Towards Holistic Continuous Software Performance Assessment

@QUDOS 2017

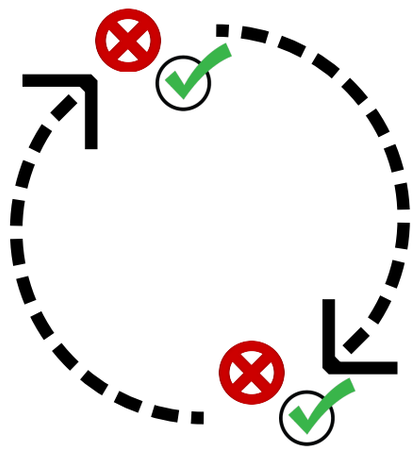
Università della Svizzera italiana

Facoltà di scienze informatiche

http://benchflow.inf.usi.ch

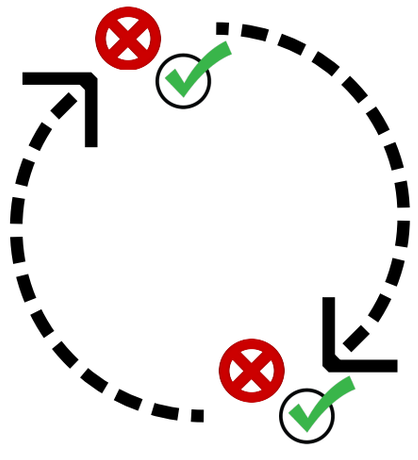
Vincenzo Ferme, Cesare Pautasso

Software Development Nowadays



Continuous
Feedback

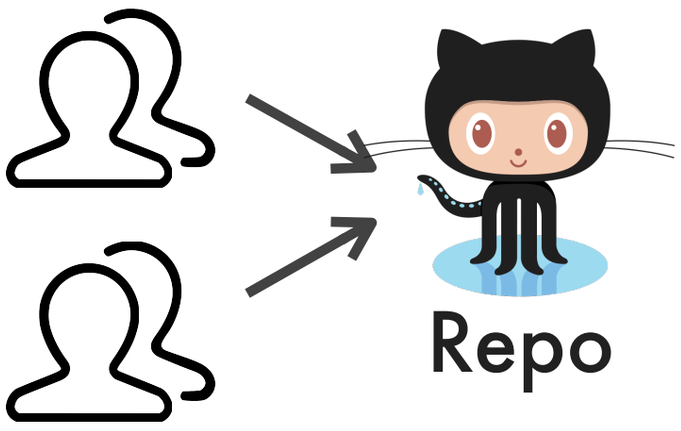
Software Development Nowadays



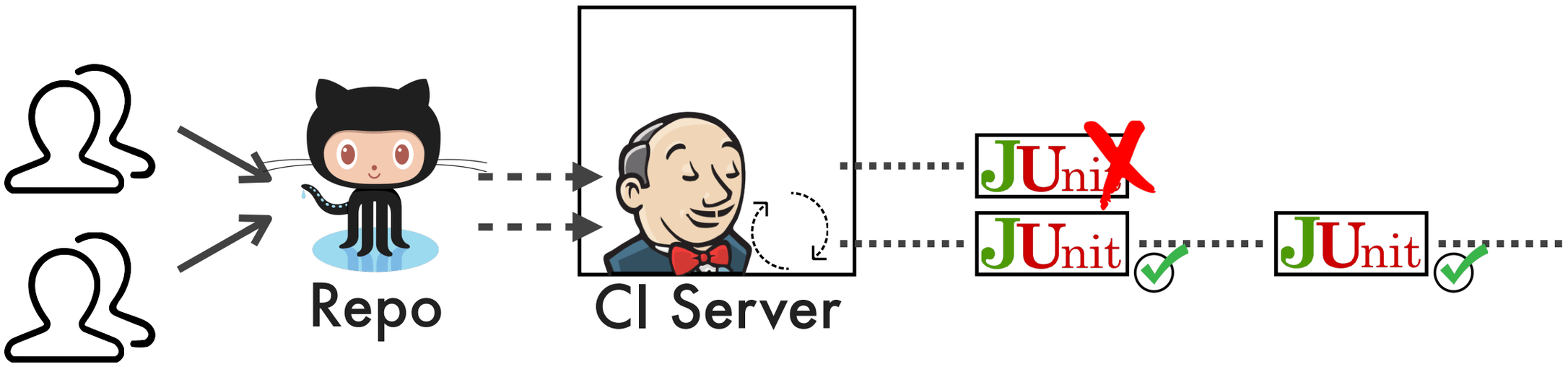
Continuous Feedback

DevOps

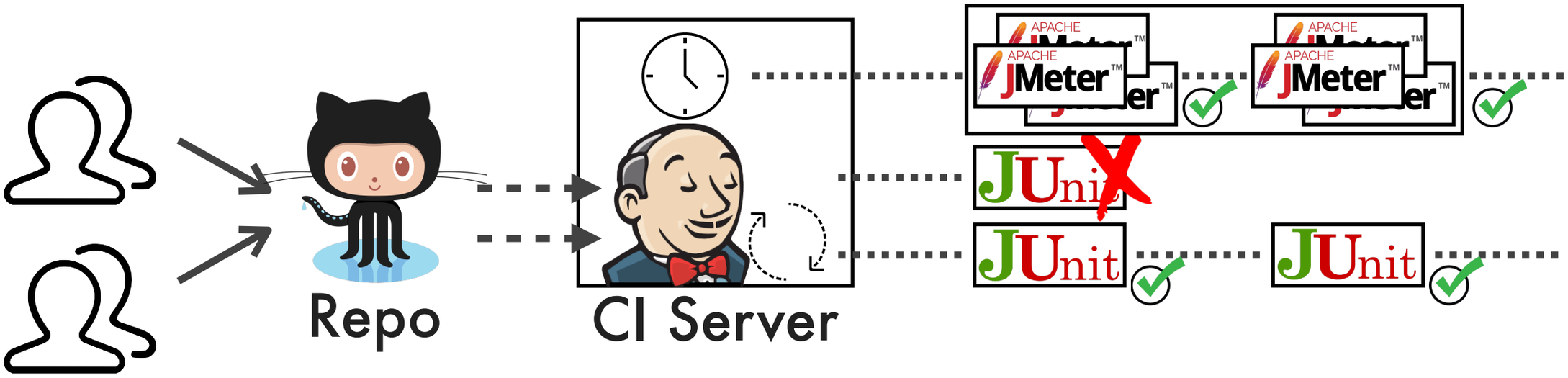
Continuous Feedback



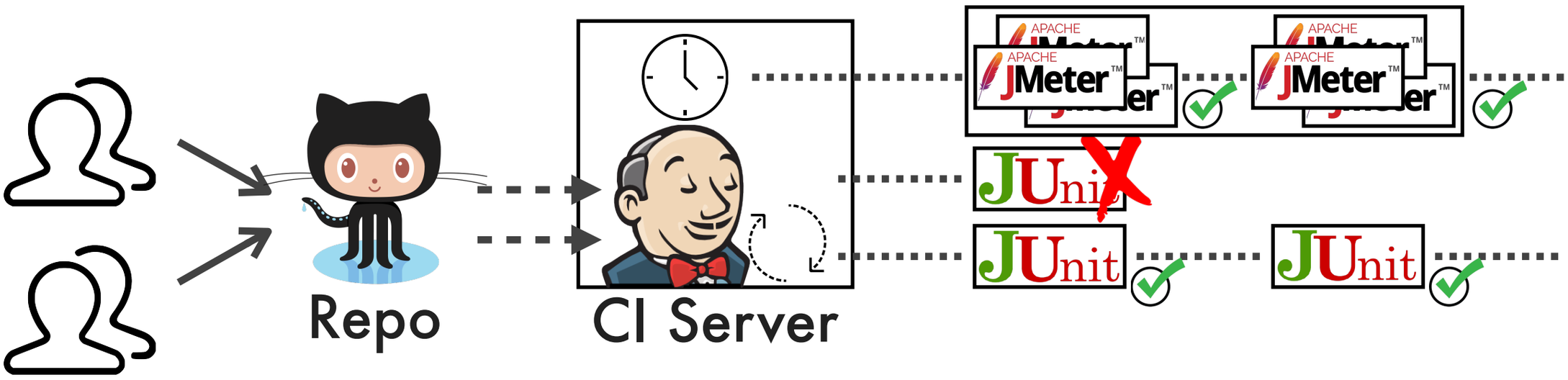
Continuous Feedback

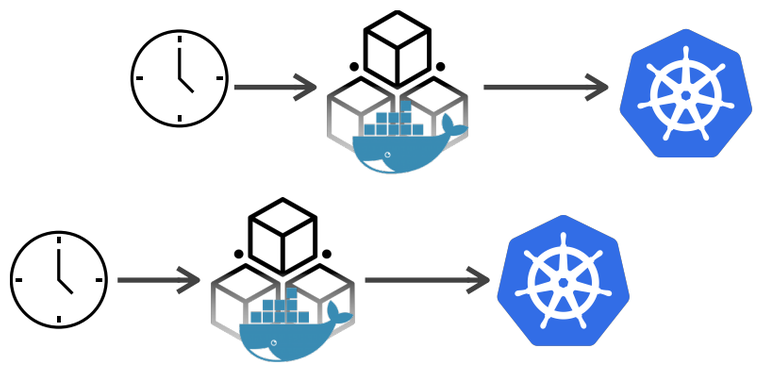


What about performance?



What about performance?





Continuous Deployment

Existing Performance Engineering Tools

particularly open-source ones

Existing Performance Engineering Tools

particularly open-source ones

Existing Performance Engineering Tools

particularly open-source ones

Existing Performance Engineering Tools

particularly open-source ones

DataMill

Existing Performance Engineering Tools

particularly open-source ones

DataMill

Cloud WorkBench

Existing Performance Engineering Tools

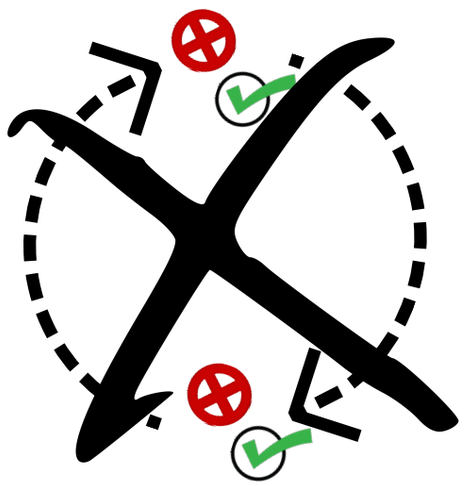
particularly open-source ones

DataMill

Cloud WorkBench

inspectIT

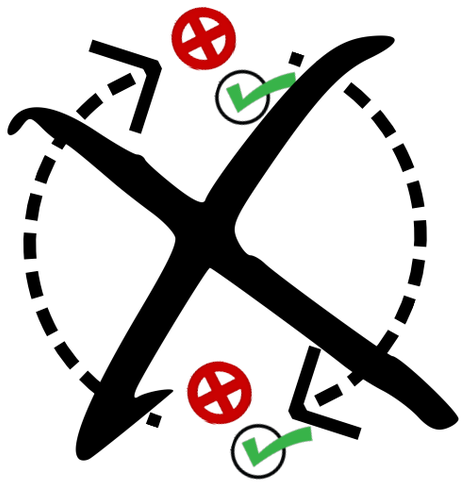
Limitations of Current Solutions



Not integrated in CSA
(E.g., Do not Leverage Continuous Feedback)

CSA: Continuous Software Assessment

Limitations of Current Solutions

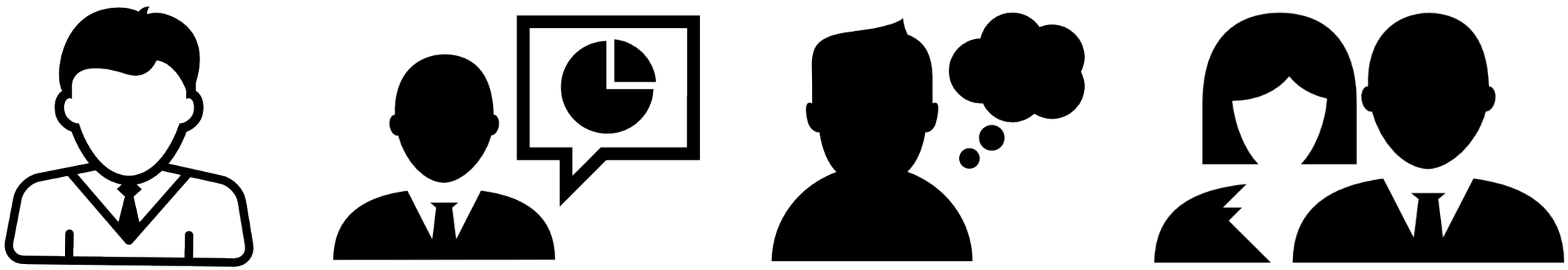


Not integrated in CSA
(E.g., Do not Leverage Continuous Feedback)

Rarely Automating the End-to-End Process

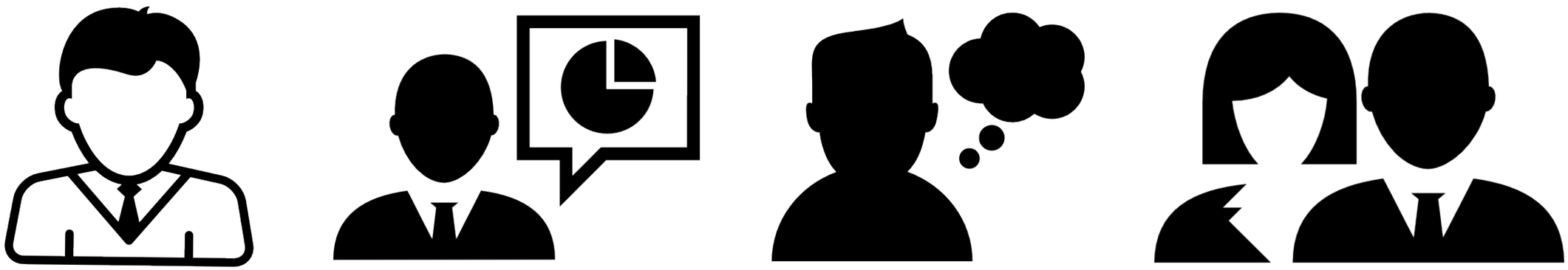
CSA: Continuous Software Assessment

DevOps: Professional Profiles Perspective

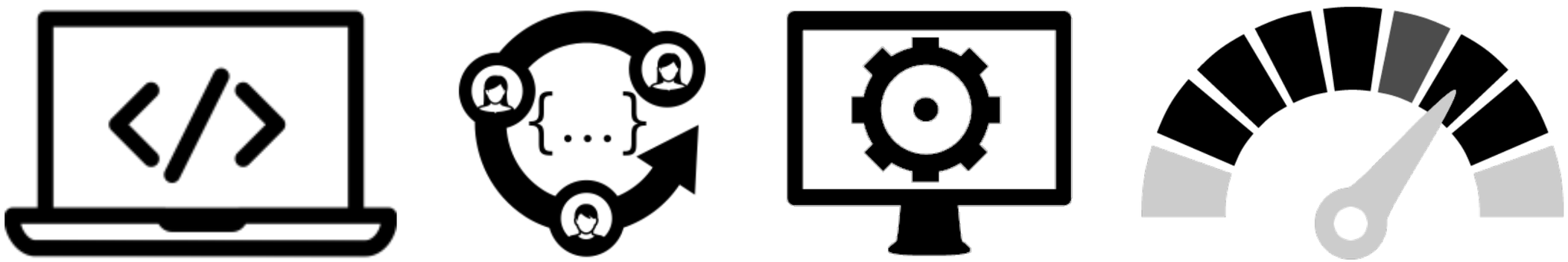


Different Professional Profiles
(e.g., Developers, Q/A and Operations Engineers)

DevOps: Professional Profiles Perspective



Different Professional Profiles
(e.g., Developers, Q/A and Operations Engineers)



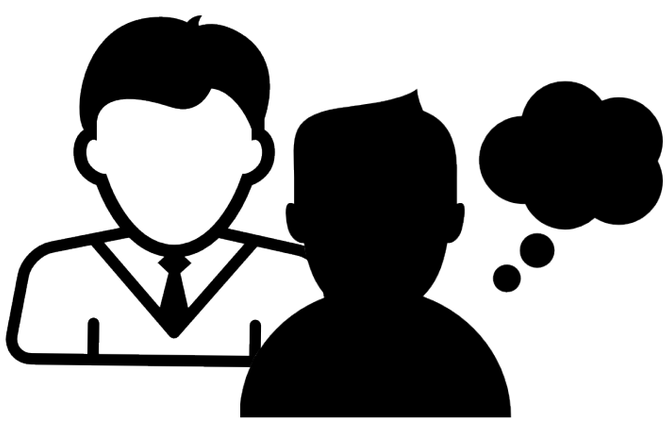
Heterogeneous and Cross-Cutting Skills

Vision

Holistic Continuous Software Performance Assessment

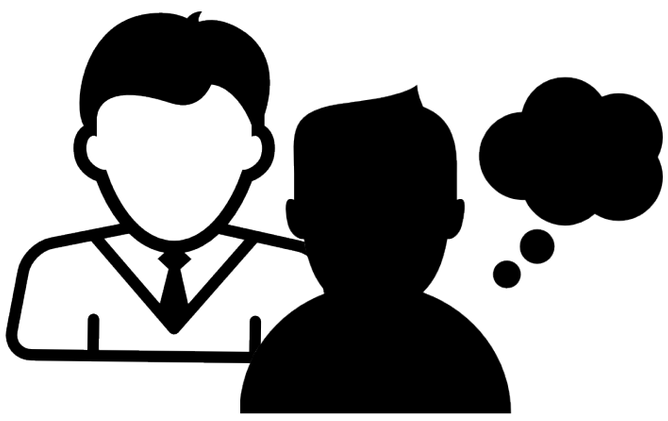
Vision

Holistic Continuous Software Performance Assessment



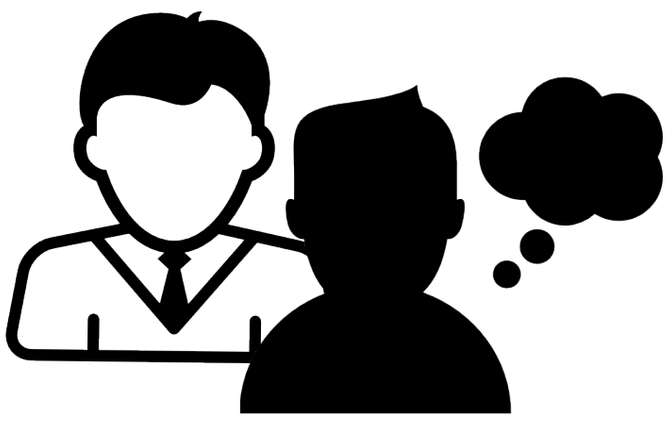
Vision

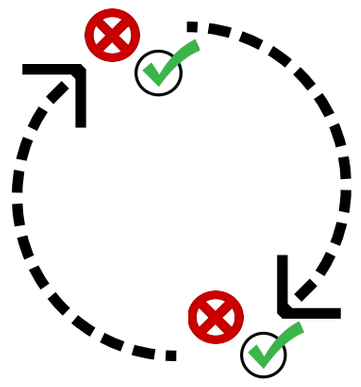
Holistic Continuous Software Performance Assessment



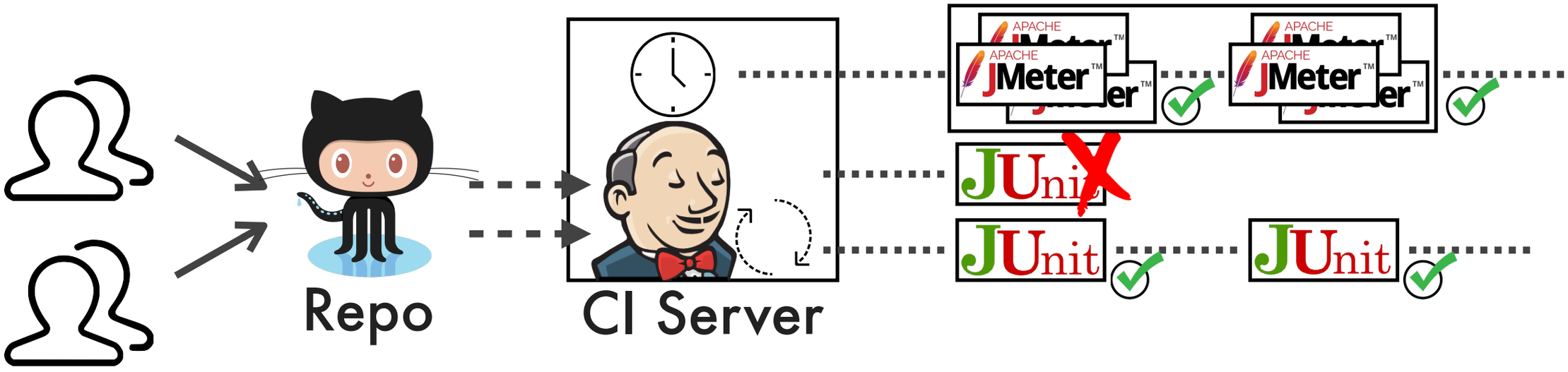
Vision

Holistic Continuous Software Performance Assessment

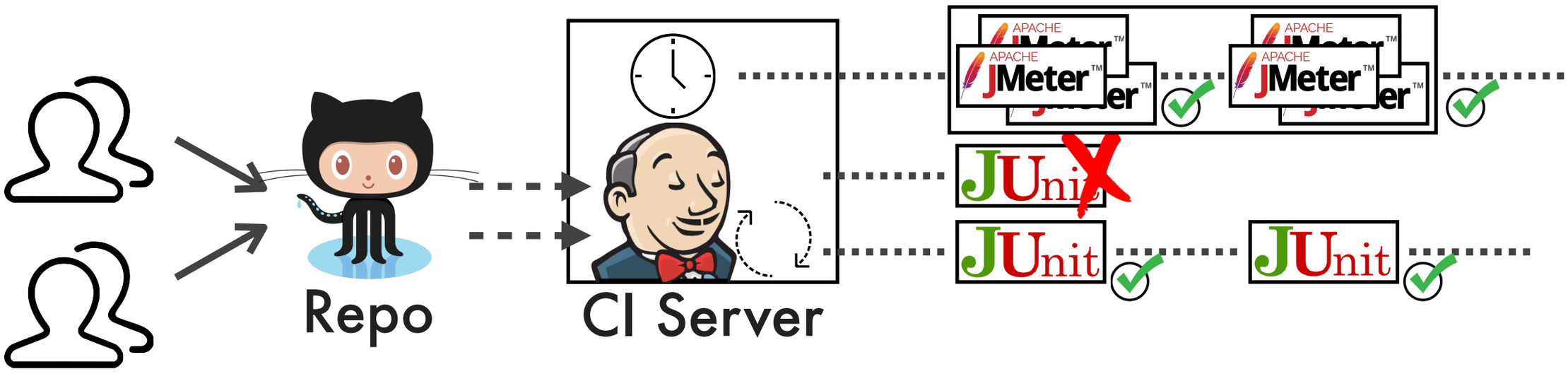




Approach Overview

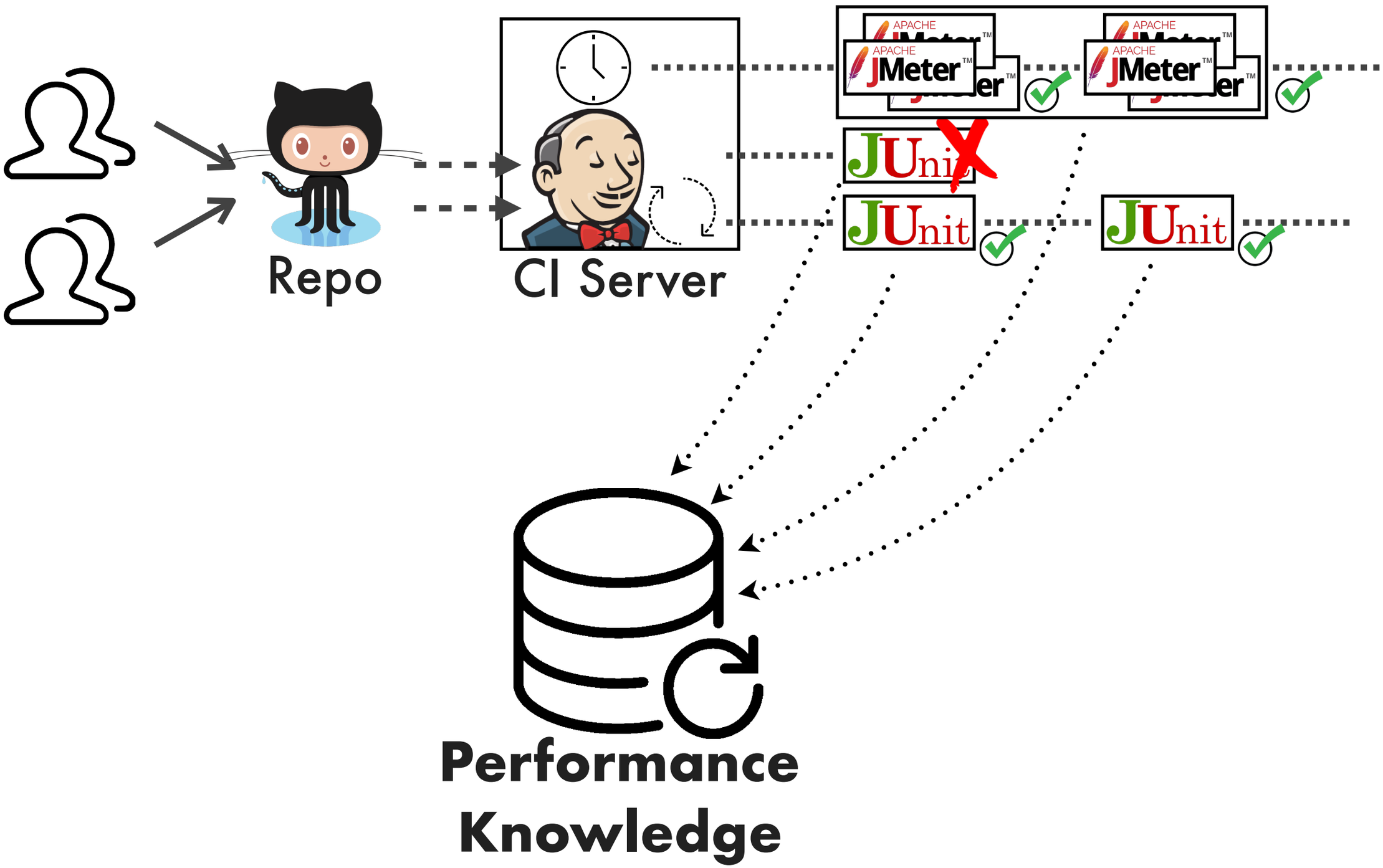


Approach Overview



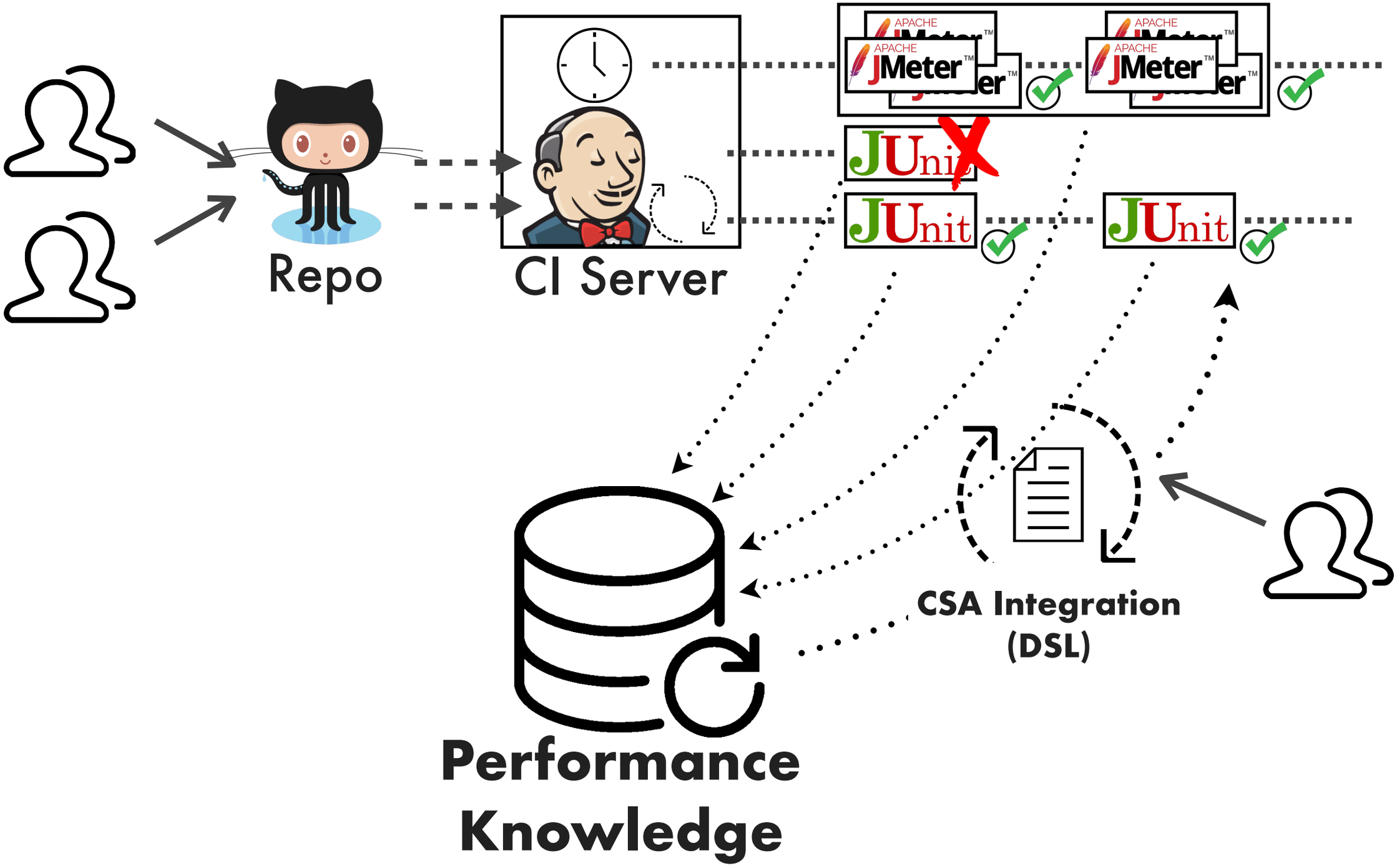
3 Main Features

Approach Overview



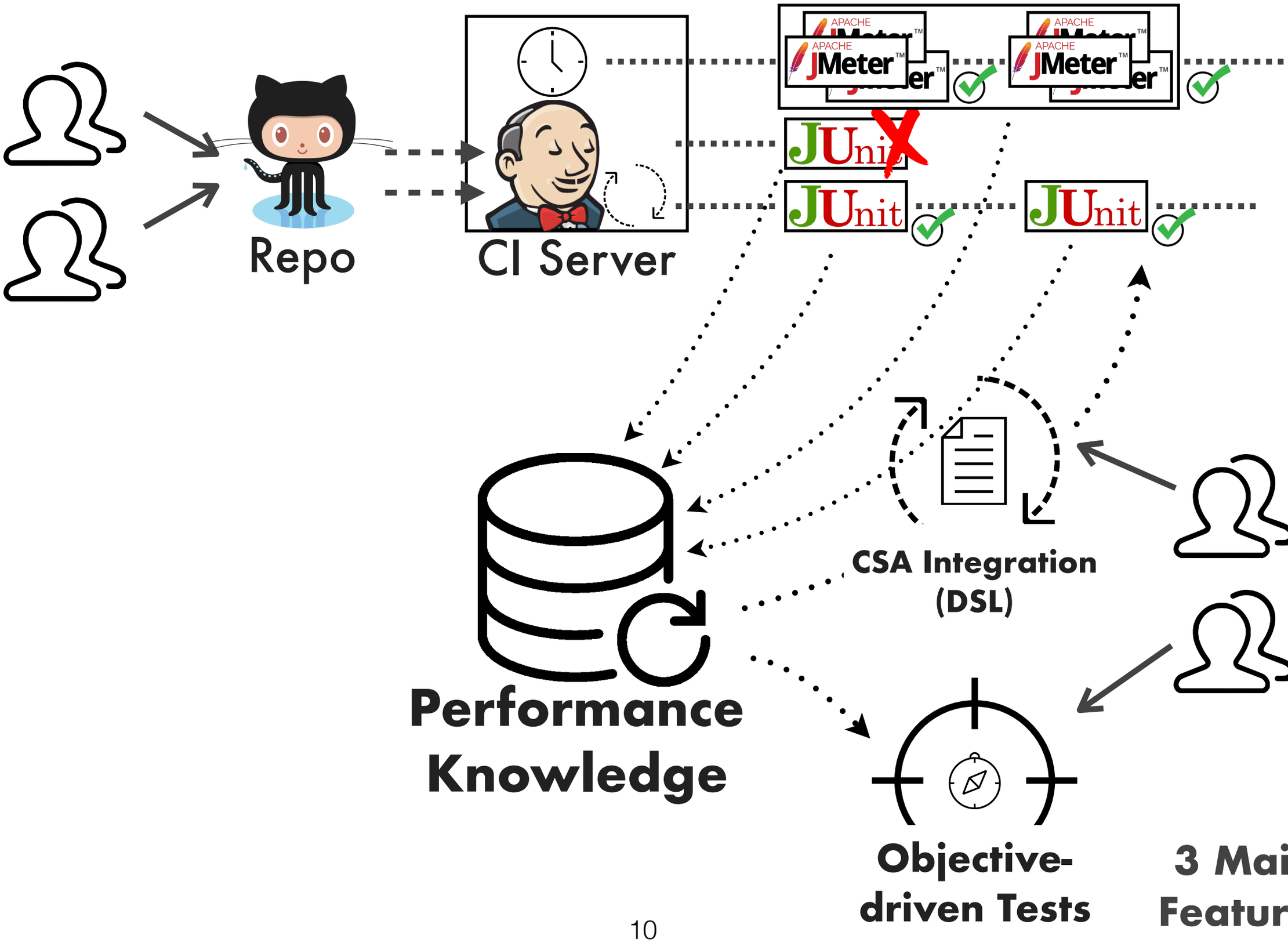
3 Main Features

Approach Overview

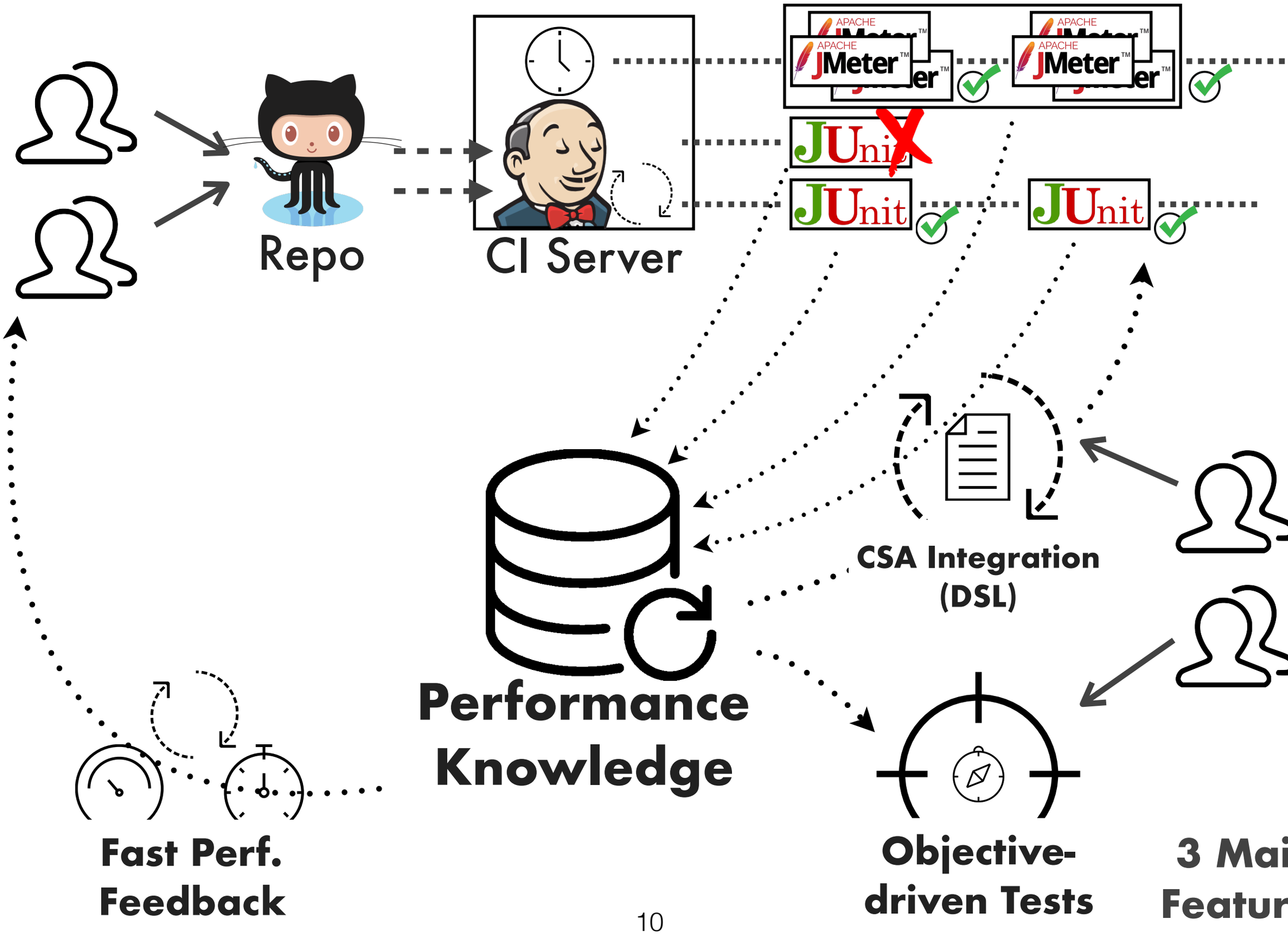


3 Main Features

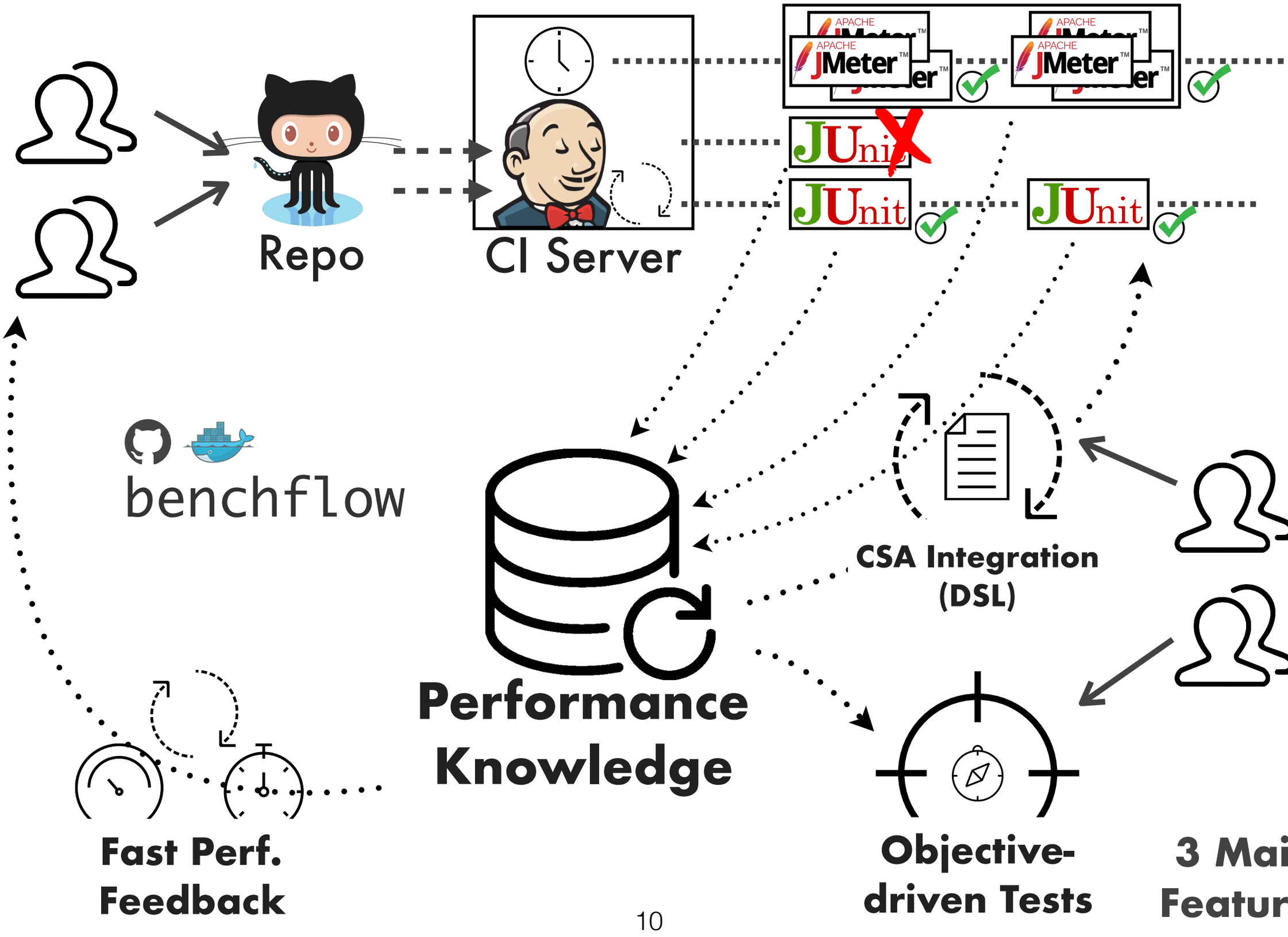
Approach Overview



Approach Overview



Approach Overview

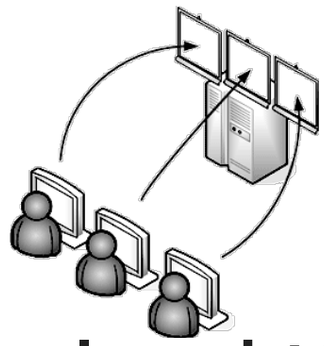


Integration in Development Lifecycles (DSL)

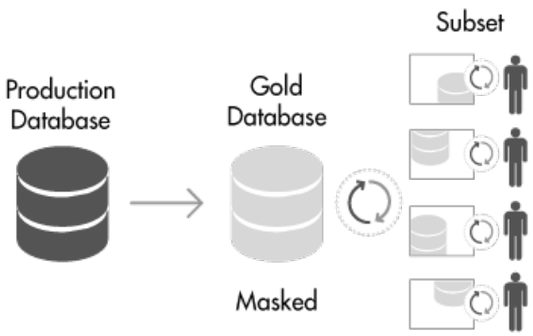
DSL Overview (Literature)

Load Functions

Workloads



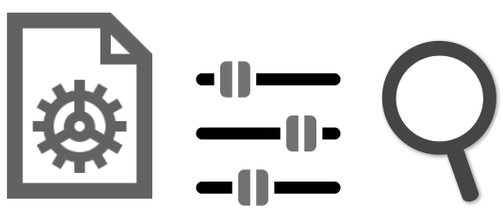
Simulated Users



Test Data

TestBed Management

Client-side Perf. Data Analysis



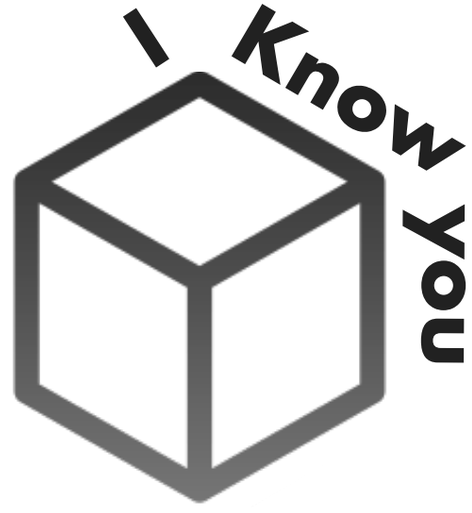
Definition of Configuration Tests

Main DSL Features

Integration in CSA

Main DSL Features

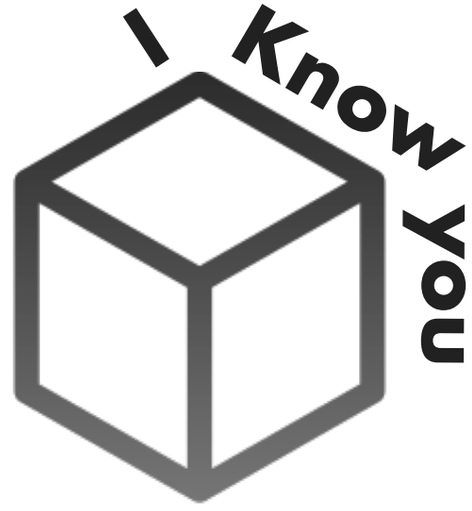
Integration in CSA



SUT-awareness

Main DSL Features

Integration in CSA

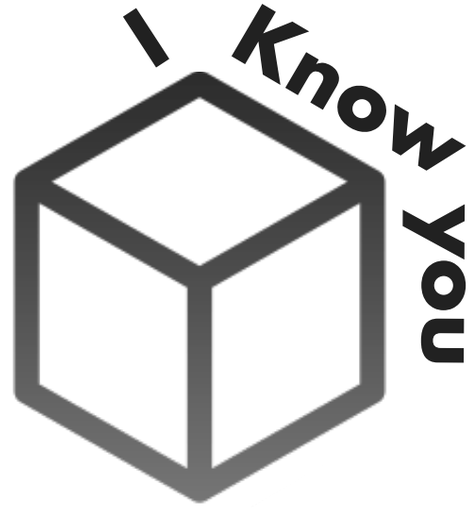


SUT-awareness

Collection and Analysis of Performance Data

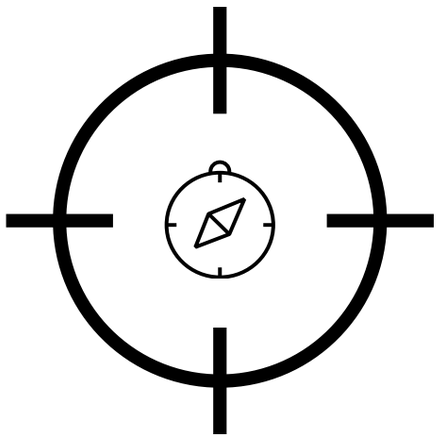
Main DSL Features

Integration in CSA



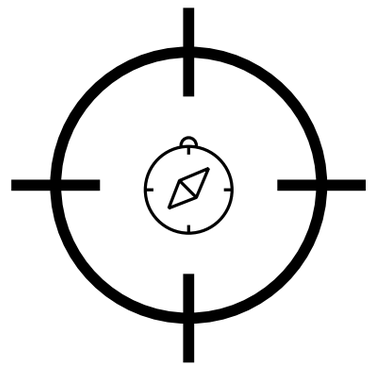
SUT-awareness

Collection and Analysis of Performance Data



Objective-Driven Performance Testing

Objective-Driven Performance Testing



Objectives Taxonomy

Base Objectives (Test Types)

standard performance tests, e.g., load test, stress test, spike test, and **configuration test**

Objectives Taxonomy

Base Objectives (Test Types)
standard performance tests, e.g., load test, stress test, spike test, and **configuration test**

Objectives
specific types of performance engineering activities, e.g., capacity planning and performance optimisations

Objectives Taxonomy

Base Objectives (Test Types)
standard performance tests, e.g., load test, stress test, spike test, and **configuration test**

Objectives
specific types of performance engineering activities, e.g., capacity planning and performance optimisations

Meta-Objectives
defined from already collected performance knowledge, e.g., comparing different systems using a benchmark

Example: Configuration Test

objective:

type: configuration

observation:

- ...

exploration_space:

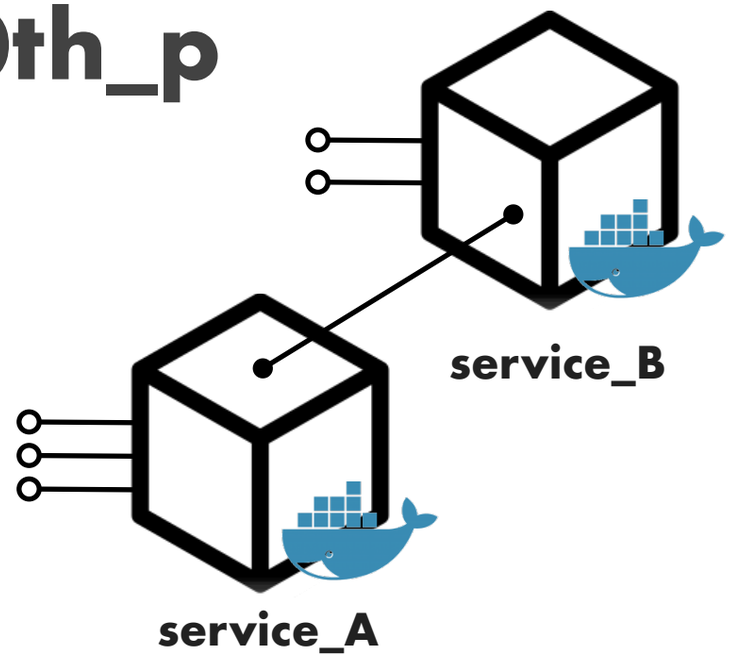
- ...

termination_criteria:

- ...

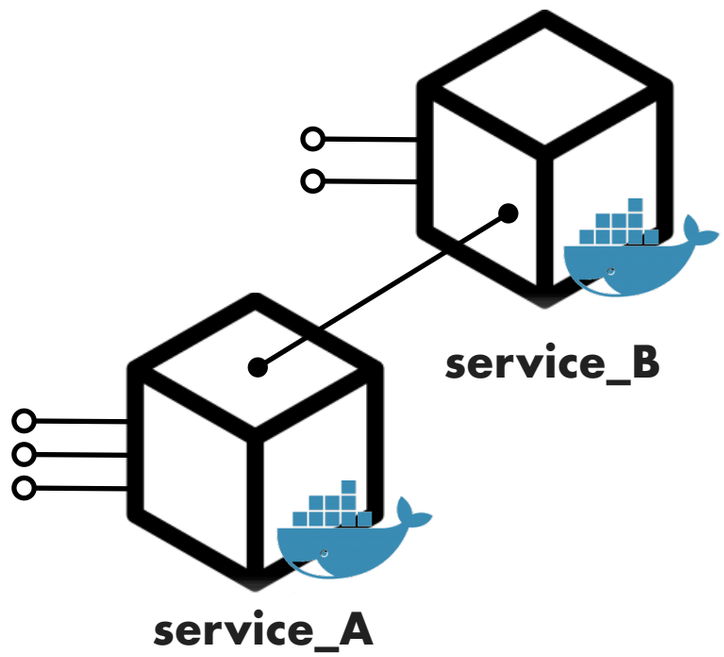
Example: Configuration Test

```
observation:
  service_A:
    - ram_avg
    - cpu_avg
    - response_time_90th_p
  service_B:
    - ram_avg
```



Example: Configuration Test

```
exploration_space:
  service_A:
    resources:
      - memory:
          range: 1GB... 5GB
          step: +1GB
      - cpus:
          range: 1...4
environment:
    - SIZE_OF_THREADPOOL:
        range: 5...100
        step: +5
  - ...
```

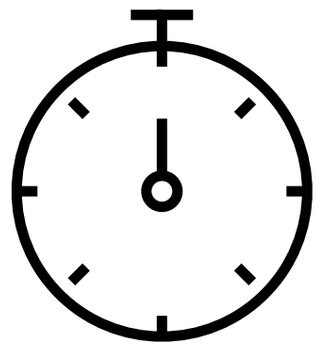


Example: Configuration Test

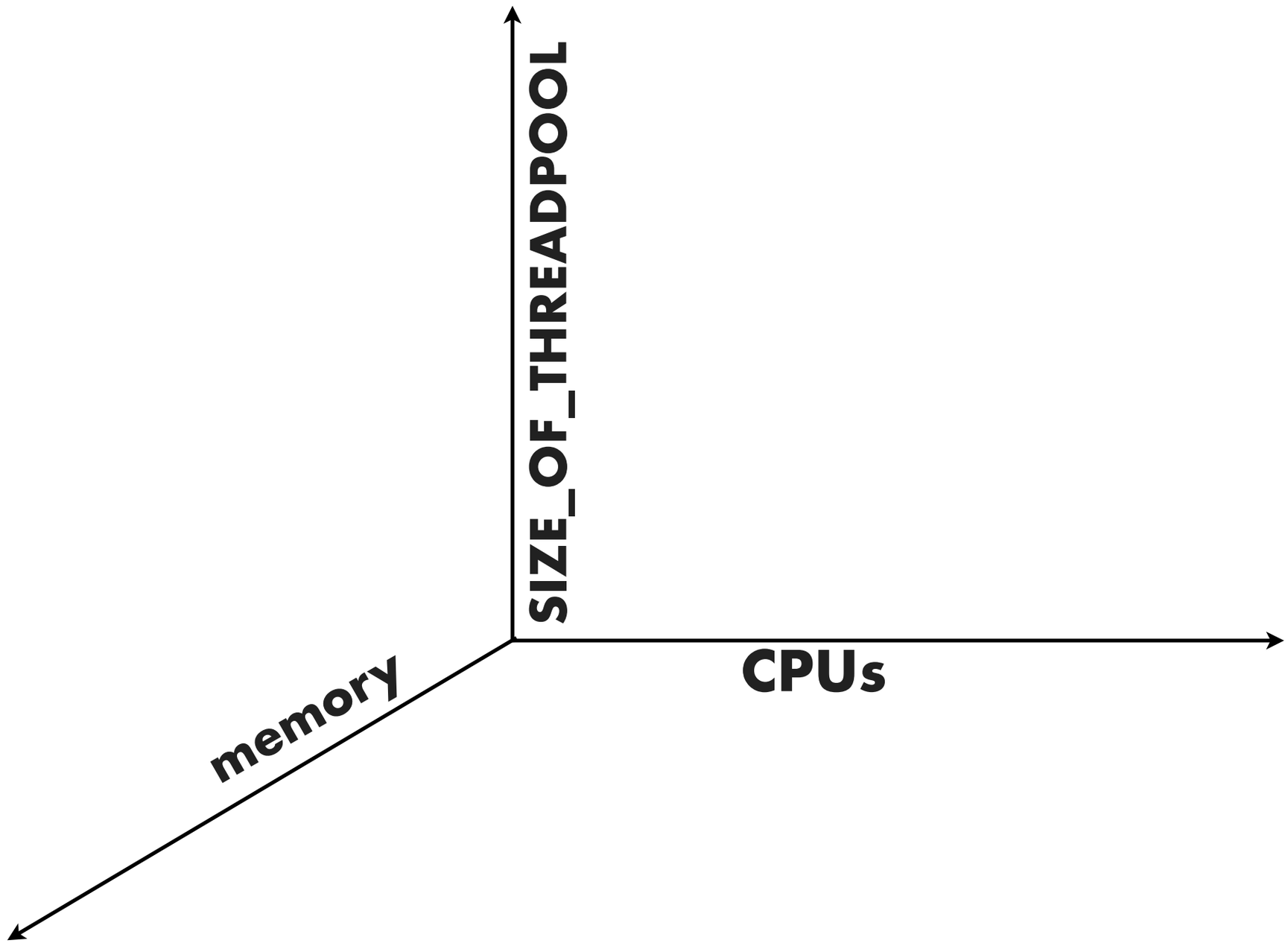
termination_criteria:

- max_exec_time = 1h

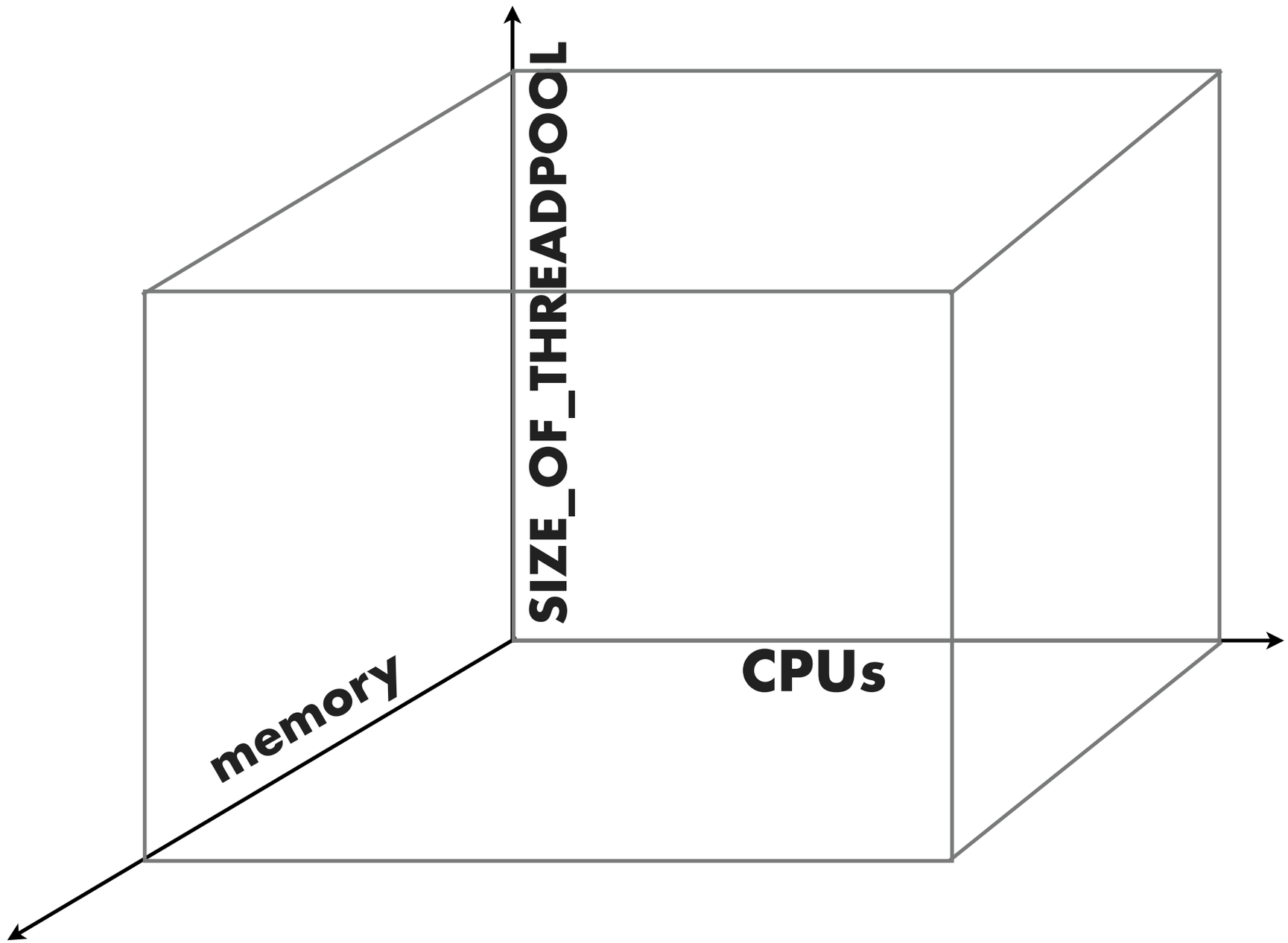
- ...



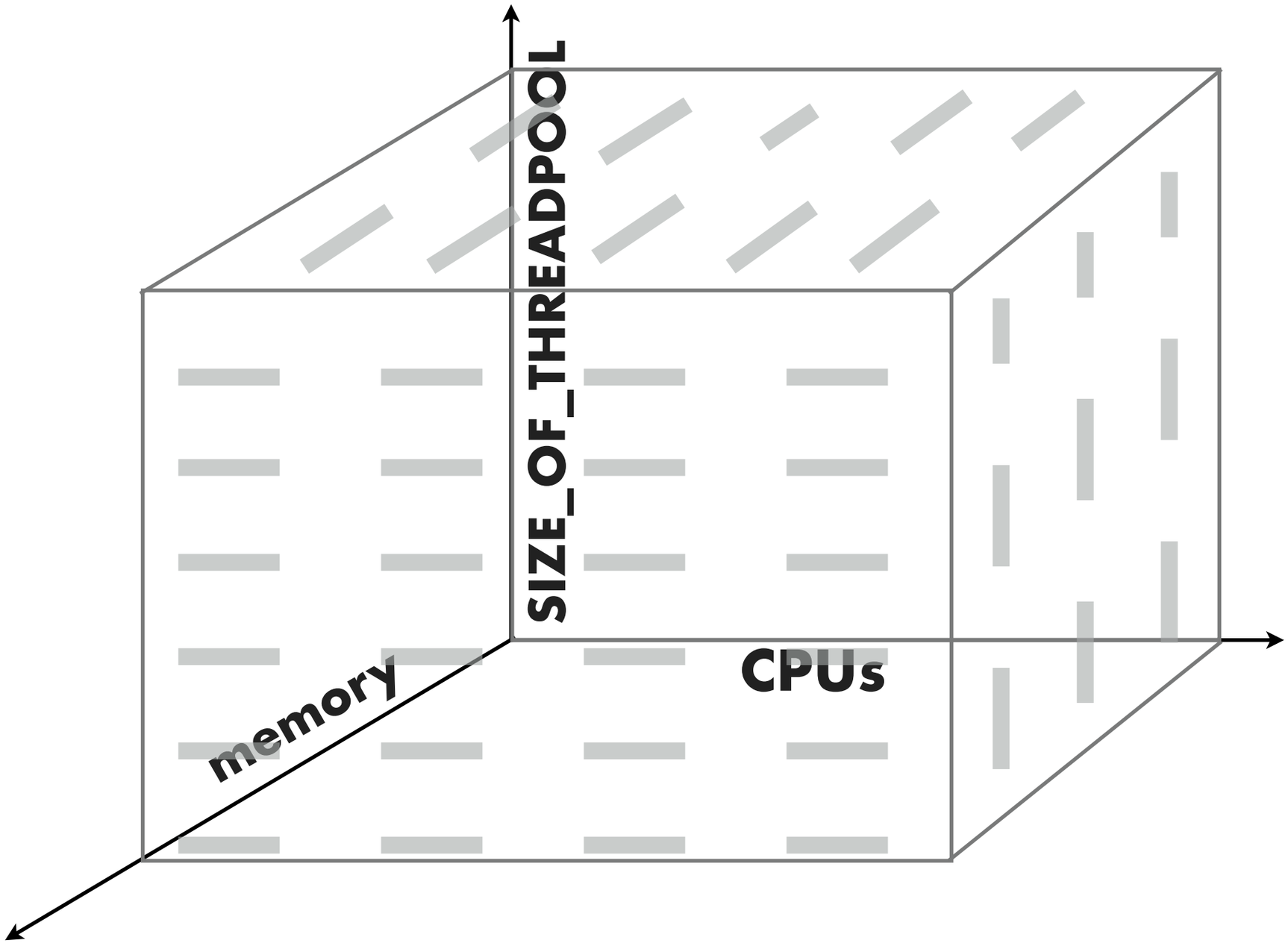
Example: Configuration Test



Example: Configuration Test



Example: Configuration Test



Example: Configuration Test

MARS

Kriging

...

observation:

service_A:

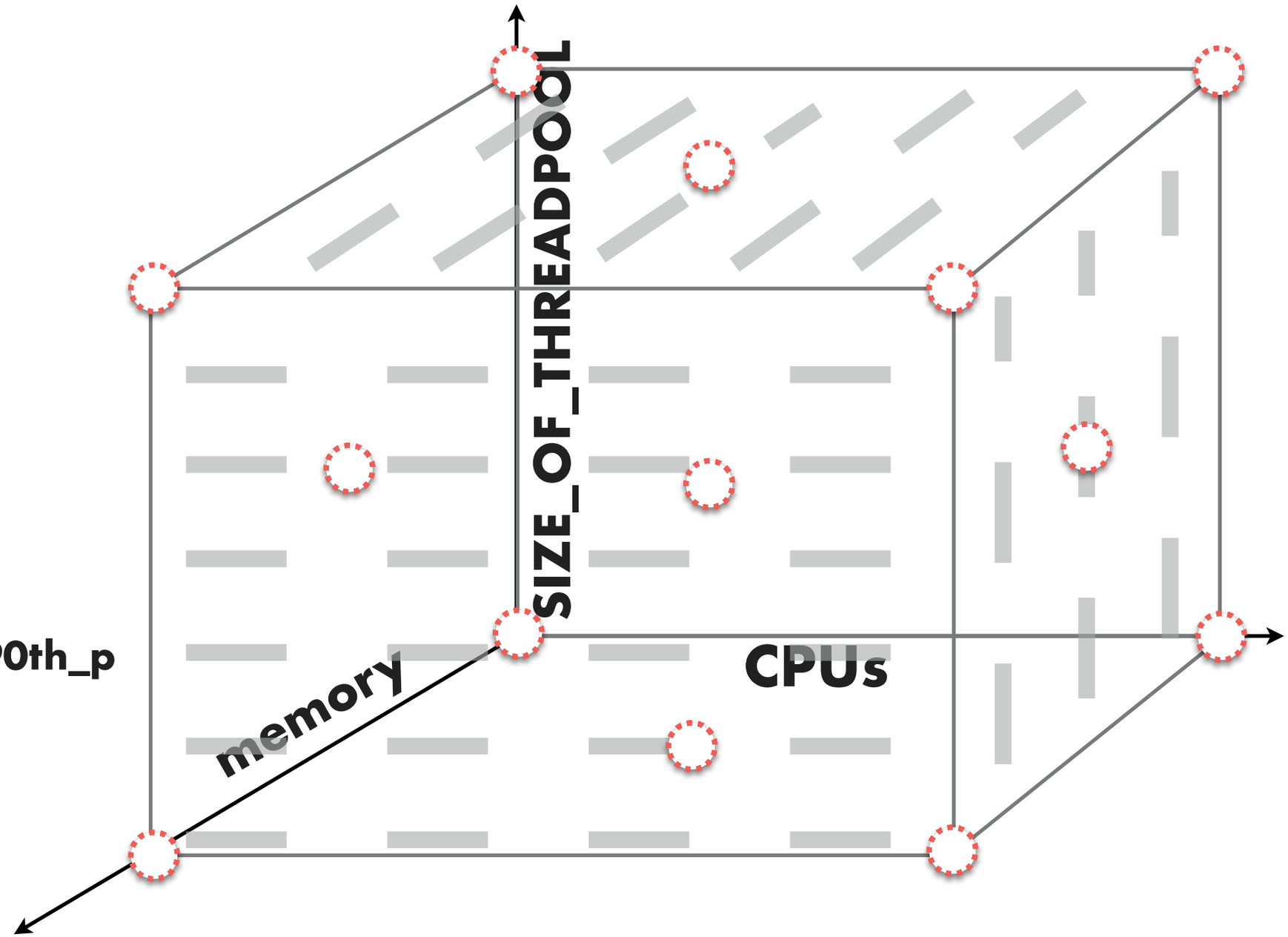
- ram_avg

- cpu_avg

- response_time_90th_p

service_B:

- ram_avg



Objectives

- **Capacity planning (also based on some constraints)**
  e.g., CPU, RAM
  why? cost of resources -> important for the business

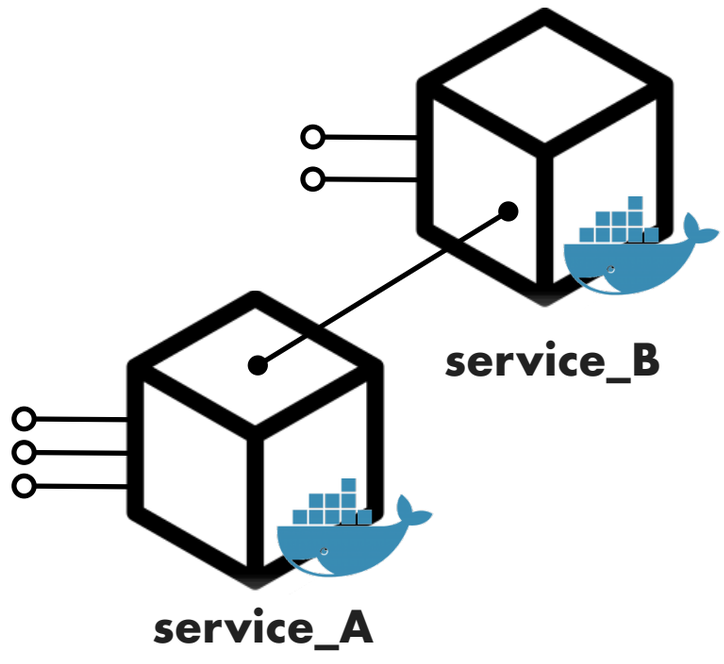
Objectives

- **Capacity planning (also based on some constraints)**
  e.g., CPU, RAM
  why? cost of resources -> important for the business

- **Performance optimisation based on some (resource) constraints**
  e.g., which configuration is optimal?
  why? responsiveness -> important for the user

Example: Performance Optimisation

```
optimisation_target:
  service_A:
      - min(response_time_90th_p)
  service_B:
      - min(memory)
...
```



Example: Performance Optimisation

optimisation_target:

service_A:

- min(response_time_90th_p)

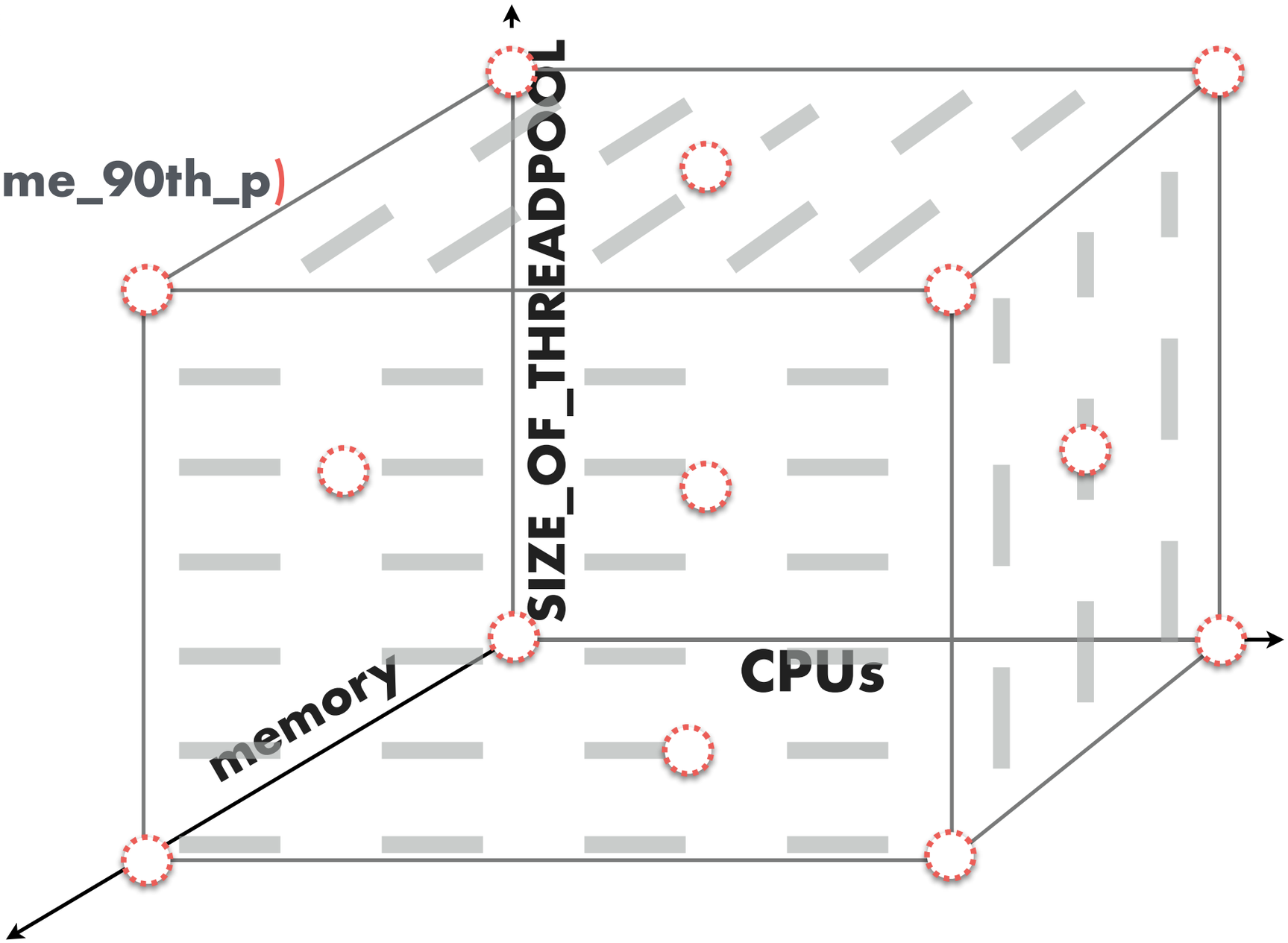
service_B:

- min(memory)

MARS

Kriging

...



Example: Performance Optimisation

optimisation_target:

service_A:

- min(response_time_90th_p)

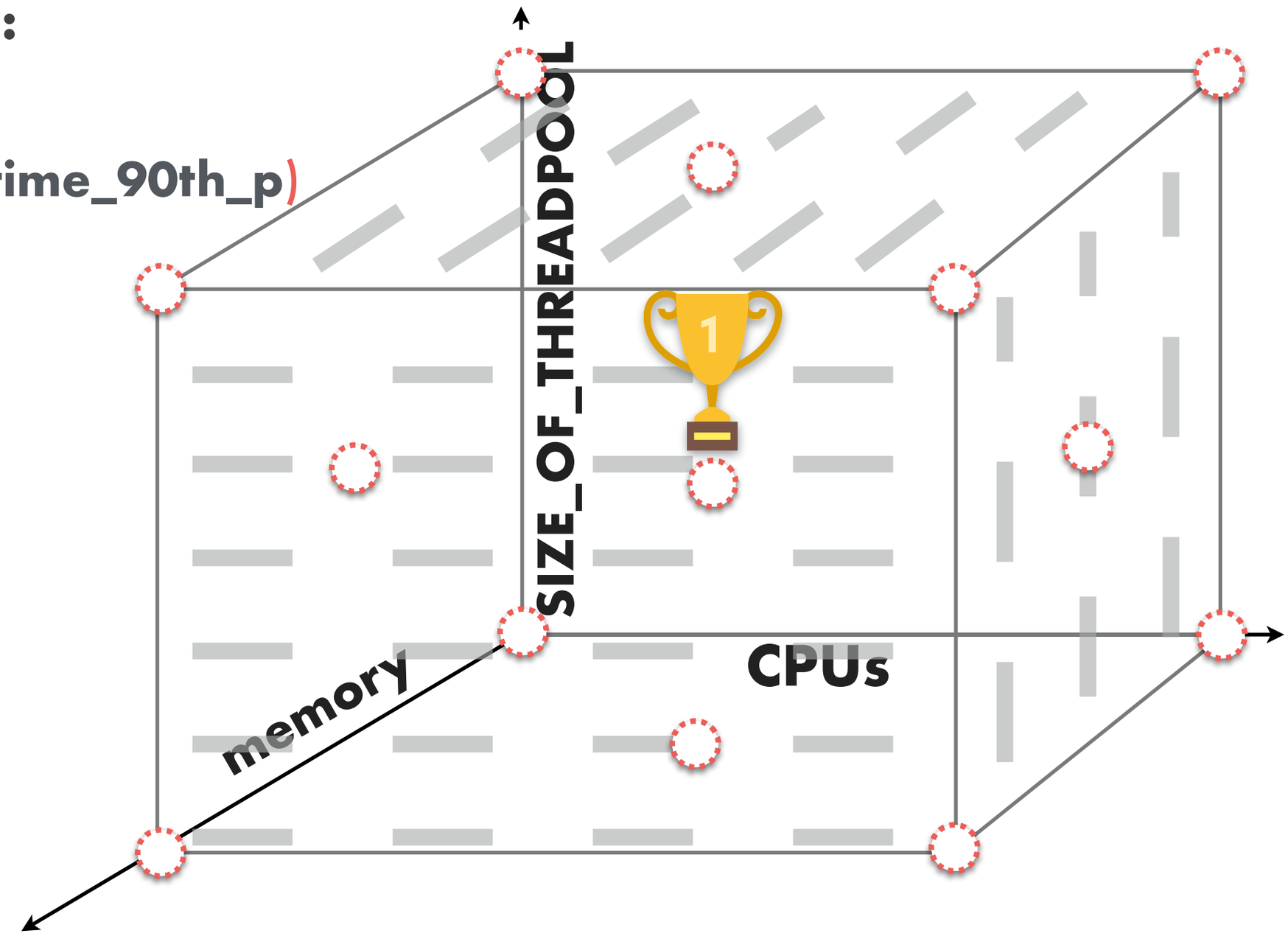
service_B:

- min(memory)

MARS

Kriging

...



Meta-Objectives

- **Regression**
  is the performance, capacity or scalability still the same as previous tests show?

Meta-Objectives

- **Regression**
  is the performance, capacity or scalability still the same as previous tests show?
- **What-If Analysis**
  what do we expect to happen to the output/dependent variables if we change some of the input/independent variables?

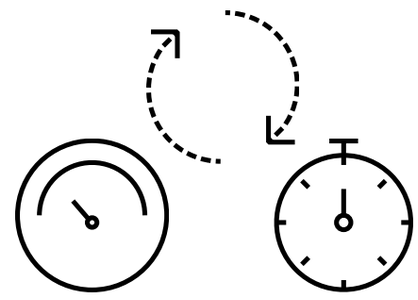
Meta-Objectives

- **Regression**
  is the performance, capacity or scalability still the same as previous tests show?
- **What-If Analysis**
  what do we expect to happen to the output/dependent variables if we change some of the input/independent variables?
- **Before-and-After**
  how has the performance changed given some features have been added?

Meta-Objectives

- **Regression**
  is the performance, capacity or scalability still the same as previous tests show?
- **What-If Analysis**
  what do we expect to happen to the output/dependent variables if we change some of the input/independent variables?
- **Before-and-After**
  how has the performance changed given some features have been added?
- **Benchmarking**
  how does the performance of different systems compare?

Fast Performance Feedback

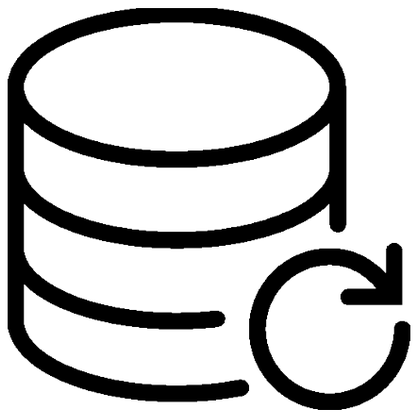


Different Types of Fast Feedback

Evaluating if the System is Ready for the Defined Performance Test Objectives and Reaches Expected State

Different Types of Fast Feedback

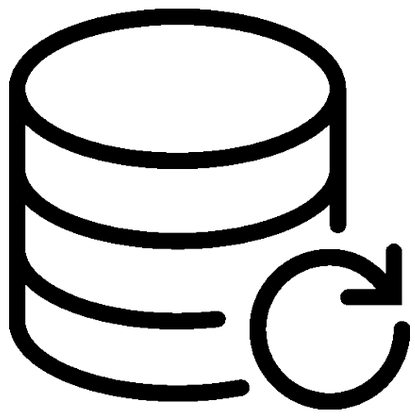
Evaluating if the System is Ready for the Defined Performance Test Objectives and Reaches Expected State



Reusing Collected Performance Knowledge

Reuse Performance Knowledge

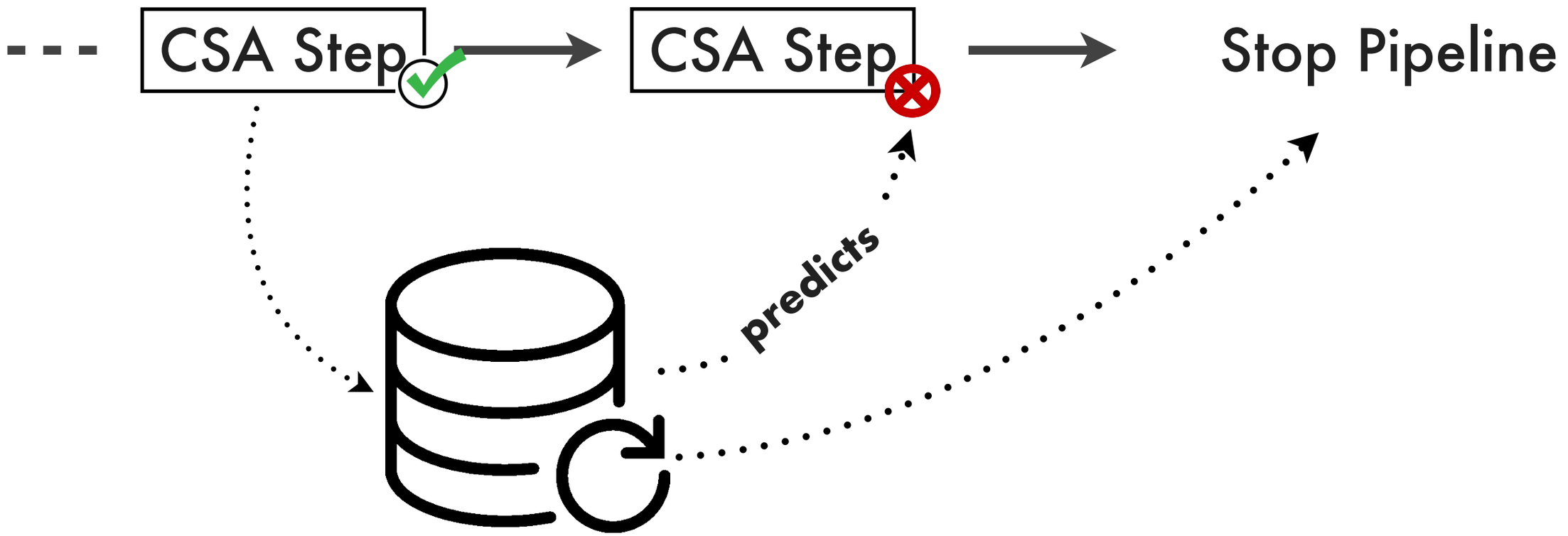
before the execution of a test



Along the Workflow/Pipeline

Reuse Performance Knowledge

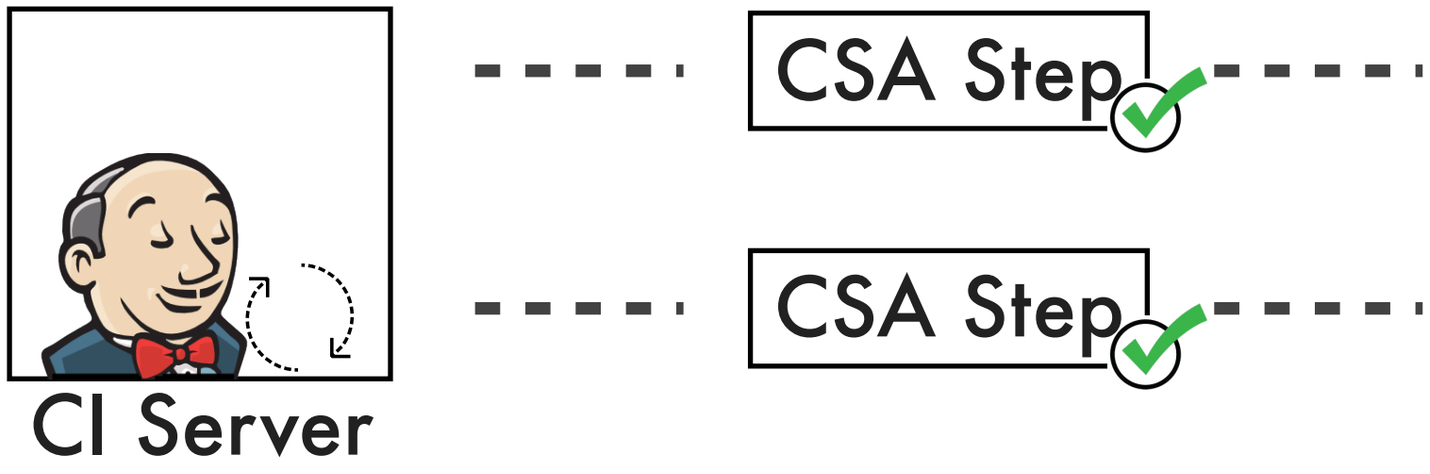
before the execution of a test

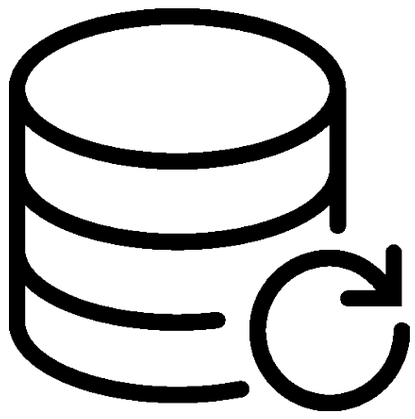


Along the Workflow/Pipeline

Reuse Performance Knowledge

during the execution of a test

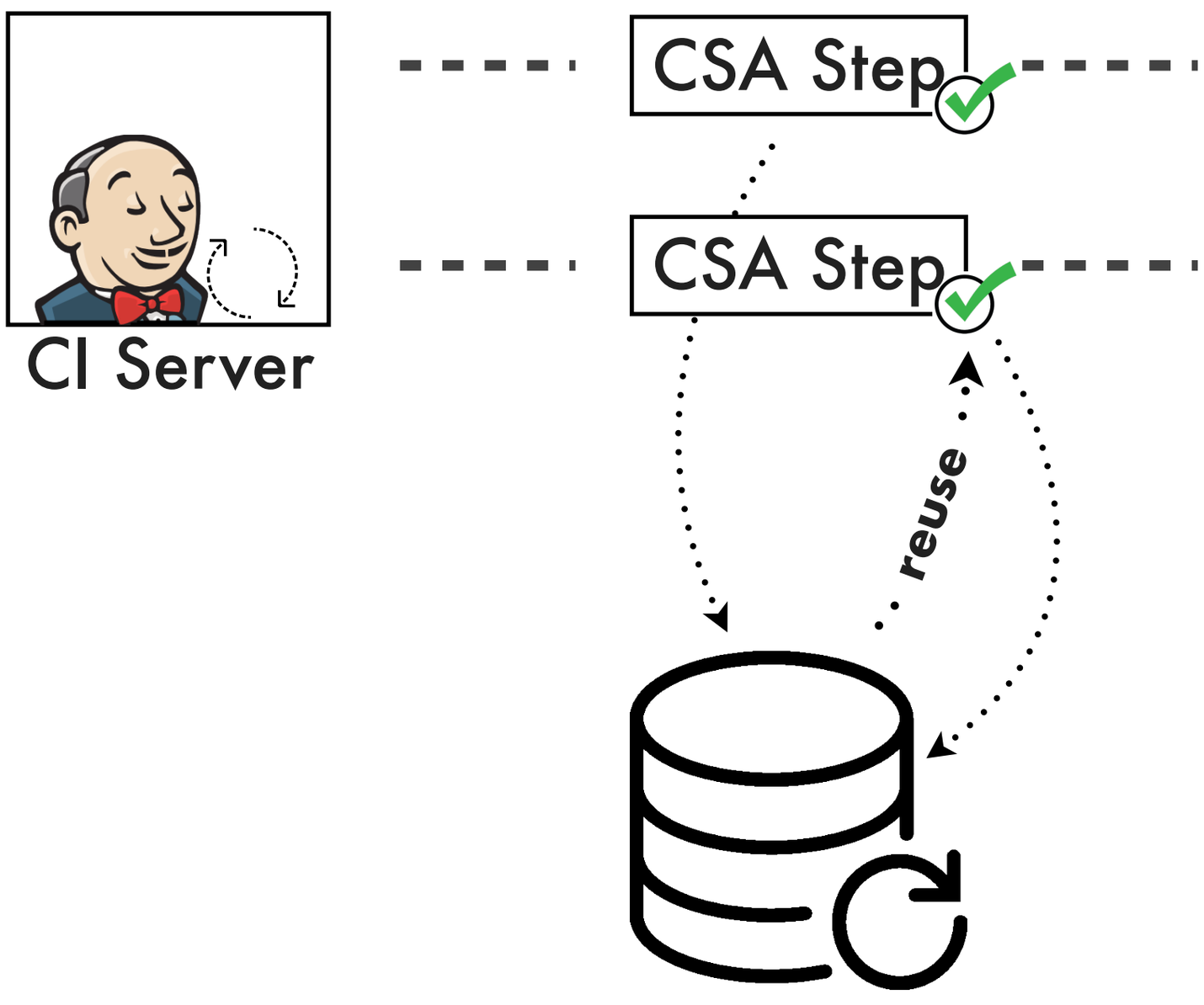




Across Different Iterations of the Same Test

Reuse Performance Knowledge

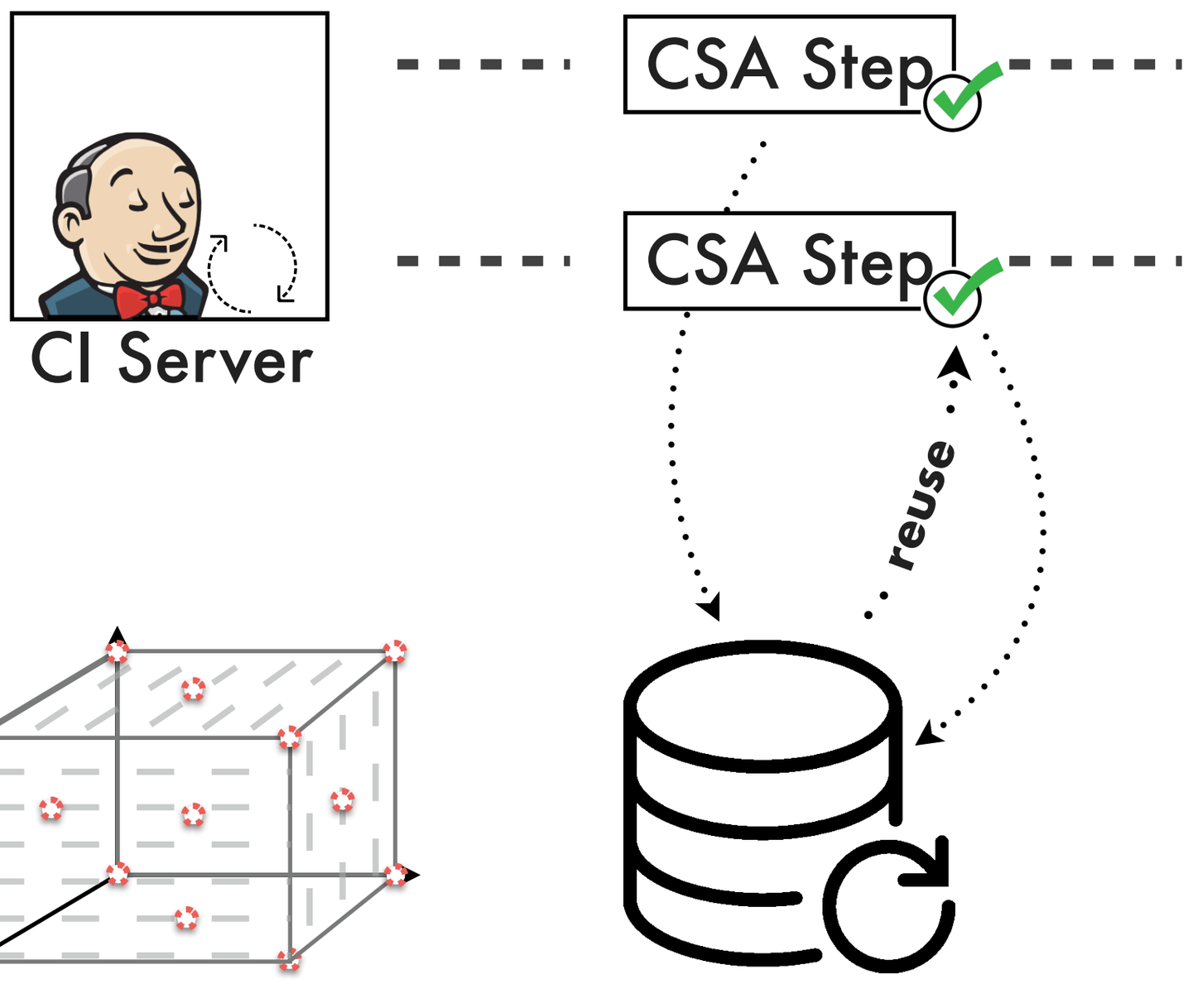
during the execution of a test



Across Different Iterations of the Same Test

Reuse Performance Knowledge

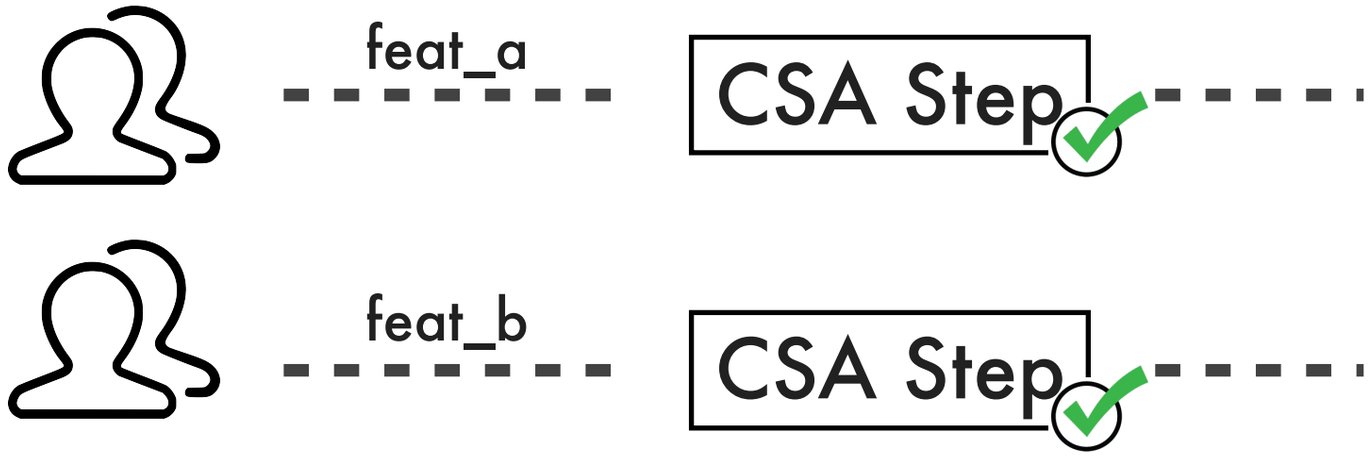
during the execution of a test

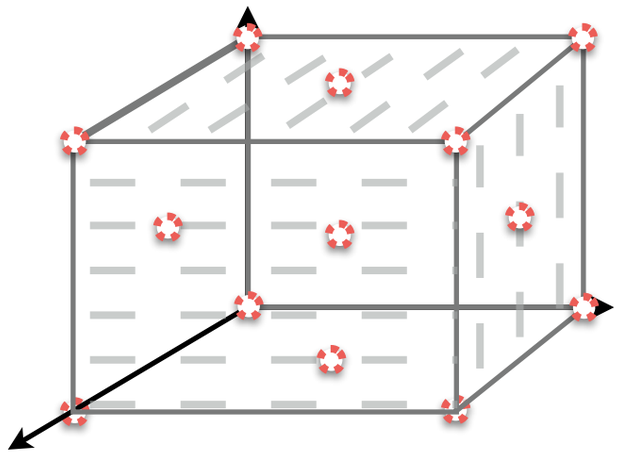


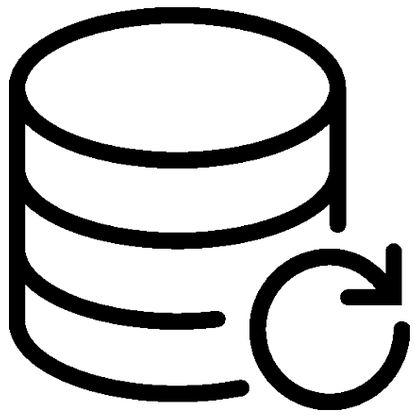
Across Different Iterations of the Same Test

Reuse Performance Knowledge

after the execution of a test



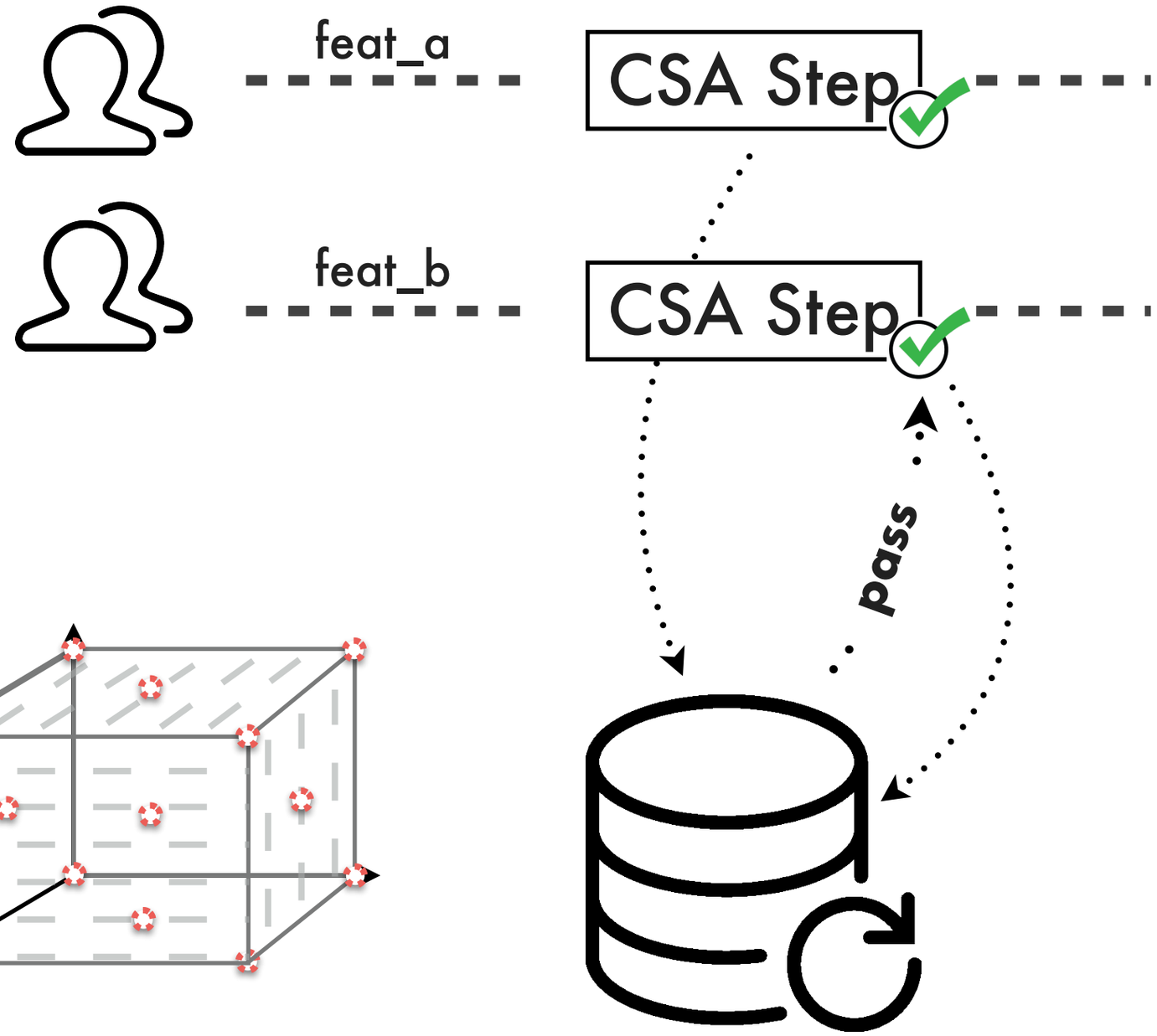




Cross Branches

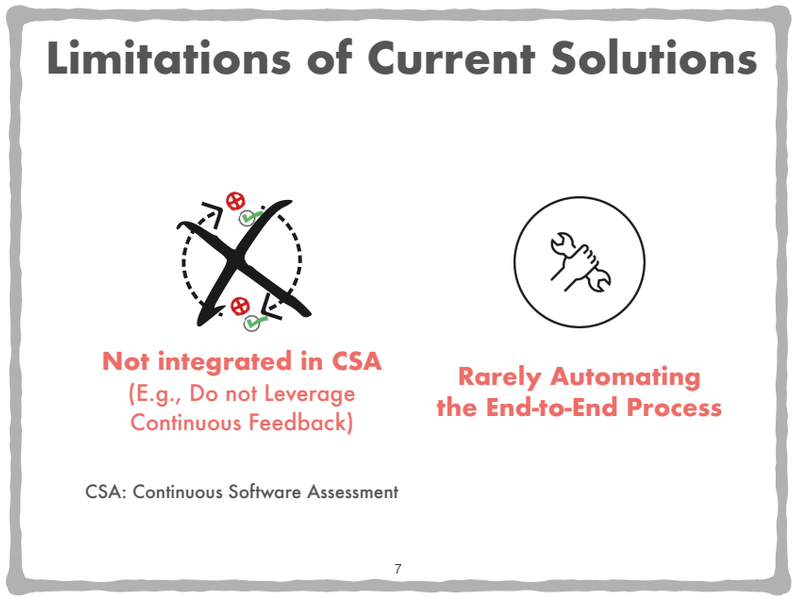
Reuse Performance Knowledge

after the execution of a test



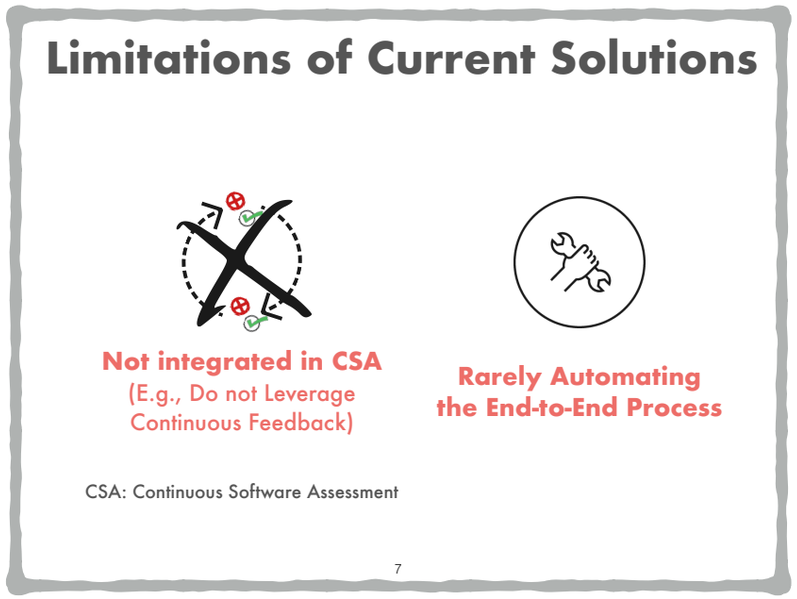
Cross Branches

Highlights

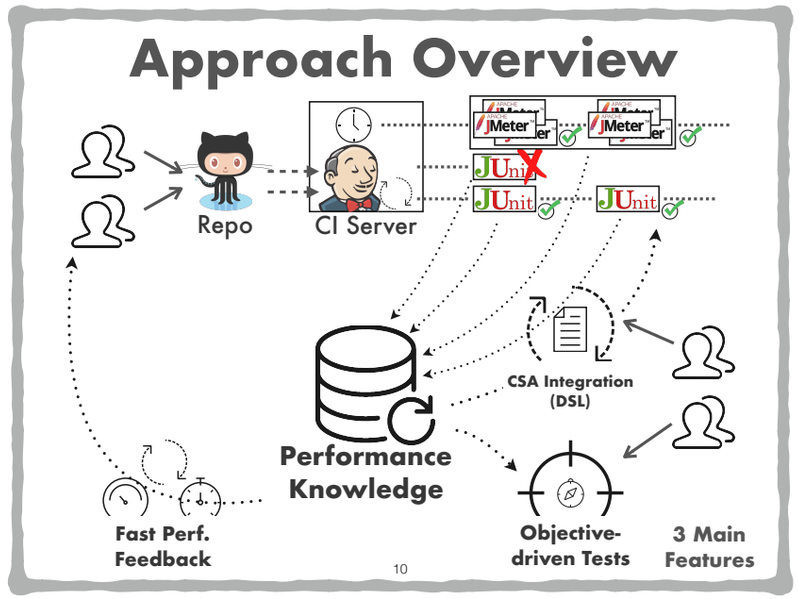


Current Solutions + Limitations

Highlights

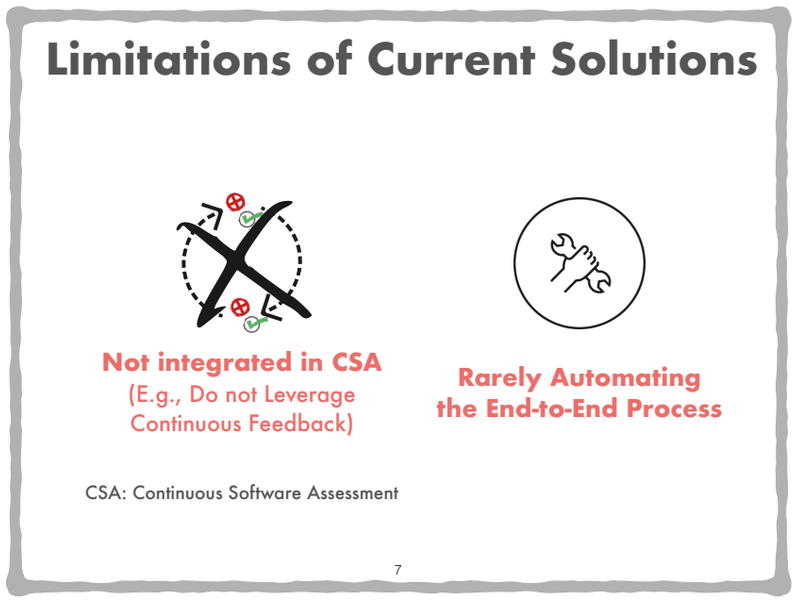


Current Solutions + Limitations

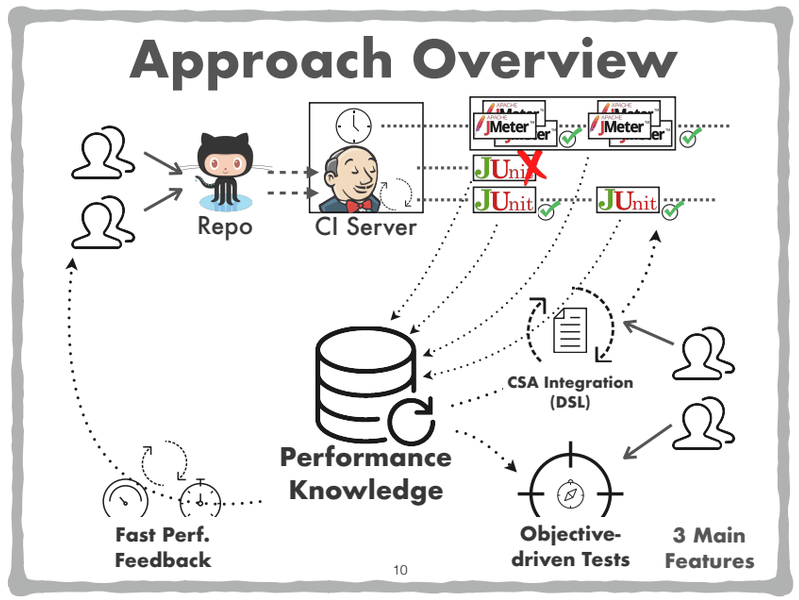


Approach Overview

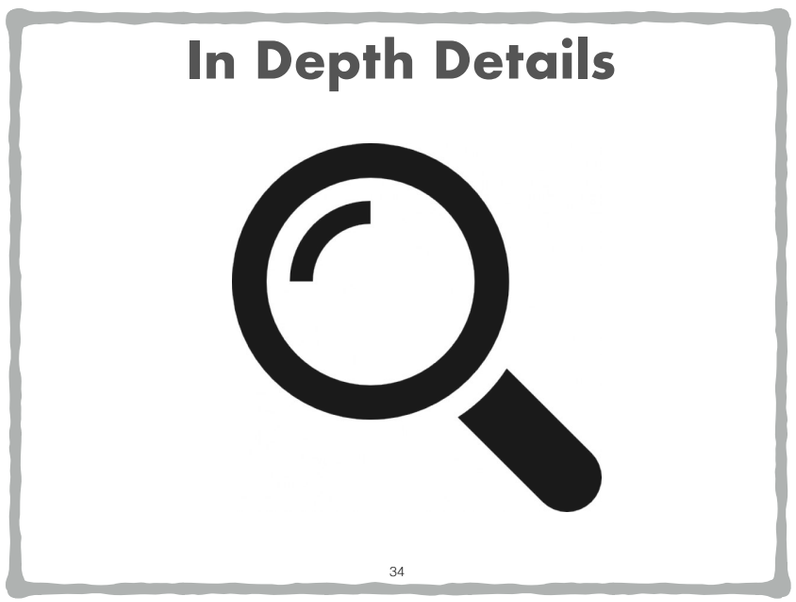
Highlights



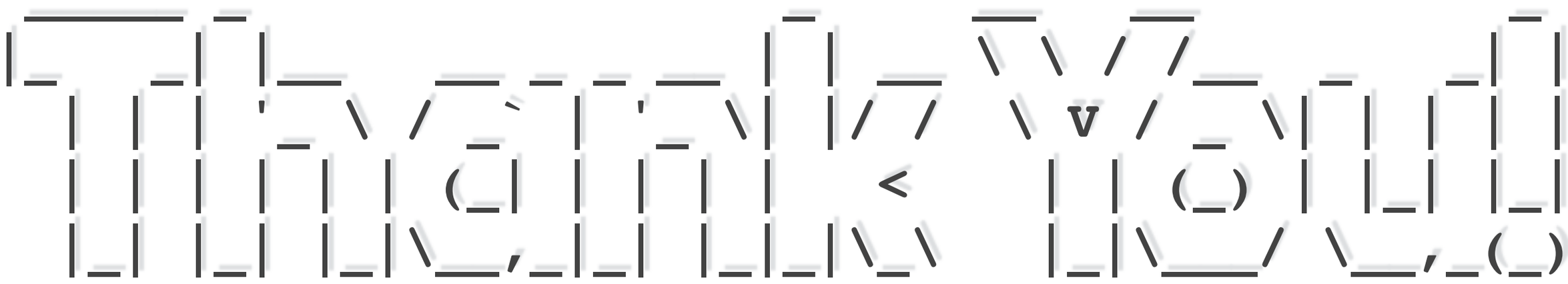
Current Solutions + Limitations



Approach Overview



Approach Details



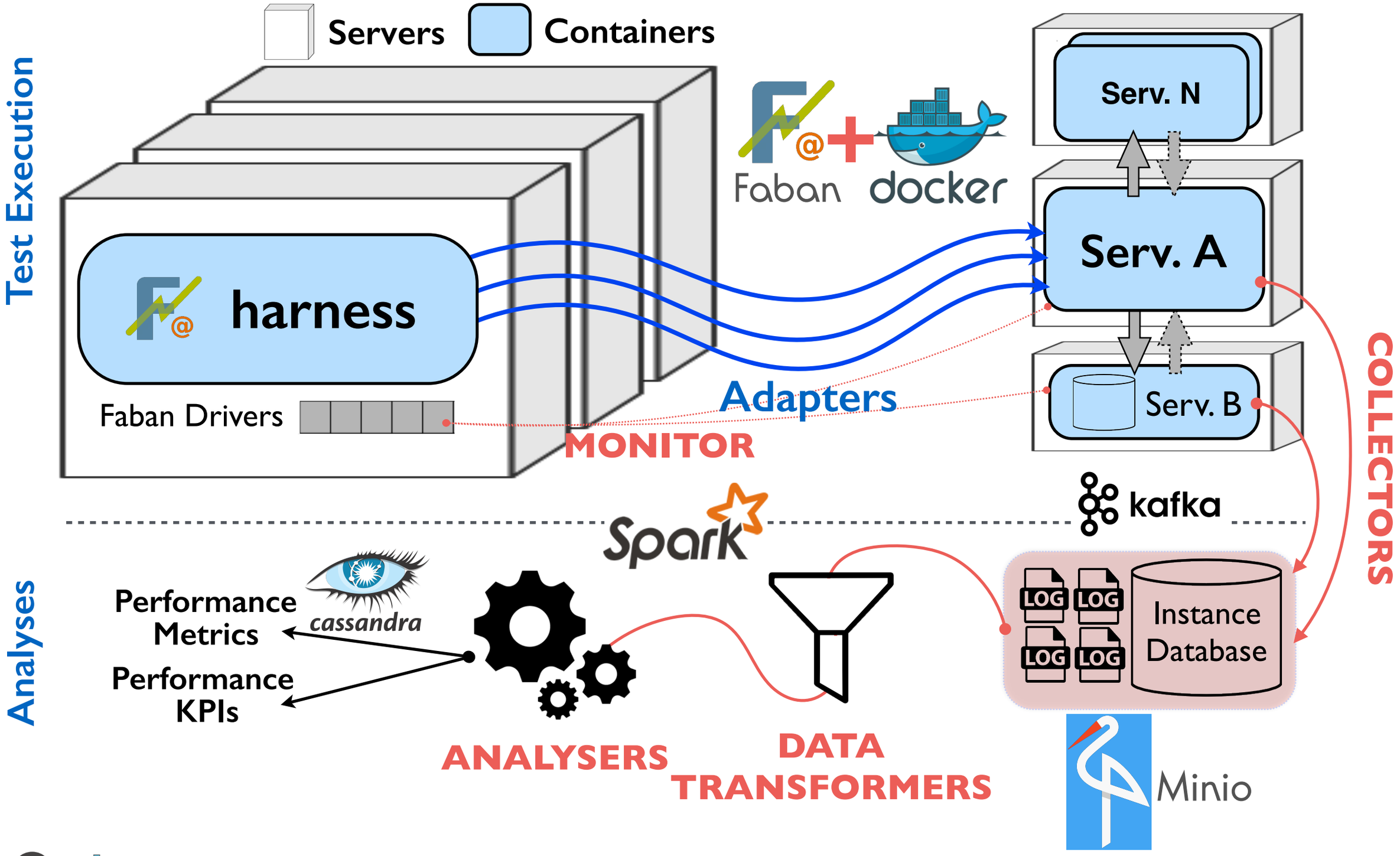
benchflow

http://benchflow.inf.usi.ch

✉ vincenzo.ferme@usi.ch

Backup Slides

BenchFlow Tool Overview



benchflow

Docker Performance

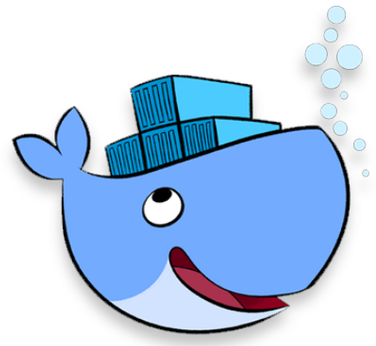
[IBM '14]
W. Felter, A. Ferreira, R. Rajamony, and J. Rubio. **An updated performance comparison of virtual machines and Linux containers.** IBM Research Report, 2014.

> "Our results show that **containers result in equal or better performance than VMs in almost all cases.** "

> "Although **containers themselves have almost no overhead**, **Docker is not without performance gotchas**. Docker volumes have noticeably better performance than files stored in AUFS. Docker's NAT also introduces overhead for workloads with high packet rates. These **features** represent a **tradeoff between ease of management and performance and should be considered on a case-by-case basis.** "

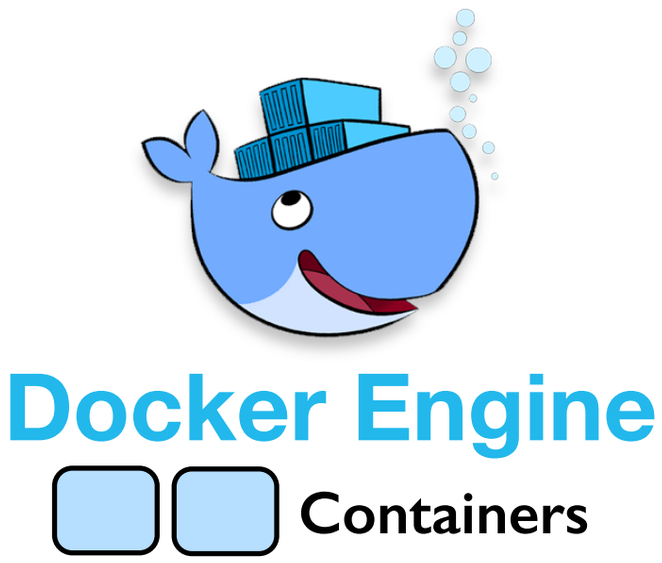
BenchFlow Configures Docker for Performance by Default

BenchFlow: System Under Test

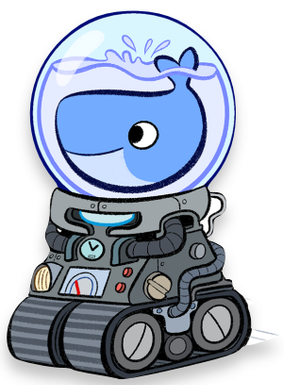


Docker Engine

BenchFlow: System Under Test

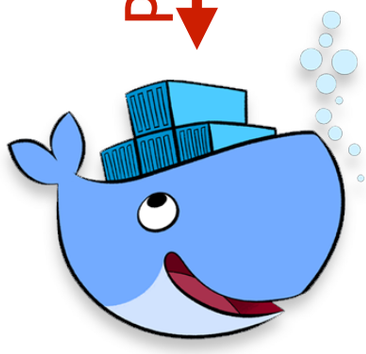


BenchFlow: System Under Test



Docker Machine

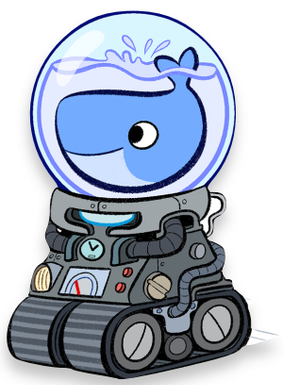
provides



Docker Engine

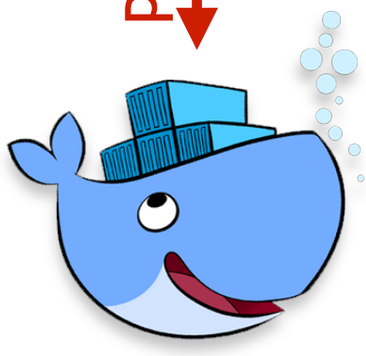
Containers

BenchFlow: System Under Test



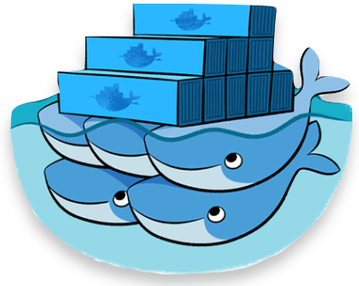
Docker Machine

provides



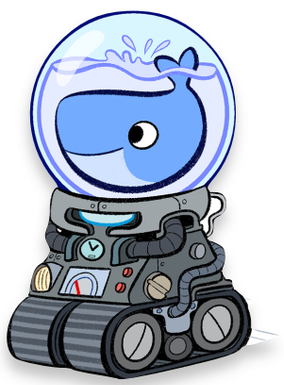
Docker Engine

Containers



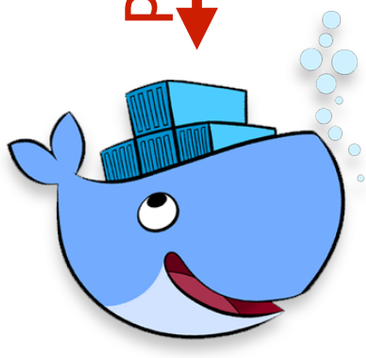
Docker Swarm

BenchFlow: System Under Test



Docker Machine

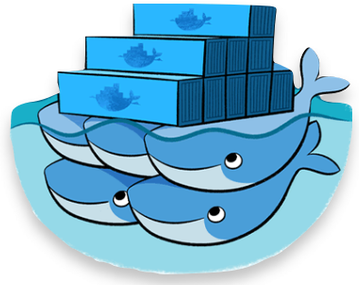
provides



Docker Engine

Containers

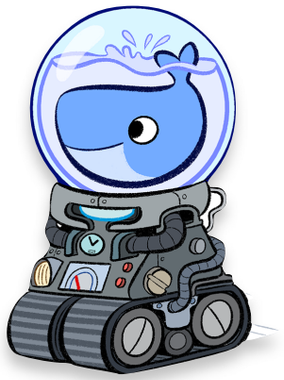
manages



Docker Swarm

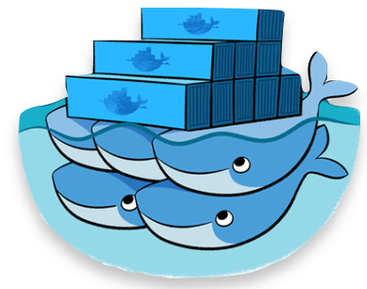
Servers

BenchFlow: System Under Test

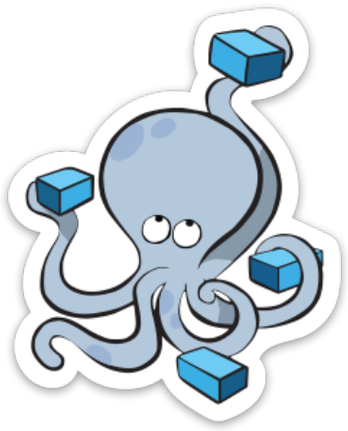


Docker Machine

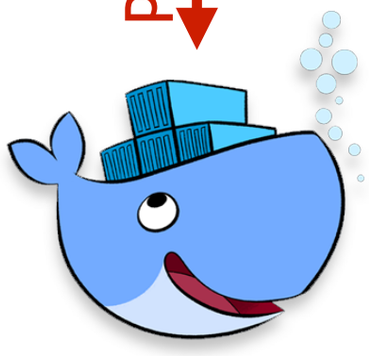
provides



Docker Swarm



Docker Compose



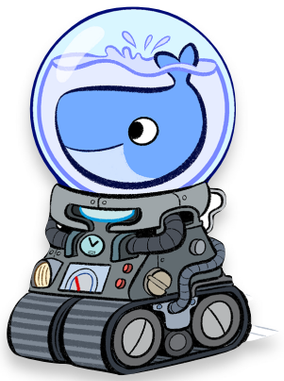
manages

Servers

Docker Engine

Containers

BenchFlow: System Under Test



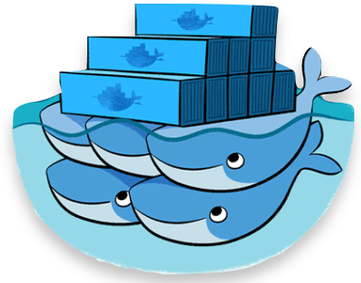
Docker Machine

Docker Compose

SUT's Deployment Conf.

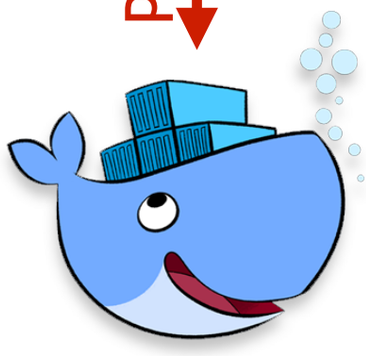
deploys

provides



Docker Swarm

Servers

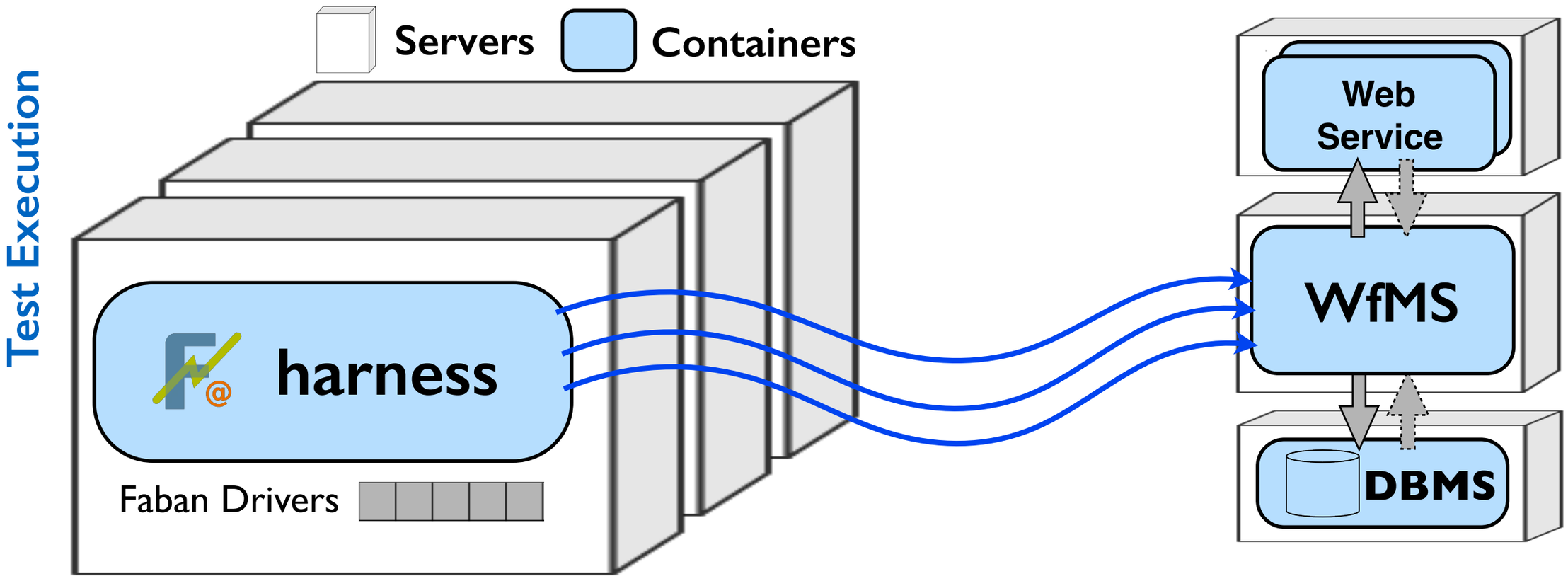


manages

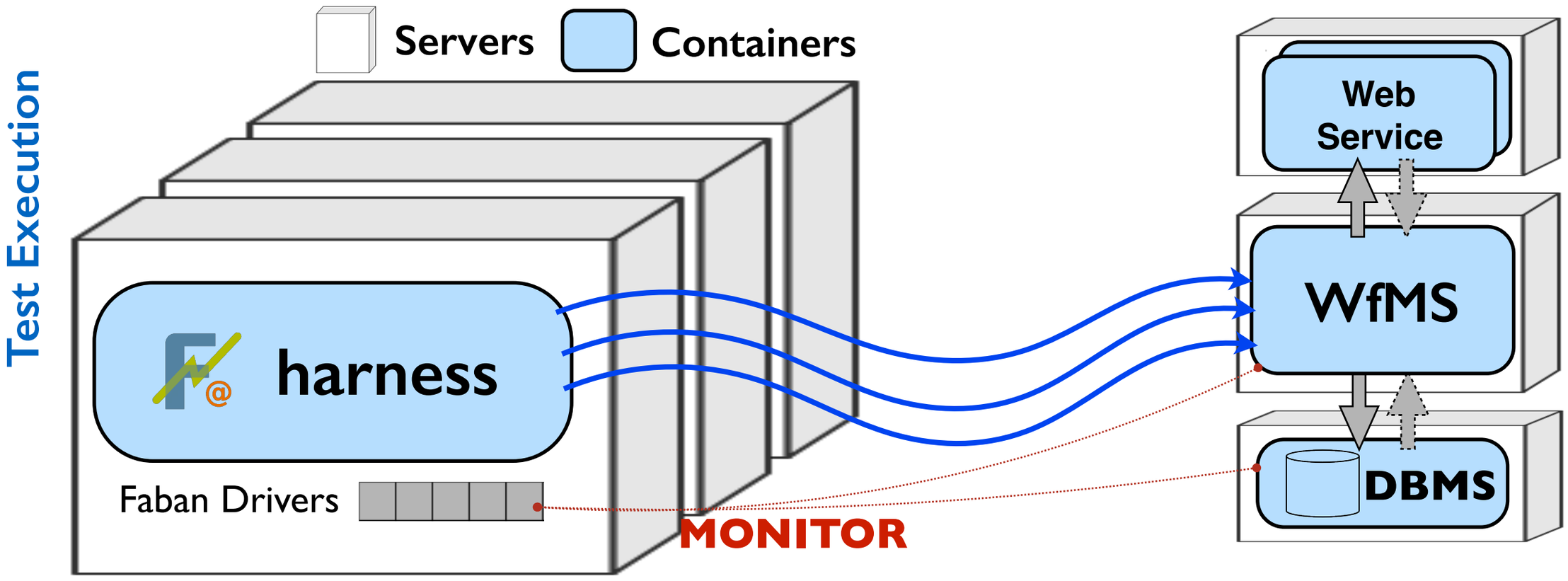
Docker Engine

Containers

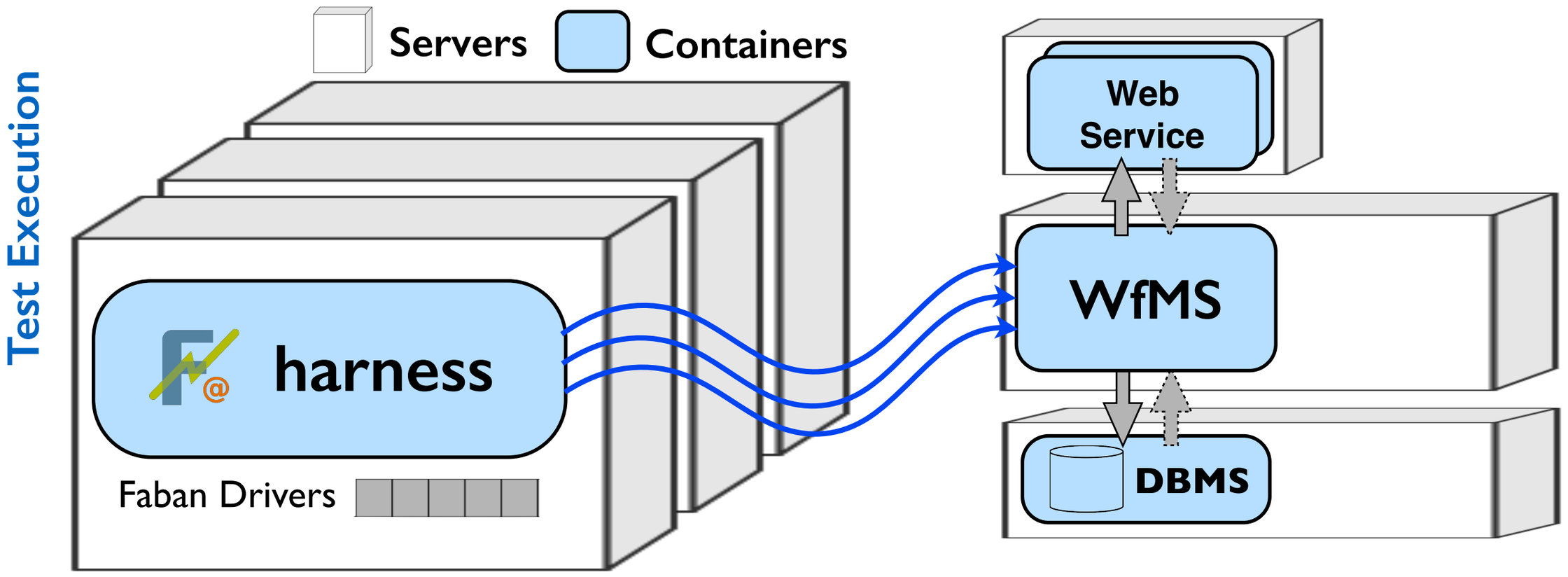
Server-side Data and Metrics Collection



Server-side Data and Metrics Collection



Server-side Data and Metrics Collection



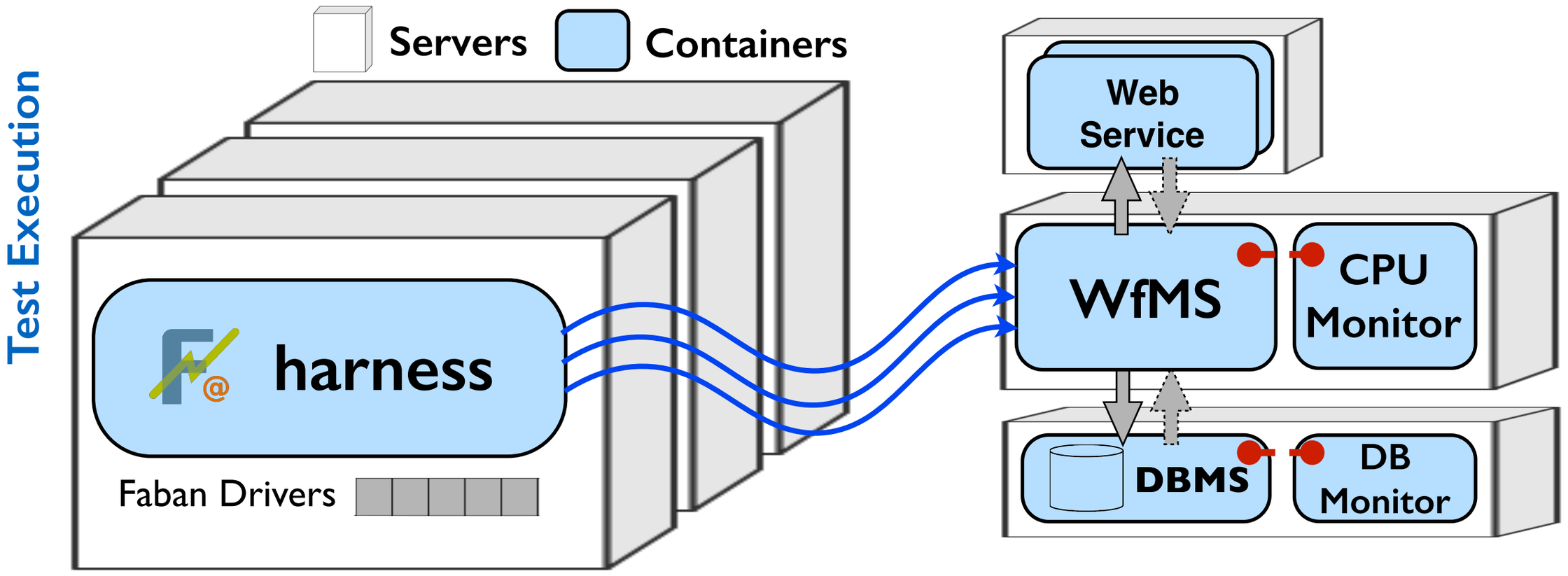
Monitors' Characteristics:

- RESTful services
- Lightweight (written in Go)
- As less invasive on the SUT as possible

Examples of Monitors:

- CPU usage
- Database state

Server-side Data and Metrics Collection



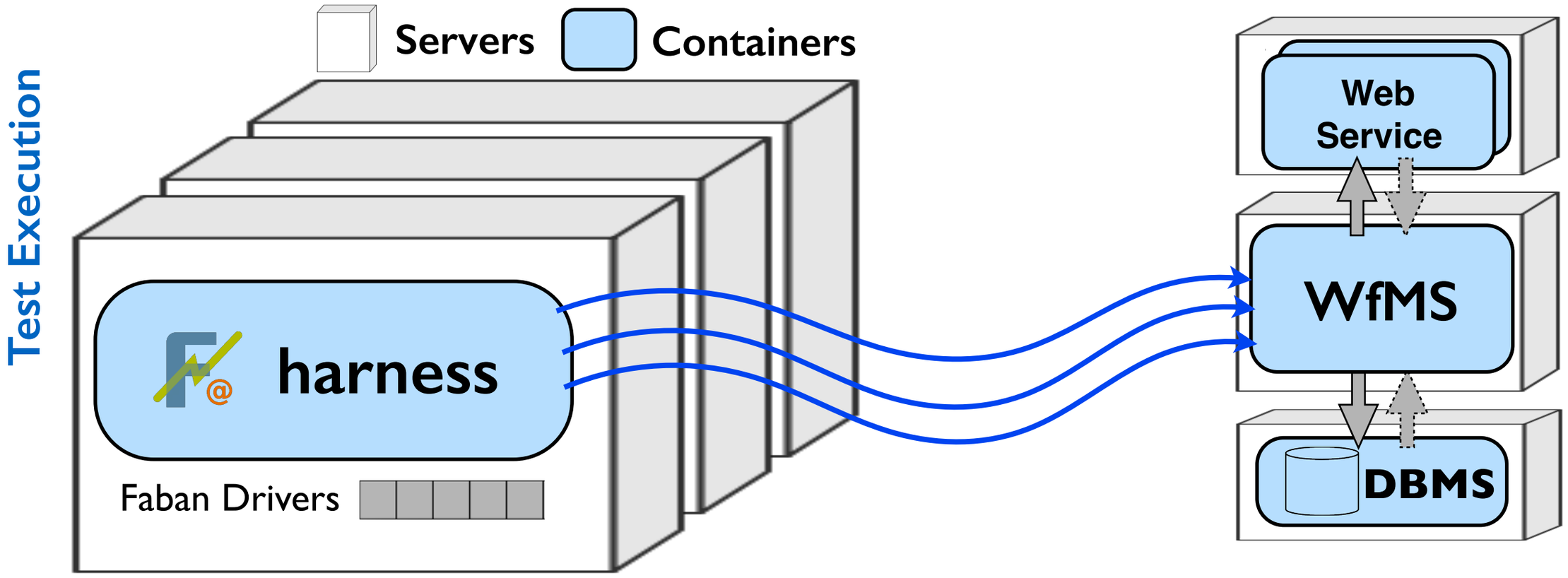
Monitors' Characteristics:

- RESTful services
- Lightweight (written in Go)
- As less invasive on the SUT as possible

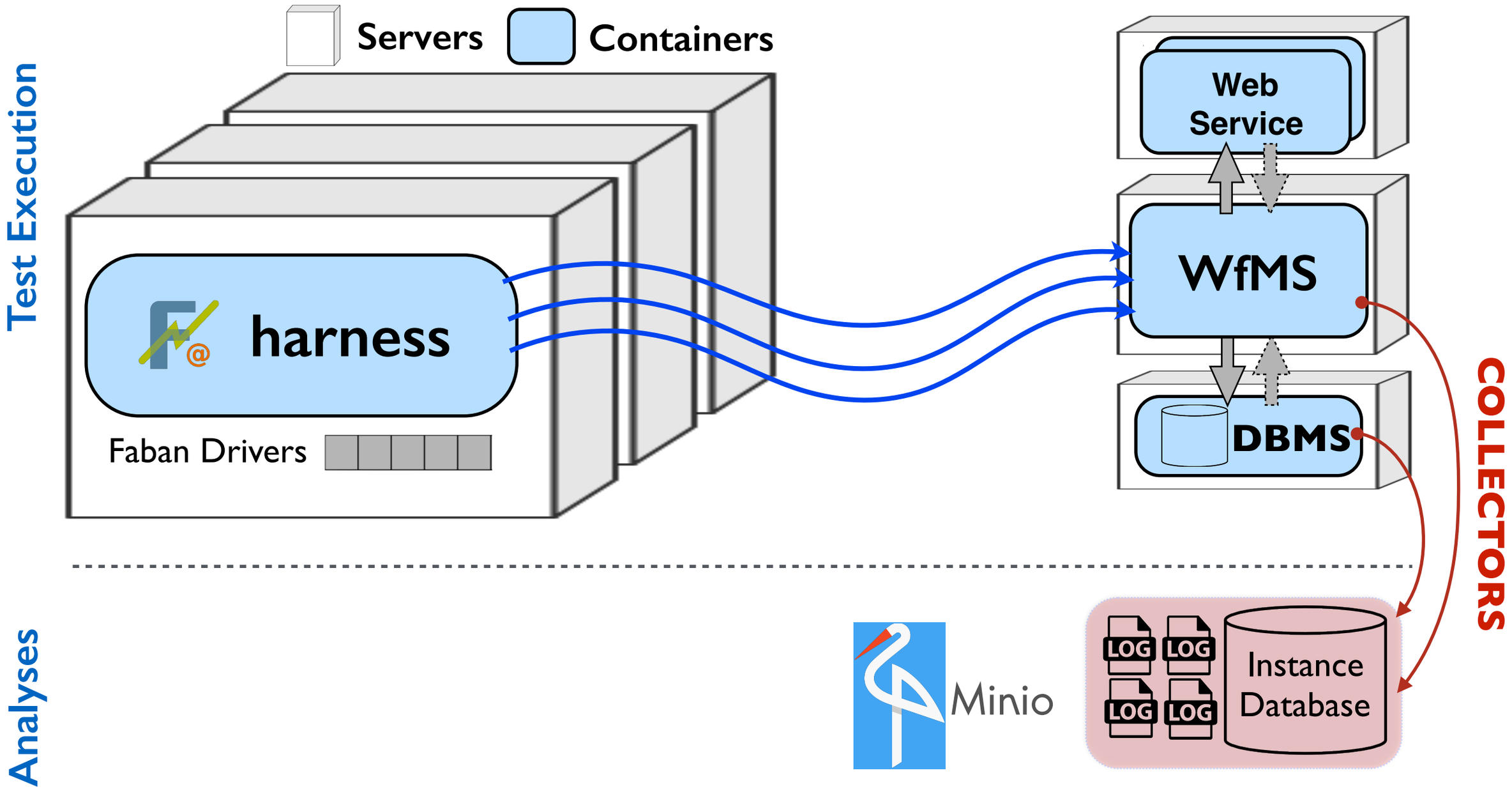
Examples of Monitors:

- CPU usage
- Database state

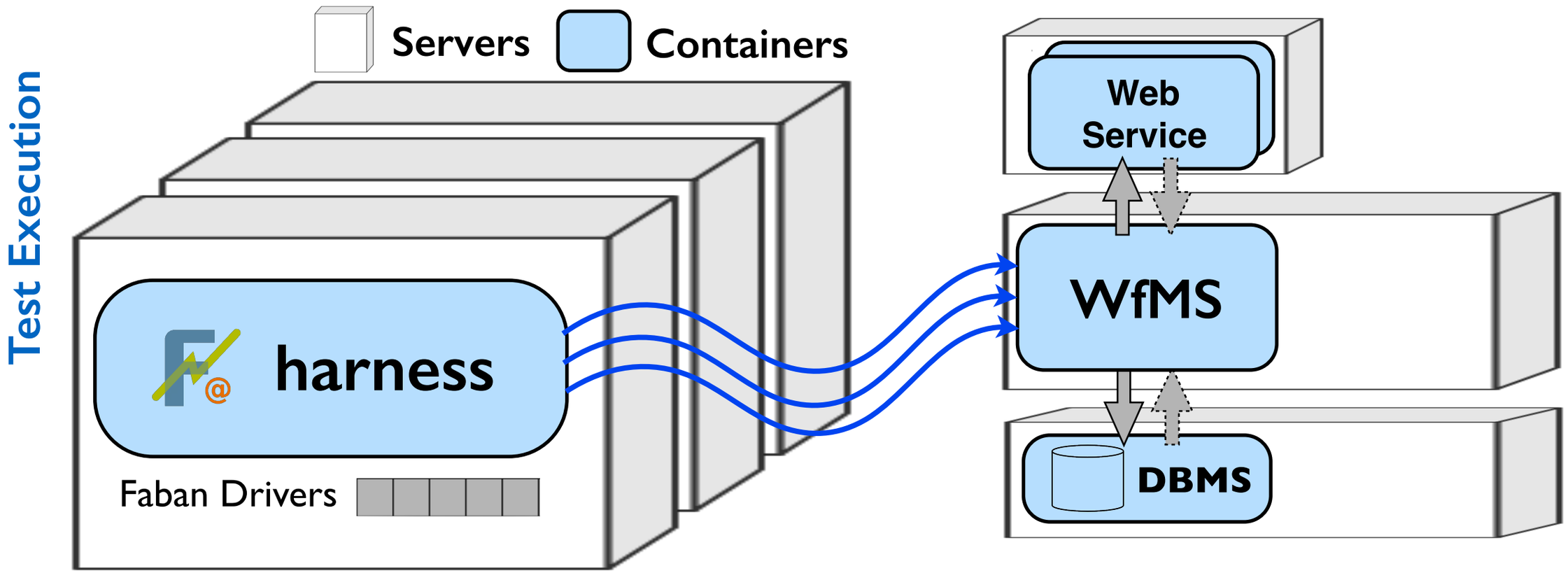
Server-side Data and Metrics Collection



Server-side Data and Metrics Collection



Server-side Data and Metrics Collection



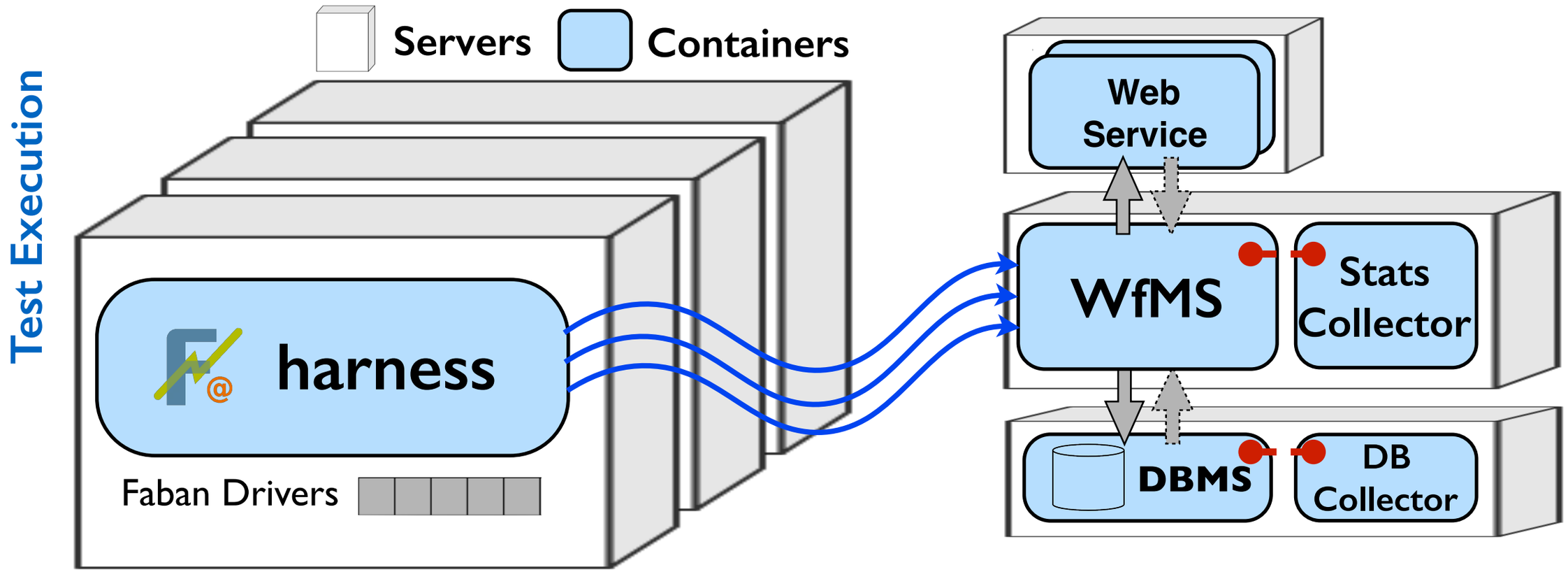
Collectors' Characteristics:

- RESTful services
- Lightweight (written in Go)
- Two types: online and offline
- Buffer data locally

Examples of Collectors:

- Container's Stats (e.g., CPU usage)
- Database dump
- Applications Logs

Server-side Data and Metrics Collection



Collectors' Characteristics:

- RESTful services
- Lightweight (written in Go)
- Two types: online and offline
- Buffer data locally

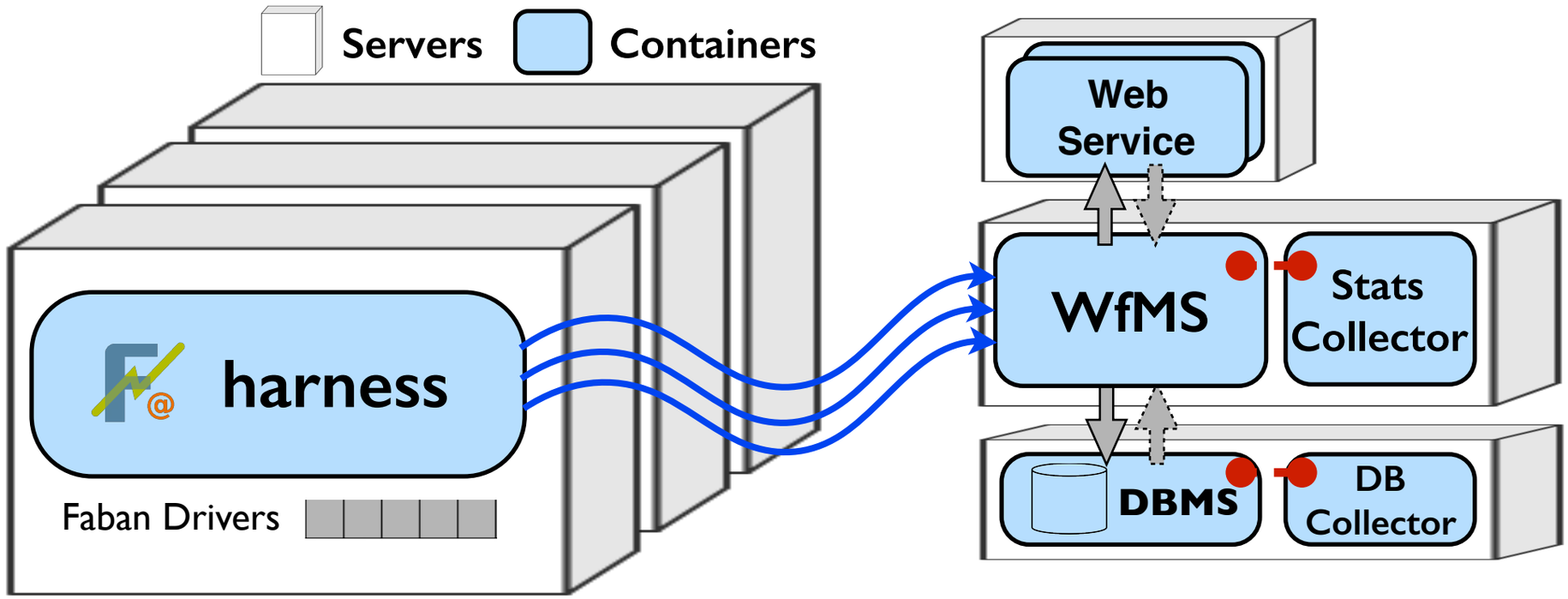
Examples of Collectors:

- Container's Stats (e.g., CPU usage)
- Database dump
- Applications Logs

Performance Metrics and KPIs

coordinate data collection and data transformation

Test Execution



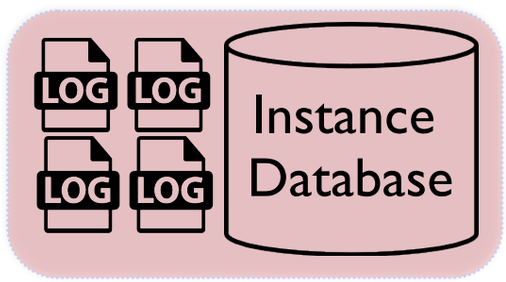
kafka

A high-throughput distributed messaging system

Analyses

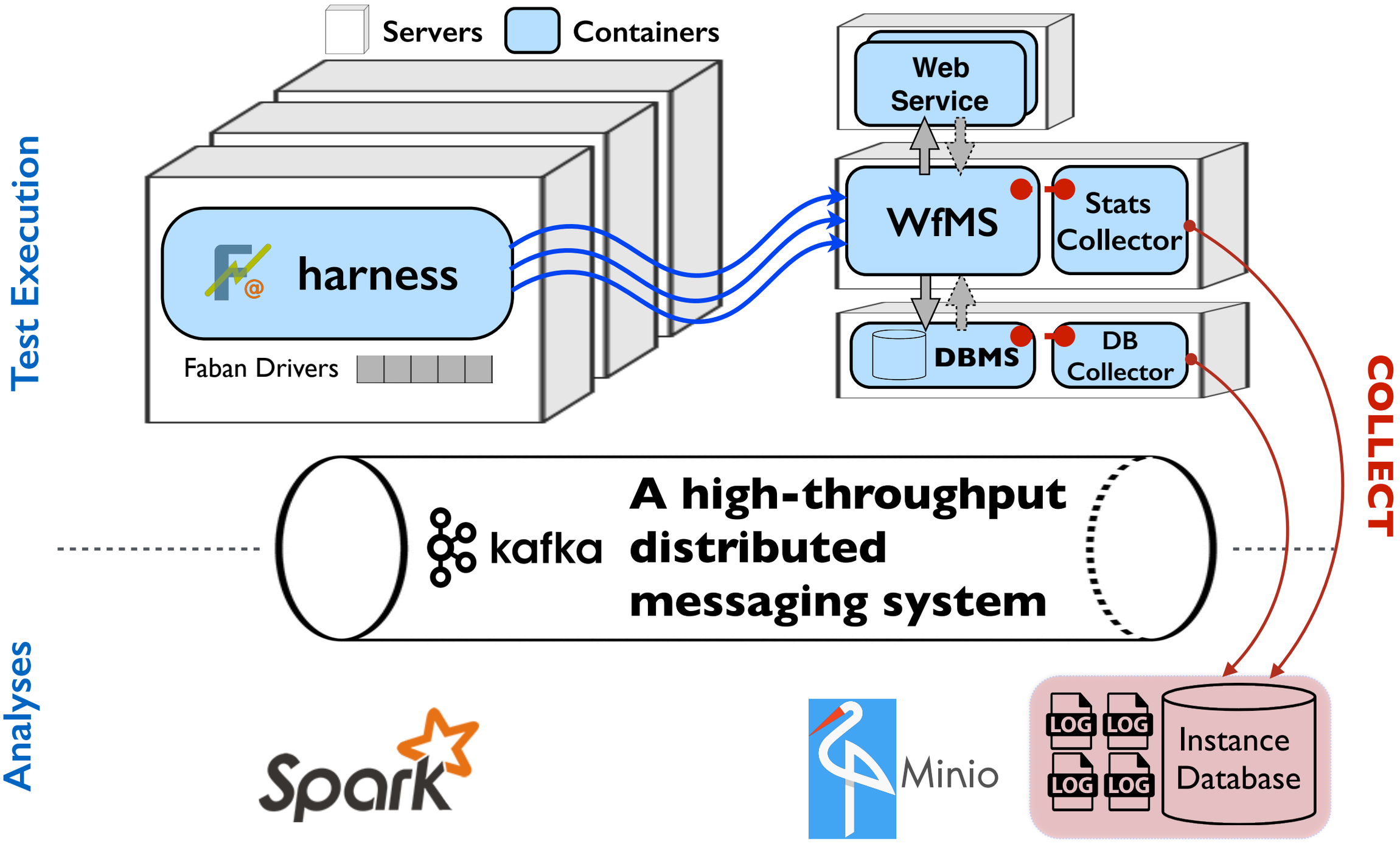
Spark

Minio



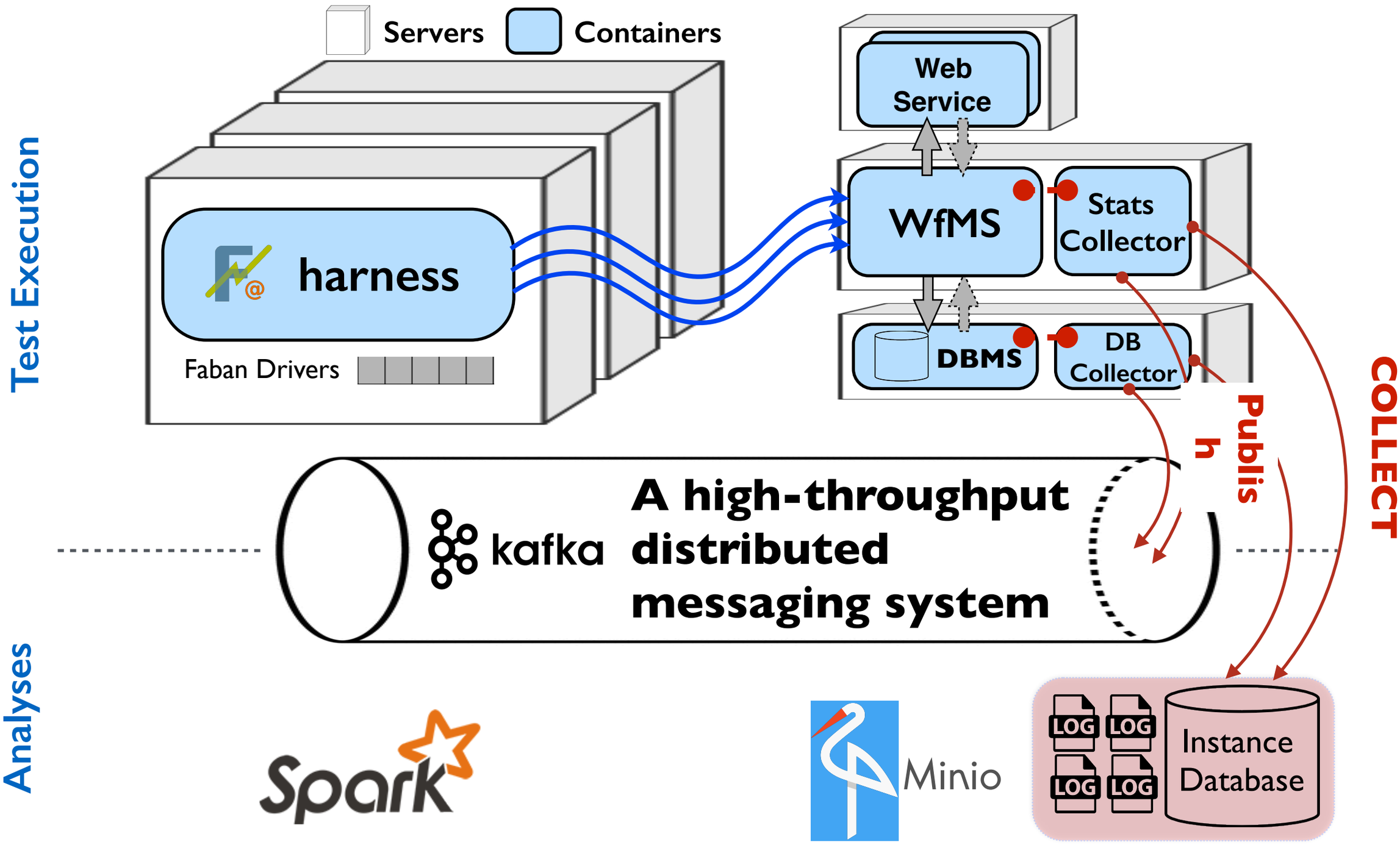
Performance Metrics and KPIs

coordinate data collection and data transformation



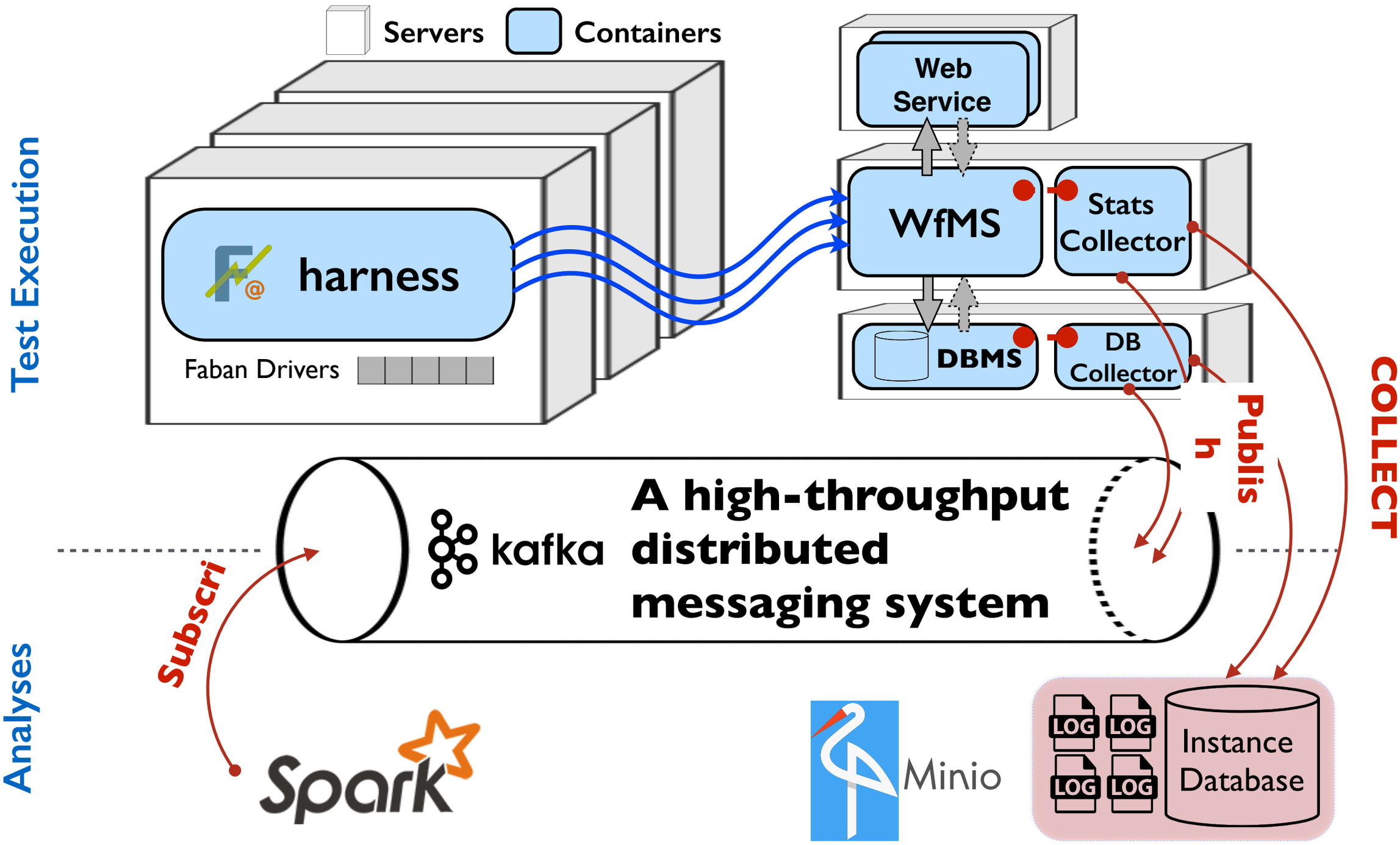
Performance Metrics and KPIs

coordinate data collection and data transformation



Performance Metrics and KPIs

coordinate data collection and data transformation



Performance Metrics and KPIs

coordinate data collection and data transformation