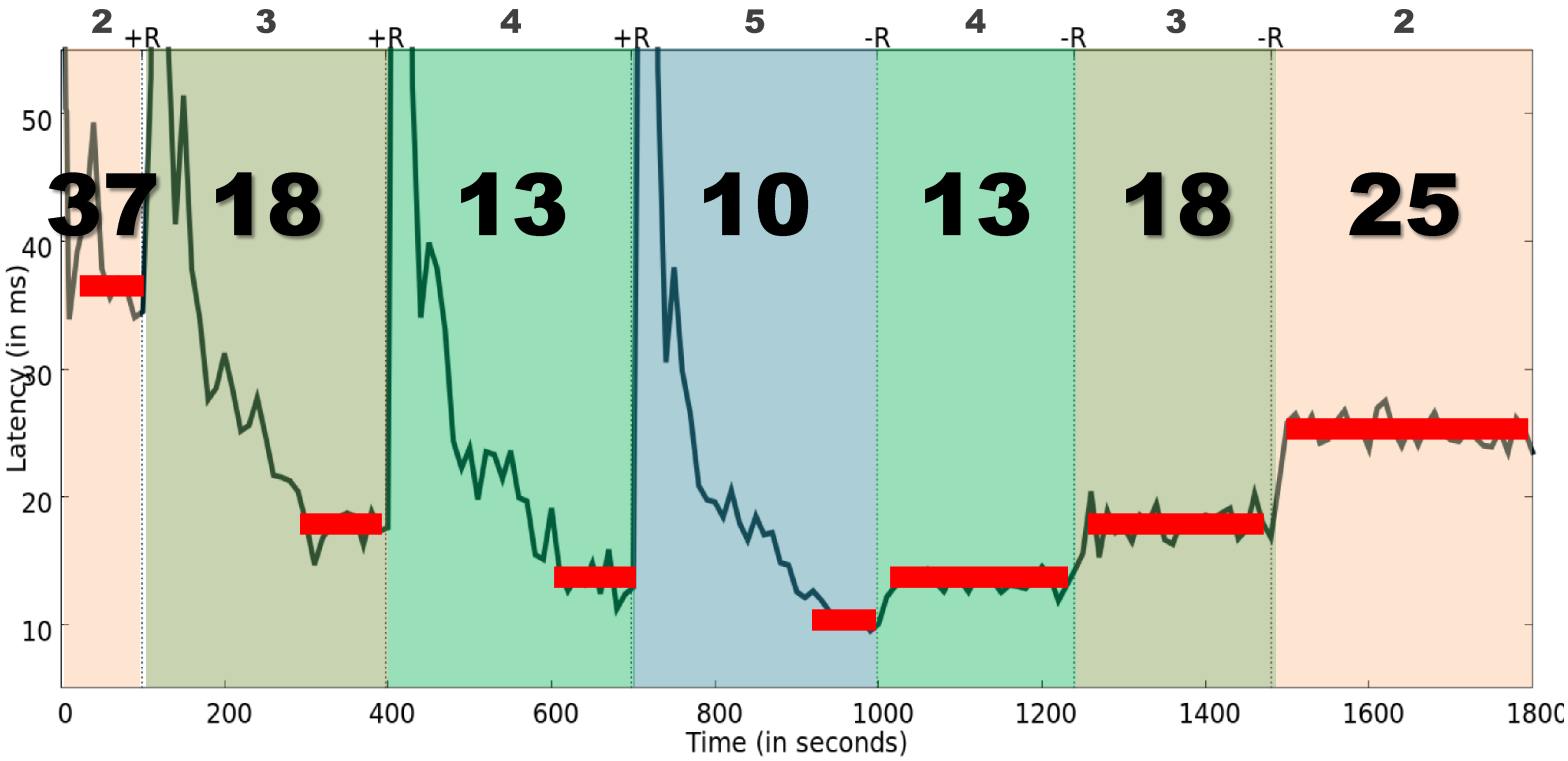
5. Experimental evaluation

Group **size**



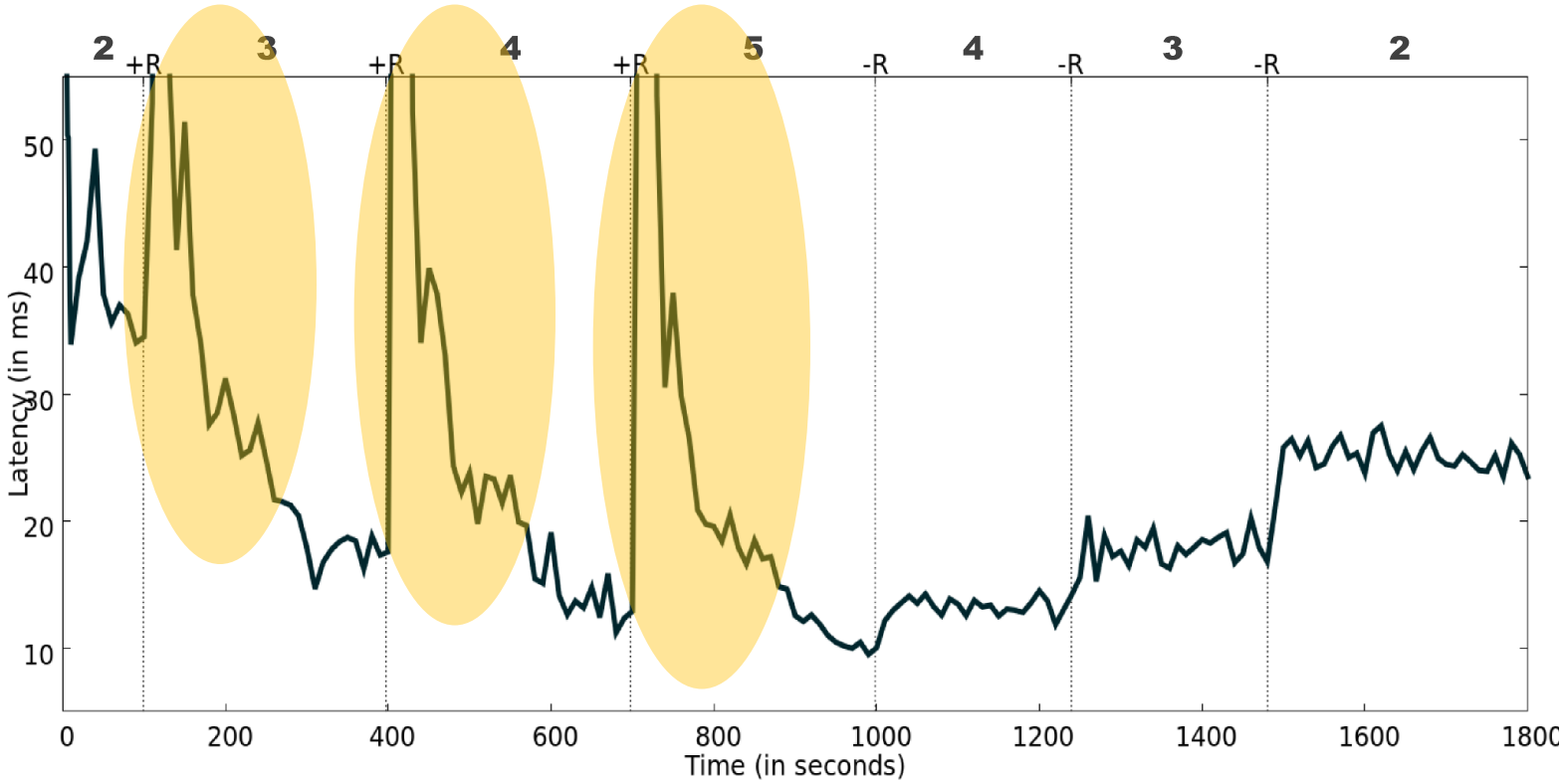
5. Experimental evaluation

Latency (in ms)



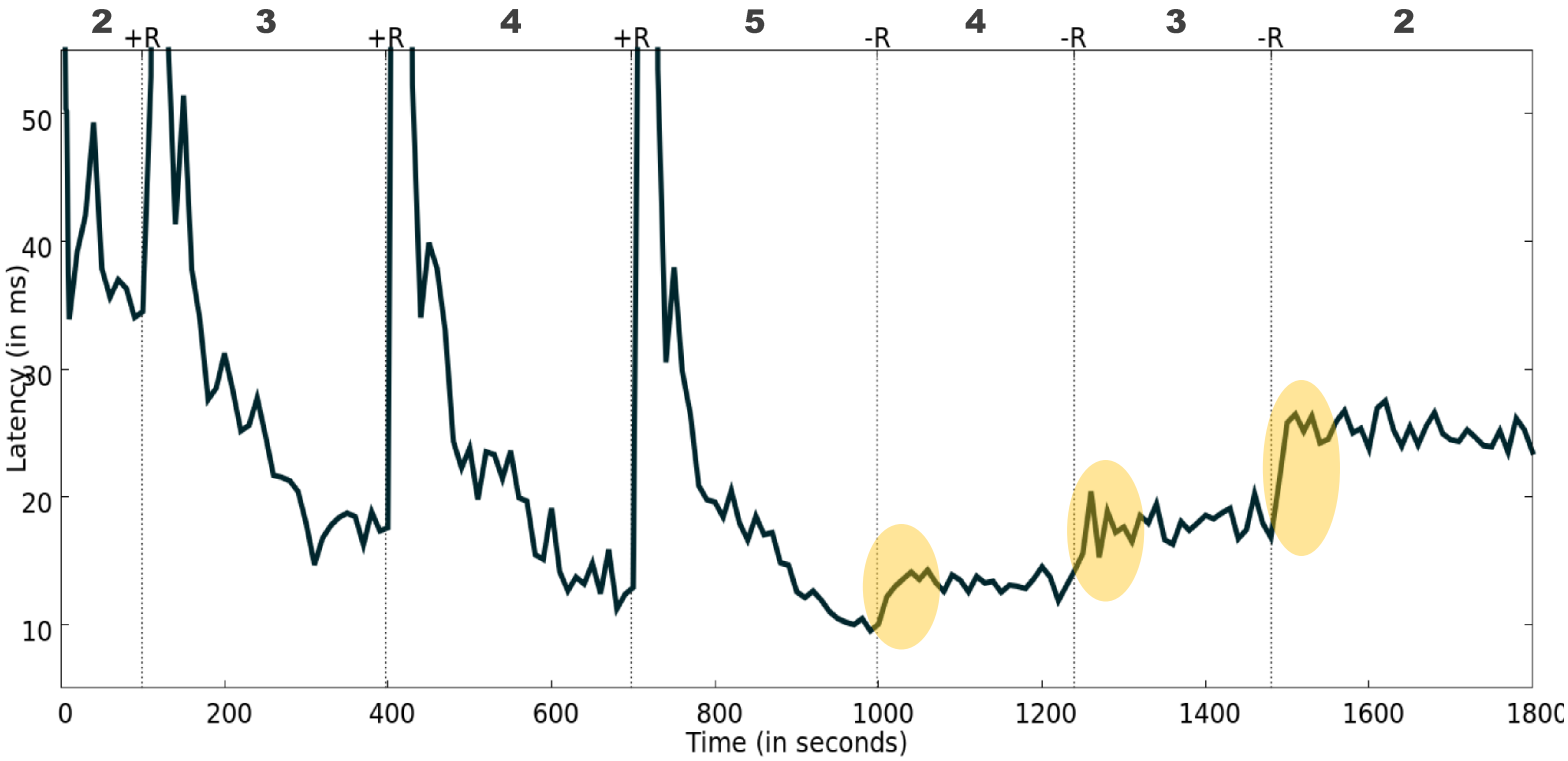
5. Experimental evaluation

Latency **spikes** when **inserting** replicas …
(Load balancer reconfig. + warm-up)



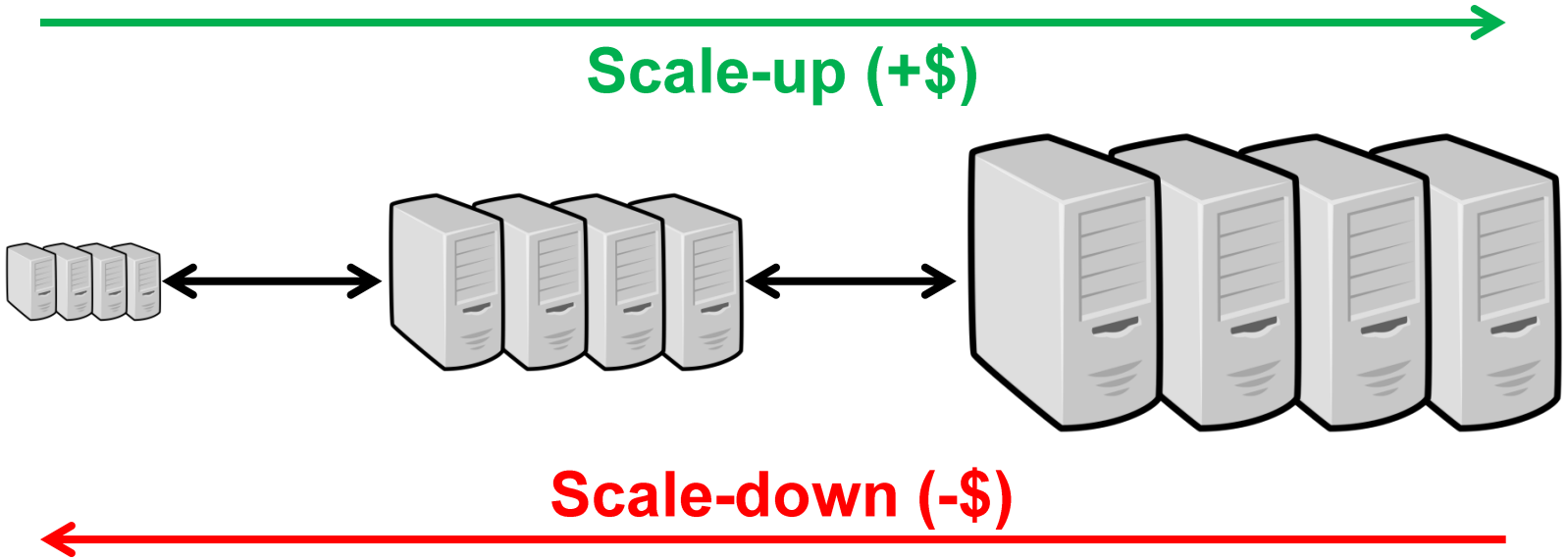
5. Experimental evaluation

… but **does not** when **removing**
(Load balancer protocol is simpler + no JIT)



5. Experimental evaluation

3rd experiment: Scale-up and scale-down



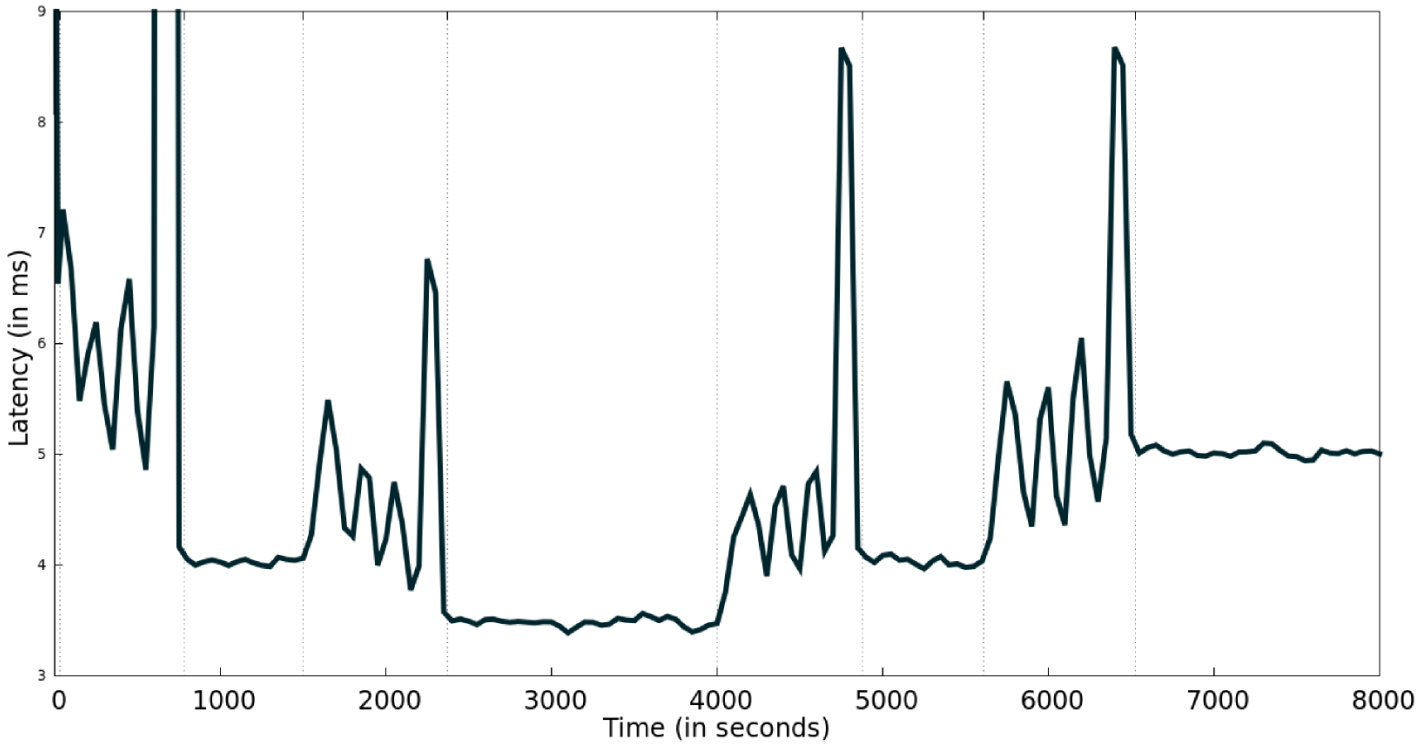
- Increase/Reduce the CPU and RAM for each replica
- Applied only in the **BFT service**

VM	ECU	RAM	US$/h (EC2)
Small	1	2 GB	0.06
Medium	2	4 GB	0.12
Large	4	8 GB	0.24

5. Experimental evaluation

Scale-up and scale-down

A BFT key-value store (stateful)



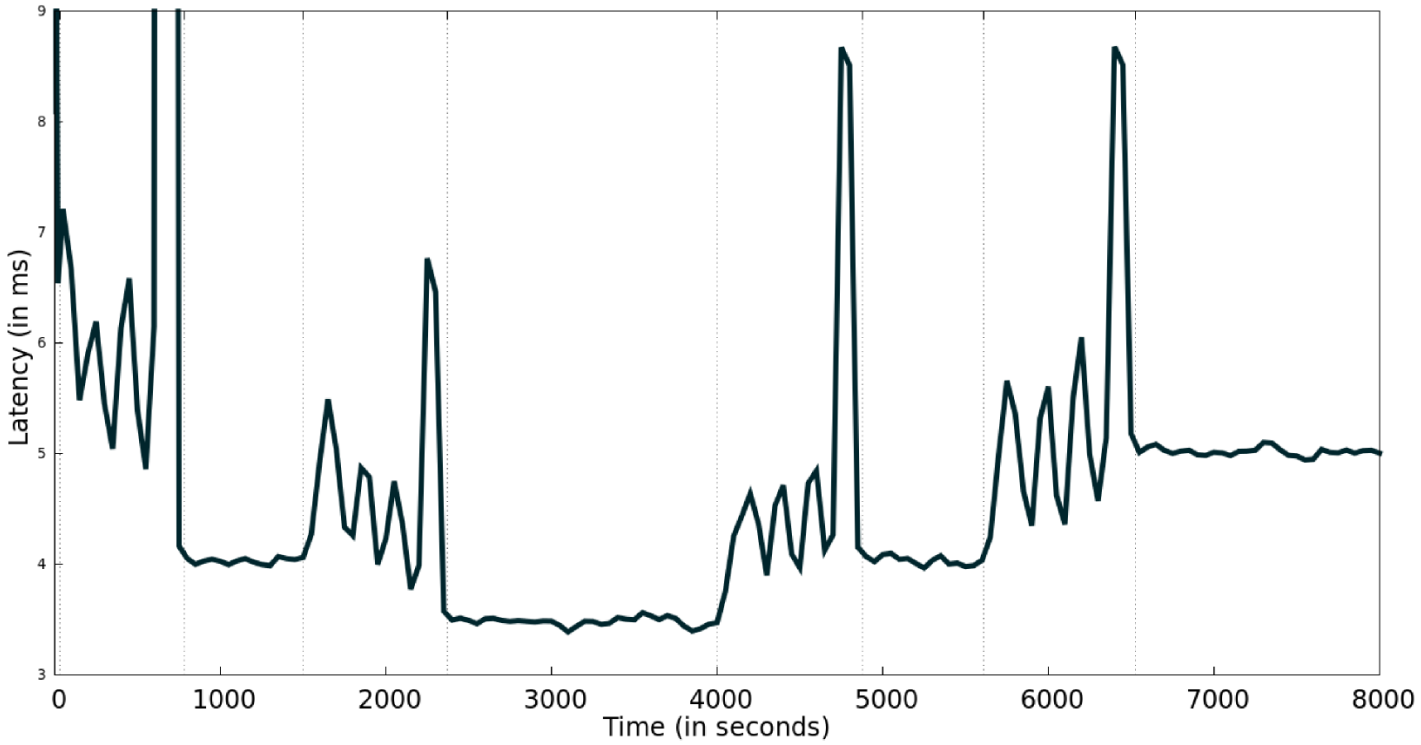
5. Experimental evaluation

4
replicas

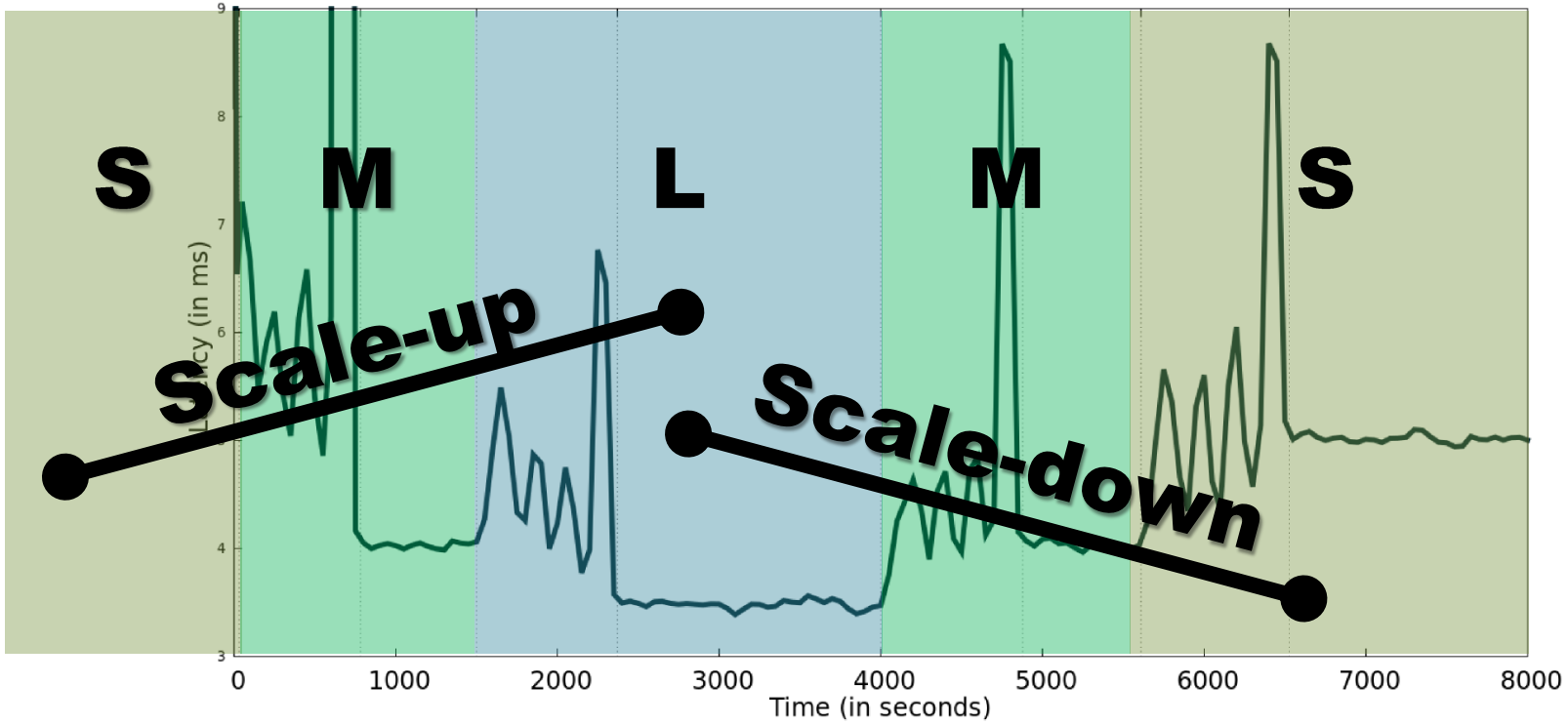
4
adaptations

2.2h
experiment time

30 min
each adaptation time

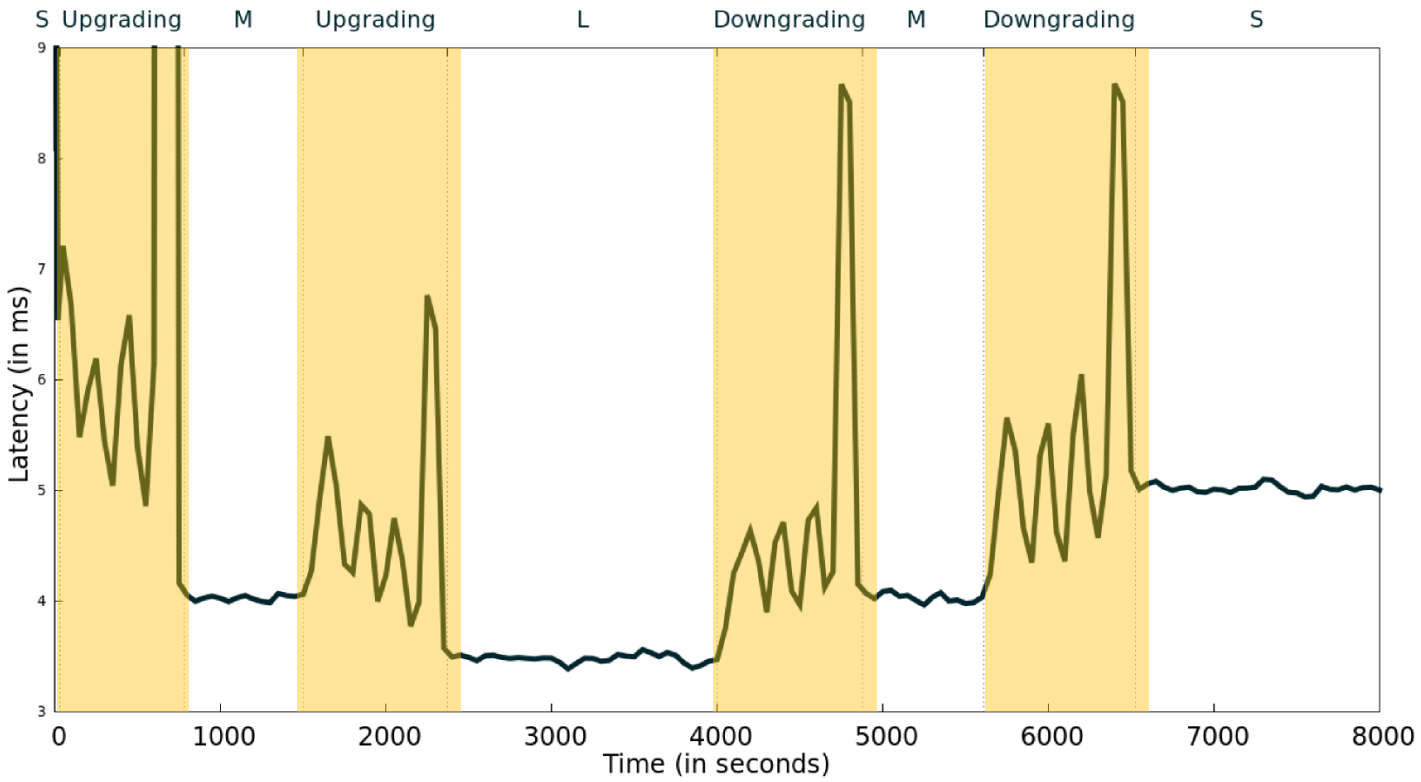


Group **types**

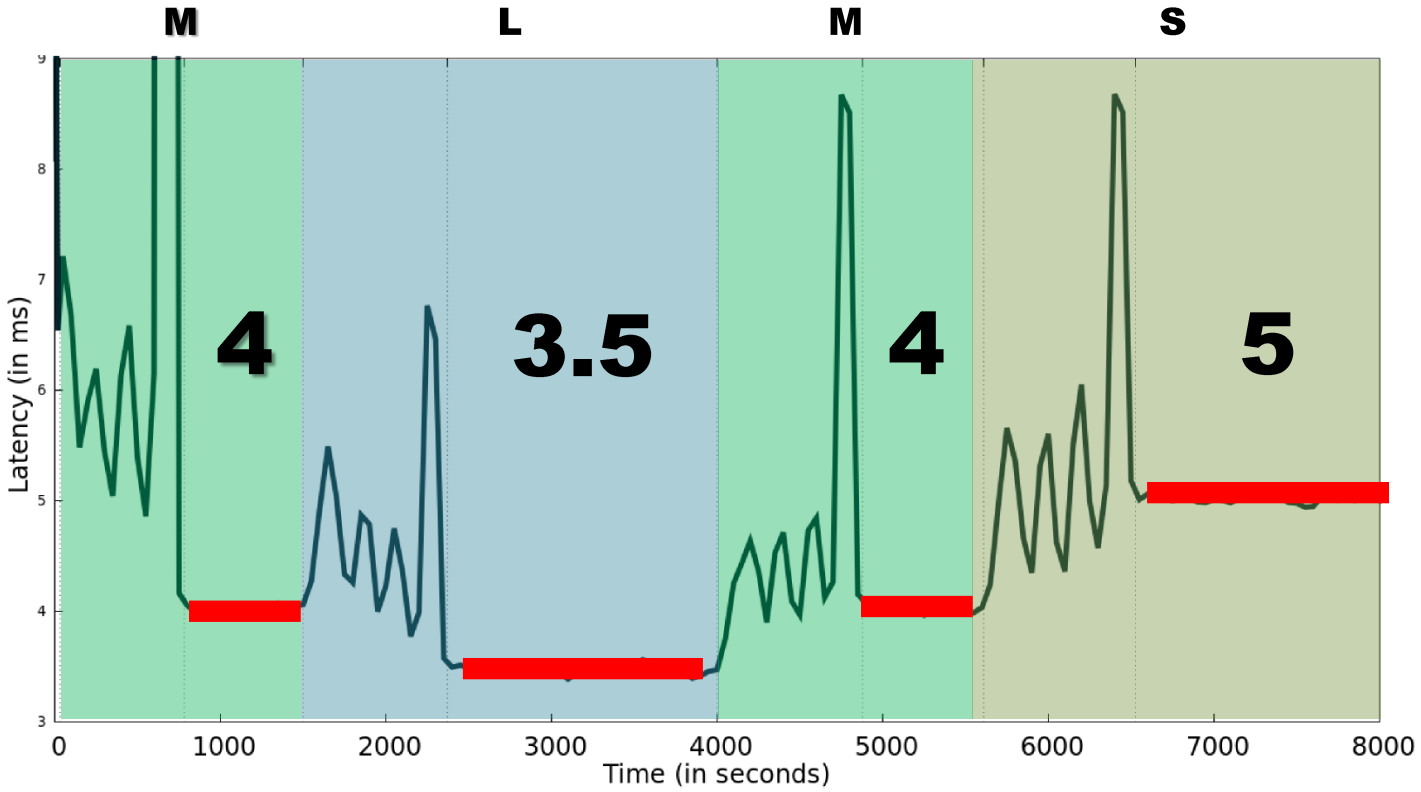

S
M
L
M
S
Scale-up
Scale-down
9
8
7
6
4
3
0
1000
2000
3000
4000
5000
6000
7000
8000
Time (in seconds)

5. Experimental evaluation

Each scale-up or scale-down is comprised of 4 replacements

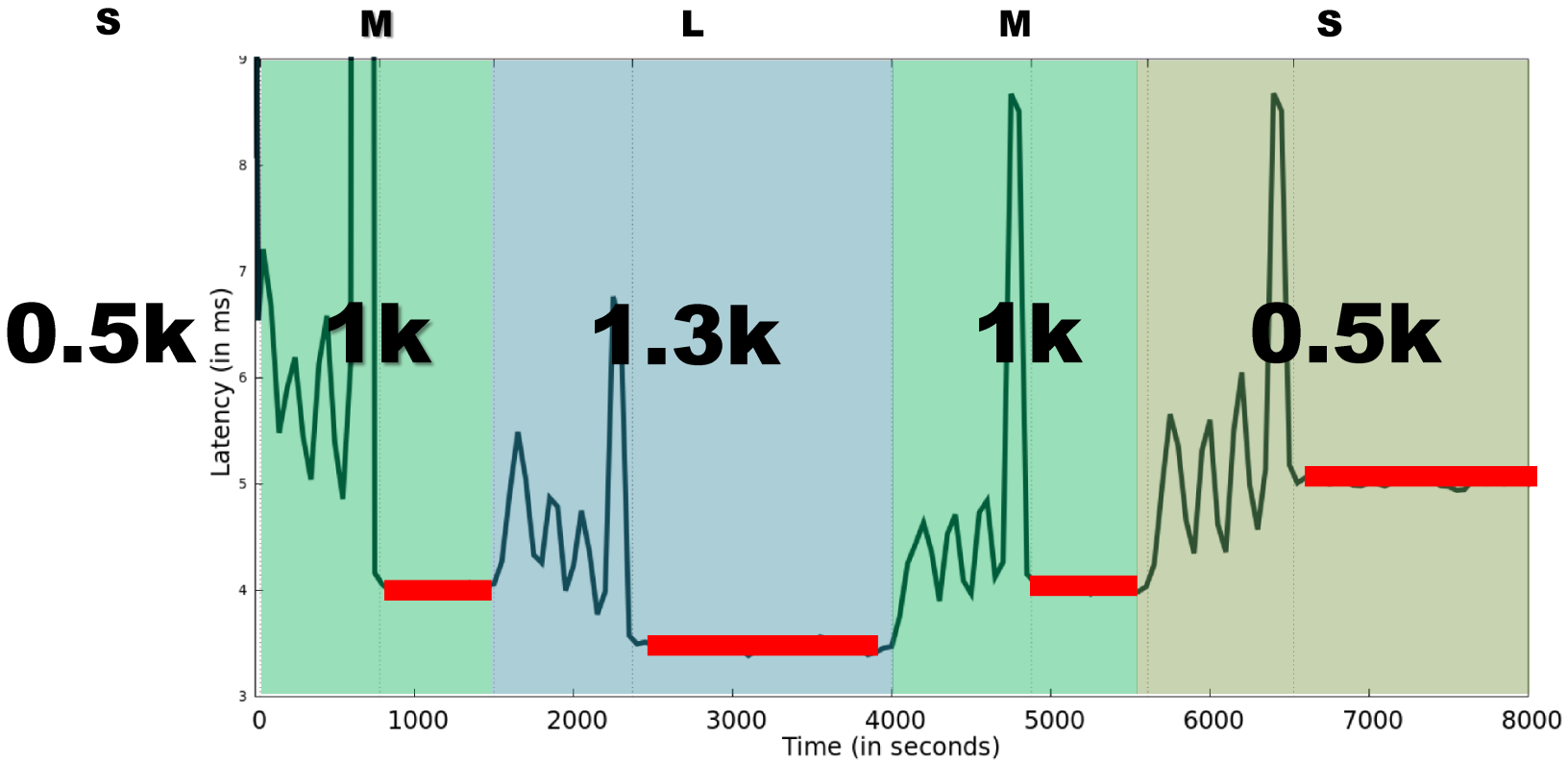


Latencies (in ms)



5. Experimental evaluation

Throughput (in ops/s)



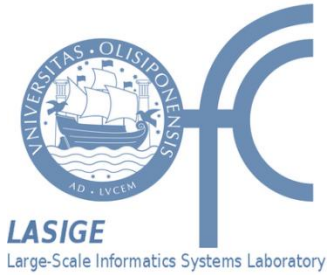
6. Conclusions

There are still **opportunities** to **improve** replicated services **performance, dependability** and **cost-efficiency** with proper usage of **cloud computing dynamism**.

- We presented **FITCH:**
  - An infrastructure to **support** the **dynamic adaptation** of **replicated services** in **cloud environments**
  - 3 basic operations (add, remove and replace replicas)
  - 2 representative services
    - **A CFT web service (stateless)**
    - **A consistent BFT key-value store (stateful)**
  - 3 adaptation scenarios
    - **Proactive recovery**
    - **Scale-out and scale-in**
    - **Scale-up and scale-down**

Thank you!

vielmo@lasige.di.fc.ul.pt
http://lasige.di.fc.ul.pt/~vielmo



LaSIGE: Large-Scale Informatics Systems Laboratory

Fundação para a Ciência e Tecnologia (FCT)

Multiannual Program

http://lasige.di.fc.ul.pt

CloudFIT: Fault-and-Intrusion Tolerance for Clouds

Fundação para a Ciência e Tecnologia (FCT)

PTDC/EIA-CCO/108299/20008

http://cloudfit.di.fc.ul.pt

Tclouds: Trustworthy Clouds

EU's 7th Framework Program

ICT-257243

http://tclouds.eu/