Modular and Reproducible Simulator Architecture for Composable Cloud Systems

Rubén Luque, José Luis Díaz

University of Oviedo

(Spain)

Joaquín Entrialgo

Technical University of Madrid, UPM

(Spain)

j.entrialgo@upm.es

SIMUL 2025

September 28 – October 2

Lisbon (Portugal)

Introduction

Introduction

Motivation

- Cloud computing: dominant paradigm for scalable services
- Challenges in modelling and evaluating cloud systems

Main contributions

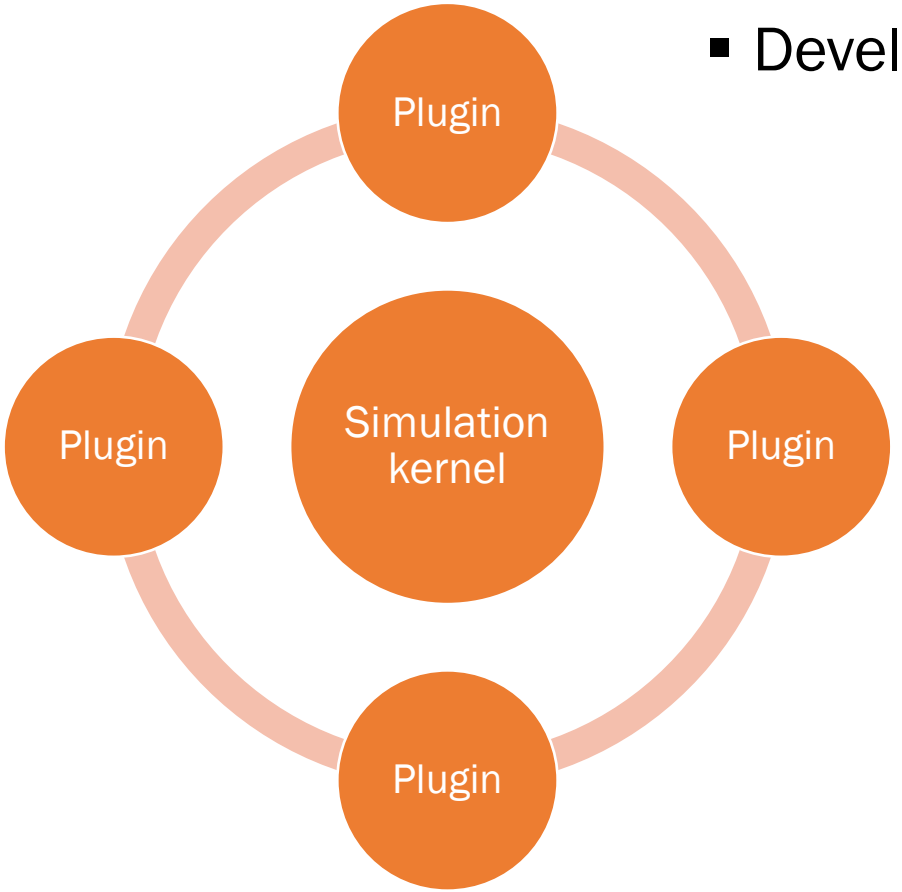
- Design of a modular and reproducible simulation architecture
- Extensible plugin system
- Validation scenario

System architecture

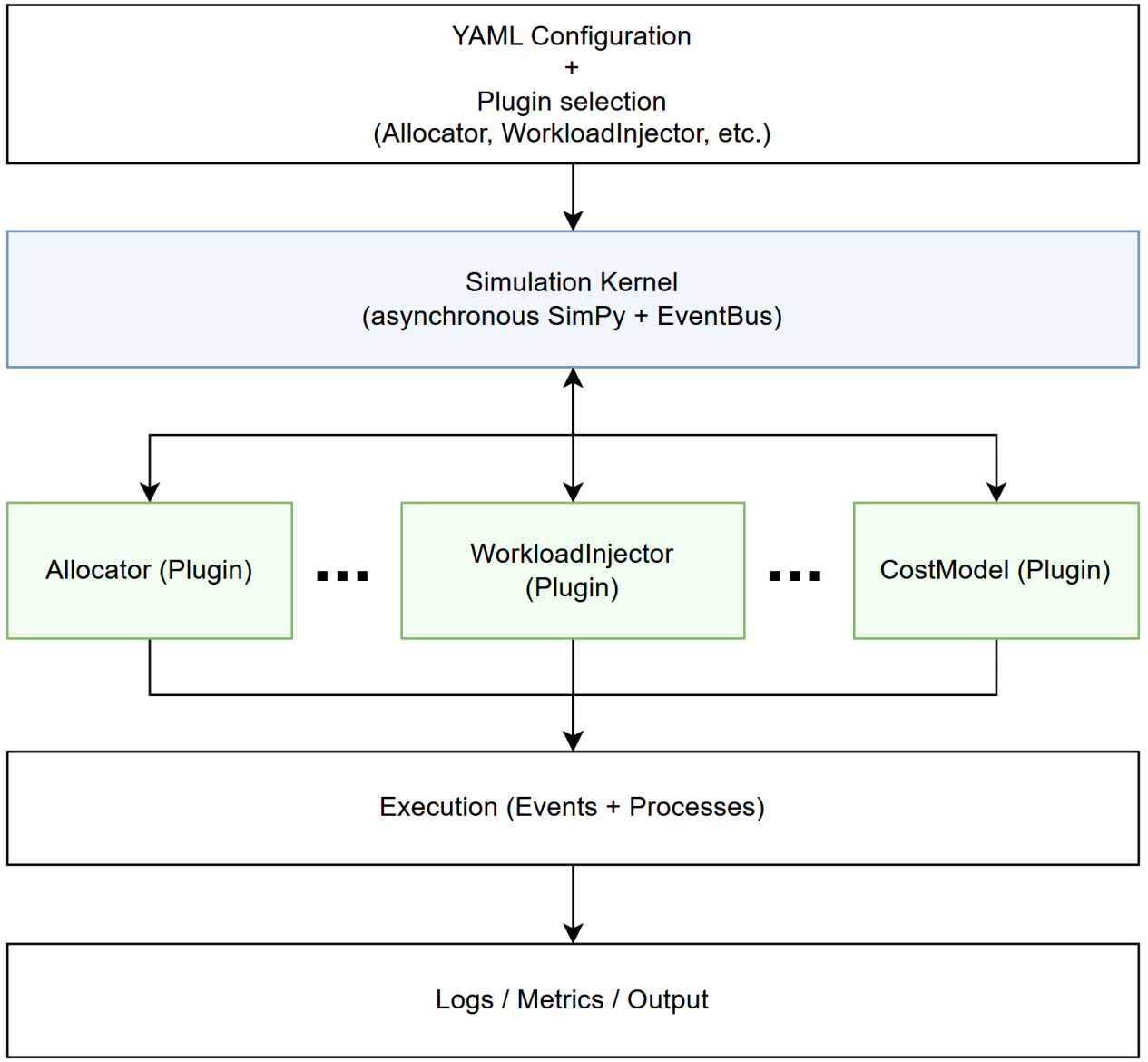
Overview

- Discrete event simulator
- Modular and extensible
- Reproducible

- Two kind of users
  - Analysts: design simulation scenarios
  - Developers: implement plugins



System architecture



Plugin System and Extensibility

Plugin System and Extensibility

- Plugin discovery and registration
  - Using the pluggy library
  - Dynamic plugin discovery
  - Multiple plugins can be available at the same time
  - Selected in YAML

Plugin System and Extensibility

- Interface contracts via Protocols
  - *hookspec*: functions required by pluggy for all plugins
  - Protocols: particular functions for each plugin type
    - e.g.: get_workloads(), apply_allocation(), compute_cost()
  - Improves code reliability, reduces integration errors and facilitates development of new components

Experimental Validation

Experimental Validation

- Goals
  - Demonstrate the proposed architecture supports scenario-based evaluation
  - Demonstrate modularity
- Nuberu
  - Prototype using this architecture
- Case study
  - Shows how the simulation can help exposing assumptions of decision-making tools

Experimental Validation

Scenario

- Cost optimization
- Allocation of VMs and containers
- Using Conlloovia
  - Inputs:
    - VM instance classes
    - Container classes
    - Throughput for each container/VM
    - Workload
  - Outputs
    - Allocation

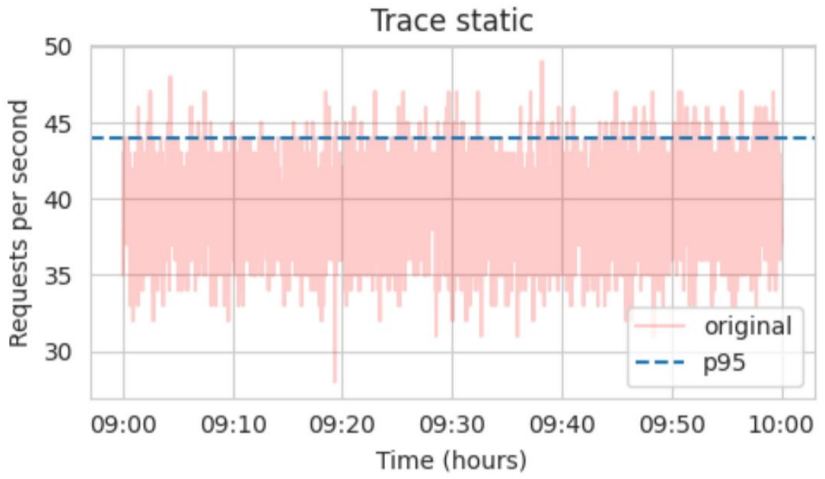
TABLE I. PERFORMANCE (REQUEST PER SECOND (RPS)) OF EACH CONTAINER CLASS ON EVERY VM INSTANCE CLASS

C. Class / VM I. Class	cc0app0	cc0app1	cc1app0	cc1app1	cc2app0	cc2app1
c5.2xlarge	2.10	0.46	4.30	0.96	6.35	1.63
c5.large	2.10	0.46	4.30	0.96	6.35	1.63
c5.xlarge	2.10	0.46	4.30	0.96	6.35	1.63
c6i.2xlarge	2.29	0.50	4.71	1.02	6.82	1.76
c6i.large	2.29	0.50	4.71	1.02	6.82	1.76
c6i.xlarge	2.29	0.50	4.71	1.02	6.82	1.76

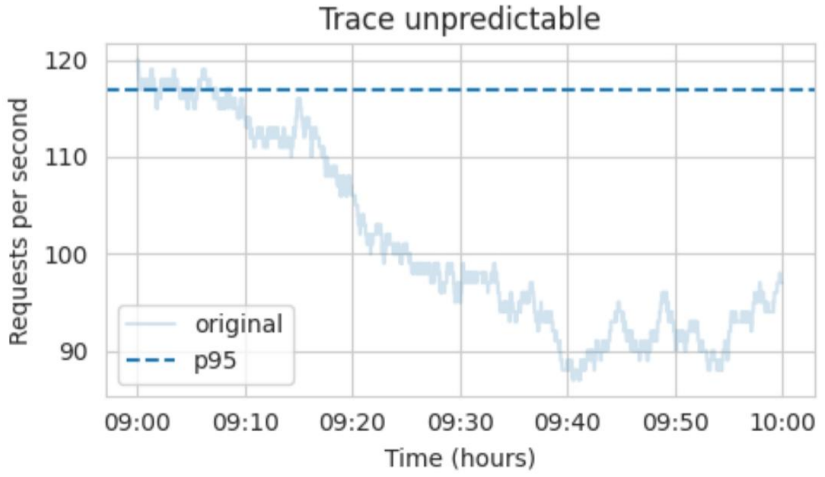
Experimental Validation

Scenario

- Cost optimization
- Allocation of VMs and containers
- Using Conlloovia
  - Inputs:
    - VM instance classes
    - Container classes
    - Throughput for each container/VM
    - Workload
  - Outputs
    - Allocation



(a) Workload for app0

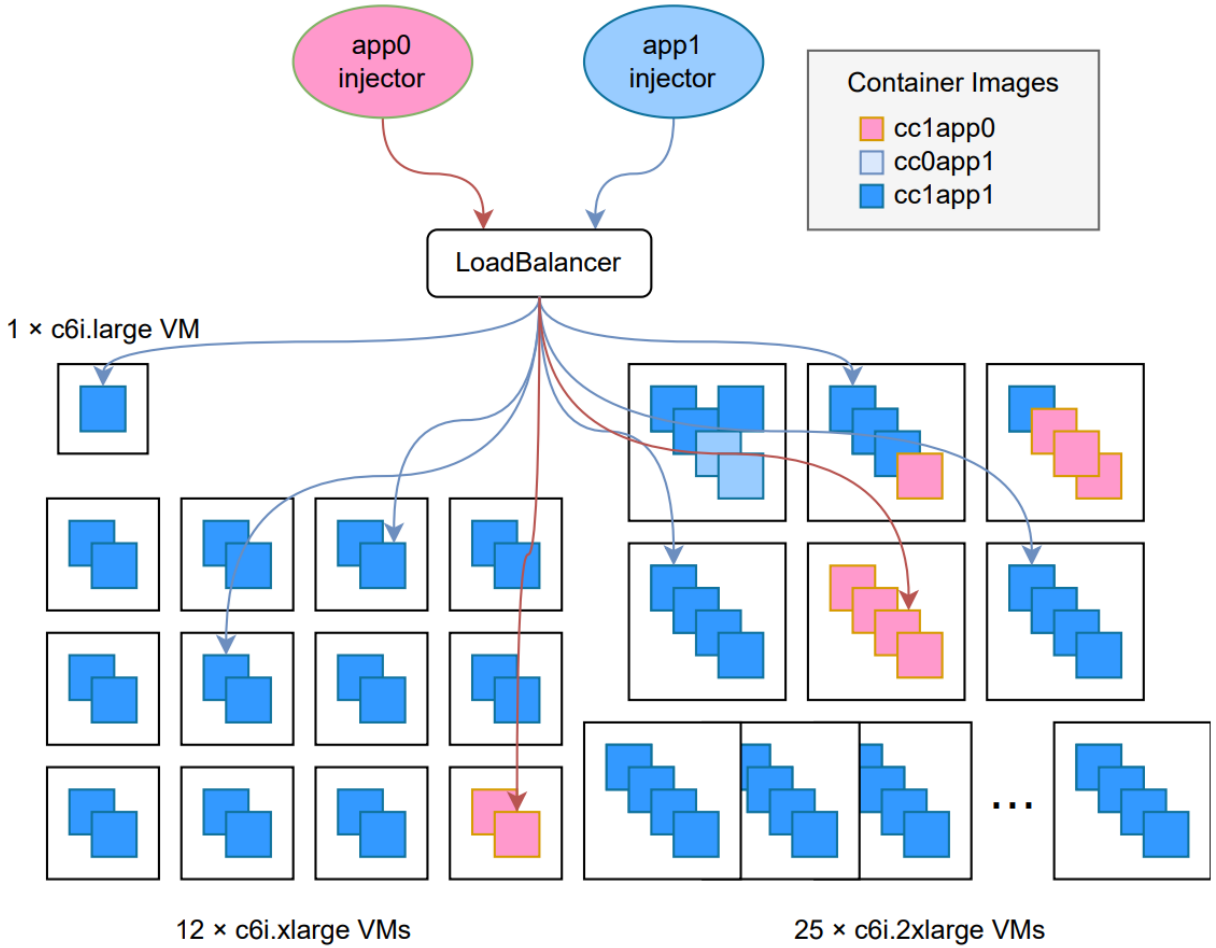


(b) Workload for app1

Experimental Validation

Scenario

- Cost optimization
- Allocation of VMs and containers
- Using Conlloovia
  - Inputs:
    - VM instance classes
    - Container classes
    - Throughput for each container/VM
    - Workload
  - Outputs
    - Allocation



Cost: $10.62

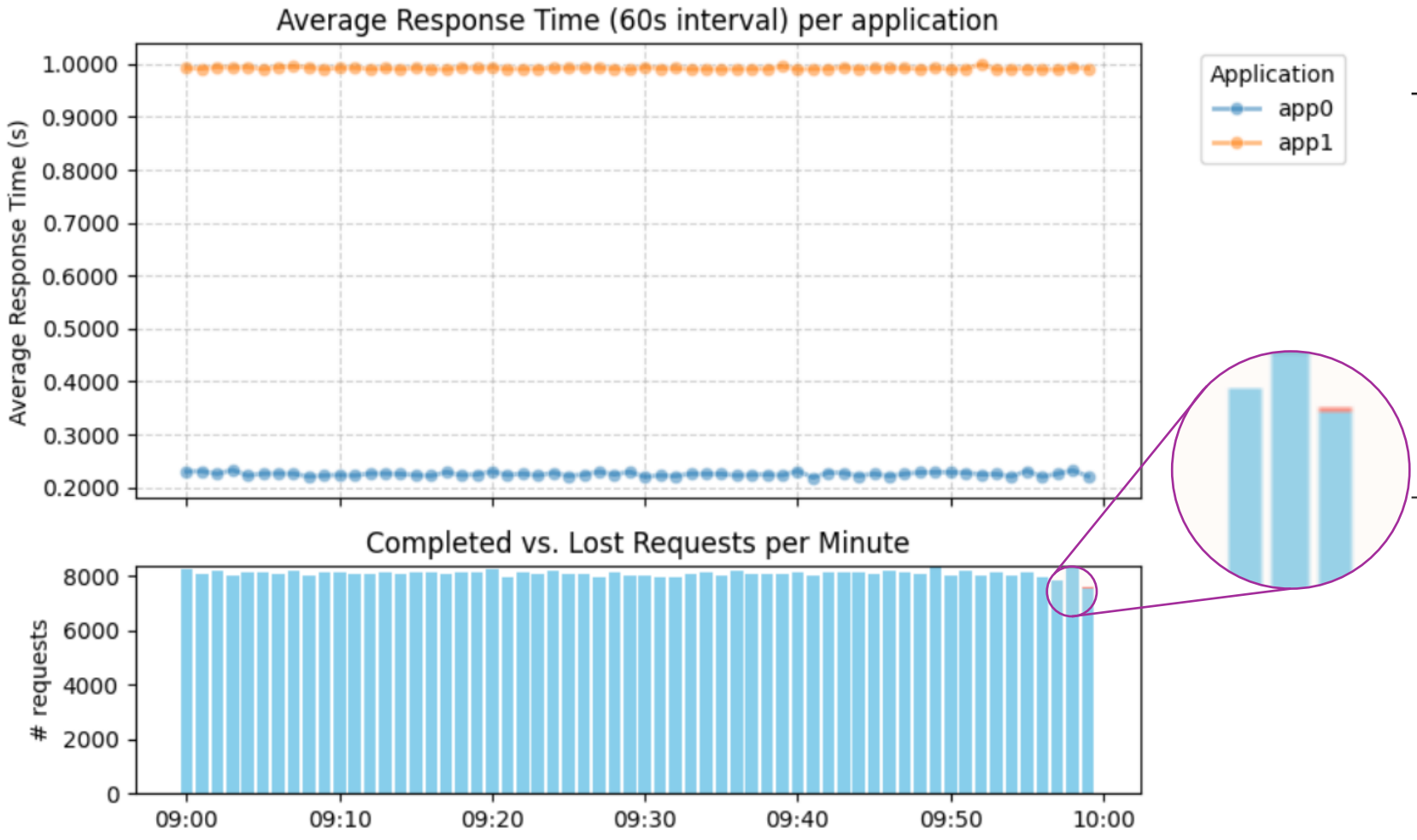
Experimental Validation

Discussion

- Queuing model (Q)
  - No queue
  - Queues of 1000 requests per container
- Termination policy (Term)
  - Hard
  - Drain
- Load Balancing (LB)
  - RR: Round-Robin
  - SWRR: Smooth Weighted Round-Robir
- Load injection (Load)
  - Poisson process with same p95 as assumed by Conlloovia
  - Original trace

Q	Term	LB	Load	app0	app1	cost
0	drain	RR	poisson	82.6%	95.0%	$10.66
			trace	100.0%	98.5%	$10.63
		SWRR	poisson	82.6%	94.8%	$10.66
			trace	100.0%	94.4%	$10.63
	hard	RR	poisson	82.6%	94.9%	$10.62
			trace	100.0%	98.5%	$10.62
		SWRR	poisson	82.6%	94.8%	$10.62
			trace	100.0%	94.3%	$10.62
1000	drain	RR	poisson	100.0%	99.8%	$16.56
			trace	100.0%	99.8%	$16.53
		SWRR	poisson	100.0%	100.0%	$10.66
			trace	100.0%	100.0%	$10.63
	hard	RR	poisson	100.0%	99.2%	$10.62
			trace	100.0%	99.2%	$10.62
		SWRR	poisson	100.0%	100.0%	$10.62
			trace	100.0%	100.0%	$10.62

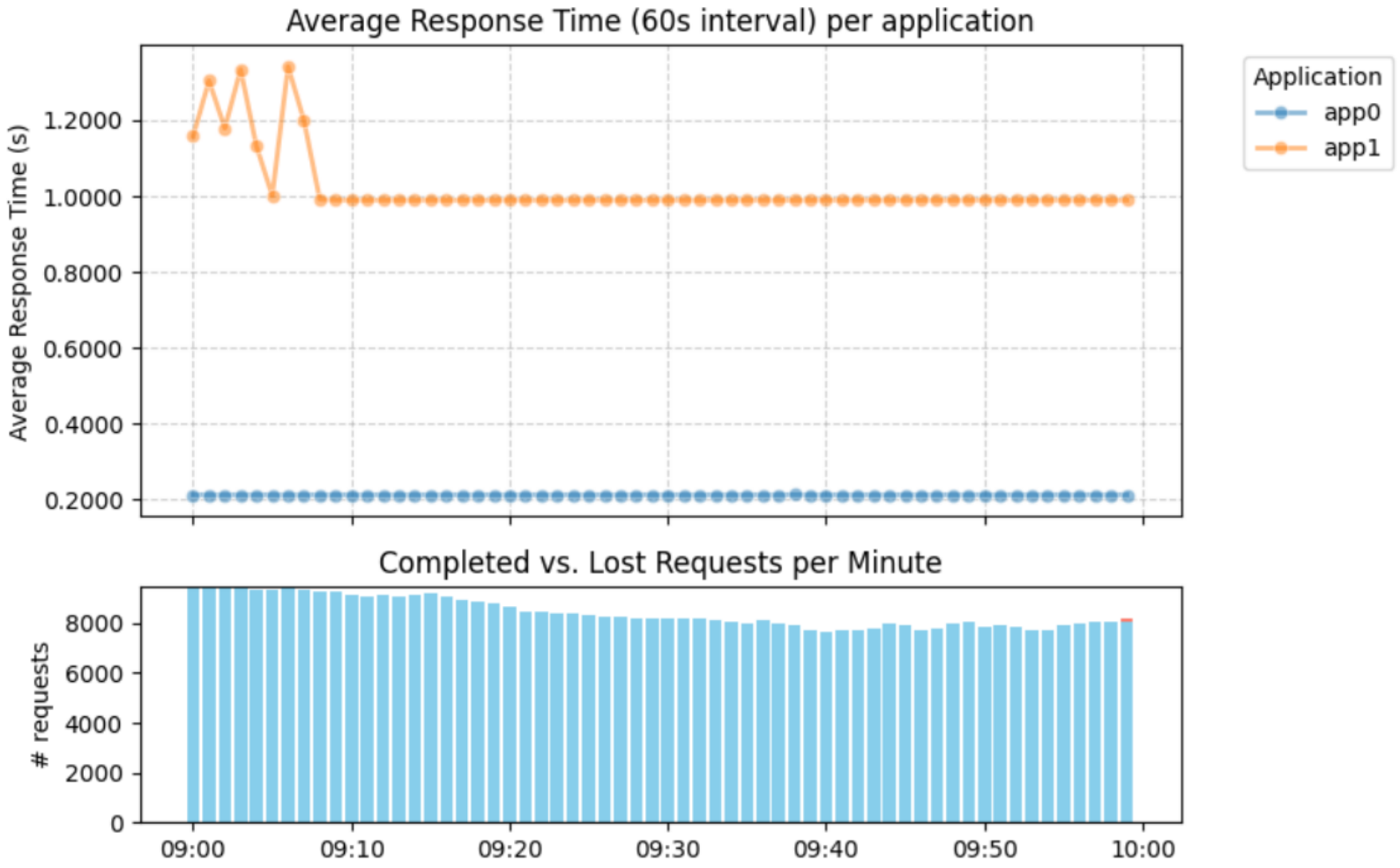
Experimental Validation



(a) Synthetic workload with Poisson arrivals

Q	Term	LB	Load
0	drain	RR	poisson trace
		SWRR	poisson trace
	hard	RR	poisson trace
		SWRR	poisson trace
1000	drain	RR	poisson trace
		SWRR	poisson trace
	hard	RR	poisson trace
		SWRR	poisson trace

Experimental Validation



(b) Trace based workload

Q	Term	LB	Load
0	drain	RR	poisson trace
		SWRR	poisson trace
	hard	RR	poisson trace
		SWRR	poisson trace
1000	drain	RR	poisson trace
		SWRR	poisson trace
	hard	RR	poisson trace
		SWRR	poisson trace

Conclusions

Conclusions

- New modular, extensible architecture for cloud simulation frameworks
  - Reproducible
  - Composable
- Experimental setup in version-controlled files
  - Aligns with FAIR (Findable, Accessible, Interoperable, and Reusable) principles
- Framework under development. Future work:
    - Network and I/O modules
    - Expand plugins: Autoscalers, LLMs, Spot instances…
    - Validate with large-scale comparative studies
    - Cloud-edge