Tool for Production and Performance Testing Based Application Monitoring (pptam)

Contents

[Glossary 3](#_Toc33194973)

[Introduction 4](#_Toc33194974)

[Project goals 4](#_Toc33194975)

[Stakeholders 4](#_Toc33194976)

[Online Material, Links, and Contacts 4](#_Toc33194977)

[Architecture Requirements 5](#_Toc33194978)

[Business Requirements 5](#_Toc33194979)

[Functional Requirements 5](#_Toc33194980)

[Quality Requirements 5](#_Toc33194981)

[Constraints 5](#_Toc33194982)

[System Overview 6](#_Toc33194983)

[Use Cases 6](#_Toc33194984)

[Context View 6](#_Toc33194985)

[Context Data 6](#_Toc33194986)

[System Decomposition 9](#_Toc33194987)

[Functional View 9](#_Toc33194988)

[Functional Data 9](#_Toc33194989)

[Functional Sequence Diagrams 10](#_Toc33194990)

[Logical View 10](#_Toc33194991)

[Logical Data 13](#_Toc33194992)

[Logical Sequence Diagrams 17](#_Toc33194993)

[Technical View 17](#_Toc33194994)

[Technology Overview 17](#_Toc33194995)

[Deployment Overview 17](#_Toc33194996)

[Technical Data 17](#_Toc33194997)

[Source Code Organization 17](#_Toc33194998)

[Technical Sequence Diagrams 17](#_Toc33194999)

[Mapping of Requirements to Solutions 17](#_Toc33195000)

# Glossary

# Introduction

## Project goals

## Stakeholders

## Online Material, Links, and Contacts

# Architecture Requirements

## Business Goals

A user uses PPTAM in order to achieve following goals:

* Anti-pattern detection.
* Attack detection.
* Overal system performance.
* Bottlenecks

## Functional Requirements

* User configures PPTAM with:
  + System under test.
  + Workload.
  + Usage profile.
* User starts PPTAM with user configuration.
* PPTAM executes user defined experiments.
* PPTAM records results about performance of the microservices in experiments.
* PPTAM analyses performance results.
* PPTAM presents analysis results in a graphical manner.

## Quality Requirements

## Constraints

* System under test must be containerized.
* The tool needs to be isolated, preventing the interaction with cloud and other external services.
* PPTAM is developed on Linux.
* User configuration should be provided in JSON format.

# System Overview

## Use Cases

## Context View



Figure 1: Context view of the PPTM system.

Users:

* PPTAM\_External User
  + Accesses the system over a web interface (Figure 1).
    - A user setups parameters of an execution order (Figure 2).
    - A user initiates execution of an execution order (Figure 2).
    - A user observes test results ().
* ~~PPTAM\_Administrator~~
  + ~~Accesses the system over a web interface (Figure 1).~~
  + ~~Accesses source code of the system (Figure 1).~~

## Context Data

Inputs for the PPTAM system are:

* An execution order (Figure 2) with the following parameters:
* Microservices - …
* Workload - …
* Attack mode - …

Outputs of the PPTAM system are:

* A test result (Figure 3) with the following parameters (an example is shown in Figure 4):
  + Type of measures:
  + Analysis conclusions:
  + Dataset - …
  + Microservice:
    - Dataset index
    - Dataset



Figure 2: Context data - execution order.



Figure 3: Context data - test result.

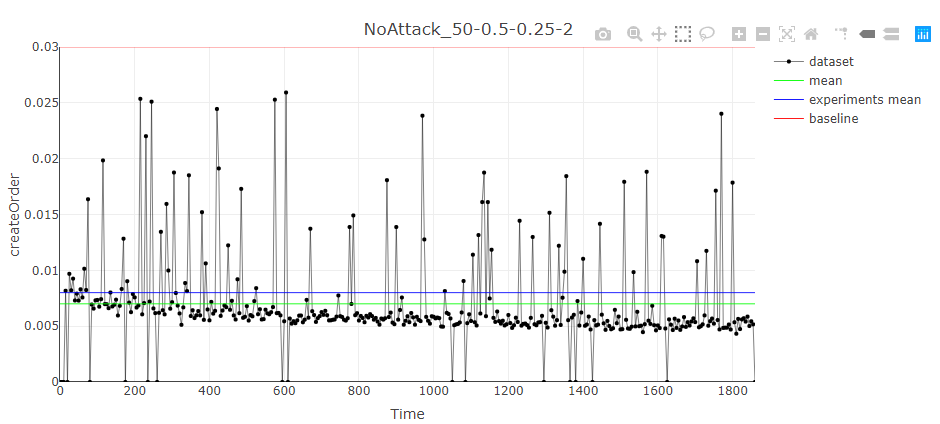


Figure 4: An example of PPTAM test results.

# System Decomposition

## Functional View



Figure 5: Functional view of the PPTAM system.

Functional components comprising the PPTAM system (Figure 4) are:

* Faban Client (*driver*, Figure 8):
  + Responsible for deploying tests to Faban Server.
  + Starting the tests on Faban Server.
  + Getting the diagnostic information of the Faban Server
* Faban Server (*test*):
  + Executes deployed tests.
  + Returns test results to Faban Client.

The communication between the *driver* and the *test* is performed via http requests and responses.

Functional components comprising the Web interface (…) are:

* …

Functional components comprising the Parser (…) are:

* …

Functional components comprising the Analyzer (…) are:

* …

## Functional Data

Inputs for the Faban Client (*driver*) are:

* …

Outputs of Faban Client (*driver*) are:

* Deploy command (Figure 6) with the following parameters:
  + Test ID - …
  + Driver - …
  + Driver configuration - …
  + Docker file - …
* Test status (Figure 6) with the following parameters:
  + Duration - …
* Undeploy command (Figure 6) with the following parameters:
  + Test ID - …

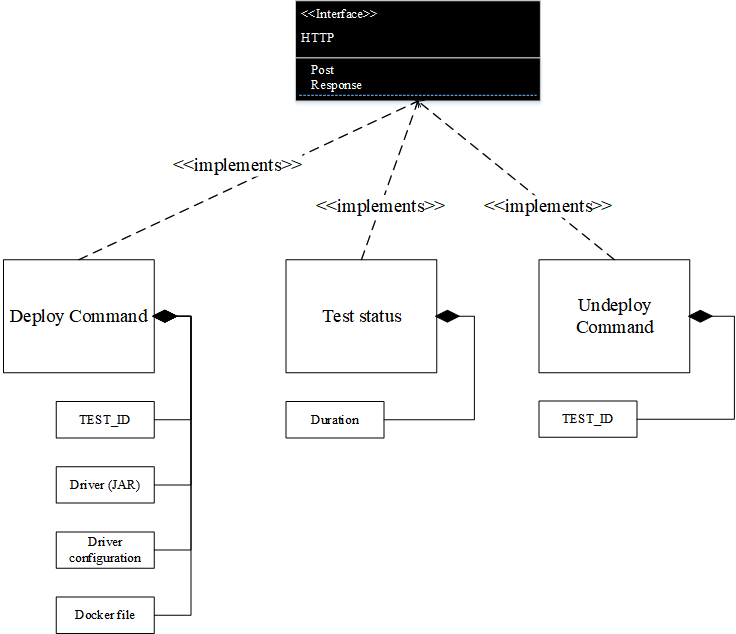


Figure 6: Functional data - HTTP commands between Faban Client and Faban Server

## Functional Sequence Diagrams

## Logical View

Logical components comprising the Faban Client (*driver*) (Figure 8) are:

* Deploy test - …
* Submit and run test - …
* Kill test - …
* Get status - …
* Get run info - …

Logical components comprising the Faban Server (*test*) (…) are:

* …

The communication between the *driver* and the *test* is performed via http requests and responses.

Logical components comprising the Parser (Figure 9) are:

* Parse all directories - …
* Parse single directory - …
* Parse operation - …

Logical components comprising the Analyzer (…) are:

* …

Logical components comprising the Web interface (…) are:

* …



Figure 7: Logical view - Faban components.



Figure 8: Logical view - Faban Client.



Figure 9: Logical view - Parser

## Logical Data

Inputs for the “deploy test” are:

* JAR file (Figure 10) - ...
* Benchmark name (Figure 10) - ...

Inputs for the “submit and run test” test are:

* Benchmark name (Figure 10) - ...
* Profile (Figure 10) - ...
* XML run configuration file (Figure 10) - ...

Inputs for the “kill test” test are:

* Run ID (Figure 10) - ...
* Run configuration (Figure 10) - ...

Inputs for the “get diagnostic” are:

* Run ID (Figure 10) - ...
* Run queue (Figure 10) - ...

Outputs of the “deploy test” are:

* Deploy status (Figure 10) - …

Outputs of the “submit and run test” are:

* Run ID (Figure 10) - …

Outputs of the “kill test” are:

* Run ID (Figure 10) - …

Outputs of the “get diagnostic” are:

* Run status (Figure 10) - …
* Run info (Figure 10) - …, with parameters:
  + Description - …
  + Scale - …
  + Metric - …
  + Submitter - …
  + Tags - …
  + Date - …
  + Result - …
* Run queue (Figure 10) - …
* Run log stream (Figure 10) - …

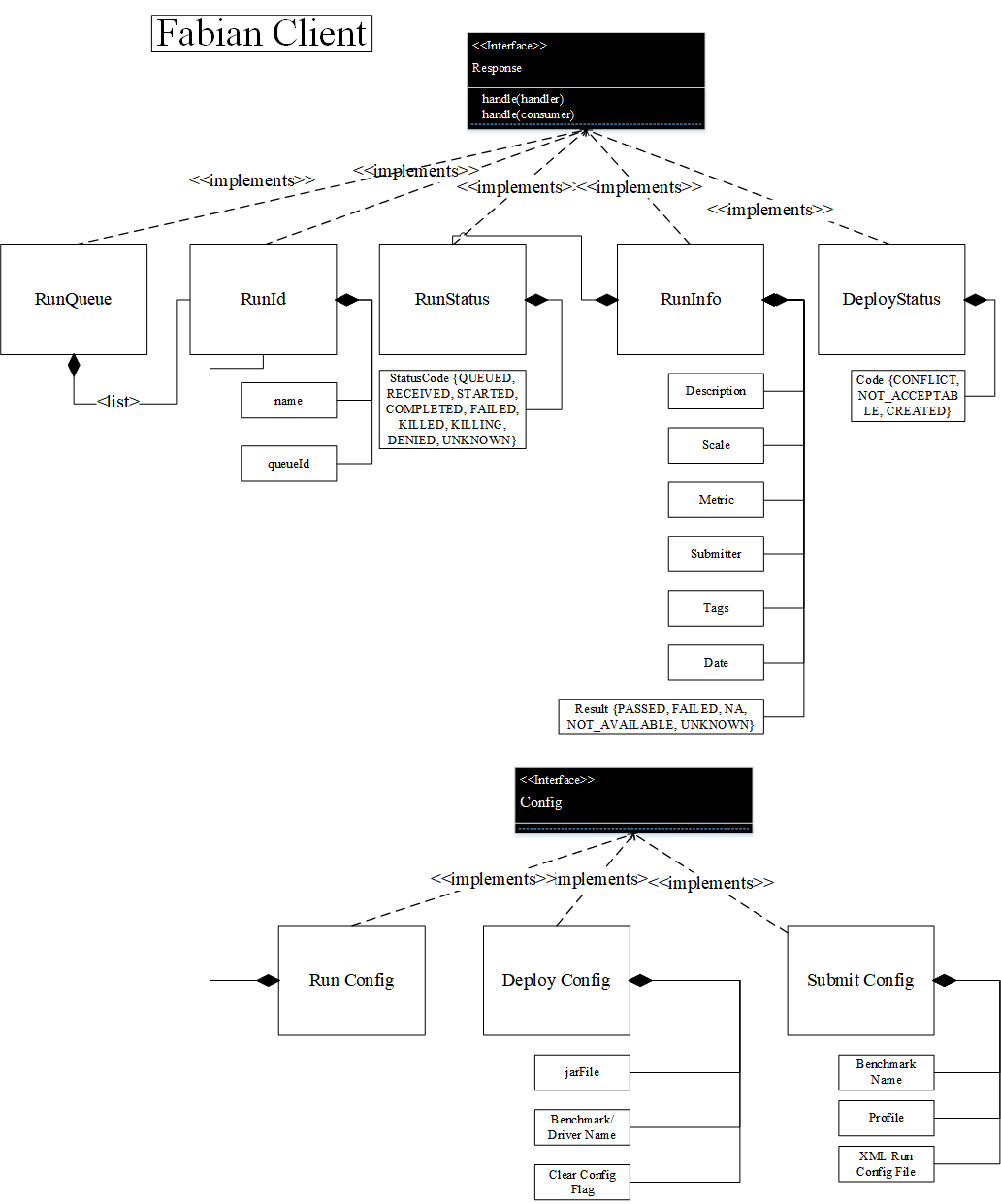


Figure 10: Logical data - messages and requests.



Figure 11: Logical data - parser.



Figure 12: Logical data - analyzer.

## Logical Sequence Diagrams



Figure 13 Logical sequence diagram, starting and running the tests.

## Technical View

### Technology Overview

### Deployment Overview

## Technical Data

## Source Code Organization

## Technical Sequence Diagrams

# Mapping of Requirements to Solutions