Trace Generation and Closed-Loop Digital/Physical Twin for the Digital Continuum

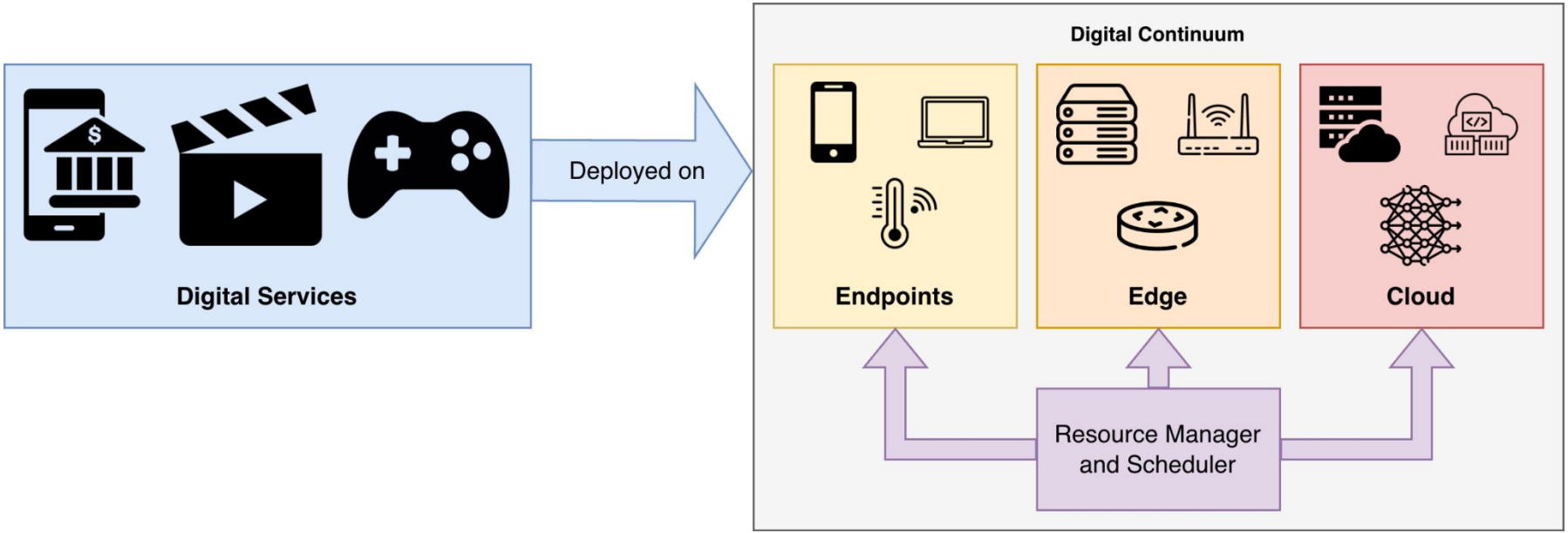
Man Yin Edward Chui

MSc Thesis Defense, 29 June 2026

1st supervisor:	Tiziano De Matteis
Daily supervisor:	Matthijs Jansen
2nd reader:	*Pending*

Context: digital services and digital continuum

Digital services are increasingly deployed across the ***digital continuum***.

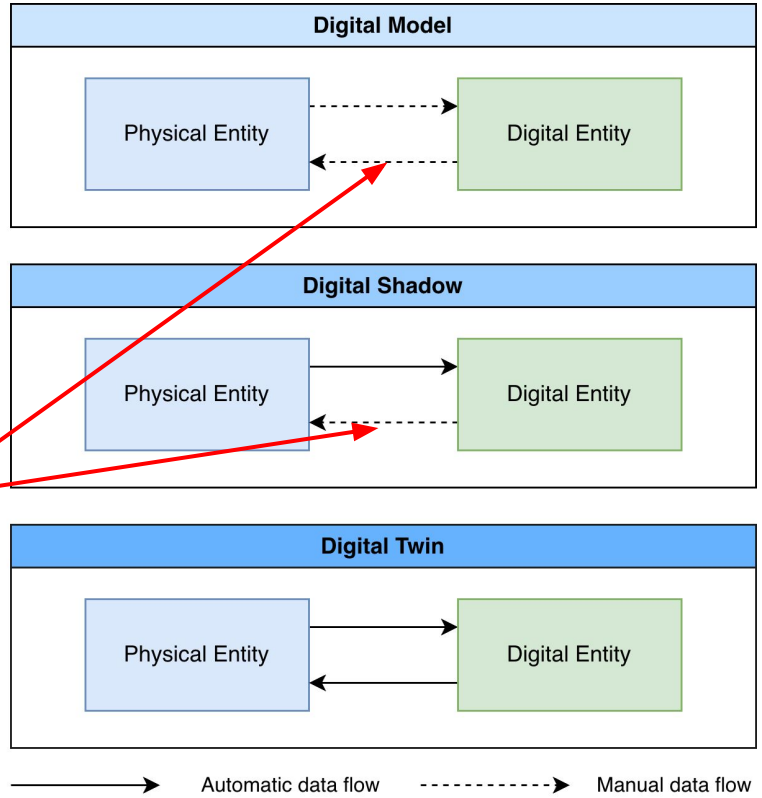


Need **evidence-based** decision support for resource management that is quick and safe.

Context: digital twin as decision support mechanism

Digital Twin (DT): connects a physical system with a virtual representation, that enables:

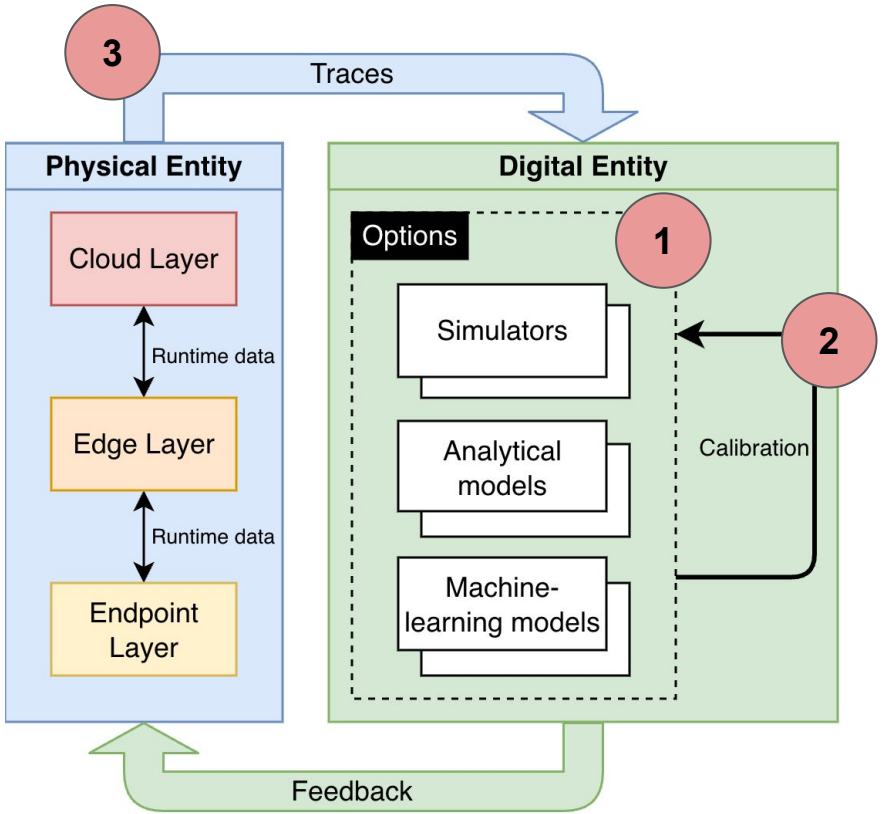
- Mirroring and monitoring
- Replaying observed execution
- Improving reproducibility
- Answer “what-if” questions



Require human-in-the-loop for feedback

Research Gap
Truly closed-loop DT remains scarce across domains.

Background and scope



1 **Simulator as the digital entity**

✅ Fidelity
✅ Cost & Speed
✅ Repeatability

2 **Calibration for simulation fidelity**

Having a simulator alone is not sufficient.
How to calibrate the simulator?

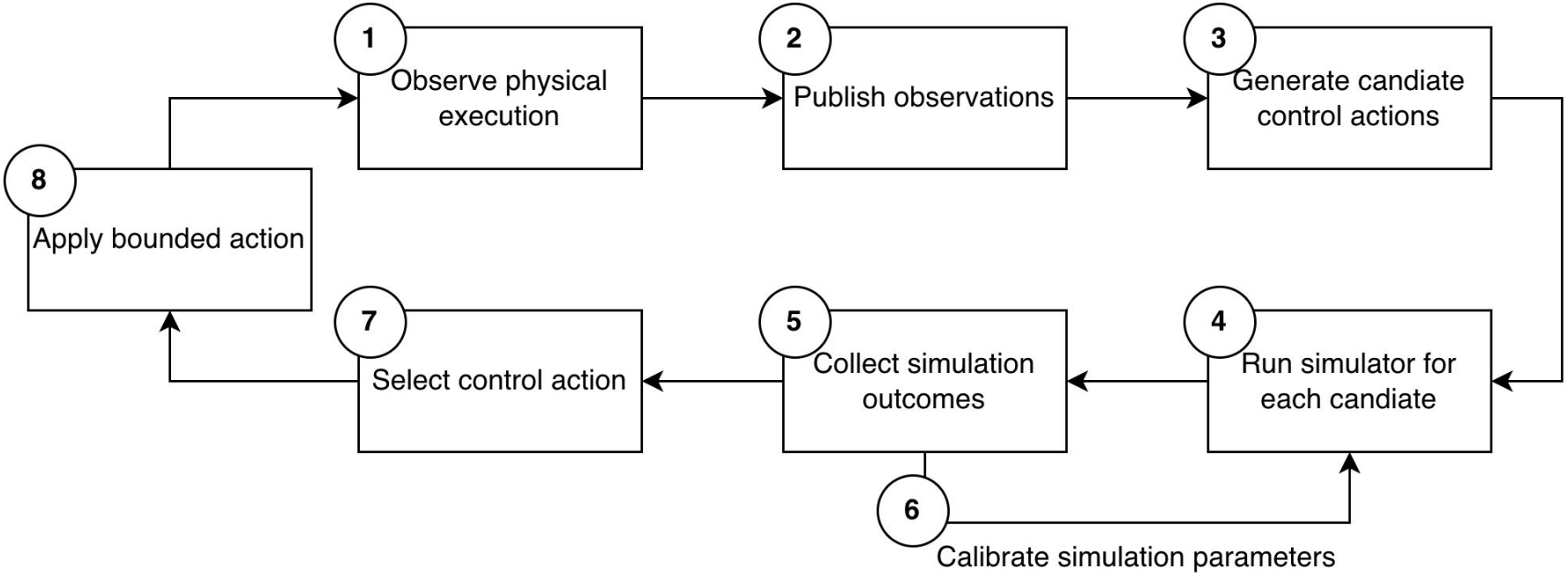
3 **Execution trace as the interface**

Expose *enough* information for reproduce and evaluate execution.
What is the semantics required?

Research questions

- **RQ1**: How to **design** a closed-loop digital/physical twin architecture for the digital continuum?
  - ➢ **C1**: Closed-loop digital twin architecture design
- **RQ2**: How to **operationalize** a trace-based simulation workflow to support closed-loop decision making?
  - **RQ2a**: What trace **schema** and semantics are required?
  - **RQ2b**: How can the simulator be **calibrated**?
  - **RQ2c**: How can simulator output answer **what-if questions** and be converted into actionable control decisions?
  - ➢ **C2**: Trace-to-simulation-to-action methodology
- **RQ3**: What is the **decision-support value** of the closed loop?
  - ➢ **C3**: Quantitative evaluation

Design process: conceptual workflow



Inspired by the **MAPE-K** (Monitor, Analyse, Plan, Execute using Knowledge) loop, we derive 8 steps for the high-level closed-loop workflow, which becomes part of the *functional requirements*.

Design requirements: non-functional requirements

Fit-for-purpose fidelity
The simulator shall be accurate enough, but not attempting to reproduce every details

Modularity, portability and extensibility
Components shall communicate through explicit interfaces so that they can be easily replaced

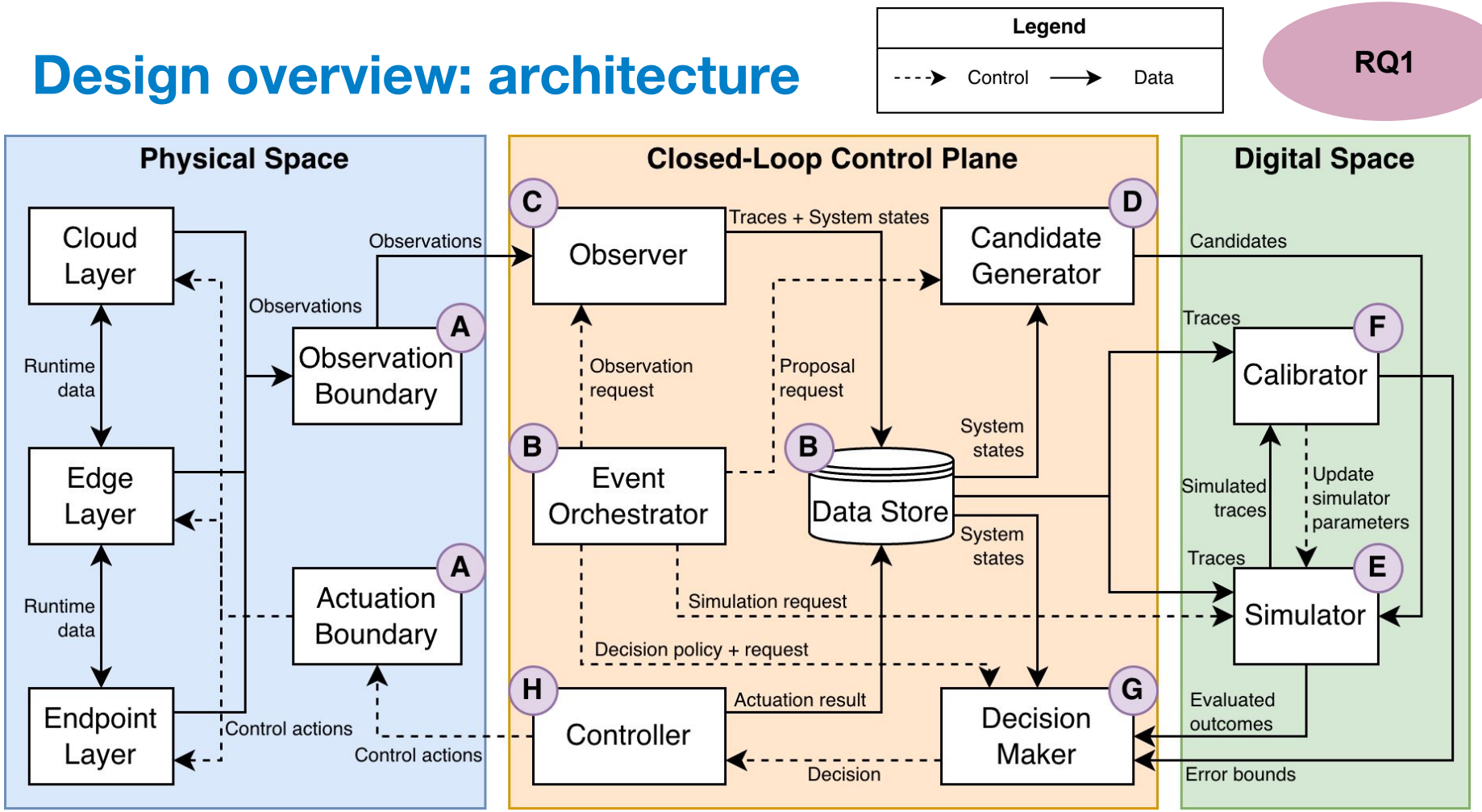
Bounded and safe actuation
Control action shall be restricted to supported operations of the physical entity

Timely and state-consistent
Stale or mismatched evaluation results shall not drive physical actions

Reproducibility
The system shall record observations and decisions to inspect and reproduce a closed-loop iteration

Design overview: architecture

RQ1



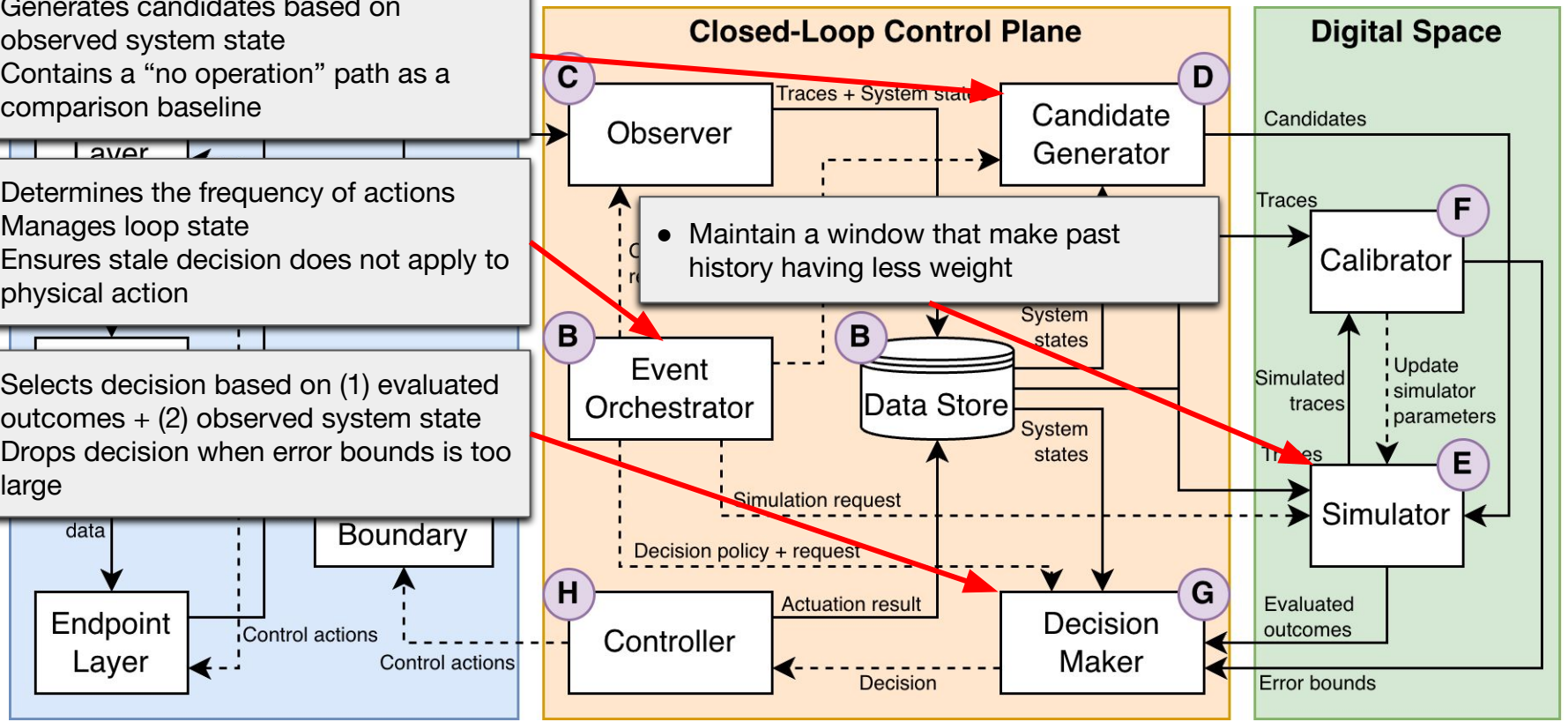
Design overview: key components highlight

RQ1

- Generates candidates based on observed system state
- Contains a “no operation” path as a comparison baseline

- Determines the frequency of actions
- Manages loop state
- Ensures stale decision does not apply to physical action

- Selects decision based on (1) evaluated outcomes + (2) observed system state
- Drops decision when error bounds is too large



Realization: framework



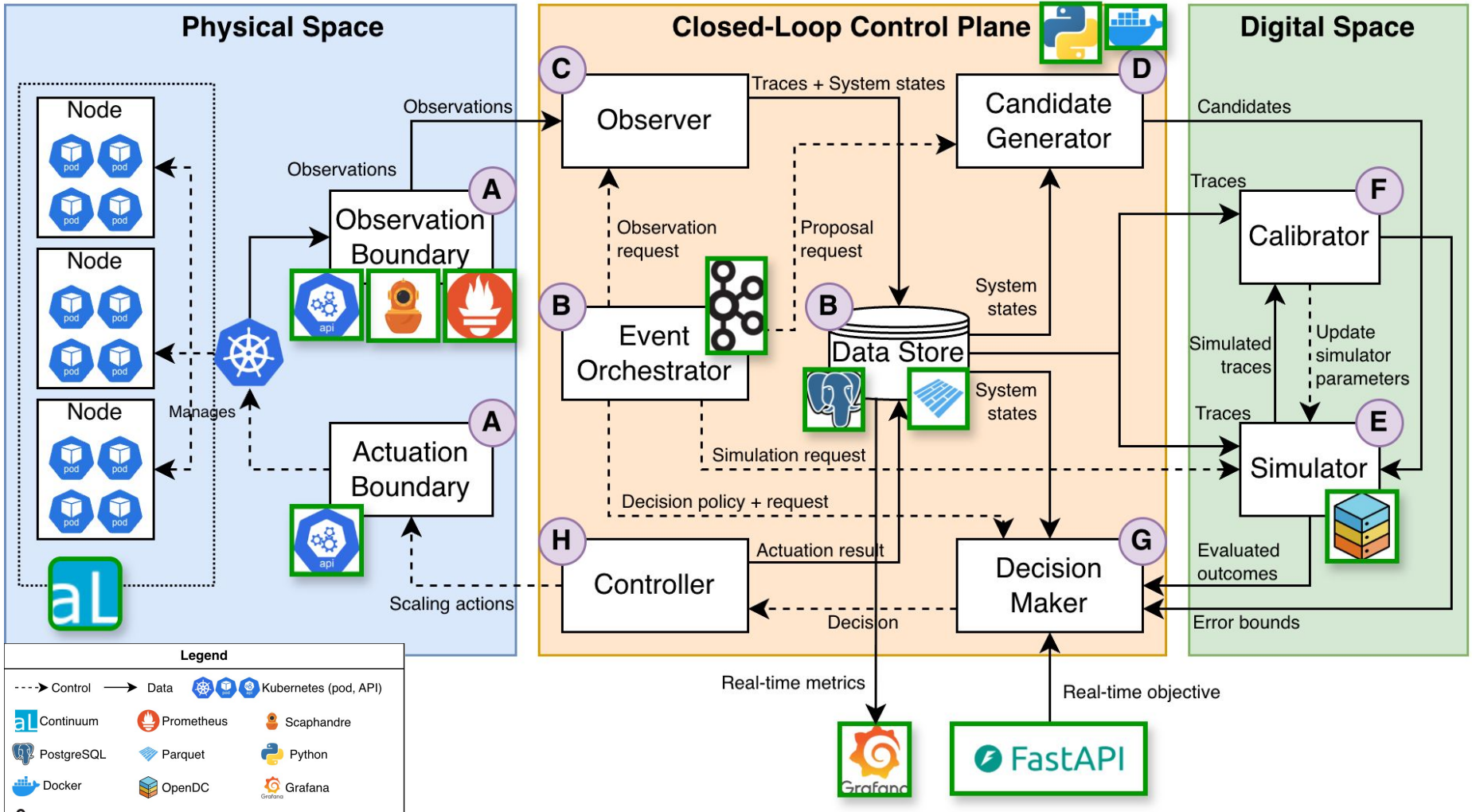

Core Technologies

- **Physical space**: Continuum + Kubernetes
- **Observation interface**: Prometheus + Scaphandre
- **Actuation interface**: Kubernetes API
- **Digital space (simulator)**: OpenDC

Other Technologies

- **Event streaming**: Kafka
- **Data store**: PostgreSQL + Parquet
- **Human interface**: Grafana, FastAPI
- Based on work of ***OpenDT***
  - It is a *digital shadow* that replay SURF's historical execution traces

What-if question to answer
If we add/remove node(s) from the system, does it improve runtime/power consumption?


Physical Space
Node
Node
Node
Observations
Observations
Observation Boundary
A
Manages
Actuation Boundary
A
Scaling actions
Closed-Loop Control Plane
C
Observer
Traces + System states
D
Candidate Generator
Observation request
Proposal request
B
Event Orchestrator
B
Data Store
System states
System states
Simulation request
Decision policy + request
H
Controller
Actuation result
G
Decision Maker
Decision
Digital Space
Candidates
Traces
F
Calibrator
Simulated traces
Update simulator parameters
Traces
E
Simulator
Evaluated outcomes
Error bounds
Real-time metrics
Real-time objective
Grafana
FastAPI
Legend
Control
Data
Kubernetes (pod, API)
Continuum
Prometheus
Scaphandre
PostgreSQL
Parquet
Python
Docker
OpenDC
Grafana
Kafaka
FastAPI
FastAPI

Realization: trace contract

RQ2a

Physical source	Trace record	Key parameters	Why it is needed
Kubernetes pod lifecycle	Task	`id, submission_time, duration, cpu_count, cpu_capacity, memory_capacity`	Replay workload arrival, duration, CPU/memory demand
Resource samples	Fragment	`task_id, duration, cpu_usage`	Capture time-varying CPU usage
Power readings	Consumption	`timestamp, power_draw`	Compare/calibrate simulated power
System state	Topology	`state_id, timestamp, host_count, core_count, memory_size`	Describe available and in-use resources for what-if simulation
Orchestrator state	State metadata	`state_id, candidate_id`	Keep simulation input reproducible and state-consistent

The schema is sufficient as the same records support *replay, calibration and what-if evaluation*.

Realization: calibration

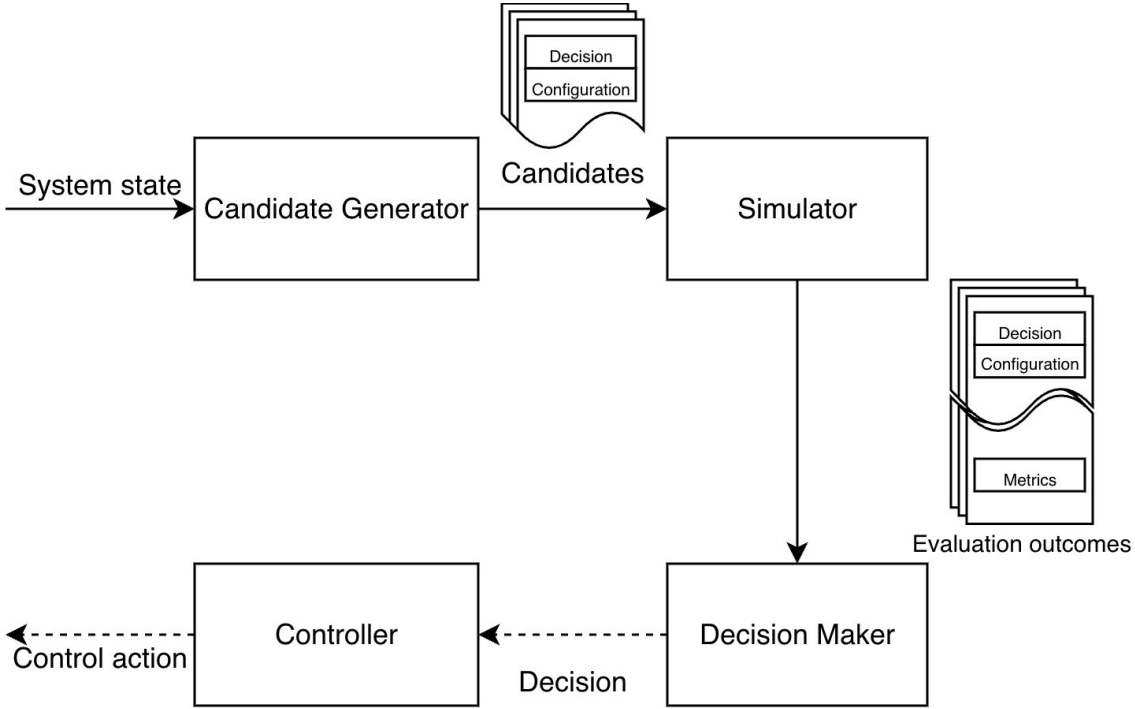
RQ2b

TODO

- Reused calibration idea from **OpenDT**
  - Grid search strategy over the parameter space
- New challenges: How to manage constantly changing topology and work with calibration at the same time
  - Only tune baseline candidate (The topology currently in use)
  - Record the calibrated parameter for each topology the calibration has visited
- Steps:
  - Identify tunable parameters
  - Gather enough data for tuning
  - Maintain (Topology, parameters) mapping

Realization: decision/action

RQ2c



TODO

- What happening in the actual system is more important than the simulation results
- Flowchart in ranking and selecting the candidate?

Evaluation setup

TODO

- Fibonacci workload as Poisson process to mimic real-word workload
- Workload from low → medium → high → medium → low
- 10 setup x 3 replications

Preliminary findings (Not in present, for self reference)

TODO

- Most experiment result matches with expectation:
  - (3) & (4), (8) & (9) shows comparable total runtime
- Some findings does not match expectation:
  - All settings with calibration, MAE increases
    - May need to fine tune the calibration parameters
  - Energy objective (5, 10) does not show significant energy reduce
    - Maybe the pending pod decision is too frequent

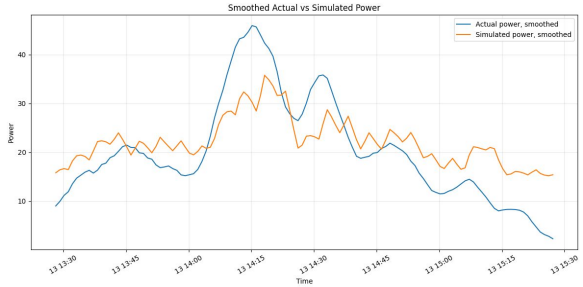
Settings	
(1) 8 Static	(6) 8 Static with calibration
(2) 4 Static	(7) 4 Static with calibration
(3) Baseline autoscaler (Scale based on pending pod)	(8) Baseline autoscaler with calibration
(4) Closed-loop OpenDT (Runtime objective)	(9) Closed-loop OpenDT with calibration
(5) Closed-loop OpenDT (Energy objective)	(10) Closed-loop OpenDT with calibration

Setting	Total time	Average wait time	Total power(J)	MAE Power
(1)	02:02:57	00:00:02.432432	145874.389727	7.31
(2)	02:15:10	00:10:30.443243	129504.668002	5.83
(3)	02:03:20	00:01:00.916216	143119.255054	8.29
(4)	02:03:39	00:02:25.140540	137011.006276	8.39
(5)	02:10:08	00:04:10.851351	142345.05035700003	6.94
(6)	02:02:55	00:00:02.481081	145611.845225	7.62
(7)	02:17:46	00:10:25.089189	129549.610739	7.65
(8)	02:03:14	00:01:00.797297	141380.612938	9.60
(9)	02:03:42	00:02:09.427027	138577.294111	8.62
(10)	02:03:59	00:04:00.391891	139578.834438	9.05

Experiment 1: simulator fidelity

TODO: real figure. Figure shall be large

- Validation Experiment: Is the simulator (OpenDC) accurate?
- Current observation:
  - **Simulator may only see what have happened (trace), but not what is happening (running execution)** → There can be drift and delay, using MAE/MAPE may not be fair comparison
    - Will fine tune for one last time
    - We may need some error measurement method that capture the shape & characteristic of the power graph to show the simulator fidelity is enough
  - The gap between actual and simulation may come from the assumption, as we simply assume all machines uses the same energy model
    - N=8, not large enough to dilute the noise and some other factors



Experiment 1: simulator fidelity

TODO

Observation 1
Simulator may only see what have happened, but not what is happening

Experiment 2: calibration

TODO: figure

- Evaluation Experiment: Does calibration design helps improve simulation accuracy?

Observation X

Experiment 3: closed-loop value

RQ3

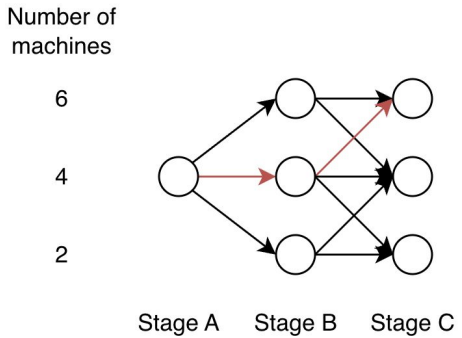
TODO: figure

- Evaluation Experiment: Does the closed-loop DT provide better/similar decision support than the baseline?

Experiment 4: decision maker

TODO: run if there is still time

- Validation Experiment: The decision maker pick the best decision out of the candidates
- Another set of setup, more shorter and simpler
- Run 3x7 combinations configuration, select the path that gives the best outcome in terms of runtime/power usage
- Then run our implementation to compare the decision making



Summary

Take home messages:

TODO

Image Sources

- Flaticon https://www.flaticon.com/
  - Authors: Freepik, SumberRejeki, Andrean Prabowo, afif fudin, Good Ware, Pixel perfect, Ary Prasetyo, Afian Rochmah Afif, Ndut_Design, SyafriStudio