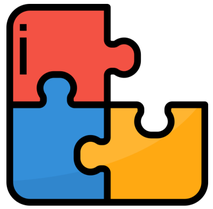
Simulating and Emulating Distributed Cloud-Edge Applications with Python

Jacopo Massa

May 29th, 2025

WHY?

Cloud-Edge continuum is ***heterogeneous***, ***dynamic*** and ***QoS-sensitive***



Simulation is ***not enough***

Lack of ***user-friendly*** and ***accessible*** tools that easily capture the environment complexity

WHAT?

DAY stributed

A Cloud-Edge ***Python platform***
for simulated/emulated runtime environments

Manage, ***monitor*** and ***test*** complex
Cloud-Edge scenarios

Allow the modelling of ***real-world***
infrastructures and application behaviours
(also ML-oriented)

ECLYPSE

GitHub

Paper

Massa J., De Caro V., Forti S., Dazzi P., Bacciu D., Brogi A.
ECLYPSE: a Python Framework for Simulation and Emulation of the Cloud-Edge Continuum.
arXiv, 2025.

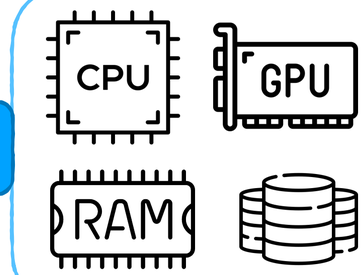
ECLYPSE: Simplifying Cloud-Edge simulations

Simulation

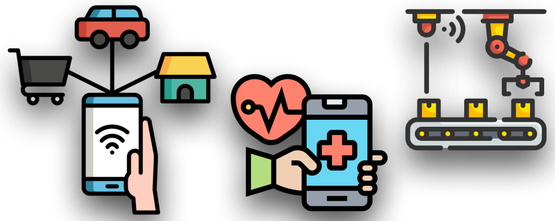
1

Assets

Node



Edge



Services | Interactions

Nodes | Links

2

Requirements & Capabilities

3

Events

Place

Update

Monitor

Report

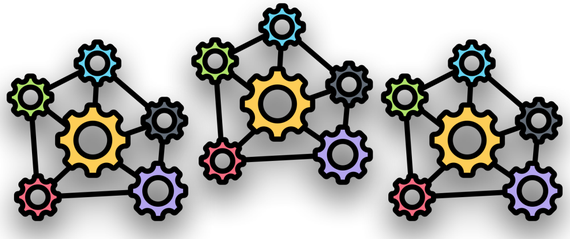
ECLYPSE: Simplifying Cloud-Edge simulations

Emulation

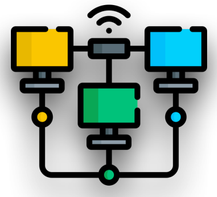
1

RemoteConfig

RAY



Logics | MPI, REST…



Actors | Delayed RMI

2

Remote Environment

3

Events

Place

Update

Remote Monitor

Report

ECLYPSE: Simplifying Cloud-Edge simulations

- GitHub repository
  - *public interface & ready-to-use examples*
- Off-the-shelf components
  - *placement strategies, update policies, metrics, assets, ...*
- Updated documentation (accessibility)
- easily extensible (modularity)
  - *customise events, metrics, strategies, policies ...*
  - *integrated with well-known third-party libraries (NetworkX, Ray, PyTorch, Pandas, ...)*

Try it now → `pip install eclypse`

Docs: eclypse.readthedocs.io

GitHub: github.com/eclypse-org/eclypse

```python
from eclypse.builders.application import get_sock_shop
from eclypse.builders.infrastructure import hierarchical
from eclypse.placement.strategies import BestFitStrategy
from eclypse.simulation import (
    Simulation,
    SimulationConfig,
)

SEED = 42

node_assets = ["cpu", "gpu", "ram", "storage"]
edge_assets = ["latency", "bandwidth", "availability", "security"]

app = get_sock_shop(seed=SEED, node_assets=node_assets, edge_assets=edge_assets)
infrastructure = hierarchical(SEED, node_assets=node_assets, edge_assets=edge_assets)
strategy = BestFitStrategy()

sim_config = SimulationConfig(
    seed=SEED,
    path="my-example",
    timeout=300,
    log_to_file=True,
)

sim = Simulation(infrastructure=infrastructure, simulation_config=sim_config)
sim.register(app, strategy)

sim.start()
sim.wait()
```

How would your algorithm behave in the real world?

Use ECLYPSE as your experimental testbed

Inject your algorithm

Test your own placement, scheduling, consensus or replication strategies.
Simulate dynamic resources, failures, latency and churn.

Prototype autonomy

Deploy services as Ray actors with local policies and partial knowledge.
What if your nodes make decisions on their own?

Shape the benchmark

Join the discussion to define meaningful, reproducible benchmarks
for distributed and parallel algorithm robustness.

ECLYPSE

THANK YOU FOR YOUR ATTENTION!

Jacopo Massa
jacopo.massa@di.unipi.it
pages.di.unipi.it/massa