D3.3: Documentation and user guidelines of the MEASURE framework

MEASURE

……………………………………………………

Executive summary

The main purpose of WP3 is the implementation of tools to measure, analyse, and visualise the metrics and methods defined in WP2 that extract and show information of the software engineering processes. The main challenge of this WP lies on the ability of working with heterogeneous software engineering processes. The tools implemented in this WP considers the work carried out in WP2 as well as the feedback coming from WPs 4 (data analytics) and 5 (evaluation), in order to refine the tools with the results obtained during the evaluation and industrial testing of the developed solutions.

In particular, WP3 has the following objectives:

- Implement the tools to automatically measure software engineering processes during the whole software lifecycle.
- survey the existing analysis tools and define and implement new ones to extract information from the performed measurements in order to guide the decision making in the software engineering processes.
- Implement visualization tools to expose the extracted results in an easy-readable fashion, so allowing a quick understanding of the situation and the possible actions that can be taken to improve the diverse stages of the software lifecycle.
- Integrate and harmonize the implemented/selected tools to be able to operate in a common framework with the ultimate objective of enhancing the quality and efficiency of software products while lowering their cost and time-to-market.
- consider the feedback obtained from WPs 4 and 5 to make the developed tools more efficient and useful for real industrial deployments.

These objectives have been fulfilled during the MEASURE project, and the developed tools have been integrated in the MEASURE solution called MEASURE platform that is to be launched to the software engineering market, guaranteeing its efficiency and profitableness.

The tools development and integration performed in this WP is completed by a detailed documentation (i.e. the current document D3.3) oriented to guide the users in the utilization of the tools and the best procedures to obtain the highest benefit when using the MEASURE framework during software engineering processes. This documentation is released at the same time of the final version of the MEASURE platform (D3.2).

Table of Contents

1. Introduction

1.1. Role of this deliverable

The deliverable D3.3 is the documentation and user guidelines of the MEASURE framework including the core platform and the measuring, analysis and visualization tools in the MEASURE project. These tools are integrated into a unique platform called the MEASURE platform. D3.2 is the software described in this document.

1.2. Structure of this document

The D3.3 document is organized in one section and set of annexes describing the installation and user guides of the different pieces of software developed in the context of MEASURE. The section introduces the MEASURE documentation plan and its packaging approach.

1.3. Relationship with others MEASURE deliverables

The D3.3 deliverable is related to the list of the following MEASURE deliverables:

- D3.2 Final release of the measuring, analysis and visualization tools that conform the MEASURE platform is the main software outcome of the MEASURE project. The D3.3 documents this software.
- D2.2 Formal specification of MEASURE metrics: The metrics specified in D2.2 will be supported by the MEASURE platform. More extensions to include more metrics are planned.
- D5.3 Intermediate evaluation results from the executed case studies: D3.1 will be the input the deliverable D5.3. The feedback from this evaluation will allow to improve the MEASURE platform.

1.4. Contributors

All the tool providers contributed to this document. Namely:

- SOFT: The MEASURE platform + Hawk Measuring tool + Quality Guard analysis tool
- MTI: MMT measuring tool + Mint analysis tool
- ICAM: EMIT measuring tool + M Elki analysis tool
- IMT: The Metric Suggester analysis tool + Mint analysis tool
- Bitdefender and UniBuc: The Stracker analysis tool

2. Measure Documentation Plan

2.1. Measure Packaging Approach

The MEASURE platform is composed of Core element of the MEASURE platform and a set of extension tools to monitor and gather more measurements or/and a set of analysis tools to perform advanced analysis of collected measures. These components are available on the Github platform following this link:

https://github.com/ITEA3-Measure

The packaging approach for the whole MEASURE platform is composed of the following features:

- Central element: The Measure Platform
  - Including set of Standards Measurements (Provided by MeasureProject)
  - Including set of Standard Application (Provided by MeasureProject)
- First kind of Extensions: measuring tools
  - The Monitoring Tool
  - Set of Measurement associated with this monitoring tool
- Second kind of Extensions: The analysis tools
  - The Analysis Tool
- Measure Catalogue: catalogue of measures from contributors
- Application Catalogue : catalogue of applications from contributors.

2.2. Documentation Publication

The documentation will be basically a public website that is planned to promote the MEASURE platform solution. This website will contain:

- Each component documentation will be
  - available in html on the website
  - downloadable in pdf

2.3. Documentation Plan

The MEASURE platform

- Installation guide (Annexe A)
- User guide (Annexe B)
- Developers guide (Annexe C)

The measuring tools

- Hawk
  - Installation guide (Annexe D)
  - User guide (Annexe E)
- MMT
  - Installation guide (Annexe F)
  - User guide (Annexe G)
- EMIT
  - Installation and user guide (Annexe H)

The analysis tools

- MINT
  - Installation guide (Annexe I)

  - User guide (Annexe J)
- Quality Guard
  - Installation guide (Annexe K)
  - User guide (Annexe L)
- Metric Suggester
  - Installation and user guide (Annexe M)
- M Elki
  - Installation and user guide (Annexe N)
- Stracker
  - Installation guide (Annexe O)
  - User guide (Annexe P)

Annexe A. Measure Platform Installation Guide

Measure Platform Installation Guide

Measurement and Data Analysis Platform

Measure Platform Overview

The measure platform is a tool dedicated to measure, analyse, and visualise the metrics and to extract and show information of the software engineering processes.

- Implement the tools to automatically measure software engineering processes during the whole software lifecycle by executing measures defined in SMM standard and extracted from a catalogue of formal and platform-independent measurements.
- Provide methodologies and tools which allow measure tools provider to develop a catalogue of formal and platform-independent measure.
- Implement storage solution dedicated to measurements resulting of measure execution in big data context.
- Implement visualization tools to expose the extracted results in an easy-readable fashion, so allowing a quick understanding of the situation and the possible actions that can be taken to improve the diverse stages of the software lifecycle.
- Implement an extension mechanism dedicated to the integration of external analysis tools will provide long terms analysis and predictive evaluations on collected measures.
- Implement an Extended API allowing to facilitate the integration on Measure Platform with external tools and services.

The platform activity is organised around its ability to collect measurement by executing measures defined by the SMM standard. SMM measures are auto-executable component, implemented externally, which can be interrogated by the platform to collect measurements.

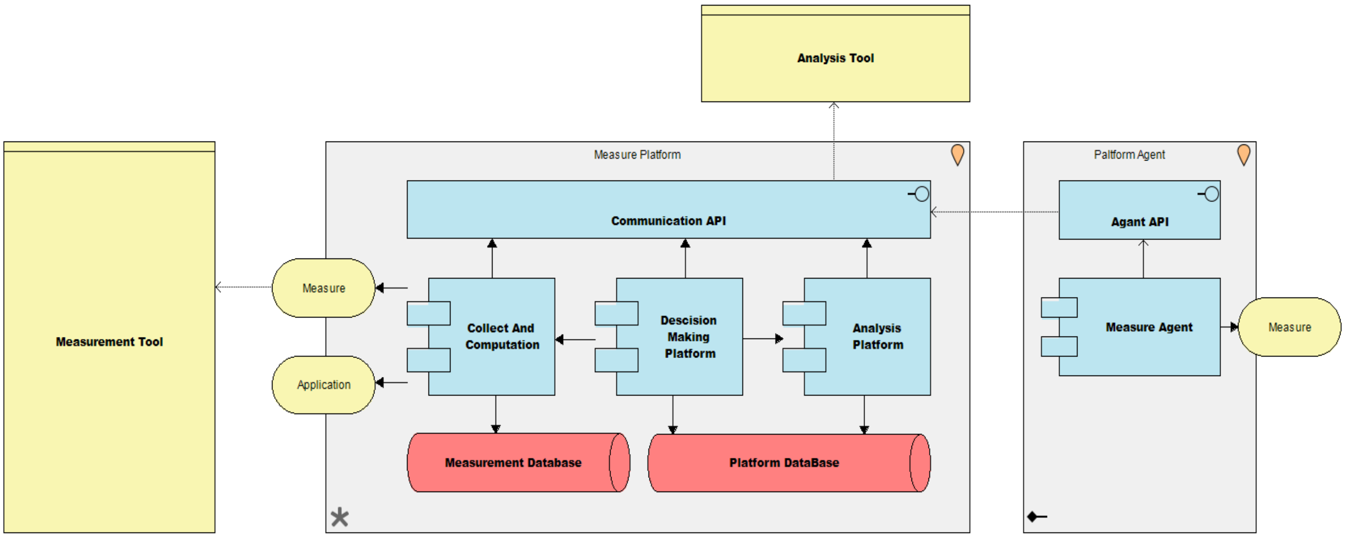


Figure 1. Architecture overview of Measure Platform

Measure Platform: Central component of this deliverable, the Measure platform provides services relate to data collection, analysis and display. It's composed of six sub-components:

Platform Agent: Measure tools on client side which collect data. The executable measure provides a way to collect data in physical world. A measure can be executed on platform side and collect physically this data through an existing measure tool. A Measure can also be directly executed on client side (a computer close to measured element) by the intermediary of a Platform Agent.

Measurement Tools: A Measurement Tool is and external tool which collects or calculates measurement from a specific source. In context of the project, 5 Measurements tools has been developed (see section 3) and a lot of existing tools in the market (such as SonarQube for code quality metrics) can be also considered as Measurement Tools.

Analysis Tools: An Analysis Tool a set of external services which work on the historical measures values in order to provide advanced and valuable analysis function to the platform. In order to support a large set of analyses services and do not limit to it a specific technology, the Analysis Tools are external processes. The analysis tool is integrated to the platform using a specific API. This integration includes embedded visualisation provided by the analysis service into the platform.

Measures: A Measure is a small and autonomous Java program based on the SMM specification which can collect measurements. A Measure can be Direct (Collect of measurement in physical world), a Proxy (Ensure communication between a Measurement Tool and the Platform) or Derived Measure (Measure calculated by the aggregation of existing Measures).

Applications: An Application is a set of Measures aggregated together in order to address a functional requirement. The application is associated with a visual dashboard which is directly integrated into the Decision-Making platform when the Application is deployed on a project.

Hardware and Software Requirements

System	**Linux, Windows**		
Installation Scenario	**Minimum Requirement**	**Standard Configuration (500 Metric Collection, Basic Analysis, 50 Users)**	**Advanced Analysis (+2000 Metrics, All Analysis Services, 200 Users)**
RAM	4 Go	8 Go	16 Go
Processor	32-bit, 4 cores	64-bit, 4 cores	64-bit, 8 cores
Hard Disk	80 GB for system drive	80 GB for system drive	200 GB for system drive

Prerequisite

Install MySQL Database

- Download MySQL Community Server ver. 5.7 or above:

  https://dev.mysql.com/downloads/mysql/

- Install MySQL using these instructions:

  https://dev.mysql.com/doc/refman/5.7/en/installing.html

- Create a new database named "measureplatform".

Using MySQL Command Line Client:

```
CREATE DATABASE measureplatform;
```

Install Elasticsearch

- Download Elasticsearch ver. 5.6 or above (as zip)

  https://www.elastic.co/downloads/elasticsearch

- Unzip the application in your tool directory.

Install Kibana

- Download Kibana ver. 5.4 or above (as zip)

  https://www.elastic.co/downloads/kibana

- Unzip the application in your tool directory

Java 1.8 Installation

- Download and install the jdk8 in your environment:

  http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

Measure Platform Installation

Download

- Download the last released version of the MeasurePlatform

  https://github.com/ITEA3-Measure/MeasurePlatform/releases

- Unzip the platform in your tool directory.

Installation and Configuration

Platform is configured using a property file named **application.properties**. This property file has to be put in the same folder of the measure-platform-1.0.0.jar binary application.

Edit the **application.properties** file:

Table 1 : General Properties of Measure Platform

Property	Description	Default Value
spring.datasource.url	JDBC URL of the database	jdbc:mysql://localhost/measureplatform
spring.datasource.username	Login of the MySQL database.	
spring.datasource.password	Password of the MySQL database.	
spring.datasource.driver-class-name	Driver JDBC for MySQL	com.mysql.jdbc.Driver
measure.repository.path	Path of an empty directory which will be used to store uploaded measures.	./storage
measure.kibana.api	Ip of the Elasticsearch installation.	localhost:9200
measure.kibana.adress	Ip of the Kibana installation.	localhost:5601
measure.kibana.version	Version of Elasticsearch / Kibana	5.6.0
server.port	Port of the MeasurePlatform web application	80

Table 2 : Mail Service Configuration of Measure Platform

Property	Description	Default Value
spring.mail.host	Url of the mail service	smtp.gmail.com
spring.mail.port	Port of the mail service	587
spring.mail.username	Login of the mail account	
spring.mail.password	Password of the mail account	
spring.mail.protocol	mail protocole	smtp
spring.mail.tls		true
spring.mail.properties.mail.smtp.auth		true
spring.mail.properties.mail.smtp.starttls.enable		true
spring.mail.properties.mail.smtp.ssl.trust		smtp.gmail.com

Example of **application.properties** file:

```
measure.repository.path=C:/work/MEASURE/Platform/PackagedPlatform/Platform/storage/
spring.datasource.url=jdbc:mysql://localhost:3306/measureplatform
spring.datasource.username=root
spring.datasource.password=root
spring.datasource.driver-class-name=com.mysql.jdbc.Driver

server.port=8085
measure.kibana.adress=localhost:5601
measure.kibana.api=localhost:9200
measure.kibana.version=5.6.0

spring.mail.username=
spring.mail.password=
spring.mail.host=smtp.gmail.com
spring.mail.port=587
spring.mail.protocol=smtp
spring.mail.tls=true
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.starttls.enable=true
spring.mail.properties.mail.smtp.ssl.trust=smtp.gmail.com
```

Running Measure Platform

1. **Start MySQL**

2. **Start Elasticsearch**

```
./elasticsearch-5.6.0/bin/elasticsearch
```

3. **Start Kibana**

```
./kibana-5.6.0/bin/kibana
```

4. **Start the Measure platform**

```
java -jar measure-platform-{version}.war
```

To access to the Platform: *http://localhost:80/#/*

Connection using the default Platform Account

At deployment, the Measure Platform is created with 2 default accounts:

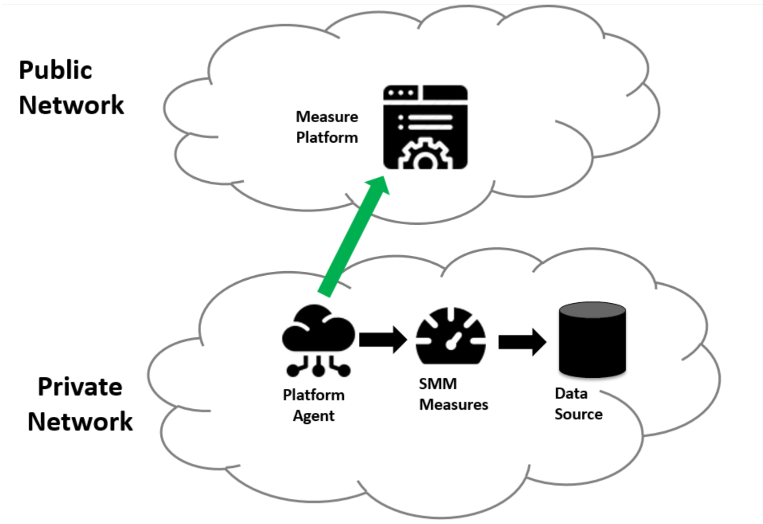
Administrator account		User account	
Login	admin	**Login**	user
Password	admin	**Password**	user

Measure Agent Installation

The Platform Agent in an autonomous application connected to the Measure Platform via the Communication API. It allows to deploy and execute measures on client side and the configuration of measure execution remains administrated remotely by the platform.

In most of the cases, measures are executed on platform side and they manage a connection with external data sources to collect required data. But it's not always possible to execute measures on platform for various reasons like network configuration, security policy, and scalability or because of the mechanisms used by the measure to collect data. To address this issue, we have developed an autonomous agent which provides the following services:

- Allow to execute direct measure on client side, closer of measured element instead of executing it on platform side.
- Ensure the scalability of the Measure platform by providing a way to execute time-consuming measures on remote computers while maintaining a centralized control of measure's execution.
- Solve common issues related to Network Security rules in Industrial context by allowing to deploy the Measure Platform different networks than where the monitored data sources is stored.



Download

- Download the last packaging of the Agent:

  https://github.com/ITEA3-Measure/MeasureAgent/releases

- Unzip the platform in your tool directory.

Installation and Configuration

Platform is configured using a property file named **application.properties**. This property file has to be put in the same folder of the measure-platform-1.0.0.jar binary application.

Edit the **application.properties** file:

Property	Description	Example
measure.repository.path	Path of local directory containing measures	C:/work/MEASURE/Agent/storage
measure.server.adress	IP of MeasurePlatform Server	localhost:80
measure.agent.name	Name of the Agent (*Must be Unique*)	Agent1

Example of **application.properties** file:

```
measure.repository.path=/home/measure/storage
measure.server.adress=xxx.xxx.xxx.xxx/measure
measure.agent.name= MyAgent
```

Deploy Measure on Agent

The Measure which can be executed by the agent must be manually deployed. During the configuration, you have defined the path of the storage folder, a directory which will contain the measures deployed on your agent.

- Unzip the packaged Measure
- Copy the unzipped Measure on storage folder.

Expected Directory Organisation Example:

```
/agent/storage
/agent/storage/MyMeasure
/agent/storage/MyMeasure/MyMeasure-1.0.0.jar
/agent/storage/MyMeasure/MeasureMetadata.xml
/agent/storage/MyMeasure/lib
```

Running Measure Platform Agent

```
java -jar measure-agent-1.0.0.jar
```

Visualise Agent on Server Side

When an agent is started, it is automatically registered on the Measure Platform. The list of agents actually registered on the Measure Platform can be consulted on the "Platform > Remote Agent" page.

Visualise Available Measure

The measures deployed on agents can be visualised on the Measure Catalogue page like other measures deployed on server. The Host indicates the name of the agent which provides this measure.

Instantiate Client-Side Measure

Like for server-side measures, you have to create an instance of a client-side measure in order to collect it. The only differences to a server-side measure is that, at this time it is not possible to execute once a client-side measure. The client-side measure instance has to be scheduled.

Annexe B. Measure Platform User Guide

Measure Platform User Guide

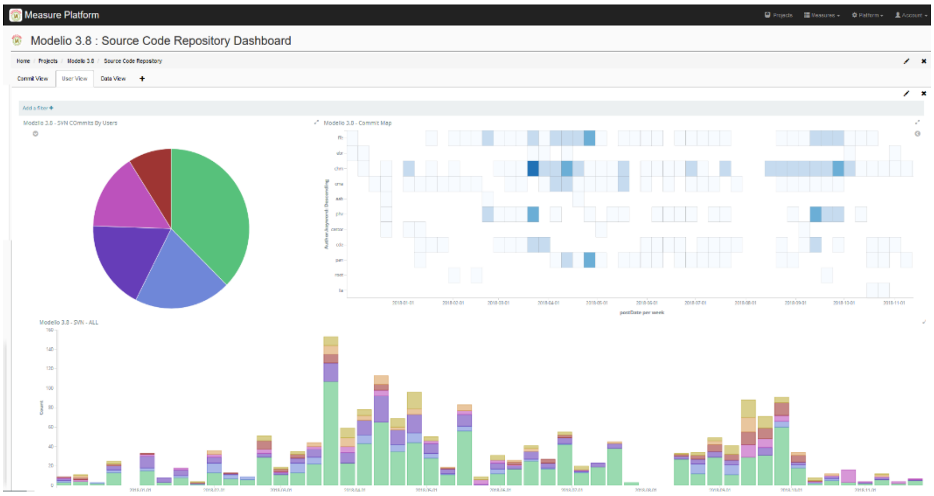
Measurement and Data Analysis Platform

Measure Platform Overview

Platform Presentation

The Measure platform is a tool dedicated to measure, analyse, and visualise the metrics to extract and show information of the software engineering processes.

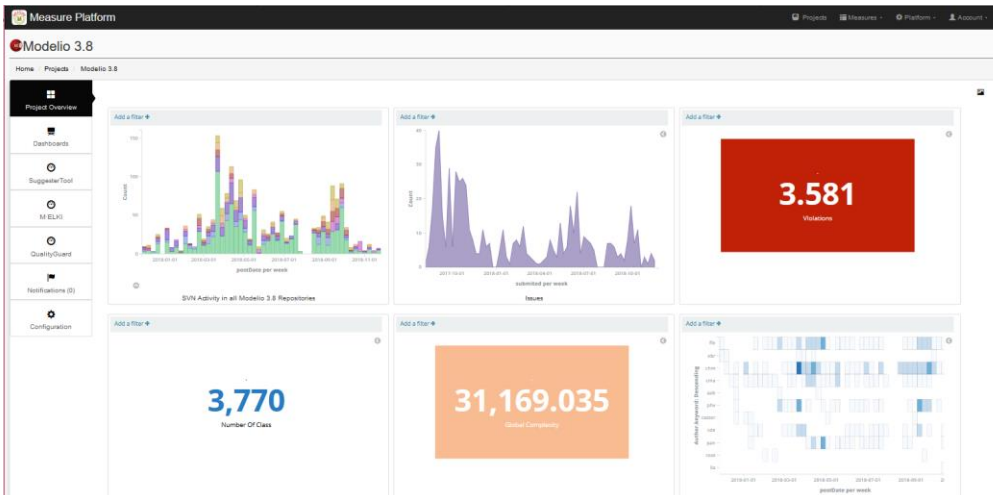
- Implement the tools for automatically measure software engineering processes during the whole software lifecycle by executing measures defined in SMM standard and extracted from a catalogue of formal and platform-independent measurements.
- Provide methodologies and tools which allow measure tools provider to develop a catalogue of formal and platform-independent measure.
- Implement storage solution dedicated to measurements resulting of measure execution in big data context.
- Implement visualization tools to expose the extracted results in an easy-readable fashion, so allowing a quick understanding of the situation and the possible actions that can be taken to improve the diverse stages of the software lifecycle.
- Implement an extension mechanism dedicated to the integration of external analysis tools will provide long terms analysis and predictive evaluations on collected measures.
- Implement of an Extended API allowing to facilitate the integration on Measure Platform with external tools and services.



The Measure platform provides services to host, configure and collect measures, to store measurements and analyse them. These measures are first defined in SMM standard using the Modelio modelling tool and its extension dedicated to SMM modelling. They are packaged under an executable format as Measure Definition (For more details related to measure execution format, please refer to the section 2.4.1 of this document).

Measures are registered and stored on Measure platform in order to initiate the collect of measurements, the next step consists on defining instance of measure based on measure definitions. A measure represents a generic data collection algorithm that has to be instantiated and configured to be applied on a specific context. For example, a measure which collects data related to an SVN repository must be configured by the URL of this repository. Next the Measure Platform can start collecting measurement (data resulting of the execution of an instantiated measure). Collected measurements are stored on a No SQL designed to be able to process a very large amount of data. To collect measurements, the direct measures can delegate the collect work to existing Measure tool.

Stored measurements are presented directly to the end user following a business structured way by the Decision-making platform, a web application which allows organising measures based on projects / software development phases and display its under various forms of charts. The measurements can also be processed by analysis tools to present consolidated results.



Measure and Monitoring Applications

The collect of measurement goes through two kinds of connectors that can be deployed on the Measure Platform: **The Measure** and the **Monitoring Applications.**

- A Measure is a small connector allow to collect measurements. A Measure can be Direct (Collect of measurement in physical world), a Proxy (Ensure communication between a Measurement Tool and the Platform) or Derived Measure (Measure calculated by the aggregation of existing Measures).

  A large set of Measure able to collect measurement form Market Tool or form specific measurements tools are provided with this Platform. This Measures can be found at this URL:

  https://github.com/ITEA3-Measure/Measures/releases

- A Monitoring Application is composed of a set of measures which address the same data source (Ex: a GitHub Repository). These measures are packaged together and, deployed a project, provide automatically a complete Visual Dashboard related to this Data Source.

Analysis Tools and Measurement Tools

The Measure Platform support two kinds of extensions which complete the functionalities provided by the Core Platform

- **Measurement Tools**: External tools that collect or calculate measurements from a specific data source. Measurements Tools are delivered with set of associated Metrics and Monitoring Applications which make the link between the Platform and the Measurement Tool.

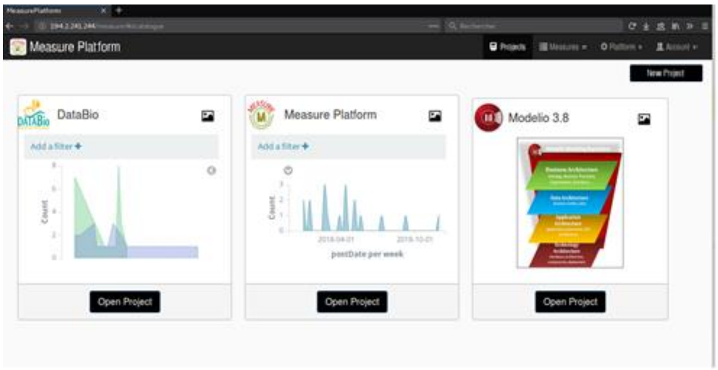
- **Analysis Tools**: Set of external services which work on the historical measures values in order to provide advanced and valuable analysis function to the platform. In order to support a large set of analyses services and do not limit to it a specific technology, the Analysis Tools are external processes. The analysis tool is integrated to the platform using a specific API. This integration includes embedded visualisation provided by the analysis service into the platform

The Analysis and Measurements Tools are packaged separately from the platform and are provided with specific installation guides and user guides.

Project Catalogue

The Project Catalogue list all Monitoring projects currently accessible by the connected user.

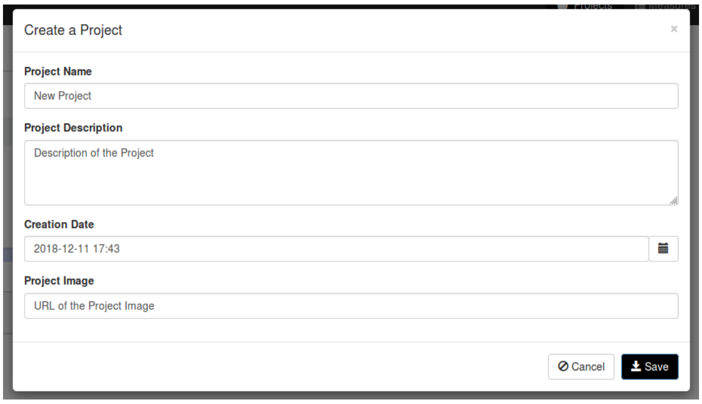
- The creation of a new project is performed in this view.
- A metric overview can be associate to the Project.



Create a Project

The **New Project** button allow to create a Project.

- Name: Name of the project
- Description: Description of the Project
- Creation Date: Creation Date of the Project
- Project Image: URL of an image which will be associate to the project



Create a Project

Project Name

New Project

Project Description

Description of the Project

Creation Date

2018-12-11 17:43

Project Image

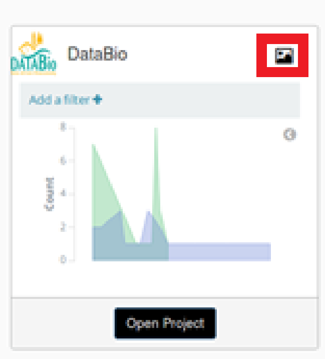
URL of the Project Image

Associate Overview to a Project

By clicking on the View Edition button of each project card, it possible to associate an overview to the project.

This overview can be:

- A metric collected by the project.
- A customised content.



Project Configuration

Configure a Project

The Project Configuration page allows to configure the list of metrics, measurement application and Analysis Tools used in the project.

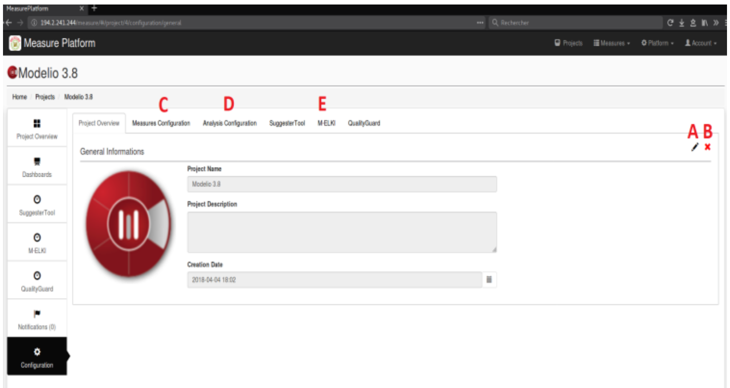
A – Edit project properties

B – Delete project

C – Configure measures and applications

D – Register analysis pools

E - Configure registered analysis tools



General Informations

Project Name: Modelio 3.8

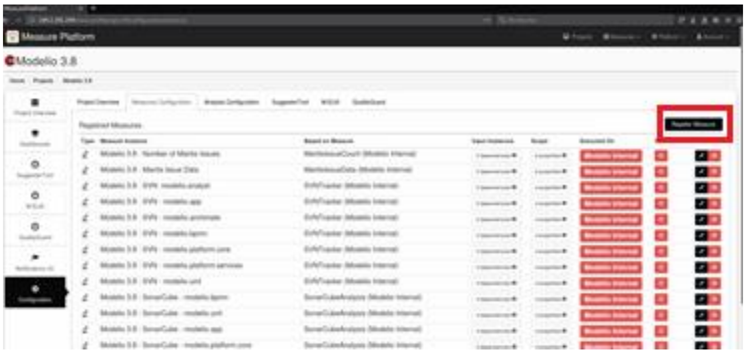
Project Description

Creation Date: 2018-04-04 18:02

Measure Configuration View

To collect measurements, a measure of the Catalogue must be added and configured in the project. The Measure Configuration view list all measure currently deployed on the project

- The **Register Measure** Button allow to create a new Measure
- Existing Measures can be **updated** or **deleted** from this view.



Configure a New Measure

To deploy a new measure, in project, it’s required to fulfil measures parameters (Ex: A Git Hub measure required the URL of the GitHub repository which will be Monitored)

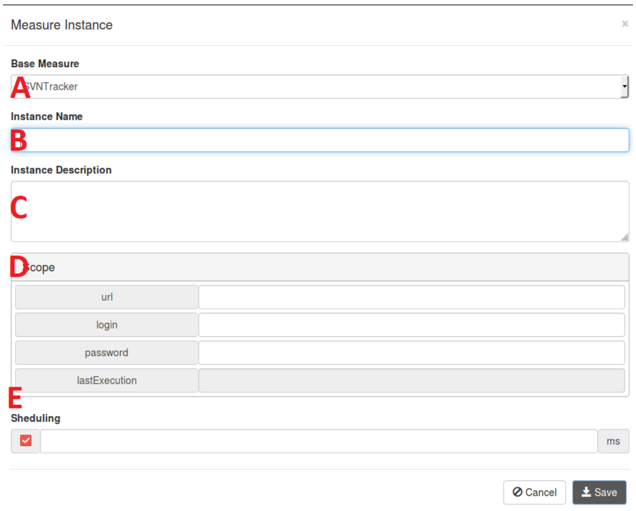
A – Select the kind of measure to deploy

B – Name of the measure

C – Description of the measures

D – Fulfil parameters required by the measure to manage connection with the data source

E – Configure measure scheduling: Measures will be executed periodically by the platform to collect measurement. This Feld allow you to collect periodically the frequencies of this collect.



Measure Instance

Base Measure: SVNTracker

Instance Name

Instance Description

Scope

url

login

password

lastExecution

Sheduling

☑ ms

Execute a Measure

Once configured, a measure cans be executed in many ways:

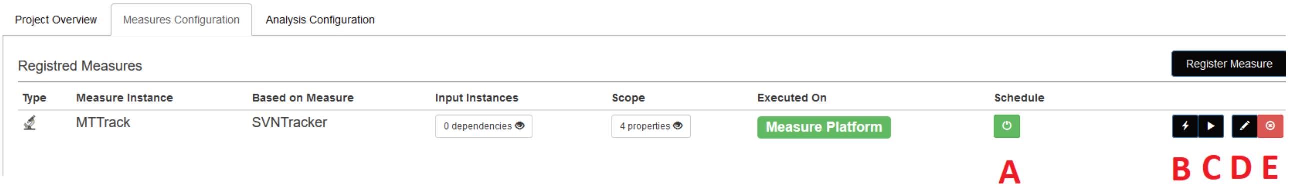
A – Activate Scheduling: If scheduling period has been configured, this button allows you to activate the scheduling. The measures will be collected periodically by the platform.

B – Test the measure: the measure will be executed once for testing purpose. The measure execution result will be displayed but not stored in the measurement database.

C – Execute the measure once. The measure execution result will be display and stored in the measurement database.

D – Edit the measure configuration.

E – Delete the measure.

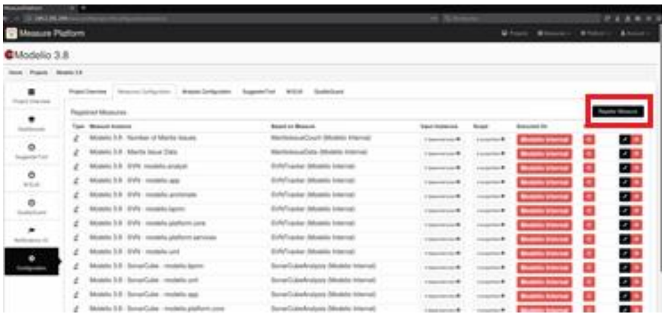


Registred Measures

Type	Measure Instance	Based on Measure	Input Instances	Scope	Executed On	Schedule
	MTTrack	SVNTracker	0 dependencies 👁	4 properties 👁	Measure Platform	

Deploy an Application in a Project

As for Individual Measure, a Measurement Application can be deployed and configured in the project. Once deployed, the measures associated with this application will be automatically configured and collected and a specific dashboard will be integrated in main project view to visualise the collected information.

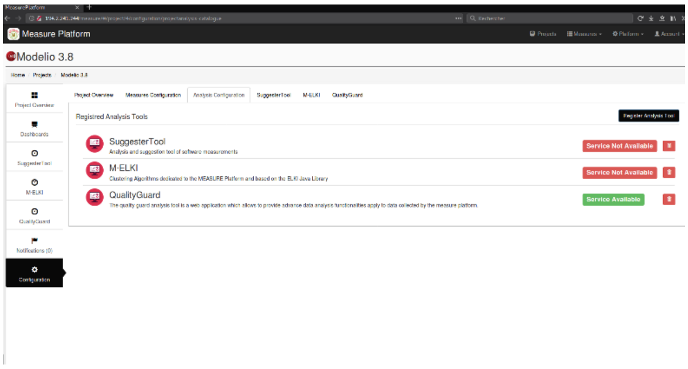


Deploy an Analysis Tool in a Project

Analysis Tools are external analysis services integrated into the Measure Platform. To use one of these services in a project:

- Click on Register Analysis Tool button
- Select the analysis tool to add

Once added, the analysis tool can be configured using the new dedicated view. Please refer to the documentation of this tools for more details about configuration of a specific analysis service.



Modelio 3.8

Registred Analysis Tools

SuggesterTool
Analysis and suggestion tool of software measurements
Service Not Available

M-ELKI
Clustering Algorithms dedicated to the MEASURE Platform and based on the ELKI Java Library
Service Not Available

QualityGuard
The quality guard analysis tool is a web application which allows to provide advance data analysis functionalities apply to data collected by the measure platform.
Service Available

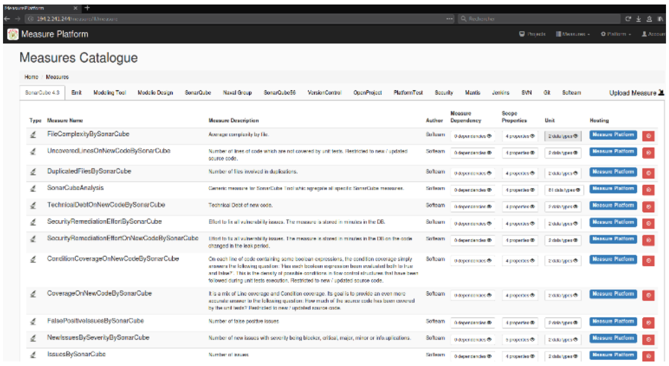
Measure and Application Catalogue

Measure Catalogue

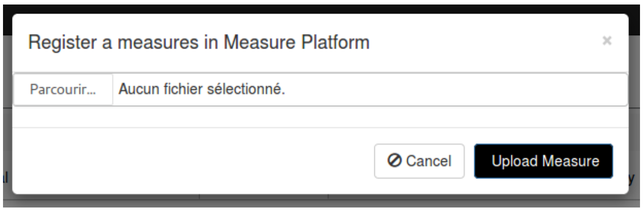
The Measure Catalogue contains the list of Measures supported by the Platform. Organised in a category, each measure is characterised by is name, a description, the scope (configuration parameters) and the data model of measurements return when the measure is executed.

The catalogue allows to:

- Upload a measure in the platform form the packaged measure as zip file.
- Remove a measure of the platform. In this case, all instances of this measures used in projects are deleted to.



Type	Measure Name	Measure Description	Author	Measure Dependency	Scope Properties	Unit	Hosting
	FileComplexityBySonarCube	Average complexity by file.	Softeam	0 dependencies	4 properties	2 data types	Measure Platform
	UncoveredLinesOnNewCodeBySonarCube	Number of lines of code which are not covered by unit tests. Restricted to new / updated source code.	Softeam	0 dependencies	4 properties	2 data types	Measure Platform
	DuplicatedFilesBySonarCube	Number of files involved in duplications.	Softeam	0 dependencies	4 properties	2 data types	Measure Platform
	SonarCubeAnalysis	Generic measure for SonarCube Tool whic agregate all specific SonarCube measures.	Softeam	0 dependencies	4 properties	81 data types	Measure Platform
	TechnicalDebtOnNewCodeBySonarCube	Technical Debt of new code.	Softeam	0 dependencies	4 properties	2 data types	Measure Platform
	SecurityRemediationEffortBySonarCube	Effort to fix all vulnerability issues. The measure is stored in minutes in the DB.	Softeam	0 dependencies	4 properties	2 data types	Measure Platform
	SecurityRemediationEffortOnNewCodeBySonarCube	Effort to fix all vulnerability issues. The measure is stored in minutes in the DB on the code changed in the leak period.	Softeam	0 dependencies	4 properties	2 data types	Measure Platform
	ConditionCoverageOnNewCodeBySonarCube	On each line of code containing some boolean expressions, the condition coverage simply answers the following question: 'Has each boolean expression been evaluated both to true and false?'. This is the density of possible conditions in flow control structures that have been followed during unit tests execution. Restricted to new / updated source code.	Softeam	0 dependencies	4 properties	2 data types	Measure Platform
	CoverageOnNewCodeBySonarCube	It is a mix of Line coverage and Condition coverage. Its goal is to provide an even more accurate answer to the following question: How much of the source code has been covered by the unit tests? Restricted to new / updated source code.	Softeam	0 dependencies	4 properties	2 data types	Measure Platform
	FalsePositiveIssuesBySonarCube	Number of false positive issues	Softeam	0 dependencies	4 properties	2 data types	Measure Platform
	NewIssuesBySeverityBySonarCube	Number of new issues with severity being blocker, critical, major, minor or info.uplications.	Softeam	0 dependencies	5 properties	2 data types	Measure Platform
	IssuesBySonarCube	Number of issues	Softeam	0 dependencies	4 properties	2 data types	Measure Platform


Register a measures in Measure Platform
×
Parcourir...
Aucun fichier sélectionné.
Cancel
Upload Measure

Application Catalogue

As for Measure Catalogue, the Application Catalogue contains the list of Measurement application supported by the Platform.

The Application Catalogue allows to:

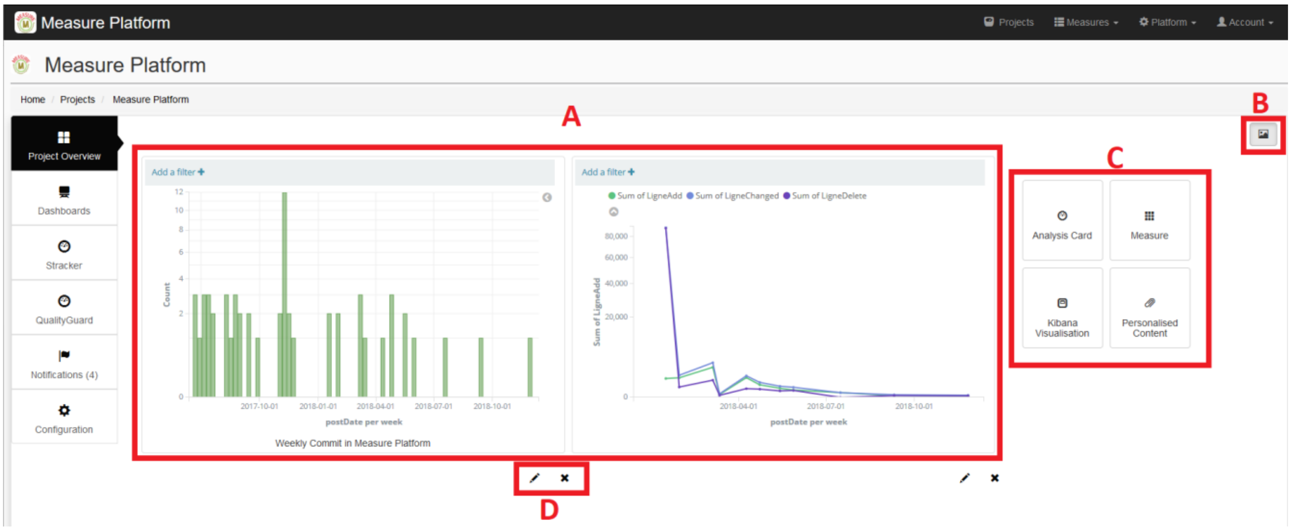
- Upload a measurement application in the platform form the packaged measure as zip file.
- Remove a measurement application of the platform. In this case, the Application will be removed from all projects which used the Application.

Visualise Measures

Each project provides a main dashboard in which collected measurements can be visualised. This dashboard is dynamic and can be configured by the project owner.

Configure Main Dashboard

The Main Project Dashboard allows to visual collects measurements. It can also display synthetics information from analysis services or customized contents.



A – Existing visualisation of measurement

B – Activate / Disactivate the edition model of the dashboard

C – Add a new visualisation in the dashboard

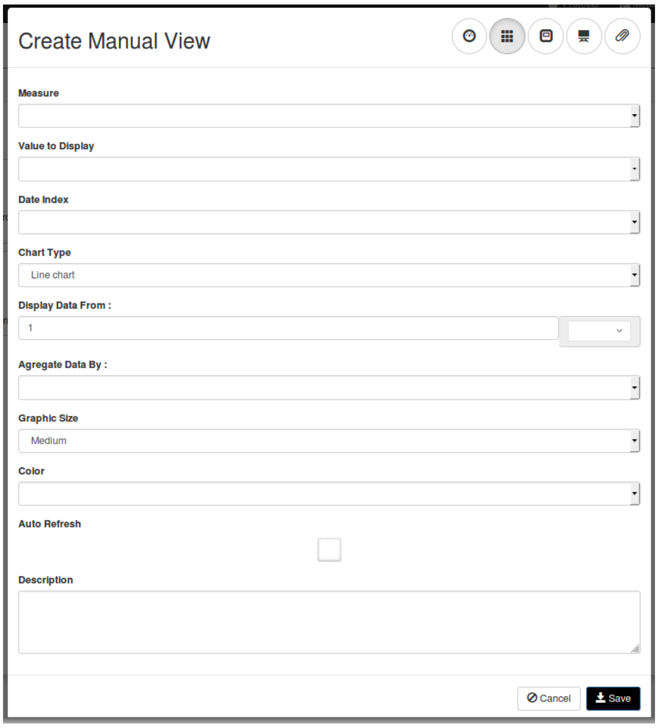
D – Update / Delete existing visualisation

Create a new Measure Visualisation

To visualise measurements collected by a measure, switch the dashboard in edition model and click on the **Add Measure** button.

To visualise the collected measurements as a chart:

- Measure: Select the Measure to visualise. The measure configure on this project will be listed yea.
- Value to Display: If the data model of the measure contained several numerical values, you can select the value you went to visualise.
- Date Index: Select the index used as x axis of the graph: If the data model of the measure contained several data, select the date used as x axis.
- Chart Type: Select the type of char used in visualisation (Bar Charts, Line Chart, Area Chart or Single Value)
- Display Date From: period of time covered by the visualisation (ex: 1 year)
- Aggregate Data By: Select how data points will be aggregated (ex: 1 dot per week)
- Graphic Size: Initial size of the visualisation.
- Color: Main color of the graph.



Create Manual View

Measure

Value to Display

Date Index

Chart Type: Line chart

Display Data From : 1

Agregate Data By :

Graphic Size: Medium

Color

Auto Refresh □

Description

This service allows to visualise simple measurements that are numerical measures periodically collocated and visualise them as Bar Charts, Line Chart, Area Chart or Single Values. To create more complex visualisations, please refer to the next section "Integrate a Kibana Visualisation".

Integrate a Kibana Visualisation

The Measure Platform used the Kibana tool to manage visualisation of collected measurements. Kibana provides a complete editor which allow to create complex visualisations from a set of measurements. If you plan to visualise complex data, you will have to create a Kibana visualisation that will be integrated into the dashboard.

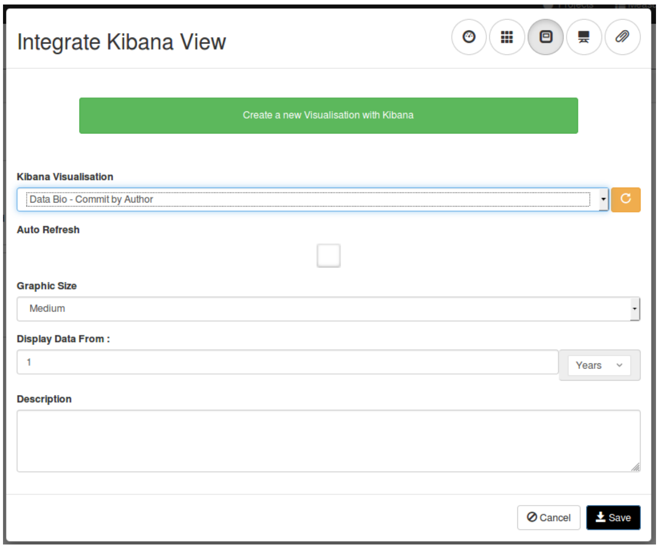
To create a Kibana visualisation, switch the dashboard in edition model and click on the **Add Kibana View** button.

The **Create new Visualisation** button will open the Kibana editors. This editor will allow you to create complex visualisation on your data. For more information related to Kibana, please refer to the documentation of the tool:

https://www.elastic.co/guide/en/kibana/current/index.html

To integrate a Kibana Visualisation:

- Kibana Visualisation: Select the Kibana visualisation to integrate: The list of existing saved Kibana visualisation will be available.
- Graphic Size: Select the size of the visualisation
- Display Date From: period of time covered by the visualisation (ex: 1 year)



Integrate Kibana View

Kibana Visualisation: Data Bio - Commit by Author

Auto Refresh: □

Graphic Size: Medium

Display Data From : 1 Years

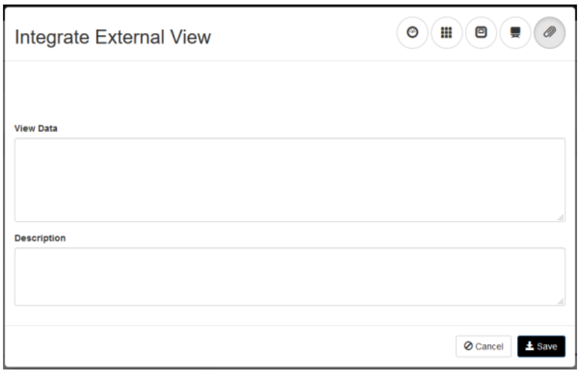
Description

Integrate a customised Content

The current configurable dashboard allows you to integrate customised content like images, descriptions or others test using the HTML 5 format.

To integrate a customised content, switch the dashboard in edition model and click on the **Add Personalised Content** button.

- View Data: integrate your HTML code which can contains texts, images, media …



Integrate External View

View Data

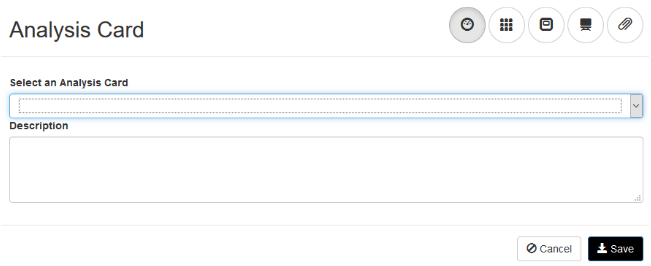
Description

Integrate Analysis Card Visualisation

Platform Analysis tool has the possibility to provide some simplified visualisation of the result of their analysis. This view is called Analysis Card. To integrate an Analysis Card, switch the dashboard in edition model and click on the **Add Analysis Card** button.

To integrate an Analysis Card:

- Select the Analysis Card form the list of cards provided by analysis tools deploy on the current project



Analysis Card

Select an Analysis Card

Description

Additional Dashboards

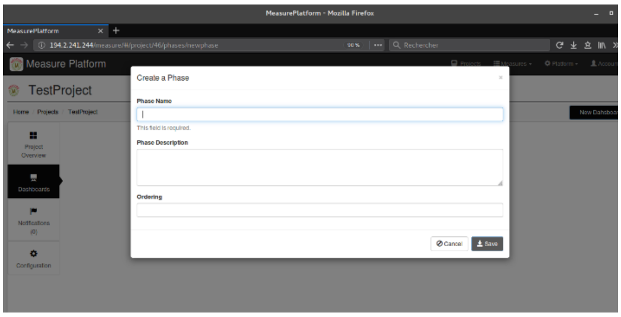
The main Project Dashboard allows you to visualise the key metrics collected on your project, but it will be often required to create new specialised dashboards to organise your metrics visualisation.

Create Additional Dashboard

To create an additional Dashboard, go the Dashboard view and click on the **New Dashboard** button.

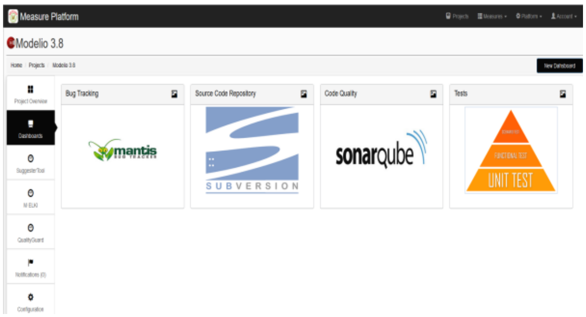
To create a Dashboard:

- Name: Name of the new Dashboard
- Description: Description associated to the dashboard



Once created, the dashboard will be visible in Dashboard View. As for Projects, an overview of the content of the dashboard can be include. To open the Dashboard, click on it.

The Dashboard is composed of several tab. Each of this tab work like the Main Project Dashboard



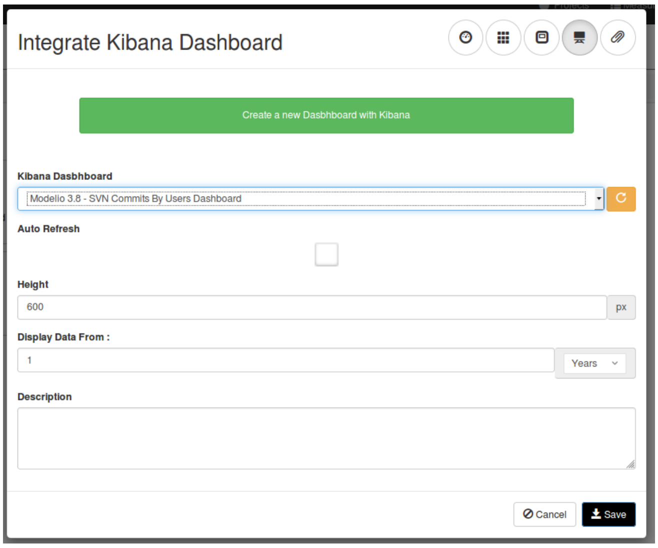
Integrate Kibana Dashboard

On additional dashboard, it is also possible to integrate directly a complete dashboard created with Kibana in the same way as for the measurement visualizations.

To integrate a Kibana Dashboard, create a new Tab using the **+** button. For more information related to dashboard creation with Kibana, please refer to the documentation of the tool:

https://www.elastic.co/guide/en/kibana/current/index.html

- Kibana Dashboard: Select the previously saved Kibana dashboard to integrate
- Height: Indicate the height of the dashboard
- Display Date From: period of time covered by the visualisation (ex : 1 year)


Integrate Kibana Dashboard

Kibana Dashboard: Modelio 3.8 - SVN Commits By Users Dashboard

Auto Refresh: □

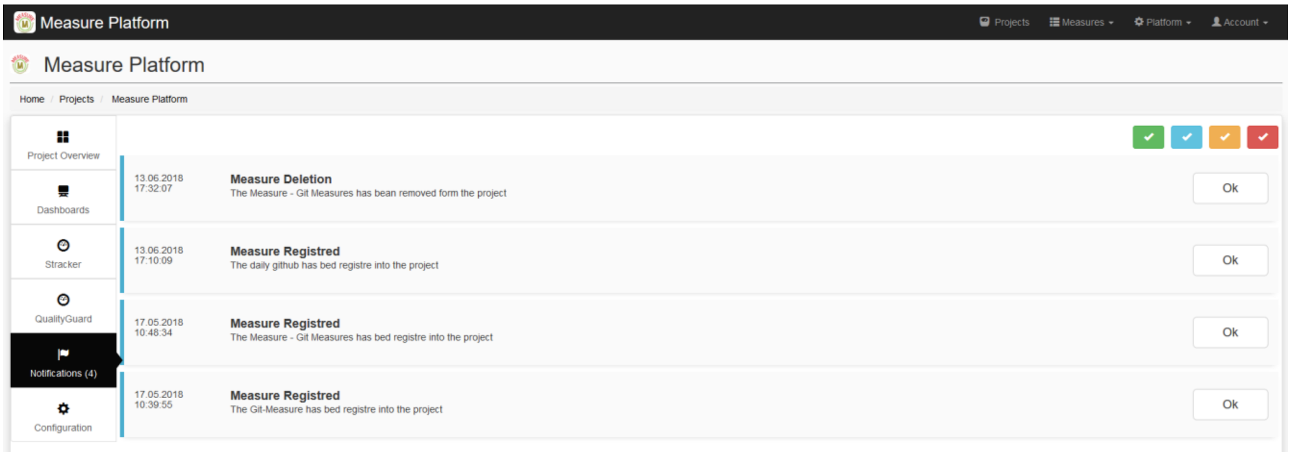
Height: 600 px

Display Data From : 1 Years

Description

Notification Service

The Measure Platform includes a notification service which allows to notify the user of critical event related to the project. Notification notifications are of various types and sent by the platform or the analysis tools.



13.06.2018
17:32:07
Measure Deletion
The Measure - Git Measures has bean removed form the project

13.06.2018
17:10:09
Measure Registred
The daily github has bed registre into the project

17.05.2018
10:48:34
Measure Registred
The Measure - Git Measures has bed registre into the project

17.05.2018
10:39:55
Measure Registred
The Git-Measure has bed registre into the project

Data Analysis

The analysis of measurements collected by the Measure Platform are delegated to Analysis Tool deeply integrated to the platform. The Analysis Tool are set of external services which work on the historical measures values in order to provide advanced and valuable analysis function to the platform. In order to support a large set of analyses services and do not limit to it a specific technology, the Analysis Tools are external processes. The analysis tool is integrated to the platform using a specific API. This integration includes embedded visualisation provided by the analysis service into the platform. The Table below contained a not exhaustive list of Analysis tool currently integrated to the Measure Platform.

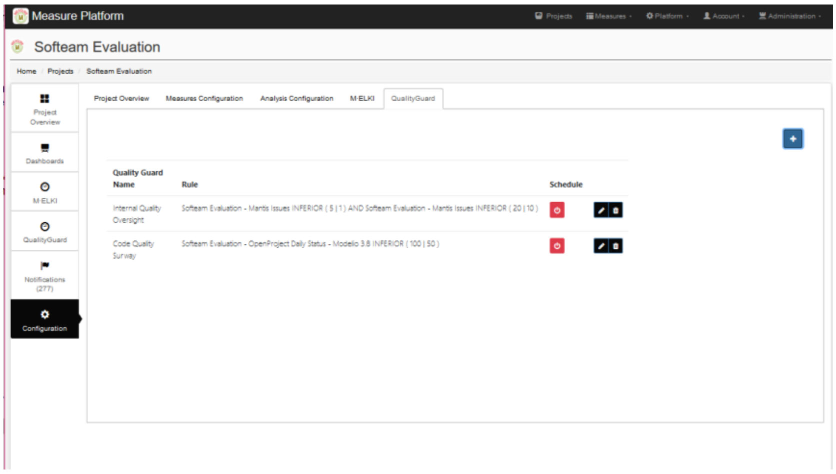
Quality Guard Notification Service	Quality Guard allow to define quality constraints which allow to compare in real-time measures collected by the platform to predefined measure thresholds and send notification if these thresholds are exceeded.
STRACKER Prediction and Forecasting	Metrics prediction, metrics correlation, and metrics forecasting services integrated to the Measure Platform
MINT Recommendation Service	The Metrics Intelligence Tool (MINT) is a software solution designed to correlate metrics from different software development life cycle in order to provide valuable recommendations to different stakeholders impacting the software development process.
Metrics Suggester Measurement Plan Automation	The Metrics Suggester provides a framework to automatize the suggestion of software metrics based on an initial measurement plan. To do so, the framework needs an initial configuration from the user to determine the metrics range to be analysed and the classifier.
M·ELKI Clustering service	M·ELKI is a set of web services that makes possible to select, configure, process and visualize results of several clustering algorithms provided from the ELKI Java library.

The Analysis and Measurements Tools are packaged separately from the platform and are provided with specific installation guides and user guides.

In order to integrate deeply the analysis tools into the Measure Platform, the analysis tools provide web pages which will be embedded to the platform. Each of these views are defined on the platform side by a specific URL. For project specific views, this URL is different for each project. You will see below the list of view which can be provided by the analysis tool and embedded into the Measure Platform.

- **Global Configuration Page (optional):** If the analysis tool requires a way to provide some configuration interface which will be shared by all project, it can provide a global configuration web page.

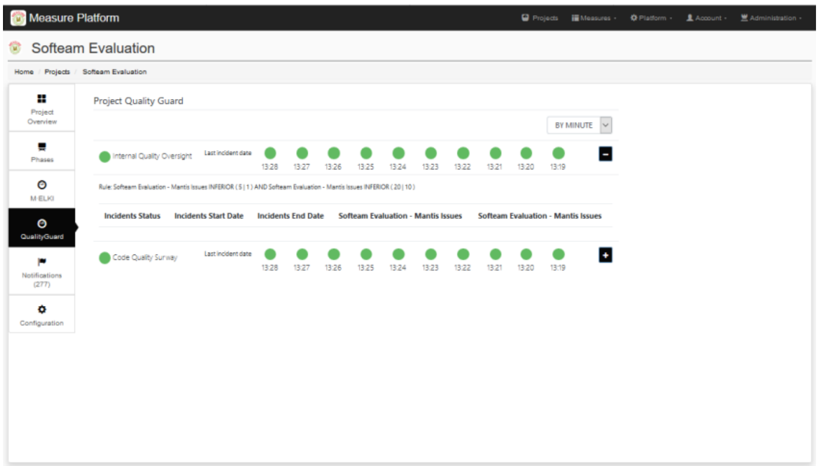
- **Project Specific Analysis Configuration page:** Configuration page which are specific for each project. This page is embedded into project configuration page and allow to configure the analysis service provided by the external analysis tool.



Quality Guard Name	Rule	Schedule
Internal Quality Oversight	Softeam Evaluation - Mantis Issues INFERIOR (5 | 1) AND Softeam Evaluation - Mantis Issues INFERIOR (20 | 10)	
Code Quality Surway	Softeam Evaluation - OpenProject Daily Status - Modelio 3.8 INFERIOR (100 | 50)	

Figure 2 : Analysis tool configuration page of Quality Guard Analysis tool

- **Analysis Tool Main View:** Main view of the analysis tool which are specific for each project. In this view, the analysis service.



Rule: Softeam Evaluation - Mantis Issues INFERIOR (5 | 1) AND Softeam Evaluation - Mantis Issues INFERIOR (20 | 10)

Incidents Status	Incidents Start Date	Incidents End Date	Softeam Evaluation - Mantis Issues	Softeam Evaluation - Mantis Issues

Figure 3 : Main view of the Quality Guard Analysis Tool

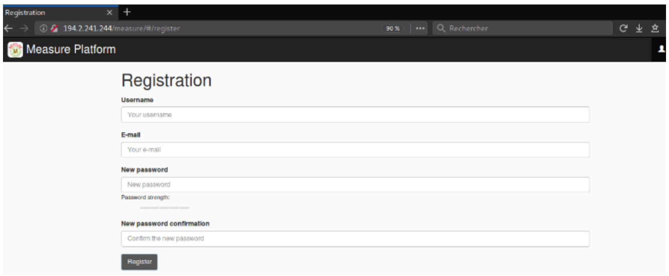
- **Dashboard Card:** Optional small view which can be integrated to projects dashboards in order to provide some key information to project managers related to the service provided by the analysis tool.

Users Management & Access Right

Create a new Account

To use the Measure Platform, you are invited to create a User Account. In front page of the platform, clink on the **Register** button.

The registration page will invite you to provide you Name, Email, and password. An email send at the provided address will allow you to complete the registration process



Registration

Username: Your username

E-mail: Your e-mail

New password: New password

Password strength:

New password confirmation: Confirm the new password

Manage Access Right to Projects

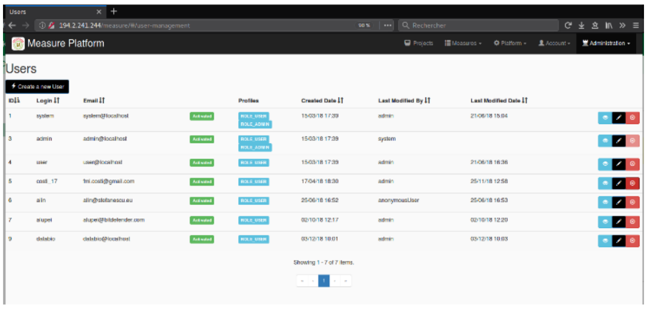
By default, a platform user has access to projects he has created and configured. It’s possible to provide access on specific project to other users registered in Measure Platform.

More information will be available soon.

Platform Administration

The Platform administrator (registered user with administrator role) has access to serval administration services to manage the server. These services are accessible via the **Administration** menu.

User Management: Service allowing to create new users, delete existing users and manage user roles.

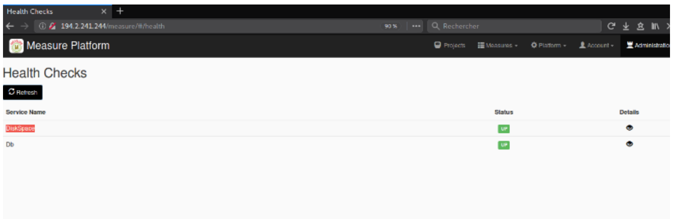


Users

ID	Login	Email		Profiles	Created Date	Last Modified By	Last Modified Date
1	system	system@localhost	Activated	ROLE_USER ROLE_ADMIN	15/03/18 17:39	admin	21/06/18 15:04
3	admin	admin@localhost	Activated	ROLE_USER ROLE_ADMIN	15/03/18 17:39	system	
4	user	user@localhost	Activated	ROLE_USER	15/03/18 17:39	admin	21/06/18 16:36
5	costi_17	fmi.costi@gmail.com	Activated	ROLE_USER	17/04/18 18:30	admin	25/11/18 12:58
6	alin	alin@stefanescu.eu	Activated	ROLE_USER	25/06/18 16:52	anonymousUser	25/06/18 16:53
7	alupei	alupei@bitdefender.com	Activated	ROLE_USER	02/10/18 12:17	admin	02/10/18 12:20
9	databio	databio@localhost	Activated	ROLE_USER	03/12/18 10:01	admin	03/12/18 10:03

Showing 1 - 7 of 7 items.

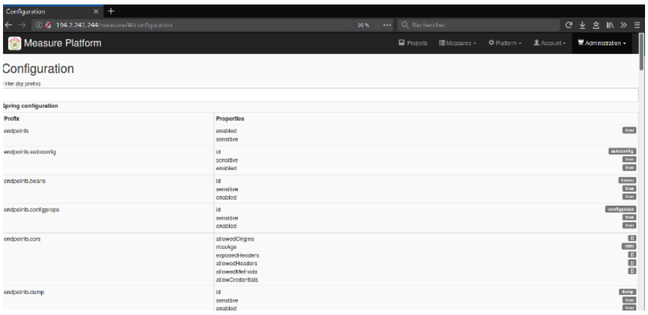
Health Check: Service allowing to check the availability of resources consumed by the server (Disk Space, Memory, etc.)



Health Checks

Service Name	Status	Details
DiskSpace	UP	
Db	UP	

Configuration: Service allowing to configure all aspects of the Measure Platform server.



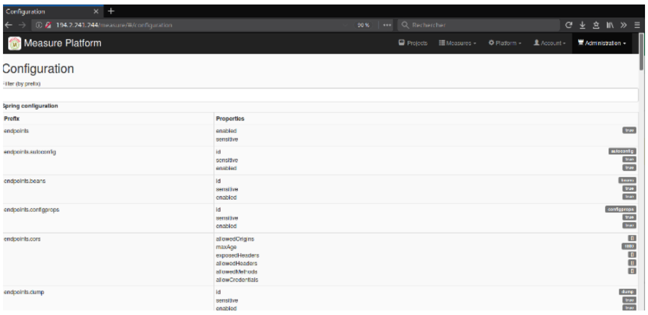
Configuration

Filter (by prefix)

Spring configuration

Prefix	Properties	
endpoints	enabled sensitive	true
endpoints.autoconfig	id sensitive enabled	autoconfig true true
endpoints.beans	id sensitive enabled	beans true true
endpoints.configprops	id sensitive enabled	configprops true true
endpoints.cors	allowedOrigins maxAge exposedHeaders allowedHeaders allowedMethods allowCredentials	[] 1800 [] [] []
endpoints.dump	id sensitive enabled	dump true true

Configuration: Service allowing to configure all aspects of the Measure Platform server.

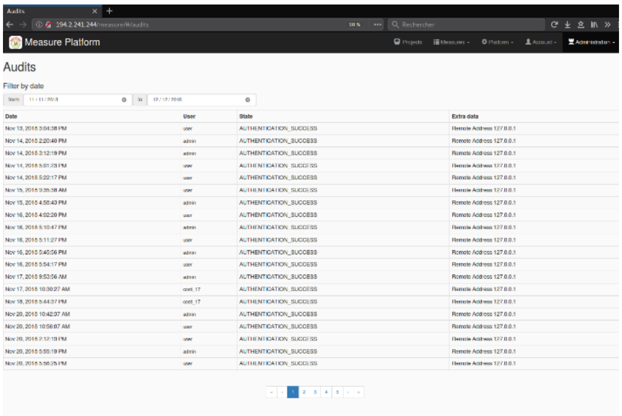

Configuration

Filter (by prefix)

Spring configuration

Prefix	Properties	
endpoints	enabled sensitive	true
endpoints.autoconfig	id sensitive enabled	autoconfig true true
endpoints.beans	id sensitive enabled	beans true true
endpoints.configprops	id sensitive enabled	configprops true true
endpoints.cors	allowedOrigins maxAge exposedHeaders allowedHeaders allowedMethods allowCredentials	[] 1800 [] [] []
endpoints.dump	id sensitive enabled	dump true true

Security Log: Logs of connections to the Measure Platform



Audits

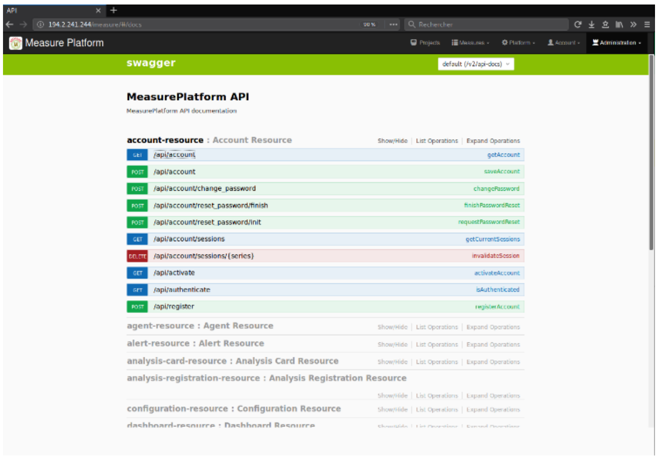
Filter by date

from 11 / 11 / 2018 to 12 / 12 / 2018

Date	User	State	Extra data
Nov 13, 2018 3:04:38 PM	user	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 14, 2018 2:20:40 PM	admin	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 14, 2018 3:12:19 PM	admin	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 14, 2018 5:01:23 PM	user	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 14, 2018 5:22:17 PM	user	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 15, 2018 9:35:38 AM	user	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 15, 2018 4:58:43 PM	admin	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 16, 2018 4:02:20 PM	user	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 16, 2018 5:10:47 PM	admin	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 16, 2018 5:11:27 PM	user	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 16, 2018 5:45:56 PM	admin	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 16, 2018 5:54:17 PM	user	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 17, 2018 9:53:56 AM	admin	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 17, 2018 10:30:27 AM	cost_17	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 18, 2018 5:44:37 PM	cost_17	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 20, 2018 10:42:37 AM	admin	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 20, 2018 10:56:07 AM	user	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 20, 2018 2:12:19 PM	user	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 20, 2018 5:55:19 PM	admin	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1
Nov 20, 2018 5:56:25 PM	user	AUTHENTICATION_SUCCESS	Remote Address 127.0.0.1

« ‹ 1 2 3 4 5 › »

Platform API: Swagger description of the REST API exposed by the Measure Platform



MeasurePlatform API

MeasurePlatform API documentation

account-resource : Account Resource — Show/Hide | List Operations | Expand Operations

GET	/api/account	getAccount
POST	/api/account	saveAccount
POST	/api/account/change_password	changePassword
POST	/api/account/reset_password/finish	finishPasswordReset
POST	/api/account/reset_password/init	requestPasswordReset
GET	/api/account/sessions	getCurrentSessions
DELETE	/api/account/sessions/{series}	invalidateSession
GET	/api/activate	activateAccount
GET	/api/authenticate	isAuthenticated
POST	/api/register	registerAccount

agent-resource : Agent Resource — Show/Hide | List Operations | Expand Operations

alert-resource : Alert Resource — Show/Hide | List Operations | Expand Operations

analysis-card-resource : Analysis Card Resource — Show/Hide | List Operations | Expand Operations

analysis-registration-resource : Analysis Registration Resource — Show/Hide | List Operations | Expand Operations

configuration-resource : Configuration Resource — Show/Hide | List Operations | Expand Operations

dashboard-resource : Dashboard Resource — Show/Hide | List Operations | Expand Operations

Extend the Platform

The Measure platform can be extended following three ways in order to support new data sources, new way to visualise collected data or now data analysis services.

Development of New Measure

A Measure is a small and autonomous java program based on the SMM specification which allow to collect measurements. The Measures make the link between a Measurement Tool, a Remote Service, a Captor or any others kind of data sources and the Measure Platform.

If you plan to monitor measures that are not currently supported by the measure platform, you will probably have to develop your own Measures.

There are two kind of Measures:

- The **Direct Measure** (Collect of measurement in physical world), a Proxy (Ensure communication between a Measurement Tool and the Platform) or
- The **Derived Measure** (Measure calculated by the aggregation of existing Measures).

For more information related to Measure development, please refer to the Measure Platform Developers Guide.

Development of New Measurement Applications

An Application is a set of Measures aggregated together in order to address a functional requirement. The application is associate with a visual dashboard which directly integrated into the Decision-Making platform when the Application is deployed on a project.

For more information related to Measurement Application development, please refer to the Measure Platform Developers Guide.

Development of a New Analysis Service

In order to support a large set of analyses services and do not limit to it a specific technology, the Analysis Tools are external processes. Although external, we wanted a deep integration between the platform and the analysis tools. We solved this issue in the following way:

- The Measure platform provides a REST API which allows an analysis tool to register it on the platform, to receive notifications from the platform and access to information related to project defined and measure collected by the platform.
- On its side, the analysis tool provides some web pages which will be embedded into the platform web application.

For more information related to Analysis Service development, please refer to the Measure Platform Developers Guide.

Annexe C. Measure Platform Developers Guide

Measure Platform Developers Guide

Measurement and Data Analysis Platform

Extend the Measure Platform

The Measure platform can be extended in order to support new data sources, new ways to visualise collected data or now data analysis services. The Measure Platform developers guide provides the required key to the development of these extensions. The Platform can be extended as follows:

Development of New Measure

A Measure is a small and autonomous java program based on the SMM specification which allow to collect measurements. The Measures make the link between a Measurement Tool, a Remote Service, a Captor or any others kind of data sources and the Measure Platform.

If you plan to monitor measures that are not currently supported by the measure platform, you will probably have to develop your own Measures.

There two kind of Measures:

- The **Direct Measure** (Collect of measurement in physical world), a Proxy (Ensure communication between a Measurement Tool and the Platform) or
- The **Derived Measure** (Measure calculated by the aggregation of existing Measures).

Development of New Measurement Applications

An Application is a set of Measures aggregated together in order to address functional requirements. The application is associated with a visual dashboard which directly integrated into the Decision-Making platform when the Application is deployed on a project.

Development of New Analysis Service

In order to support a large set of analyses services and do not limit to it a specific technology, the Analysis Tools are external processes. Although external, we wanted a deep integration between the platform and the analysis tools. We solved this issue in the following way:

- The Measure platform provides a REST API which allows an analysis tool to register it on the platform, to receive notification from the platform and access to information related to project defined and measure collected by the platform.
- On its side, the analysis tool provides some web pages which will be embedded into the platform web application.

Measure Development

Measure Architecture

A SMM Measure is a small and independent software component which allows retrieving or calculating a measurement. The Measure make the link between a remote measurement or service and the MeasurePlatform. The implementation of a measure is based on a library developed in parallel with the Measure Platform: the “SMMMeasureApi”. We identify two main kinds of measures: The Direct Measures and the Derived Measures.

- A **Direct Measure** is used to collect data in physical world. This kind of measure can be executed on the platform on the Client Side. To define a Direct Measure, the IDirectMeasure has to be implemented. This interface will be called by the Measure Platform to retrieve the measurements.

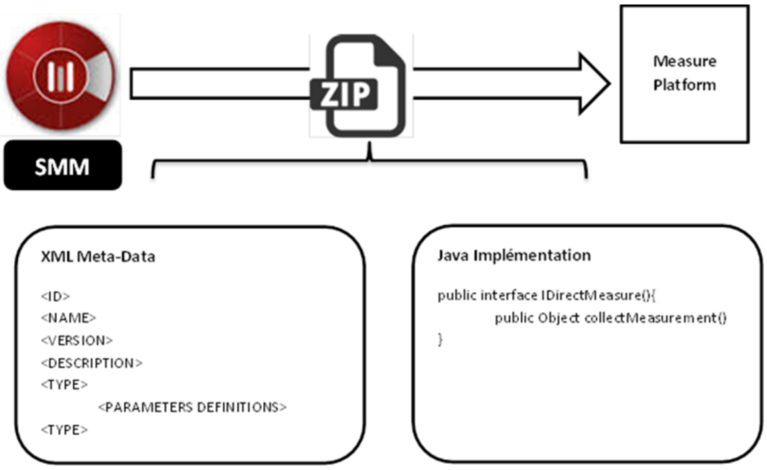
- A **Derived Measure** is used to define a combined measure which can calculate new measurements using one or more measurements stored on the Measure platform. To define a Derived Measure, the IDerivedMeasure has to be implemented.

In this section, we will describe how to specify, implement and package a new measure which will be deployed on the Measure platform. The Modelio Modeling tool can be used for the specification, implementation and packaging of measures, but it is also possible to implements the measure manually.

In SMM, a direct or derived measure definition is associated with an Operation which represents the implementation of the measure. These operations can be expressed in natural language or may contain executable code. In order to be able to collect direct measures and to execute calculated measure, we have to choose a common executable language. For that, we currently support Java.

An SMM Measure is a zip file containing:

- A **Jar file**: The Java implementation of the measure
- A **lib folder**: Java libraries used by the measure implementation
- A **MetaData.xml file**: Metadata related to the measure



The Jar file contained the implementation the measure itself. In order to be executed by the platform, this implementation is based on the SMMMeasureApi available at this URL:

https://github.com/ITEA3-Measure/SMMMeasureApi/

The metadata file is an xml file containing several information's related to the measure and used by the measure platform to load dynamically the Measure. It allows to define the scope (dynamic properties provided wen the measure is deploy on a project of the Measure Platform) and the data model returned by the measure when executed.

Develop a Measure Using Maven

In order to help you to start the development of a new Measure, a Maven Archetype is available on our Maven repository: *http://repository.modelio.org*

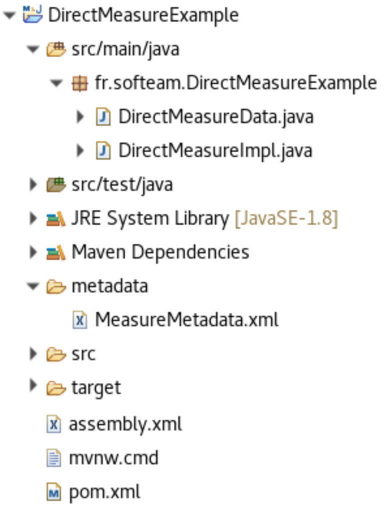
To create the implementation project:

- Create a new Maven project using an Archetype

- Register the Modelio maven repository ass new remote maven catalogue http://repository.modelio.org

- Select the **DirectMeasure** Archetype or the **DerivedMeasure** Archetype depending of the kind of measure you which to implement.

- Create your Java measure implementation project



```
DirectMeasureExample
  src/main/java
    fr.softeam.DirectMeasureExample
      DirectMeasureData.java
      DirectMeasureImpl.java
  src/test/java
  JRE System Library [JavaSE-1.8]
  Maven Dependencies
  metadata
    MeasureMetadata.xml
  src
  target
  assembly.xml
  mvnw.cmd
  pom.xml
```

Use an existing project template which will allow you to start the implementation of a new measure is also available at this address:

https://github.com/ITEA3-Measure/Measures/tree/master/_Examples/TemplateMeasure

Once the implementation of the measure is completed, you can package your measure as ZIP in a format compatible with the Measure Platform using the Maven Install compilation target.

Develop a Measure using the Modelio Modelling Tool

The Modelio Modeling tool supports the Structured Metrics Model (SMM) standard. This specification defines a meta-model for representing measurement information related to any model-based information with an initial focus on software, its operation, and its design. Referred to as the Structured Metrics Meta-model (SMM), this specification is an extensible meta-model for exchanging both measures and measurement information concerning artefacts contained or expressed by structured models, such as MOF.

The SMMLibrary Module is an extension for Modelio 3.4 tool, which allows to model, specify, implement, and package new catalogue of measures in SMM format.

1. **Download the Modelio Open Source 3.4.1**: *https://www.modelio.org/downloads/download-modelio.html* OR Direct link to 3.4.1 : *https://sourceforge.net/projects/modeliouml/files/3.4.1/*
2. **Download the last SMMDesigner Module**: *https://github.com/ITEA3-Measure/SMM-Designer/releases/tag/0.3.00* Download file : SMM_0.3.00.jmdac
3. **Start Modelio and create a new Project**.
4. **Add the** SMM_0.3.00.jmdac module into the project

This module will allow you to:

- Specify scope, data model and dependency of the measure using Models

- Generate a Maven implementation project based on this specification.

- Help you to implement the measure using Model Driven Development Approach

- Package the measure in a format supported by the Measure Platform

Please refer to the documentation of the SMM Module for more details about the development of measures using Modelio.

Measure Metadata File

The MetaData.xml file contains meta-data related to the SMM Measure:

- Name, description, category and provider of the Measure
- Type of the Measure

- Unite (data model) of the measure
- List of properties of the measure
- List of references for Derived Measure (inputs form other measures)

Element	**Owner**	**Attribute**	**Description**
Measure			The Measure
		name	Name / Id of the Measure
		type	SMM Type of the measure: [DIRECT,COLLECTIVE,RACKING,GRADE,BINARY,COUNTING,ESCALED,RATIO]
		category	Classification of the measure by category
		provider	People / Entity which developed the measure
description	Measure (1)		Description of the Measure
unite	Measure (1)		Data Model of measurements returned by the measure
fields	Unite (*)		A Field of the measure unite
		fieldName	Name of the field
		fieldType	Type of the field: [u_text,u_integer,u_long,u_date,u_boolean,u_float,u_geo_point,...]
scopeProperties	Measure (*)		A property user to configure the execution of the measure
		name	Name of the property
		defaultValue	Default value of the property
		type	Type of the property: :[STRING,INTEGER,FLOAT,DATE,ENUM,PASSWORD,DESABLE]
description	scopeProperties (1)		Description of the scope property
enumType	scopeProperties (1)		Emum definition for scopeProperties of type ENUM

enumvalue	enumType (*)		Emum values
		label	label of the enum entry
		value	value of the enum entry
references	Measure (*)		References to inputs required by Derived Measures
		measureRef	Name of the required measure
		number	Default Number of instances of this input required
		expirationDelay	Filter old measurement
references-role	References (1)		Role of impute in current Measurement. This role allows to identify several instance of the same Measure in a DerivedMeasure.

Example of a MeasureMetaData.xml file

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Measure name="RandomMeasure" type="DIRECT"  category=”Test” provider=”MeasurePlatform”>
   <description>Return a random measure and his variation</description>
   <scopeProperties defaultValue="100" name="MaxRange" type="INTEGER">
      <description>MaxRange</description>
   </scopeProperties>
   <scopeProperties defaultValue="0" name="MinRange" type="FLOAT">
      <description>MinRange</description>
   </scopeProperties>
   <scopeProperties defaultValue="0" name="PreviousValue" type="DESABLE">
      <description>PreviousValue</description>
   </scopeProperties>
   <scopeProperties defaultValue="Borned" name="Kind" type="ENUM">
      <description>Kind</description>
      <enumType>
         <enumvalue label="Is Borrned" value="Borned"/>
         <enumvalue label="Not Borrned" value="UnBorned"/>
      </enumType>
   </scopeProperties>
   <scopeProperties name="TestDate" type="DATE">
      <description>TestDate</description>
   </scopeProperties>
   <scopeProperties name="TestPassword" type="PASSWORD">
      <description>TestPassword</description>
   </scopeProperties>
   <scopeProperties name="TestString" type="STRING">
      <description>TestString</description>
   </scopeProperties>
   <unit name="RandomMeasurement">
      <fields fieldName="FinalValue4" fieldType="u_double"/>
      <fields fieldName="Variation4" fieldType="u_integer"/>
      <fields fieldName="myDate" fieldType="u_date"/>
   </unit>
</Measure>
```

Direct Measure Implementation

A direct measure is used to collect data in physical world. This kind of measure can be executed on the platform on Client Side. To define a Direct Measure, implement the IDirectMeasure interface. This interface will be called by the MeasurePlatform to retrieve the measurements.

```
public interface IDirectMeasure {
        public List<IMeasurement> getMeasurement() throws Exception;
        public Map<String,String> getProperties();
}
```

To implement a direct measure, please extend the DirectMeasure class:

- **getMeasurement():** calculates and returns a list of measurements.
- **getProperties():** provides a way for the Measure platform to communicate properties to DirectMeasure implementation.

Example: RandomGenerator, a toy measure which returns a random number between MinRange and MaxRange value at each call.

```
public class RandomGenerator extends DirectMeasure {

        @Override
        public List<IMeasurement> getMeasurement() throws Exception {
                List<IMeasurement> result = new ArrayList<>();

                // Retrive Platform Properties by her name
                int maxRange =  Integer.valueOf(getProperty("MaxRange"));
                int minRange = Integer.valueOf(getProperty("MinRange"));

                // Collect Measure
                Random gen = new Random();
                int value = gen.nextInt(maxRange - minRange) + minRange;

                // Create Measurement :  In this case, a simple IntegerMeasurement
                IntegerMeasurement measurement = new IntegerMeasurement();
                measurement.setValue(value);
                result.add(measurement);
                return result;
        }
}
```

Derived Measure implementation

A derived measure is used to define a combined measure which calculates new measurements using one or more measurements stored on the Measure platform. To define a derived measure, implement the IDerivedMeasure interface. This interface will be called by the Measure Platform to calculate the measurement.

```
public interface IDerivedMeasure {
        public List<IMeasurement> calculateMeasurement() throws Exception;
        public void addMeasureInput(String reference,String role, IMeasurement value);
        public Map<String,String> getProperties();
}
```

To implement a derived measure, please extend the **DerivedMeasure** class:

- **calculateMeasurement():** Calculate and return a list of measurements based on provided measurement imputs.
- **addMeasureInput():** Provide a way for the Measure Platform to communicate input measurements to the DerivedMeasure implementation.
- **getProperties():** Provide a way for the Measure Platform to communicate properties to the Derived Measure implementation.

A Derived Measure allows to combine measurement provided by other measures (Direct or Derived). Required inputs measure are defined on the MetaData.xml. These references are identified by a **measureRef** and a **role**.

- The **measureRef** is the id of the measure which can provide a measurement as input.
- The **role** is the role of the imput in current measurement. This role allows to identify several instances of the same measure of a Derived Measure.
- The **expirationDelay** property allows to filter as input the measures which has been calculated recently
- The **number** property allows to select the number of inputs of this type which will be communicated to the derived measure implementation by the platform.

```
<references expirationDelay="60000" measureRef="RandomGenerator" number="1">
  <role>RandomNumber A</role>
</references>
<references expirationDelay="60000" measureRef="RandomGenerator" number="1">
  <role>RandomNumber B</role>
</references>
```

Inputs are defined when an instance on the measure is deployed on the Measure Platform.

Example: RandomBinaryMeasure, a toy measure which returns the result of a binary operation between two RandomGenerator result

```
public class RandomBinaryMeasure extends DerivedMeasure {

        @Override
        public List<IMeasurement> calculateMeasurement() {
                Integer result = 0;

                // Retrive input Measurements by her Role
                List<IMeasurement> op1 = getMeasureInputByRole("RandomNumber A");
                List<IMeasurement> op2 = getMeasureInputByRole("RandomNumber B");

                // Calculate result
                if(op1.size() == 1 && op2.size() == 1){
                        String oper = "+";

                         // Retrive the operator as Property
                        oper = getProperty("Operation");

                        Integer val1 =   (Integer) op1.get(0).getValues().get("value");
                        Integer val2 =   (Integer) op2.get(0).getValues().get("value");

                        if(oper.equals("+")){
                                result = val1 + val2;
                        }else if(oper.equals("-")){
                                result = val1 - val2;
                        }else if(oper.equals("*")){
                                result = val1 * val2;
                        }else if(oper.equals("/")){
                                result = val1 / val2;
                        }
                }

                // Return result as  new IntegerMeasurement
                IntegerMeasurement measurement = new IntegerMeasurement();
                measurement.setValue(result);

                List<IMeasurement> measurements = new ArrayList<>();
                measurements.add(measurement);

                return measurements;
        }
}
```

Example: RandomSumMeasure, a toy measure which returns the sum of measurements provided by the RandomGenerator measure.

```
public class RandomSumImpl extends DerivedMeasure {
    @Override
    public List<IMeasurement> calculateMeasurement() throws Exception {
        Integer result = 0;
        for (IMeasurement operande : getMeasureInputByRole("RandomNumber")) {
            try {
                result = result + (Integer) operande.getValues().get("value");
            } catch (NumberFormatException e) {
                System.out.println("Non Numeric Operande");
            }
        }

        IntegerMeasurement measurement = new IntegerMeasurement();
        measurement.setValue(result);

        List<IMeasurement> measurements = new ArrayList<>();
        measurements.add(measurement);
        return measurements;
    }
}
```

Measurement

A Measurement is a data model used as input and output of SMM measure. A measurement has to extend the IMeasurement interface. A measurement is presented as set of Java elements which can be accessed via a Map. All values are accessed using a String identifier defined in MetaData.xml file.

```
public interface IMeasurement {
    public Map<String, Object> getValues();
    public String getLabel();
}
```

Predefined Measurements: The API provides some predefined measurements which can be used in the measure implementation

- IntegerMeasurement: allows to manipulate numbers in the measure implementation

```
int value = 10;
IntegerMeasurement measurement = new IntegerMeasurement();
measurement.setValue(value);
```

Custom Measurements Example: A Measure developer can define custom measurements to manage her own set of data.

The **SVNMeasurement** is used by a measure which collects COMMIT information provided by an SVN repository. It manages data related to the author of the commit, the message on the commit and the date of the commit.

```
public SVNMeasurement(String author,String message,Date postDate){
    super();
    this.valueMap.put("Author", author);
    this.valueMap.put("Message", message);
    this.valueMap.put("postDate",new Date(postDate.getTime()));
}
```

Measures Example

More than 200 measures are available on open source on the GitHub of the Measure project:

https://github.com/ITEA3-Measure/Measures

Measurement Application Development

NB: The implementation of Measurement Application support is not finalised, this documentation will be update.

Analysis Tool Development

The main objective of the analysis platform is to implement analytics algorithms, to correlate the different phases of software development and perform the tracking of metrics and their value. The platform also connects and assure interoperability among the tools and define actions for improvement and possible countermeasures.

Integration Mechanism of Analysis Tool into

In order to ensure the integration of various kind of analysis tool into the measure platform, the Analysis component provide an integration mechanism of external tool to register it into the measure platform, to access to measurement data, to received notification form the platform and finally to provide analysis results. This integration is based on a set of REST services used to manage communication between analysis Tools and the Platform.

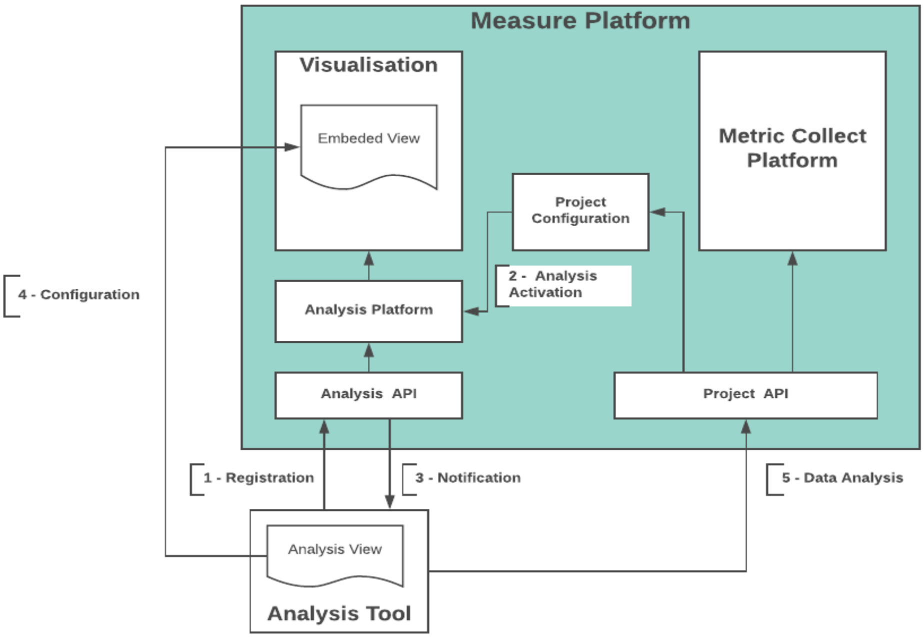


Figure 4 : Registration process of an external analysis tool into the platform

- **Registration**: At start-up of the Analysis Tool, it must register itself to the platform using the Registration service. This would allow the project to activate the analysis tools.

Registration Rest Service:

PUT	/api/analysis/register

Input Data (json)	`{`
	`  "configurationURL": "string",`
	`  "description": "string",`
	`  "name": "string"`
	`}`

- **Wait for Notifications**: The Analysis Tool must listen to notifications from the platform in order to know when a project requests the usage of the analysis tool. The notification (Alert) system is based on pooling system. The Analysis tool pool the platform periodically using the alert service to received notifications. The Platform send several kinds of notifications listed below:

Alert Type	Description	Properties
ANALYSIS_ENABLE	A Project sends an activation request for the Analysis Tool. It's not required for analysis tool to subscribe to this alert, the subscription is automatic.	• ANALYSISID: Id of the instance of analysis associated with this request on platform side
ANALYSIS_DESABLE	A Project indicate that the analysis service is not required anymore. It's not required for analysis tool to subscribe to this alert, the subscription is automatic.	• ANALYSISID: Id of the instance of analysis associated with this request on platform side.
MEASURE_ADDED	A new Measure is added the the project	• MEASUREID: Id of the Measure
MEASURE_REMOVED	A Measure is removed from the project	• MEASUREID: Id of the Measure
MEASURE_SCHEDULED	A Measure is not collected periodically for the project	• MEASUREID: Id of the Measure
MEASURE_UNSCHEDULED	A Measure is not collected anymore by the project	• MEASUREID: Id of the Measure

By default, all register project subscribe automatically to ANALYSIS_ENABLE and ANALYSIS_DESABLE Notifications.

Retrieve Platform Alerts REST Service: This service retrieves the alerts form the platform for a specific analysis tool

<table>
<tr><td><b>GET</b></td><td>/api/analysis/alert/list/{AnalysisToolName}</td></tr>
<tr><td><b>Parameter</b></td><td>AnalysisToolName : Name of the Analysis Tool (provided in registration service)</td></tr>
<tr><td><b>Output Data (json)</b></td><td><pre>
{
  "alerts": [
    {
      "alertType": "string",
      "projectId": 0,
      "properties": [
        {
          "property": "string",
          "value": "string"
        }
      ]
    }
  ],
  "from": "2018-03-13T12:16:33.164Z"
}
</pre></td></tr>
</table>

- **Configure Analysis**: When a project activates an analysis tool, the analysis tool must configure it for the project and provide URLs for the project-specific configuration page, the project main view and optionally the dashboard cards.

Configuration REST Service

Warning: The analysis configuration input data required a projectAnalysisId. This id is provided by the platform as properties of the ANALYSIS_ENABLE and ANALYSIS_DESABLE notification message.

PUT	/api/analysis/configure

Input Data (json)	{
	"cards": [
	{
	"cardUrl": "string",
	"label": "string",
	"preferedHeight": 0,
	"preferedWidth": 0
	}
	],
	"configurationUrl": "string",
	"projectAnalysisId": 0,
	"viewUrl": "string"
	}

- **Analyse the Project**: When configured, the analysis tool can start its analysis work for the specific project. In order to perform this work, the analysis tool can explore the project configuration using the various services provided by the Measure platform. It can also configure new Alerts to receive notifications when the project configuration has changed.

Alert Subscription REST Service: This service allows an analysis tool to subscribe to a new alert related to a specific project

PUT	PUT /api/analysis/alert/subscribe

Input Data (json)	{
	"analysisTool": "string",
	"eventType": "ANALYSIS_ENABLE",
	"projectId": 0,
	"properties": [
	{
	"property": "string",
	"value": "string"
	}
	]
	}

Alert Unsubscribe REST Service : This service allows the analysis tool to unsubscribe to an alert.

PUT	PUT /api/analysis/alert/unsubscribe

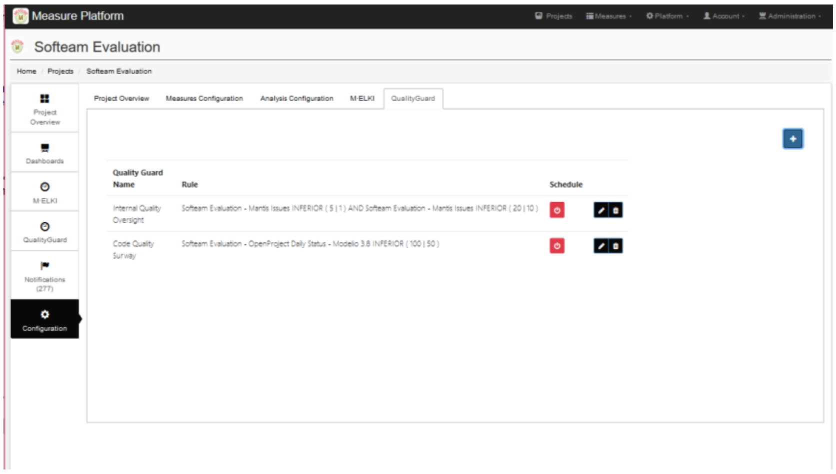
Input Data (json)	{
	"analysisTool": "string",
	"eventType": "ANALYSIS_ENABLE",
	"projectId": 0,
	"properties": [
	{
	"property": "string",
	"value": "string"
	}
	]
	}

User Interface integration using Embedded View

In order to integrate deeply the analysis tool to the Measure Platform, the analysis tools have to provide some web pages which will be embedded to the platform web application. Each of these views are defined on the platform side by a specific URL. For project specific views, this URL is different for each project. You will see below the list of view which can be provided by the analysis tool and embedded into the Measure Platform.

- **Global Configuration Page (optional):** If the analysis tool requires a way to provide some configuration interface which will be shared by all project, it can provide a global configuration web page.

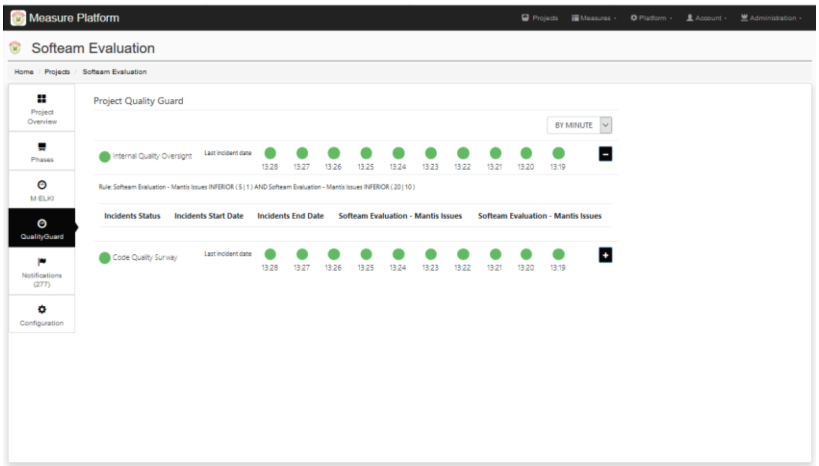
- **Project Specific Analysis Configuration page:** Configuration page which are specific for each project. This page is embedded into project configuration page and allow to configure the analysis service provided by the external analysis tool.



Quality Guard Name	Rule	Schedule
Internal Quality Oversight	Softeam Evaluation - Mantis Issues INFERIOR (5 | 1) AND Softeam Evaluation - Mantis Issues INFERIOR (20 | 10)	
Code Quality Surway	Softeam Evaluation - OpenProject Daily Status - Modelio 3.8 INFERIOR (100 | 50)	

Figure 5 : Analysis tool configuration page of Quality Guard Analysis tool

- **Analysis Tool Main View:** Main view of the analysis tool which are specific for each project. In this view, the analysis service.



Rule: Softeam Evaluation - Mantis Issues INFERIOR (5 | 1) AND Softeam Evaluation - Mantis Issues INFERIOR (20 | 10)

Incidents Status	Incidents Start Date	Incidents End Date	Softeam Evaluation - Mantis Issues	Softeam Evaluation - Mantis Issues

Figure 6 : Main view of the Quality Guard Analysis Tool

- **Dashboard Card:** Optional small view which can be integrated to projects dashboards in order to provide some key information to project managers related to the service provided by the analysis tool.

Platform Querying Services

The platform provides several other services which can be used by the analysis tools to retrieve platform and project configurations data, information related to measures and measurements and more.

The list of available services can be consulted via Swagger directly on deployed Measure platform. To access this specification, one must be connected as Administrator to the platform. The complete API specification is available on Administration > API menu

Some example of available HTTP services:

- GET /api/measure/findall : List all measures
- GET /api/measure/{id} : information related to a specific measure
- GET /api/measure-properties/{id} : List of scope properties associated with one measure
- GET /api/projects : List all projects
- GET /api/projects/{id} : Information related to a specific project
- GET /api/phases/byproject/{id} : Get phases of a specific project
- GET /api/phases/{id} : Information of a specific phase
- GET /api/measure-instances : List of all measure instances
- GET /api/measure-instances/{id} : Information of a specific measure instance
- GET /api/project-measure-instances/{id} : List of measure instances of a specified project
- GET /api/measure-instance/scheduling/execute/{id} : Execute a specific measure
- GET /api/measure-instance/scheduling/start/{id} : Activate scheduling of a specific measure
- GET /api/measure-instance/scheduling/stop/{id} : Deactivate scheduling of a specific measure

Run Measure Platform From Source

The measure platform is an open source product. The current section of the documentation present how to run the platform in developer mode from source.

Prerequisites

The Measure Platform can be executed both on Linux or Windows systems. For that, the platform requires the installation of: MySQL, Elasticsearch, Kibana and Java 1.8.

MySQL Installation

- Download MySQL Community Server 5.7 or above : *https://dev.mysql.com/downloads/mysql/*
- Install MySQL using the folloing instruction : *https://dev.mysql.com/doc/refman/5.7/en/installing.html*
- Create a new database named "measureplatform".

Elasticsearch Installation

- Downloade Elasticsearch 5.6 (as zip): *https://www.elastic.co/downloads/elasticsearch*
- Unzip the application in your tool directory.

Kibana Installation

- Downloade Kibana v 5.6 (as zip): *https://www.elastic.co/downloads/kibana*
- Unzip the application in your tool directory.

Java 1.8 Installation

- Download and install the jdk8 in youe environment: *http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html*

Eclipse IDE

- Download and install the last version of Eclipse IDE for Java EE Developers: *https://www.eclipse.org/downloads/eclipse-packages/*

Retrieve the MeasurePlatform Source Code

The Measure Platform source code is hosted on GitHub. To retrieve it, you can:

- Download it as zip file : *https://github.com/ITEA3-Measure/MeasurePlatform*
- Clone the Git repository
  - Install git: *https://git-scm.com/downloads*
  - Clone the repository: `git clone https://github.com/ITEA3-Measure/MeasurePlatform.git`

You can now import the Measure Platform as a new Maven project in Eclipse.

Start the Application in developers Mode

1. **Start MySQL**
2. **Start Elasticsearch :** ./elasticsearch-5.4.0/bin/elasticsearch
3. **Start Kibana:**./kibana-5.4.0/bin/kibana
4. **Start the Measure platform:**

- **From Eclipse IDE**: Select the "MeasurePlatformApp.java" file and Right Click > Run as > Java Application

Hawk Installation Guide

Monitoring Tool of MEASURE Platform

Hardware Requirements

- Recommended memory capacity: 8 Go
- OS: Hawk should work on Linux, Windows and Mac

Prerequisite

- Java must be installed. We recommend the 1.8 version.

Hawk Installation

- Download hawk-server-nogpl_xxxxxxxx-linux.gtk.x86_64.zip the latest version of "Hawk Server at the link:

  https://github.com/mondo-project/mondo-hawk/releases

- Extract the archive in your computer
- Provide configuration files for the instances to be executed on Hawk Server. These instances are described in XML files and must be added in the configuration directory at the root of hawk server before running hawk-server.

Configuration

- You must provide a configuration file for each instance you want to execute on Hawk server.
- The configuration files must be located in the configuration folder at the root of the hawk server project. The configuration files can have any name and must be in XML format.
- You can run many instances in Hawk Server at the same time and must add configuration files for each of these instances.
- In the configuration file we specify
  - The name of the Hawk instance
  - The backend (database engine) in whitch the instance will be running
    - `"org.hawk.orientdb.OrientDatabase"` for normal queries.
    - `"org.hawk.greycat.LevelDBGreycatDatabase"` for temporal queries
  - Delay for automatic updating (synchronization) of the model in the database. You can specify Min and Max values for Delay in milliseconds. If Max=0, it means the updating of the indexed model will be manual.
  - Plugins needed to execute the instance. We must specify the necessary plugins depending on the used backend (see the examples below).
  - The path of Modelio metamodel descriptor. The modelio metamodel can be found in Modelio model repository in "admin" folder.
  - The monitored repositories: they can be local (type="org.hawk.localfolder.LocalFolder") or SVN links (type="org.hawk.svn.SvnManager"). See the example below for more information.

- More details on configuration file can be found in this link: *https://github.com/mondo-project/mondo-hawk/wiki/File-based-instance-configuration*

- Here is an example of configuration file for an instance running on time awre backend in order to perform temporal queries.

Databio_instance.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<hawk backend="org.hawk.greycat.LevelDBGreycatDatabase" name="DataBio"
factory="org.hawk.timeaware.factory.TimeAwareHawkFactory">
    <delay max="0" min="0"/>
    <plugins>
        <plugin name="org.hawk.modelio.exml.listeners.ModelioGraphChangeListener"/>
        <plugin name="org.hawk.modelio.exml.metamodel.ModelioMetaModelResourceFactory"/>
        <plugin name="org.hawk.modelio.exml.model.ModelioModelResourceFactory"/>
        <plugin name="org.hawk.timeaware.graph.TimeAwareModelUpdater"/>
    </plugins>
    <metamodels>
        <metamodel location="/home/.../metamodel_descriptor.xml" uri=""/>
    </metamodels>
    <repositories>
        <repository frozen="false"
location="https://rd.constellation.modeliosoft.com/svn/.../trunk/model" pass="***"
type="org.hawk.svn.SvnManager" user="***"/>
    </repositories>
</hawk>
```

- Here is an example of configuration file for an instance with non-temporal queries

Project2_instance.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<hawk backend="org.hawk.orientdb.OrientDatabase" name="Measure">
    <delay max="0" min="0"/>
    <plugins>
        <plugin name="org.hawk.modelio.exml.listeners.ModelioGraphChangeListener"/>
        <plugin name="org.hawk.modelio.exml.metamodel.ModelioMetaModelResourceFactory"/>
        <plugin name="org.hawk.modelio.exml.model.ModelioModelResourceFactory"/>
        <plugin name="org.hawk.graph.updater.GraphModelUpdater"/>
    </plugins>
    <metamodels>
        <metamodel location="/home/.../metamodel_descriptor.xml" uri=""/>
    </metamodels>
    <repositories>
        <repository frozen="false" location="file:///home/.../design/model/"
type="org.hawk.localfolder.LocalFolder" user="" pass=""/>
        <repository frozen="false" location="file:///home/.../model2/model/"
type="org.hawk.localfolder.LocalFolder" user="" pass=""/>
    </repositories>
</hawk>
```

Running Hawk Server

- To run Hawk Server in Linux, execute in the terminal the command:

```
./run-server.sh
```

- Before executing any queries wait for the indexation process to be finished. A message indicating the completion of this task will be displayed:

```
Updated Hawk instance (success)
```

- If you haven't specified an automatic update in the configuration file, you can run the command:

```
./sync-server.sh
```

- You can stop Hawk server by typing "exit" in the console or by pressing: CTRL + C

Hawk User Guide

Monitoring Tool of MEASURE Platform

Hawk Server Overview

Hawk is a model indexing solution that can take models written with various technologies and turn them into graph-based databases for easier and faster querying. The indexing solution integrates Modelio modeling tool which contains different types of models such as UML, Archimate, BPMN, Analyst, etc.

- Hawk allows to perform fast queries on models (on modelio models in particular) using EOL language. EOL (*Epsilon Object Language*) is a programming language for creating, querying and modifying EMF models (Eclipse Modeling Framework).
- Hawk provide querying engines for executing EOL queries on models such as Modelio models.
- Here is an example of EOL query that return the total number of classes: “return Class.all.size;”. For more information see the following lik: *https://github.com/mondo-project/mondo-hawk/wiki/Example-queries-on-Modelio-based-UML-models*.
- Hawk is integrated in the monitoring and analysis tool “Measure Platform”. It allows developers to collect measures associated to the design phase in the development process.

Running Hawk Server

- Hawk Server is executed in command line interface by executing the script haw-server.sh in the root folder.
- After running the server, you must wait until the indexation of the models of the repositories is completed.
- After the indexation is finished, you will see the following message if the indexation is successful:

```
Updated Hawk instance DataBio (success). 13831 s 341 ms
```

- If the indexation is failed, you will see the following message

```
Updated Hawk instance DataBio (failed). 13831 s 341 ms
```

- The output of the server execution is displayed in console and stored in “hawk.log” file. It can be useful to see the errors that caused the updating failure.
- If you haven’t specified an automatic update in the configuration file, you can run the command ./sync-server.sh to synchronize the indexed model in graph database with the model of the updated local or SVN repository.

Hawk Server Commands

Console Client Commands

You can run commands in the console client or via web services:

- `hawkListBackends`: Lists the available Hawk backends
- `hawkListInstances`: Lists the available Hawk instances
- `hawkStartInstance <name>`: Starts the instance with the provided name

- `hawkStopInstance <name>`: Stops the instance with the provided name
- `hawkSyncInstance <name> [waitForSync:true|false]` : Requests an immediate sync on the instance with the provided name

See the following link for more details: *https://github.com/mondo-project/mondo-hawk/wiki/Console-client*

Web Services

Hawk allows to execute POST http queries with common URL “http://localhost:8080/thrift/hawk/json” and specific body messages to each command.

Here is a list of commands with the specific body message:

- List of instances

  [1,"listInstances",1,1,{}]

- Start an instance

  [1,"startInstance",1,1,{"1":{"str":"<name>"}}]

- Stop an instance

  [1,"stopInstance",1,1,{"1":{"str":"<name>"}}]

- Synchronize an instance and wait for response

  [1,"syncInstance",1,1,{"1":{"str":"<name>"},"2":{"tf":1}}]

- Synchronize an instance and don’t wait for response

  [1,"syncInstance",1,1,{"1":{"str":"<name>"},"2":{"tf":0}}]

- Queries
  - Non-temporal query on “Measure” instance

```
[1,"query",1,1,{"1":{"str":"Measure"},"2":{"str":"return
Class.all.size;"},"3":{"str":"org.hawk.epsilon.emc.EOLQueryEngine"},"4":{"rec":{
}}}]
```

  - Temporal query on “DataBio” instance

```
[1,"query",1,1,{"1":{"str":"DataBio"},"2":{"str":"return
Model.allInstancesNow.size;"},"3":{"str":"org.hawk.timeaware.queries.TimeAwareEO
LQueryEngine"},"4":{"rec":{}}}]
```

For more information about temporal queries see the following link : *https://github.com/mondo-project/mondo-hawk/wiki/Temporal-queries-in-Hawk* .

You can run these commands via a navigator extension like “Postman” or with Linux “curl” command.

Here is an example of a curl commands

- Synchronize “DataBio” instance

```
curl -s -H 'Content-Type: application/json' -d
'[1,"syncInstance",1,1,{"1":{"str":"DataBio"},"2":{"tf":1}}]'
http://localhost:8080/thrift/hawk/json
```

- List of instances

```
curl -s -H 'Content-Type: application/json' -d '[1,"listInstances",1,1,{}]'
http://localhost:8080/thrift/hawk/json
```

Querying Language

All the queries are written in the *Epsilon Object Language* which is an imperative programming language for creating, querying and modifying EMF models (Eclipse Modeling Framework).

Examples of EOL queries supported by Hawk :

- Returns the number of instances of "Class" in the index:

```
return Class.all.size;
```

- Temporal query : Returns the number of instances of "Class" of the last revision:

```
return Class.latest.all.size;
```

- Advanced example: loops, variables and custom operations

```
var counts = Sequence {};

var i = 0;var n = count(0);

while (n > 0) {  counts.add(Sequence {">" + i, n});  i = i + 1;  n = count(i);}

return counts;

operation count(n) {  return Class.all.select(c|c.ownedOperationCount > n).size;}
```

Associated Measures

A set of metrics dedicated to the Measure Platform has been developed specifically to make the link between the platform and the Hawk measurement tools.

Hawk allow us to perform queries over all Modelio models including UML, Archimate, Analyst, etc. Some of the measures are basic like the number of UML classes, methods, interfaces, etc. In addition to this, Hawk allows to do complex queries in short time such as “Class Complexity Index”, “Package Dependencies Ratio”, etc.

Measure	Type	Description
HawkMeasure	Generic Measure	Generic Measure which allow to execute a Hawk query provided as parameter of the measure.
NumberOfRequirement	Modelio Requirements	Total number of Requirement defined in the selected scope.
NumberOfTests	Modelio Requirements	Total number of Tests defined in the selected scope.
NumberOfBusinessRule	Modelio Requirements	Total number of Business Rule defined in the selected scope.
NumberOfGoals	Modelio Requirements	Total number of Goals defined in the selected scope.
NumberOfRisks	Modelio Requirements	Total number of Risks defined in the selected scope.
RequirementTracabilityToImplementationIndice	Modelio Requirements	The % of requirement of tracing an implementation model.

RequirementTracabilityToTestIndice	Modelio Requirements	The % of requirement of tracing a test model.
RequirementCoverageIndice	Modelio Requirements	The average number of requirements tracing an architecture model.
RequirementComplexityIndice	Modelio Requirements	The average number of sub requirements defined to refine an existing requirement.
RequirementsSatisfactionQualityIndice	Modelio Requirements	Percentage of requirements that have been satisfied.
SoftwareComponentDecomposition	Modelio UML	The number of software components identified in an application architecture.
ClassDependeciesRatio	Modelio UML	The average number of dependencies from a class.
PackageDependeciesRatio	Modelio UML	The average number of dependencies from a package.
ModelAbstractnessIndex	Modelio UML	The % of abstract classes (and interfaces) divided by the total number of types in a package.
ClassComplexityIndex	Modelio UML	Moy of direct subclasses of a class. A class implementing an interface counts as a direct child of that interface.
InterfaceByComponentRatio	Modelio UML	The average number of dependencies from a class.
NumberOfClasses	Modelio UML	Total number of classes in the selected scope
NumberOfFields	Modelio UML	Total number of fields defined in the selected scope.
NumberOfInterfaces	Modelio UML	Total number of interfaces in the selected scope.
NumberOfMethods	Modelio UML	Total number of methods defined in the selected scope.
NumberOfComponent	Modelio UML	Total number of Components defined in the selected scope.
NumberOfPackage	Modelio UML	Total number of Packages defined in the selected scope.
NumberOfUseCase	Modelio UML	Total number of UseCases defined in the selected scope.
NumberOfActors	Modelio UML	Total number of Actores defined in the selected scope.

Annexe F. MMT Installation Guide (Measurement Tool)

MMT Installation Guide

Monitoring Tool of MEASURE Platform

•••

Hardware Requirements

- **Hardware**

  MMT can run easily on a (personal) computer to process low network traffic, i. e., less than 10 Mbps (mega bit per second). The minimal requirement is that the computer has at least 2 cores of CPU, 2 GB of RAM and 4GB of free hard drive disk. The computer must possess also one NIC (Network Interface Cards) on which MMT captures network traffic. Higher network bandwidth requires stronger computer.

  To be able to exploit fully capacity of MMT to process very high network bandwidth, one need suitable hardware configurations. MMT can be deployed on a single server or split on separate servers. This section specifies the hardware requirements for the former. For those of the latter, please contact us.

- **Network Interface Cards:**

  MMT needs at least 2 NICs: one for capturing network traffic and another for administrating. If the probe is to be an active network element (i.e., receives, processes and re-transmits the communication packets) then at least 3 NICs are necessary.

  To achieve the best performance, the capturing NIC should be either Intel X710 or Intel X520 card. These are recommended because they support DPDK that will considerably improve the packet processing. For other hardware architectures, adaptations and tests would need to be performed.

- **CPU**

  For the best performance, the use of Intel Xeon class server system is recommended, such as Ivy Bridge, Haswell or newer. The larger the CPU cache, the better the performance are obtained.

- **RAM**

  We recommend using the fastest memory one can buy; and, having one DIMM per channel. One should avoid having 2 or more DIMMs per channel to make sure all memory channels are used to the fullest. It is more expensive to buy 2x16GB than 4x8GB, but with the later, memory access latency increases and the frequency and throughput decreases.

- **Hard Disk Drive**

  If MMT is to write meta data on a hard disk drive, it will do it using a database system (e.g., MongoDB) or files.

  We recommend using a Solid-State Drive with at least 20 GB of free space.

- **BIOS Settings**

  The following are some recommendations on BIOS settings. Different platforms will have different BIOS naming, so the following is mainly for reference:

  1. Before starting, consider resetting all BIOS settings to their default.
  2. Disable all power saving options such as: Power performance tuning, CPU P-State, CPU C3 Report and CPU C6 Report.
  3. Select Performance as the CPU Power and Performance policy.
  4. Disable Turbo Boost to ensure the performance scaling increases with the number of cores.

5. Set memory frequency to the highest available number, NOT auto.
6. Disable all virtualization options when you test the physical function of the NIC and turn on VT-d if VFIO is required.

Prerequisite

- Ubuntu TLS16.04: This Ubuntu version is recommended to run MMT as MMT has been carefully tested on it. Another Ubuntu version can also run MMT. To run MMT on the other Linux distros and Windows, please contact us. The example terminal commands used in this manual to prepare MMT running environment is suitable for a machine running Ubuntu LTS 16.04.
- MongoDB: MMT stores meta data using MongoDB database. For the use of other database systems please contact us. To install MongoDB follow steps described in the following tutorial https://docs.mongodb.com/manual/tutorial/install-mongodb- on-ubuntu/. To resume, for Ubuntu 16.04 LTS, one needs to do the following:

  1. Import the public key used by the package management system:

     ```
     sudo apt-key adv --key server hkp://keyserver.ubuntu.com:80 -- recv 0C49F3730359A14518585931BC711F9BA15703C6
     ```

  2. Create a list file for MongoDB and reload the local package database:

     ```
     echo "deb [ arch = amd64 , arm64 ] http://repo.mongodb .org/apt/ ubuntuxenial/ mongodb-org/3.4 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-3.4.list
     ```

     ```
     sudo apt - get update
     ```

  3. Install the MongoDB package:

     ```
     sudo apt - get install -y mongodb - org
     ```

  4. Start MongoDB:

     ```
     sudo service mongod start
     ```

  5. Verify that the mongod process has started successfully by checking that the log file /var/log/mongodb/mongod.log contains the line:

     ```
     [ initandlisten ] waiting for connections on port 27017
     ```

NodeJS: The MMT-Operator runs using NodeJS. To install NodeJS, please follow the steps described in https://nodejs.org/en/download/package-manager/. To resume, for Ubuntu 16.04 LTS, one needs to do the following:

1. Create a list file for NodeJS and reload the local package database:

   ```
   curl - sL https://deb.nodesource.com/setup_ 6.x  | sudo -E bash -
   ```

2. Install NodeJS:

   ```
   sudo apt - get install -y nodejs
   ```

3. Verify that NodeJS was successfully installed:

   ```
   node -v
   ```

- Option: One might need to install the REDIS or KAFKA message bus server if the MMT-Operator is to receive meta data from the MMT-Probe via this type of publish and subscribe systems. MMT supports the use of the Data Plane Development Kit (DPDK) for high performance capturing of network traffic. To be able to use MMT with other capturing libraries, e. g., netmap, native socket, etc., please contact us. Before running MMT with DPDK, we need to set the “huge pages” option. The commands below will reserve 16 GB for MMT-Probe. Note that this configuration needs to be done after each reboot. For doing it permanently one needs to modify the arguments of the kernel command line.

```
#Each huge page has 2MB so we need 8192 pages.

 #Forasingle-nodesystem:

echo8192 > /sys/kernel/mm/hugepages/hugepages -2048kB/nr_hugepages

#ForaNUMAsystem:

echo4096 > /sys/devices/system/node/node0/hugepages/hugepages-2048kB/nr_hugepages

echo4096   >   /sys/devices/system/node/node1/hugepages/hugepages-2048kB/nr_hugepages
```

MMT Installation

- Installing MMT:

To install MMT do the following steps:

1. Install MMT:

```
#Install MMT by order:

#sudo dpkg -i mmt-dpi-<version>.deb

#sudo dpkg -i mmt-security-<version>.deb

#sudo dpkg -i mmt-probe-<version>.deb

#sudo dpkg -i mmt-operator-<version>.deb

#For example:

sudo   dpkg   -i   mmt-dpi_1.6.7.3_Linux_x86_64.deb

sudo   dpkg   -i   mmt-security_1.1.2_7d30208_Linux_x86_64.deb

sudo   dpkg   -i   mmt-probe_1.2.0_7a73b6b_Linux_x86_64.deb

sudo   dpkg   -i   mmt-operator_1.6.2_aa3d383.deb

#Update library path

sudo ldconfig
```

2. Verify that MMT was correctly installed:

```
ls -R  /opt/mmt

#There must be 6 folders: dpi, examples, operator, plugins, probe, security
```

3. Refer to the next section to configure MMT as needed. The configuration files of MMT-Operator and MMT-Probe are located at /opt/mmt/ operator/config.json and /opt/mmt/probe/conf/online.conf respectively.

For example, to use 2 threads of MMT-Probe to capture traffic at the first NIC binded to DPDK, one should update in /opt/mmt/ probe/conf/online.conf the following parameters:

```
# Enable output results to files
file - output {
enable = 1
...
}#
Using 2 threads of MMT - Probe
thread -nb = 2
# Enable MMT - Security
security2 {
enable = 1
...
}
session - report report_session {
enable = 1
...
}
condition_report report_web {
enable = 1
...
}
condition_report report_ssl {
enable = 1
...
}

enable -proto - without - session - stat = 1
```

- Start MMT:

Execute the following commands to launch the MMT-Operator and the MMT-Probe.

```
# Start MMT - Operator
sudo mmt - operator

# Start MMT - Probe to monitor traffic on NIC eth0

# run this command on a new terminal

sudo mmt - probe -c / opt / mmt / probe / conf / online . conf -i eth0
# Run MMT with DPDK for high bandwidth :
# sudo mmt - probe -c <CORE_MASK > -- -c < configuration file >
# For instance , to use 3 cores (2 threads + main ) and the configuration file online . conf to
capture traffic on the first port :
# sudo mmt - probe -c 0xB -- -c / opt / mmt / probe / conf / online . conf -i 0
```

For further information about the use of DPDK and CORE_MASK, please refer to *http://dpdk.org/doc/guides/linux_gsg/build_sample_apps.html#logical-core-use-by-applications*

- View MMT graphical:

Open a Web browser, then go to to address http://IP_of_server, in which, IP_of_server is IP of administrator NIC of the server.

Log in to MMT-Operator Web interface on the browser using admin/mmt2nm as username/password. We recommend changing this default password once logged in.

- Uninstalling MMT:

Execute the following commands to completely remove MMT:

```
sudo dpkg -r mmt - operator mmt - probe mmt - security mmt -sdk
# Do the following for completely removing all of MMT 's
execution logs :

                         sudo rm -rf / opt/ mmt
```

MMT-Probe

MMT-Probe requires a configuration file for setting different options. MMT-Probe can operate in two modes PCAP and DPDK. The PCAP mode uses libpcap library, whereas, the DPDK mode uses dpdk library, for packet capturing. There are two input-mode available for PCAP, i.e. "offline" and "online", while in DPDK, only "online" input-mode is available. The "offline" input-mode uses PCAP trace file while, the "online" input-mode uses network interface to capture the packets.

The different available configuration options are listed in the following:

- *Probe identifier:* The probe-id-number indicates the identifier of the MMT-Probe. All output report formats contain this identifier.
  - probe-id-number = 1
- *License file:* The *license_file_path* indicates the path where the license information is present.
  - license_file_path = "/opt/mmt/probe/bin/license.key "
- *Thread number:* The thread-nb indicates the number of threads the MMT-Probe will use for processing packets. It must be a positive number.
  - thread-nb = 1
- *Thread queue length:* The thread-queue indicates the maximum number of packets that can be queued for processing by a single thread. For an unlimited queue size, set the value to 0. This value is used only if PCAP mode is used. Other modes are DPDK and offline analysis of PCAP files.
  - thread-queue = 0
- *Thread queue data volume:* The thread-data indicates the maximum amount of data that can be queued for processing by a single thread. For an unlimited queue volume, set the value to 0. This value is used only if PCAP mode is used.
  - thread-data = 0
- *Packet snaplen:* The option snap-len allows indicating the maximum packet size to cap- ture. A value of 0 will set the default value (65535). This value is used only if PCAP mode is used.
  - snap-len = 0
- *Log file:* The logfile indicates the location where the log messages will be writ- ten.
  - logfile = "/opt/mmt/probe/log/online/log.data
- *Session timeout:* The session-timeout configuration bloc specifies the session timeout in seconds for different types of applications. The default-session-timeout, long-session-timeout, short-session-timeout, and live-session-timeout indicate, respectively, the session timeout period for default session, long lived session, short lived session and live session.

```
session - timeout

{

        # 0 means default value = 60 seconds. For default session:
        default - session - tim eout  =  40
        #0 means default value = 600 seconds. This is reasonable for Web and SSL connections especially when
        long polling is used. Usually applications have a long polling period of about 3~5 minutes.
        long-session-timeout  =  0
        # 0 means default value = 15 seconds. For short live sessions:
```

```
short-session- timeout   =   0
#0 means default value = 1500 seconds. For persistent connections like messaging applications and so
on:
live-session-timeout        =   0

}
```

MMT-Probe can capture the packets from a NIC or a trace file (PCAP file). Hence, input-mode and input-source needs to be specified ac- cording to the requirements:

- *Online:* This allows near real-time analysis of network traffic. In PCAP mode, the NIC's network interface name needs to be identified, whereas in DPDK mode, the interface port number needs to be identified using the input-source option.

```
input - mode = " online "
# For PCAP it is interface name
input - source = " eth0 "
# For DPDK it is interface port number
input - source = "0"
```

- *Offline:* This allows analysis of a PCAP trace file. For offline mode, the input-source identifies the name of the trace file. The offline analysis is only available for the PCAP mode.

```
input - mode = " offline "
input - source = " wow . PCAP "
```

- *Output file:* The file-output configuration bloc indicates the file where the MMT- Probe will write the reports. The data-file indicates the name of the file, location specifies the folder, retain-files indicates how many files to retain at one time, and sample_report indicates the output file sampling rate (i.e., every x seconds). If sampled_report is enabled, then a separate output file is created every x seconds (sampling time), otherwise only a single output file will be created. The sampling time is determined by file-output-period.

```
# Indicates where the traffic reports will be dumped ( CSV file )
file - output
{
# Set to zero to disable file output ( CSV ), 1 to enable
enable = 0:
# File name where the reports will be dumped:
data - file = " dataoutput . csv "
# Location where files are written:
location = "/ opt / mmt / probe / result / report / online /"
# Retains the last x sampled files , set to 0 to retain all files ( note that the value of retain - files must
be greater than the value of thread_nb + 1):
retain - files = 10
# Set to 1 if one needs a sampled file every x seconds or 0 if one needs a single file:
sampled_report = 1
 }
```

- *Output to REDIS:* The redis-output configuration bloc indicates that MMT-Probe should

use REDIS to publish the output. The hostname and the port indicate the address (IP-address) and the port of the redis-server respectively. The redis-server needs to be started beforehand.

```
# Indicates REDIS output:
redis - output
{
# Set to zero to disable publishing to REDIS, 1 to enable publishing to REDIS:
enabled = 0
# Hostname of REDIS server:
hostname = " localhost "
# Port number of REDIS server:
port = 6379
}
```

- *Output to KAFKA:* The kafka-output configuration bloc indicates that MMT-Probe should use KAFKA to publish the output. The hostname and the port indicate the address (IP-address) and the port of the kafka-server respectively. The kafka-server needs to be started beforehand.

```
# Indicates KAFKA output :
kafka - output
{
# Set to zero to disable publishing to KAFKA, 1 to enable publishing to KAFKA:
enabled = 0
# Hostname of KAFKA server:
hostname = " localhost "
# Port number of KAFKA server:
port = 9092
}
```

- *Output to socket:* The socket configuration bloc indicates that the MMT-Probe should send the output reports using sockets. The socket domain can be UNIX, Internet or both when set to 0, 1 or 2 respectively. The socket descriptor is required for the UNIX domain to specify the location of a socket file descriptor. The server address and port are required for the Internet domain to specify the address and the port of a socket. The one socket server is set to 1 to indicates that only one port is available for communication, i.e., all the threads will send reports to the same port. If it is set to 0, then multiple socket will be created, i.e., every thread will have a unique port number to send the reports and the port option should contain a number of ports equal to the number of threads).

```
# Socket configurations
socket
{
# Set to 1 to enable , 0 to disable :
enable = 0
# 0 for Unix domain , 1 for Internet domain , and 2 for both :
domain = 0
# Required for UNIX domain . Folder location where socket file descriptor is created :
socket - descriptor = "/ opt / mmt / probe / bin/"
# Required for Internet domain . If one - socket - server is set to 0 then the number of port addresses
should be equal to the number of threads ( thread_nb ). If one - socket - server is set to 1, the
number of port address should be only one :
port = {5000}
# Required for Internet domain . IP address of ip_host 1, ip_host 2...
server - address = {" localhost "} # If set to 0 the server contains multiple sockets to
receive the reports . If set to 1 only one socket will receive the reports :
one - socket - server = 1
}
```

- *Statistics reporting period:* The stats-period indicates that MMT-Probe should report statistics every x seconds. A report will be sent every stats-period number of seconds.

```
# Indicates the periodicity for reports :
stats - period = 5 # period in seconds
```

- *File output period:* The file-output-period indicates the CSV (Common Separated Values) file output period in seconds. A new file will be generated every file-output-period number of seconds.

```
# Indicates the periodicity for reporting output file :
file - output - period = 5 # sampled reporting
```

- *Reports per message(socket):* The num-of-report-per-msg indicates the number of reports included in one socket transaction (message). A socket message will contain num-of-report-per-msg messages in one transaction.

```
# Indicates the number of report per message ( sockets ).
Default is 1. This option is only available for MMT - Security using sockets :
```

```
num -of - report -per -msg = 5
```

- *Cache-size for reporting:* The cache-size-for-reporting indicates the maximum number of reports can be cached before flushing to a file.

```
# A value of 0 means that MMT will decide how many packets to cache ( default = 300000) :
cache -size -for - reporting = 300000
```

- *Protocols without sessions:* The enable-proto-without-session-stat enables or disables the reporting of protocols that do not belong to any session (e.g., ARP).

```
# A value of 1 will enable and 0 will disable the protocol statics :
enable -proto - without - session - stat = 0
```

- *Protocols with sessions:* The session-report configuration bloc enables or disables the reporting of protocols that belong to a session. The output reports can be reported to a file, redis, kafka or a combination of these channels. This is specified in the output-channel field. If nothing is specified, the default channel used will be a file.

```
# indicates session based reporting
session - report report_session
{
# Set to 1 for reporting session reports , 0 to disable :
enable = 0
# Reports are sent to the output channel . The default value is a file .
# More than one output channel is possible . In this case , the value should be a set of comma -
separated values ; for example : {redis , kafka , file }.
# Reports are sent to output channels only when the global parameters are enabled (e.g., file -
output . enable = 1, redis - output . enable = 1, kafka - output . enable = 1).
output - channel = {}
}
```

- *IP fragmentation:* The enable-IP-fragmentation-report configures the reporting of IP fragmentation information.

```
# Set to 1 to enable , 0 to disable
enable -IP - fragmentation - report = 0
```

- *Micro flows:* The micro-flows configuration bloc specifies the criteria to use to determine if a flow is a micro flow. A single micro flow statistic will not be reported separately, statistics from several micro flows will be aggregated and reported together. Aggregating micro flows statistics reduces the number of reports; however, one will lose fine grained information about each flow. The include-packet-count indicates the packet count threshold to be used to determine that a flow is a micro flow. The include-byte-count indicates the data volume threshold in KB to be used to determine if a flow is a micro flow. The report-packet-count indicates the packet count threshold to be used for creating micro flows aggregated statistics reports. The report-byte-count indicates the data volume threshold in KB to be used for creating micro flows aggregated statistics reports. The report-flow-count indicates the flows count threshold to be used for creating micro flows aggregated statistics re- ports. The output reports can be reported to a file, redis, kafka or a combination of these channels. This is specified in the output-channel field. If nothing is specified, the default channel used will be a file.

```
micro - flows
{
# Set to 1 to enable , 0 to disable :
enable = 0
# Packets count threshold to consider a flow as a micro flow :
include - packet - count = 0
# Data volume threshold in KB to consider a flow as a micro flow :
include -byte - count = 0
# Packets count threshold to report micro flows aggregated statistics :
report - packet - count = 10000
# Data volume threshold in KB to report micro flows aggregated statistics :
```

```
report -byte - count = 5000
# Flows count threshold to report micro flows aggregated statistics :
report -flow - count = 10
# Reports are sent to the output channel . The default value is a file .
# More than one output channel is possible . In this case , the value should be a set of comma -
separated values ;
for example : {redis , kafka , file }.
# Reports are sent to output channels only when the global parameters are enabled (e.g., file -
output .
enable = 1, redis - output . enable = 1, kafka - output .
enable = 1).
output - channel = {redis , kafka , file }

}
```

- *Data output:* The data-output is intended for defining the criteria to be used regarding the reporting of specific meta-data. For the time being, it only includes criteria to indicate when to include user agent parsing. The include-user-agent indicates the threshold in terms of data volume for parsing the user agent in Web traffic. This configuration bloc will be extended in the future when new reporting needs arise.

```
data - output
# Indicates the threshold in terms of data volume for parsing the user agent in Web traffic .
# The value is in kiloBytes (kB). If the value is zero , this indicates that the parsing of the user agent
should be done.
# To disable the user agent parsing , set the threshold to a negative value ( -1).
include -user - agent = 32
}
```

- *Custom event-based reports:* The event_report configuration bloc allows to indicate what protocol attributes should be reported when a certain event is triggered. The event indicates the condition that should be satisfied in order to report the attributes; and, the attributes indicate the application (protocol) attributes that need to be reported when an event occurs. Events and attributes should be in "protocol.attribute" format. There can be multiple event_report configuration blocs. Each event_report bloc is uniquely identified by its name; for instance, event_report report1. The output reports can be reported to a file, redis, kafka or a combination of these channels, as specified in the output-channel field. However, if nothing is specified, the default value is a file.

```
event_report report1
{
# Set to 1 to enable , 0 to disable :
enable = 0
# Indicates the event :
event = " rtp . burst_loss "
# Indicates the list of attributes that are reported when a event is triggered :
attributes = {" rtp . timestamp ", "rtp . jitter "}
# Reports are sent to the output channel . The default value is a file .
# More than one output channel is possible . In this case , the value should be a set of comma -
separated values ;
for example : {redis , kafka , file }.
# Reports are sent to output channels only when the global parameters are enabled (e.g., file -
output .
enable = 1, redis - output . enable = 1, kafka - output .
enable = 1). output - channel = {}

}
```

- *Custom condition-based* reports: The condition_report bloc configures the reports on different application (WEB, FTP, RTP, SSL, etc) that are a part of the session report. These reports are generated and sent to the output channels only when session-report configuration bloc is enabled. The condition_report.report_web, condition_report.report_ftp, condition_report.report_rtp, and condition_report.report_ssl configuration blocs specify the reports to be generated for HTTP (web), FTP, RTP and SSL applications, respectively. The condition indicates the condition that a flow should satisfy (for example, if a flow corresponds to WEB traffic). The attributes and handlers indicate the application (protocol) attributes and their corresponding handlers that need to be registered in order to generate the reports. The attributes field should be in the "protocol.attribute" format.

WEB report

```
 condition_report report_web

{
# Reports are sent to output channels only when session -report . enable = 1.
# Set to 1 to enable , 0 to disable :
enable = 1
# Indicates the condition to be satisfied .
condition = "WEB "
# Indicates the list of attributes for reporting .
attributes = {" http . uri"," http . method "," http . response ","
http . content_type "," http . host ", " http . user_agent ","
http . referer "," http . xcdn_seen "," http . content_len "}
# Indicates the list of handlers corresponding to the above attributes .
handlers = {" uri_handle "," http_method_handle ","
http_response_handle "," mime_handle "," host_handle ","
useragent_handle "," referer_handle "," xcdn_seen_handle ",
" content_len_handle "}
}
```

File Transfer Protocol (FTP)

```
 condition_report report_ftp

{
# Reports are sent to output the output channels only when session - report . enable = 1.
# Set to 1 to enable , 0 to disable :
enable = 1
# Indicates the condition to be satisfied :
condition = "FTP "
# Indicates the list of attributes for reporting :
attributes = {" ftp . data_direction "," ftp. p_payload "," ftp.
packet_type "," ftp . packet_payload_len ", "ftp . data_type "
, " ftp . file_name "," ftp . packet_request ", " ftp.
packet_request_parameter ","ftp . packet_response_code ","
ftp . packet_reponse_value "," ftp . transfer_type "," ftp .
ftp_session_mode ","ftp . data_direction "," ftp .
file_last_modified "," ftp. session_connection_type "," ftp
. user_name "," ftp . password "," ftp . last_command "," ftp.
last_response_code "," ftp. file_size "}
# Indicates the list of handlers corresponding to the above attributes :
handlers = {" ftp_data_direction_handle "," NULL "," NULL ","
NULL "," NULL "," ftp_file_name_handle ","
ftp_packet_request_handle "," NULL "," NULL ","
ftp_response_value_handle "," NULL "," NULL "," NULL "," NULL "
," ftp_session_connection_type_handle ","
ftp_user_name_handle "," ftp_password_handle "," NULL ", "
ftp_response_code_handle "," ftp_file_size_handle "}
}
```

Real-time Transport Protocol

```
 condition_report report_rtp

{
# Reports are sent to the output channels only when session - report . enable = 1.
# Set to 1 to enable , 0 to disable :
enable = 0
# Indicates the condition to be satisfied :
condition = "RTP "
# Indicates the list of attributes for reporting :
attributes = {" rtp . version ","rtp . jitter "," rtp . loss ","rtp .
order_err "," rtp . burst_loss "}
# Indicates the list of handlers corresponding to the above attributes :
handlers = {" rtp_version_handle "," rtp_jitter_handle ","
rtp_loss_handle "," rtp_order_error_handle ","
rtp_burst_loss_handle "}
}
```

Secure Sockets Layer (SSL)

```
 condition_report report_ssl

{
# Reports are sent to output channels only when session -report . enable = 1.
# Set to 1 to enable , 0 to disable :
enable = 0
# Indicates the condition to be satisfied :
condition = "SSL "
# Indicates the list of attributes for reporting :
attributes = {" ssl . server_name "}
# Indicates the list of handlers corresponding to the above attributes :
handlers = {" ssl_server_name_handle "}
}
```

- *HTTP reconstruction:* The condition_report.reconstruct_http configuration bloc indicates that the MMT-Probe should reconstruct the HTTP data. To enable the reconstruction, the session-report configuration bloc should also be enabled.

  The location indicates the location where the files are reconstructed; condition indicates the condition that a flow should satisfy for reconstruction; and, attributes and handlers indicate the application (protocol) attributes and their corresponding handlers that need to be registered in order to reconstruct the data. The attributes field should be in the "protocol.attribute" format. Note that currently, if enabled, only HTTP files are reconstructed, and no HTTP reports are generated.

```
# Set reconstruct_http . enable = 1 and session - report . enable = 1 for http_reconstruction .
condition_report reconstruct_http
{
# Set to 1 for HTTP reconstruction , 0 to disable .
enable = 0
# Indicates the condition .
condition = "HTTP - RECONSTRUCT "
# Location where the files are reconstructed .
location = ""
# Indicates the list of attributes .
attributes = {" tcp . payload_len ", " tcp . p_payload ", " http .
msg_start ", " http . header ", " http . headers_end ", " http .
data ", " http . msg_end ", " http . method ", " http . response ",
" http . content_type "," http . uri "}
# Indicates the list of handlers corresponding to the above attributes .
handlers = {" NULL ", " NULL ", " http_message_start_handle ",
" http_generic_header_handle ", " http_headers_end_handle
"," http_data_handle ", " http_message_end_handle ", " NULL
", " NULL ", " NULL ", " NULL ", " NULL "}

}
```

- *Radius:* The radius-output bloc configures the reports for the RADIUS proto- col. The include-msg indicates the message one needs to report, and include-condition indicates the condition to be met in order to report. The output reports can be reported to a file, redis, kafka or a combination of these channels, as specified in the output-channel field. However, if nothing is specified, the default value is a file.

```
# Indicates the strategy for RADIUS reporting
```

```
radius - output
{
# Set to 1 to enable , 0 to disable :
enable = 0
# Indicates the message one needs to report .
# Set to 0 to report all messages .
# Set to a number greater than 0 to indicate the message to report (1 for message , 2 for
conditions ):
include - msg = 0
# Indicates the condition to be met in order to report a message .
# Condition set to 1 indicates that the report should be generated iff the IP to MSISDN
mapping is present .
This is the only supported condition for this version .
Condition set to 0 to eliminate the condition .
include - condition = 0
# Reports are sent to the output channel . The default value is a file .
# More than one output - channel is possible . In this case , the value should be a set of
comma - separated values ;
for example : {redis , kafka , file }.
# Reports are sent to the output channels only when the global parameters are enabled (file
- output . enable = 1, redis - output . enable = 1, kafka - output . enable = 1).
output - channel = {}

}
```

- *<u>Security:</u>* The security output configuration blocs allow specifying the security re- porting. There are three different modules for security which are listed below:

  Module 1: The security1 is for low bandwidth single threaded operation. The results-dir indicates where the security reports will be written; and, the properties-file specifies the location of an XML file where the security rules have been specified. This module is only available for the PCAP mode.

```
Security1
{
# Set to 1 to perform security analysis , 0 to disable it:
enable = 0
# Folder where the detected breaches will be reported :
results - dir = "/ opt/ mmt/ probe / result / security / online /"
# File containing security properties and rules :
properties - file = " test_files / properties_acdc . xml "
}
```

- **Module 2**: The security2 module is for high bandwidth multi-threaded operation. The thread-nb indicates the number of security threads for each probe thread; the exclude-rules indicates the range of rules to be excluded from the verification; and, the rules-mask indicates the range of rules to be verified by each security thread. The generated reports can be written to a file, redis, kafka or a combination of these channels, as specified in the output-channel field. However, if nothing is specified, the default value is a file.

```
security2
{
# Set to 1 enable , 0 to disable it:
enable = 0
# The number of security threads per one probe thread , e.g., if we have 16 probe threads
and thread -nb = x, then x *16 security threads will be used .
# If set to zero this means that the security analysis will be done in the same threads used
by the probe .
thread -nb = 1
# Range of rules to be excluded from the verification :
exclude - rules = ""
# Mapping of rules to the security threads :
# Format : rules - mask = ( thread - index :rule - range );
# Thread - index = a number greater than 0;
# rule - range = number greater than 0, or a range of numbers greater than 0.
# Example : If we have thread -nb = 3 and "(1:1 ,2 ,4 -6) (2:3) ",
# thread 1 verifies rules 1 ,2 ,4 ,5 ,6;
# thread 2 verifies only rule 3; and
# thread 3 verifies the rest
rules - mask = ""
# Reports are sent to the output channels . The default value is a file .
# More than one output channel is possible . In this case , the value should be a set of
comma - separated values ; for example : {redis , kafka , file }.
# Reports are sent to output channels only when the global parameters are enable (file -
output . enable = 1, redis - output . enable = 1, kafka - output . enable = 1)
output - channel = {}
}
```

- **Module 3**: The security-report is for high bandwidth multi-threaded operation. The MMT-Probe and the MMT-Security exist separately for this mode (i.e., two separate executables). The communication between the MMT-Probe and the MMT-Security modules is done through sockets. The socket should be configured beforehand. The attributes indicate the list of protocol attributes that need to be reported.

```
security - report localhost {
# Set to 1 to enable and 0 to disable:
enable = 0
# Indicates the list of attributes that are reported when an event is triggered :
attributes = { "ip. dst ","ip.src "," tcp . dest_port "," tcp .
flags "," arp . ar_op "," arp . ar_sha ","arp . ar_sip "," ethernet
. dst "," ethernet .src "," tcp . seq_nb "," tcp. ack_nb ","ip.
mf_flag ","ip. frag_offset ","ip. tot_len "," meta .
packet_len ","tcp . payload_len "," tcp . src_port ","ip.
identification "," tcp . urg "," tcp. fin"," tcp . psh " }
}
```

- *The behavior* : The behavior configuration bloc indicates that the MMT-Probe should produce behavior reports. The location indicates the folder where the reports will be written.

```
{
# Set to 1 to enable , 0 to disable :
enable = 0
# Folder to write the output :
location = "/ opt / mmt / probe / result / behaviour / online /"
}
```

- *FTP reconstruction:* The reconstruct-ftp output configuration bloc indicates that the MMT-Probe should reconstruct FTP files and generate FTP reports. To enable the reconstruction, the session-report configuration bloc and session-report.ftp_report.enable should also be enabled. The location indicates where the FTP file will be stored. The reports can be written to a file, redis, kafka or a combination of these channels, as specified in the output-channel field. However, if nothing is specified, the default value is a file. The FTP reconstruction is only available for the single threaded operation.

```
# Indicates that FTP data reconstruction should be done .
# To enable the reconstruction , also enable the options session - report . enable and
session - report . ftp_report . enable

reconstruct - ftp
{
# Set to 1 to enable , 0 to disable it: enable = 0
# Indicates the folder where the output file is created :
location = "/ opt / mmt /"
# Reports are sent to the output channel . The default value is a file .
# More than one output channel is possible . In this case , the value should be a set of
comma - separated values ; for example : {redis , kafka , file }.
# Reports are sent to output channels only when the global parameters are enabled (e.g.,
file - output . enable = 1, redis - output . enable = 1, kafka - output . enable = 1).
output - channel = {}
    }
```

- *CPU and memory usage:* The cpu-mem-usage configuration bloc defines the CPU and memory usage reports to be generated. The frequency indicates the time-interval for reporting. The output reports can be reported to a file, redis, kafka or a combination of these channels, as specified in the output-channel field. However, if nothing is specified, the default value is a file.

```
# CPU and memory usage monitor

cpu -mem - usage
{
# Set to 1 to perform cpu - mem reporting , 0 to disable it:
enable = 0
# Time - interval for reporting
frequency = 5
# Reports are sent to the output channel . The default value is a file .
# More than one output channel is possible . In this case , the value should be a set of
comma - separated values ; for example : {redis , kafka , file }.
# Reports are sent to output channels only when the global parameters are enabled (e.g.,
file - output . enable = 1, redis - output . enable = 1, kafka - output . enable = 1):
output - channel = {}
}
```

- *Security multi-session:* The security-report-multisession bloc configures the report for security multi-session. The attributes indicate the list of protocol attributes to be reported. The output reports can be reported to a file, redis, kafka or a combination of these channels, as specified in the output-channel field. However, if nothing is specified, the default value is a file.

```
# This report is for security multi - session security :
security - report - multisession remote {
# Set to 1 to perform multi - session reporting , 0 to disable it:
enable = 0
# Indicates the list of attributes that are reported :
attributes = { "nfs . file_name "}
# Reports are sent to the output channel . The default value is a file .
# More than one output channel is possible . In this case , the value should be a set of
comma - separated values ; for example : {redis , kafka , file }.
# Reports are sent to output channels only when the global parameters are enabled (e.g.,
file - output . enable = 1, redis - output . enable = 1, kafka - output . enable = 1).
output - channel = {}
}
```

MMT-Operator

The configuration for the MMT-Operator is defined in the /opt/mmt/ operator/config.json file. Its structure is as follows:

```
{
" database_server ": {
" host ": " localhost ",
" port ": 27017
},
" redis_server ": {
" host ": " localhost ",
" port ": 6379
},
" kafka_server ": {
" host ": " localhost ",
" port ": 2181
},
" data_folder ": "/ opt / mmt / probe / result / report / online /",
" delete_data ": true ,
" input_mode ": " kafka ",
" probe_analysis_mode ": " online ",
" probe_stats_period ": 5,
" local_network ": [
{
"ip": "192.168.0.0" ,
" mask ": "255.255.0.0"
},
{
"ip": "172.16.0.0" ,
" mask ": "255.240.0.0"
},{
"ip": "10.0.0.0" ,
" mask ": "255.0.0.0"
},{
"ip": " fe80 ::" ,
" mask ": "8"
},{
"ip": "0.0.0.0" ,
" mask ": "255.255.255.255"
},{
"ip": "255.255.255.255" ,
" mask ": "255.255.255.255"
},{
"ip": "127.0.0.1" ,
" mask ": "255.255.255.255"
}
],
" buffer ": {
" max_length_size ": 50000 ,
" max_interval ": 30
},
" micro_flow ":{
" packet ": 1,
" byte " : 64
},
" retain_detail_report_period ": 604800 ,
" port_number ": 80,
" log_folder ": "/ tmp /",
" is_in_debug_mode ": true

    }
```

The content of config.json can be divided into the following groups:

- *Input:* This group specifies how MMT-Operator can receive meta-data generated by the MMT-Probe. The value of input_mode can be one of the following:
  - "file": MMT-Operator will read meta-data from a file in the folder data_folder. After reading a file, MMT-Operator can delete the file and its semaphore depending on the value defined by delete_data.
  - "redis": MMT-Operator will receive meta-data from a REDIS sever defined by redis_server
  - "kafka": MMT-Operator will receive meta-data from a KAFKA sever defined by kafka_server
- *Behaviour:* This group configures the behaviour of the MMT-Operator:
  - port_number is a port number used to connect to the MMT- Operator web application.
  - local_network is an array indicating IP ranges of local networks. Each network is given by an IP and network mask.
  - buffer limits the size of buffers of MMT-Operator either by (i) the number of elements, max_length_size, or by the timestamp, max_interval. With a small buffer, MMT-Operator must flush data more frequently to the database and to the Web browser clients.

    - micro_flow indicates retaining detailed information on small flows. This parameter set defines how the MMT-Operator defines micro- flows: maximum number of packets in a flow and maximum number of bytes of data in a flow.
    - probe_stats_period is a number, in seconds, indicating the rate of statistic reports from the MMT-Probe. This number must be the same as the value given by stats_period in the configuration file of MMT-Probe.
    - probe_analysis_mode represents the executing mode of MMT- Probe. Its value must be the same as the value given by input-mode in the configuration file of MMT-Probe, e. g., either "online" or "offline".
- *Maintain Database:* This group contains some thresholds used for the maintenance of the historical database.
    - retain_detail_report_period is an interval in seconds indicating how long must the MMT-Operator retain information in the collection called detail in the database. This collection contains detailed information on each report generated by the MMT-Probe. For example, in the file config.json above, MMT-Operator will delete this detailed information if it is older than 7 days. Please note that the MMT-Operator does not delete any information related to other statistics, such as, protocols/applications, bandwidth, etc.
- *Execution log:* This group configures the execution logs of the MMT-Operator.
    - log_folder is the folder that will contain the execution logs of the MMT-Operator.
    - is_in_debug_mode determines whether the MMT-Operator is running in debug mode, i. e., producing detailed log information. How- ever, this will slow down the MMT-Operator. Its value can be true or false.

MMT-Security

We can modify the default thresholds of the MMT-Security module in the file /opt/mmt/security/mmt-security.conf. The file must be created if it does not already exist.

Input:

```
# Maximum size , in bytes , of a report received from the MMT - Probe :
input . max_message_size 3000
```

Security Engine:

```
# Maximum number of reports that will be stored in a ring buffer:
security .smp . ring_size 1000
# Maximum number of instances of a rule:
security . max_instances 100000
```

Alert:

```
# Number of consecutive alerts for one rule without the full description. The first alert of a rule always contains a description of its rule. However, to avoid huge outputs, a certain number of consecutive alerts of that rule can be sent without the full description. For instance, if value is 20 then the first alert will contain the full description and the next 20 alerts generated by the same rule will not.
# Set value to 0 to include the description in all alerts.
output . ignore_description 20
```

A. MMT-Probe

MMT-Probe requires a configuration file for setting different options.

MMT-Probe can operate in two modes PCAP and DPDK. The PCAP mode uses libpcap library, whereas, the DPDK mode uses dpdk library, for packet capturing. There are two input-mode available for PCAP, i.e. "offline" and "online", while in DPDK, only "online" input-mode is available. The "offline" input-mode uses PCAP trace file while, the "online" input-mode uses network interface to capture the packets.

The different available configuration options are listed in the following:

- **Probe identifier**: The probe-id-number indicates the identifier of the MMT-Probe. All output report formats contain this identifier.

  ```
  probe -id - number = 1
  ```

- **License file**: The license_file_path indicates the path where the license information is present.

  ```
  license_file_path = "/ opt / mmt / probe /bin / license .key "
  ```

- **Thread number:** The thread-nb indicates the number of threads the MMT-Probe will use for processing packets. It must be a positive number.

  ```
  thread -nb = 1
  ```

- **Thread queue length:** The thread-queue indicates the maximum number of packets that can be queued for processing by a single thread. For an unlimited queue size, set the value to 0. This value is used only if PCAP mode is used. Other modes are DPDK and offline analysis of PCAP files.

  ```
  thread - queue = 0
  ```

- **Thread queue data volume**: The thread-data indicates the maximum amount of data that can be queued for processing by a single thread. For an unlimited queue volume, set the value to 0. This value is used only if PCAP mode is used.

  ```
  thread - data = 0
  ```

- **Packet snaplen**: The option snap-len allows indicating the maximum packet size to capture. A value of 0 will set the default value (65535). This value is used only if PCAP mode is used.

  ```
  snap - len = 0
  ```

- **Log file:** The logfile indicates the location where the log messages will be written.

logfile = "/ opt / mmt / probe /log / online / log . data

- **Session timeout**: The session-timeout configuration bloc specifies the session timeout in seconds for different types of applications. The default-session-timeout, long-session-timeout, short-session-timeout, and live-session-timeout indicate, respectively, the session timeout period for default session, long lived session, short lived session and live session.

  ```
  session - timeout
  {
  # 0 means default value = 60 seconds . For default session:

  default - session - timeout = 40

  # 0 means default value = 600 seconds . This is reasonable for Web and SSL connections especially
  when long polling is used . Usually applications have a long polling period of about 3~5 minutes .
  long - session - timeout = 0
  # 0 means default value = 15 seconds . For short live sessions :
  short - session - timeout = 0
  # 0 means default value = 1500 seconds . For persistent connections like messaging applications and
  so on:
  live - session - timeout = 0

  }
  ```

MMT-Probe can capture the packets from a NIC or a trace file (PCAP file). Hence, input-mode and input-source needs to be specified according to the requirements:

- **Online:** This allows near real-time analysis of network traffic. In PCAP mode, the NIC's network interface name needs to be identified, whereas in DPDK mode, the interface port number needs to be identified using the input-source option.

```
input - mode = " online "
# For PCAP it is interface name
input - source = " eth0 "
# For DPDK it is interface port number
input - source = "0"
```

- **Offline:** This allows analysis of a PCAP trace file. For offline mode, the input-source identifies the name of the trace file. The offline analysis is only available for the PCAP mode.

```
input - mode = " offline "
input - source = " wow . PCAP "
```

- **Output to file:** The file-output configuration bloc indicates the file where the MMTProbe will write the reports. The data-file indicates the name of the file, location specifies the folder, retain-files indicates how many files to retain at one time, and sample_report indicates the output file sampling rate (i.e., every x seconds). If sampled_report is enabled, then a separate output file is created every x seconds (sampling time), otherwise only a single output file will be created. The sampling time is determined by file-output-period.

```
# Indicates where the traffic reports will be dumped ( CSV file )
file - output
{
# Set to zero to disable file output ( CSV ), 1 to enable
enable = 0:
# File name where the reports will be dumped :
data - file = " dataoutput . csv "
# Location where files are written :
location = "/ opt / mmt / probe / result / report / online /"
# Retains the last x sampled files , set to 0 to retain all files ( note that the value of retain - files must
be greater than the value of thread_nb + 1):
retain - files = 10
# Set to 1 if one needs a sampled file every x seconds or 0 if one needs a single file :
sampled_report = 1
}
```

- **Output to REDIS:** The redis-output configuration bloc indicates that MMT-Probe should use REDIS to publish teh output. The hostname and the port indicate the address (IP-address) and the port of the redis-server respectively.

The redis-server needs to be started beforehand.

```
# Indicates REDIS output :
redis - output
{
# Set to zero to disable publishing to REDIS , 1 to enable publishing to REDIS :
enabled = 0
# Hostname of REDIS server :
hostname = " localhost "
# Port number of REDIS server :
port = 6379

}
```

- **Output to KAFKA**: The kafka-output configuration bloc indicates that MMT-Probe should use KAFKA to publish the output. The hostname and the port indicate the address (IP-address) and the port of the kafka-server respectively.

  The kafka-server needs to be started beforehand.

```
# Indicates KAFKA output :
```

```
kafka - output
{
# Set to zero to disable publishing to KAFKA , 1 to enable publishing to KAFKA :
enabled = 0
# Hostname of KAFKA server :
hostname = " localhost "
# Port number of KAFKA server :
port = 9092

}
```

- **Output to socket**: The socket configuration bloc indicates that the MMT-Probe should send the output reports using sockets. The socket domain can be UNIX, Internet or both when set to 0, 1 or 2 respectively. The socket-descriptor is required for the UNIX domain to specify the location of a socket file descriptor. The server-address and port are required for the Internet domain to specify the address and the port of a socket. The one-socket-server is set to 1 to indicates that only one port is available for communication, i.e., all the threads will send reports to the same port. If it is set to 0, then multiple socket will be created, i.e., every thread will have a unique port number to send the reports and the port option should contain a number of ports equal to the number of threads).

```
# Socket configurations
socket
{
# Set to 1 to enable , 0 to disable :
enable = 0
# 0 for Unix domain , 1 for Internet domain , and 2 for both :
domain = 0
# Required for UNIX domain . Folder location where socket file descriptor is created :
socket - descriptor = "/ opt / mmt / probe / bin/"
# Required for Internet domain . If one - socket - server is set to 0 then the number of port addresses
should be equal to the number of threads ( thread_nb ). If one - socket - server is set to 1, the
number of port address should be only one :
port = {5000}
# Required for Internet domain . IP address of ip_host 1, ip_host 2...
server - address = {" localhost "}
# If set to 0 the server contains multiple sockets to receive the reports . If set to 1 only one socket
will receive the reports :
one - socket - server = 1
}
```

- **Statistics reporting period:** The stats-period indicates that MMT-Probe should report statistics every x seconds. A report will be sent every stats-period number of seconds.

```
# Indicates the periodicity for reports :
stats - period = 5 # period in seconds
```

- **File output period:** The file-output-period indicates the CSV (Common Separated Values) file output period in seconds. A new file will be generated every file-output-period number of seconds.

```
# Indicates the periodicity for reporting output file :
file - output - period = 5 # sampled reporting
```

- **Reports per message (socket):** The num-of-report-per-msg indicates the number of reports included in one socket transaction (message). A socket message will contain num-of-report-per-msg messages in one transaction.

```
# Indicates the number of report per message ( sockets ).
Default is 1. This option is only available for MMT -Security using sockets :

num -of - report -per -msg = 5
```

- **Cache-size for reporting:** The cache-size-for-reporting indicates the maximum number of reports that can be cached before flushing to a file.

```
# A value of 0 means that MMT will decide how many packets to cache ( default = 300000) :
cache -size -for - reporting = 300000
```

- **Protocols without sessions:** The enable-proto-without-session-stat enables or disables the reporting of protocols that do not belong to any session (e.g., ARP).

```
# A value of 1 will enable and 0 will disable the protocol statics :
enable -proto - without - session - stat = 0
```

- **Protocols with sessions:** The session-report configuration bloc enables or disables the reporting of protocols that belong to a session. The output reports can be reported to a file, redis, kafka or a combination of these channels. This is specified in the output-channel field. If nothing is specified, the default channel used will be a file.

```
# indicates session based reporting
session - report report_session
{
# Set to 1 for reporting session reports , 0 to disable :
enable = 0
# Reports are sent to the output channel . The default value is a file .
# More than one output channel is possible . In this case , the value should be a set of comma -
separated values ; for example : {redis , kafka , file }.
# Reports are sent to output channels only when the global parameters are enabled (e.g., file -
output . enable = 1, redis - output . enable = 1, kafka - output . enable = 1).
output - channel = {}

}
```

- **IP fragmentation:** The enable-IP-fragmentation-report configures the reporting of IP fragmentation information.

```
# Set to 1 to enable , 0 to disable
enable -IP - fragmentation - report = 0
```

- **Micro flows:** The micro-flows configuration bloc specifies the criteria to use to determine if a flow is a micro flow. A single micro flow statistic will not be reported separately, statistics from several micro flows will be aggregated and reported together. Aggregating micro flows statistics reduces the number of reports; however, one will lose fine grained information about each flow. The include-packet-count indicates the packet count threshold to be used to determine that a flow is a micro flow. The include-byte-count indicates the data volume threshold in KB to be used to determine if a flow is a micro flow. The report-packet-count indicates the packet count threshold to be used for creating micro flows aggregated statistics reports. The report-byte-count indicates the data volume threshold in KB to be used for creating micro flows aggregated statistics reports. The report-flow-count indicates the flows count threshold to be used for creating micro flows aggregated statistics reports. The output reports can be reported to a file, redis, kafka or a combination of these channels. This is specified in the output-channel field. If nothing is specified, the default channel used will be a file.

```
micro - flows
{
# Set to 1 to enable , 0 to disable :
enable = 0
# Packets count threshold to consider a flow as a micro flow:
include - packet - count = 0
# Data volume threshold in KB to consider a flow as a micro flow:
include -byte - count = 0
# Packets count threshold to report micro flows aggregated statistics:
report - packet - count = 10000
# Data volume threshold in KB to report micro flows aggregated statistics:
report -byte - count = 5000
# Flows count threshold to report micro flows aggregated statistics:
report -flow - count = 10
# Reports are sent to the output channel. The default value is a file.
# More than one output channel is possible. In this case, the value should be a set of comma -
separated values; for example: {redis, kafka , file }.
# Reports are sent to output channels only when the global parameters are enabled (e.g., file -
output. enable = 1, redis - output. enable = 1, kafka - output. enable = 1).
output - channel = {redis , kafka , file }

}
```

- **Data output:** The data-output is intended for defining the criteria to be used regarding the reporting of specific meta-data. For the time being, it only includes criteria to indicate when to include user agent parsing. The include-user-agent indicates the threshold in terms of data volume for parsing the user agent in Web traffic. This configuration bloc will be extended in the future when new reporting needs arise.

```
data - output
{
# Indicates the threshold in terms of data volume for parsing the user agent in Web traffic .
# The value is in kiloBytes (kB). If the value is zero , this indicates that the parsing of the user agent
should be done .
# To disable the user agent parsing , set the threshold to a negative value ( -1).
include -user - agent = 32

}
```

- **Custom event based reports:** The event_report configuration bloc allows to indicate what protocol attributes should be reported when a certain event is triggered. The event indicates the condition that should be satisfied in order to report the attributes; and, the attributes indicate the application (protocol) attributes that need to be reported when an event occurs. Events and attributes should be in "protocol.attribute" format. There can be multiple event_report configuration blocs. Each event_report bloc is uniquely identified by its name; for instance, event_report report1. The output reports can be reported to a file, redis, kafka or a combination of these channels, as specified in the output-channel field. However, if nothing is specified, the default value is a file.

```
event_report report1
{
# Set to 1 to enable , 0 to disable :
enable = 0
# Indicates the event :
event = " rtp . burst_loss "
# Indicates the list of attributes that are reported when
a event is triggered :
attributes = {" rtp . timestamp ", "rtp . jitter "}
# Reports are sent to the output channel . The default value is a file .
# More than one output channel is possible . In this case , the value should be a set of comma -
separated values ; for example : {redis , kafka , file }.
# Reports are sent to output channels only when the global parameters are enabled (e.g., file -
output .
enable = 1, redis - output . enable = 1, kafka - output .
enable = 1). output - channel = {}

}
```

- **Custom condition based reports:** The condition_report bloc configures the reports on different application (WEB, FTP, RTP, SSL, etc) that are a part of the session-report. These reports are generated and sent to the output channels only when session-report configuration bloc is enabled.

The condition_report.report_web, condition_report.report_ftp, condition_report.report_rtp, and condition_report.report_ssl configuration blocs specify the reports to be generated for HTTP (web), FTP, RTP and SSL applications, respectively. The condition indicates the condition that a flow should satisfy (for example, if a flow corresponds to WEB traffic). The attributes and handlers indicate the application (protocol) attributes and their corresponding handlers that need to be registered in order to generate the reports. The attributes field should be in the "protocol.attribute" format.

WEB report

```
condition_report report_web
{
# Reports are sent to output channels only when session -report . enable = 1.
# Set to 1 to enable , 0 to disable :
 enable = 1
# Indicates the condition to be satisfied .
condition = "WEB "
# Indicates the list of attributes for reporting .
attributes = {" http . uri"," http . method "," http . response ","
http . content_type "," http . host ", " http . user_agent ","
http . referer "," http . xcdn_seen "," http . content_len "}
# Indicates the list of handlers corresponding to the above attributes .
handlers = {" uri_handle "," http_method_handle ","
http_response_handle "," mime_handle "," host_handle ","
useragent_handle "," referer_handle "," xcdn_seen_handle ",
" content_len_handle "}

}
```

File Transfer Protocol (FTP)

```
condition_report report_ftp
{
# Reports are sent to output the output channels only when session - report . enable = 1.
# Set to 1 to enable , 0 to disable :
enable = 1
# Indicates the condition to be satisfied :
condition = "FTP "
# Indicates the list of attributes for reporting :
attributes = {" ftp . data_direction "," ftp. p_payload "," ftp.
packet_type "," ftp . packet_payload_len ", "ftp . data_type "
, " ftp . file_name "," ftp . packet_request ", " ftp.
packet_request_parameter ","ftp . packet_response_code ","
ftp . packet_reponse_value "," ftp . transfer_type "," ftp .
ftp_session_mode ","ftp . data_direction "," ftp .
file_last_modified "," ftp. session_connection_type "," ftp
. user_name "," ftp . password "," ftp . last_command "," ftp.
last_response_code "," ftp. file_size "}
# Indicates the list of handlers corresponding to the above attributes :
handlers = {" ftp_data_direction_handle "," NULL "," NULL ","
NULL "," NULL "," ftp_file_name_handle ","
ftp_packet_request_handle "," NULL "," NULL ","
ftp_response_value_handle "," NULL "," NULL "," NULL "," NULL "
," ftp_session_connection_type_handle ","
ftp_user_name_handle "," ftp_password_handle "," NULL ", "
ftp_response_code_handle "," ftp_file_size_handle "}
}
```

Real-time Transport Protocol

```
condition_report report_rtp
{
# Reports are sent to the output channels only when session - report . enable = 1.
# Set to 1 to enable , 0 to disable :
enable = 0
# Indicates the condition to be satisfied :
condition = "RTP "
# Indicates the list of attributes for reporting :
attributes = {" rtp . version ","rtp . jitter "," rtp . loss ","rtp .
order_err "," rtp . burst_loss "}
# Indicates the list of handlers corresponding to the
above attributes :
handlers = {" rtp_version_handle "," rtp_jitter_handle ","
rtp_loss_handle "," rtp_order_error_handle ","
rtp_burst_loss_handle "}
}
```

Secure Sockets Layer (SSL)

```
condition_report report_ssl
{ # Reports are sent to output channels only when session - report . enable = 1.
# Set to 1 to enable , 0 to disable :
enable = 0
# Indicates the condition to be satisfied :
condition = "SSL "
# Indicates the list of attributes for reporting :
attributes = {" ssl . server_name "}
# Indicates the list of handlers corresponding to the above attributes :
handlers = {" ssl_server_name_handle "}

                    }
```

- **HTTP reconstruction:** The condition_report.reconstruct_http configuration bloc indicates that the MMT-Probe should reconstruct the HTTP data. To enable the reconstruction, the session-report configuration bloc should also be enabled. The location indicates the location where the files are reconstructed; condition indicates the condition that a flow should satisfy for reconstruction; and, attributes and handlers indicate the application (protocol) attributes and their corresponding handlers that need to be registered in order to reconstruct the data. The attributes field should be in the "protocol.attribute" format. Note that currently, if enabled, only HTTP files are reconstructed and no HTTP reports are generated.

```
# Set reconstruct_http . enable = 1 and session - report . enable = 1 for http_reconstruction .
condition_report reconstruct_http
{
# Set to 1 for HTTP reconstruction , 0 to disable .
enable = 0
# Indicates the condition .
condition = "HTTP - RECONSTRUCT "
# Location where the files are reconstructed .
location = ""
# Indicates the list of attributes .
attributes = {" tcp . payload_len ", " tcp . p_payload ", " http .
msg_start ", " http . header ", " http . headers_end ", " http .
data ", " http . msg_end ", " http . method ", " http . response ",
" http . content_type "," http . uri "}
# Indicates the list of handlers corresponding to the above attributes .
handlers = {" NULL ", " NULL ", " http_message_start_handle ",
" http_generic_header_handle ", " http_headers_end_handle
"," http_data_handle ", " http_message_end_handle ", " NULL
", " NULL ", " NULL ", " NULL ", " NULL "}

}
```

- **Radius:** The radius-output bloc configures the reports for the RADIUS protocol. The include-msg indicates the message one needs to report, and include-condition indicates the condition to be met in order to report. The output reports can be reported to a file, redis, kafka or a combination of these channels, as specified in the output-channel field. However, if nothing is specified, the default value is a file.

```
# Indicates the strategy for RADIUS reporting
radius - output
{
# Set to 1 to enable , 0 to disable :
enable = 0
# Indicates the message one needs to report .
# Set to 0 to report all messages .
# Set to a number greater than 0 to indicate the message to report (1 for message , 2 for conditions
):
include - msg = 0
# Indicates the condition to be met in order to report a message .
# Condition set to 1 indicates that the report should be generated iff the IP to MSISDN mapping is
present . This is the only supported condition for this version . Condition set to 0 to eliminate the
condition .
include - condition = 0
# Reports are sent to the output channel . The default value is a file .
# More than one output - channel is possible . In this case , the value should be a set of comma -
separated values ; for example : {redis , kafka , file }.
# Reports are sent to the output channels only when the global parameters are enabled (file - output
. enable = 1, redis - output . enable = 1, kafka - output . enable = 1).
output - channel = {}

}
```

- **Security:** The security output configuration blocs allow specifying the security reporting. There are three different modules for security which are listed below:

Module 1: The security1 is for low bandwidth single threaded operation. The results-dir indicates where the security reports will be written; and, the properties-file specifies the location of an XML file where the security rules have been specified. This module is only available for the PCAP mode.

```
security1
{
# Set to 1 to perform security analysis , 0 to disable it:
enable = 0
# Folder where the detected breaches will be reported :
results - dir = "/ opt/ mmt/ probe / result / security / online /"
# File containing security properties and rules :
properties - file = " test_files / properties_acdc . xml "

}
```

Module 2: The security2 module is for high bandwidth multi-threaded operation. The thread-nb indicates the number of security threads for each probe thread; the exclude-rules indicates the range of rules to be excluded from the verification; and, the rules-mask indicates the range of rules to be verified by each security thread. The generated reports can be written to a file, redis, kafka or a combination of these channels, as specified in the output-channel field. However, if nothing is specified, the default value is a file.

```
{
# Set to 1 enable , 0 to disable it:
enable = 0
# The number of security threads per one probe thread , e.g., if we have 16 probe threads and
thread -nb = x, then x *16 security threads will be used .
# If set to zero this means that the security analysis will be done in the same threads used by the
probe .
thread -nb = 1
# Range of rules to be excluded from the verification :
exclude - rules = ""
# Mapping of rules to the security threads :
# Format : rules - mask = ( thread - index :rule - range );
# Thread - index = a number greater than 0;
# rule - range = number greater than 0, or a range of numbers greater than 0.
# Example : If we have thread -nb = 3 and "(1:1 ,2 ,4 -6) (2:3) ",
# this means that :
# thread 1 verifies rules 1 ,2 ,4 ,5 ,6;
# thread 2 verifies only rule 3; and
# thread 3 verifies the rest
rules - mask = ""
# Reports are sent to the output channels . The default value is a file .
# More than one output channel is possible . In this case , the value should be a set of comma -
separated values ; for example : {redis , kafka , file }.
# Reports are sent to output channels only when the global parameters are enable (file - output .
enable = 1,
redis - output . enable = 1, kafka - output . enable = 1)
output - channel = {}

}
```

Module 3: The security-report is for high bandwidth multi-threaded operation. The MMT-Probe and the MMT-Security exist separately for this mode (i.e., two separate executables). The communication between the MMT-Probe and the MMT-Security modules is done through sockets. The socket should be configured beforehand. The attributes indicate the list of protocol attributes that need to be reported.

```
security - report localhost {
# Set to 1 to enable and 0 to disable :
enable = 0
# Indicates the list of attributes that are reported when an event is triggered :
attributes = { "ip. dst ","ip.src "," tcp . dest_port "," tcp .
flags "," arp . ar_op "," arp . ar_sha ","arp . ar_sip "," ethernet
. dst "," ethernet .src "," tcp . seq_nb "," tcp. ack_nb ","ip.
mf_flag ","ip. frag_offset ","ip. tot_len "," meta .
packet_len ","tcp . payload_len "," tcp . src_port ","ip.
identification "," tcp . urg "," tcp. fin"," tcp . psh " }

}
```

- **Behavior:** The behavior configuration bloc indicates that the MMT-Probe should produce behavior reports. The location indicates the folder where the reports will be written.

```
behaviour
{
# Set to 1 to enable , 0 to disable :
enable = 0
# Folder to write the output :
location = "/ opt / mmt / probe / result / behaviour / online /"

}
```

- **FTP reconstruction:** The reconstruct-ftp output configuration bloc indicates that the MMT-Probe should reconstruct FTP files and generate FTP reports. To enable the reconstruction, the session-report configuration bloc and session-report.ftp_report.enable should also be enabled. The location indicates where the FTP file will be stored. The reports can be written to a file, redis, kafka or a combination of these channels, as specified in the output-channel field. However, if nothing is specified, the default value is a file. The FTP reconstruction is only available for the single threaded operation.

```
# Indicates that FTP data reconstruction should be done .
# To enable the reconstruction , also enable the options session - report . enable and session - report
. ftp_report . enable.
reconstruct - ftp
{
# Set to 1 to enable , 0 to disable it:
 enable = 0
# Indicates the folder where the output file is created :
location = "/ opt / mmt /"
# Reports are sent to the output channel . The default value is a file .
# More than one output channel is possible . In this case , the value should be a set of comma -
separated values ; for example : {redis , kafka , file }.
# Reports are sent to output channels only when the global parameters are enabled (e.g., file -
output .
enable = 1, redis - output . enable = 1, kafka - output .
enable = 1).
output - channel = {}

}
```

- **CPU and memory usage:** The cpu-mem-usage configuration bloc defines the CPU and memory usage reports to be generated. The frequency indicates the time-interval for reporting. The output reports can be reported to a file, redis, kafka or a combination of these channels, as specified in the output-channel field. However, if nothing is specified, the default value is a file.

```
# CPU and memory usage monitor
cpu -mem - usage
{
# Set to 1 to perform cpu - mem reporting , 0 to disable it:
enable = 0
# Time - interval for reporting
frequency = 5
# Reports are sent to the output channel . The default value is a file .
# More than one output channel is possible . In this case , the value should be a set of comma -
separated values ;
for example : {redis , kafka , file }.
# Reports are sent to output channels only when the global parameters are enabled (e.g., file -
output .
enable = 1, redis - output . enable = 1, kafka - output .
enable = 1):
output - channel = {}

}
```

- **Security multi-session:** The security-report-multisession bloc configures the report for security multi-session. The attributes indicate the list of protocol attributes to be reported. The output reports can be reported to a file, redis, kafka or a combination of these channels, as specified in the output-channel field. However, if nothing is specified, the default value is a file.

```
# This report is for security multi - session security :
security - report - multisession remote {
# Set to 1 to perform multi - session reporting , 0 to disable it:
enable = 0
# Indicates the list of attributes that are reported :
attributes = { "nfs . file_name "}
# Reports are sent to the output channel . The default value is a file .
# More than one output channel is possible . In this case , the value should be a set of comma -
separated values ; for example : {redis , kafka , file }.
# Reports are sent to output channels only when the global parameters are enabled (e.g., file -
output . enable = 1, redis - output . enable = 1, kafka - output .
enable = 1).
output - channel = {}

}
```

B. MMT-Operator

The configuration for the MMT-Operator is defined in the /opt/mmt/ operator/config.json file. Its structure is as follows:

```
{
" database_server ": {
" host ": " localhost ",
" port ": 27017
},
" redis_server ": {
" host ": " localhost ",
" port ": 6379
},
" kafka_server ": {
" host ": " localhost ",
" port ": 2181
},
" data_folder ": "/ opt / mmt / probe / result / report / online /",
" delete_data ": true ,
" input_mode ": " kafka ",
" probe_analysis_mode ": " online ",
" probe_stats_period ": 5,
" local_network ": [
{
"ip": "192.168.0.0" ,
" mask ": "255.255.0.0"
},{
"ip": "172.16.0.0" ,
" mask ": "255.240.0.0"
},{
"ip": "10.0.0.0" ,
" mask ": "255.0.0.0"
},{
"ip": " fe80 ::" ,
" mask ": "8"
},{
"ip": "0.0.0.0" ,
" mask ": "255.255.255.255"
},{
"ip": "255.255.255.255" ,
" mask ": "255.255.255.255"
},{
"ip": "127.0.0.1" ,
" mask ": "255.255.255.255"
}
],
" buffer ": {
" max_length_size ": 50000 ,
" max_interval ": 30
},
" micro_flow ":{
" packet ": 1,
" byte " : 64
},
" retain_detail_report_period ": 604800 ,
" port_number ": 80,
" log_folder ": "/ tmp /",
" is_in_debug_mode ": true

}
```

The content of config.json can be divided into the following groups:

- **Input:** This group specifies how MMT-Operator can receive meta-data generated by the MMT-Probe. The value of input_mode can be one of the following:
  - "file": MMT-Operator will read meta-data from a file in the folder data_folder. After reading a file, MMT-Operator can delete the file and its semaphore depending on the value defined by delete_data.
  - "redis": MMT-Operator will receive meta-data from a REDIS sever defined by redis_server
  - "kafka": MMT-Operator will receive meta-data from a KAFKA sever defined by kafka_server
- **Behaviour:** This group configures the behaviour of the MMT-Operator:
  - port_number is a port number used to connect to the MMTOperator web application.
  - local_network is an array indicating IP ranges of local networks. Each network is given by an IP and network mask.

  - buffer limits the size of buffers of MMT-Operator either by (i) the number of elements, max_length_size, or by the timestamp, max_interval. With a small buffer, MMT-Operator must flush data more frequently to the database and to the Web browser clients.

  - micro_flow indicates retaining detailed information on small flows. This parameter set defines how the MMT-Operator defines microflows: maximum number of packets in a flow and maximum number of bytes of data in a flow.

  - probe_stats_period is a number, in seconds, indicating the rate of statistic reports from the MMT-Probe. This number must be the same as the value given by stats_period in the configuration file of MMT-Probe.

  - probe_analysis_mode represents the executing mode of MMTProbe. Its value must be the same as the value given by input-mode in the configuration file of MMT-Probe, e. g., either "online" or "offline".

- **Maintain Database:** This group contains some thresholds used for the maintainance of the historical database.

  - retain_detail_report_period is an interval in seconds indicating how long must the MMT-Operator retain information in the collection called detail in the database. This collection contains detailed information on each report generated by the MMT-Probe. For example, in the file config.json above, MMT-Operator will delete this detailed information if it is older than 7 days. Please note that the MMT-Operator does not delete any information related to other statistics, such as, protocols/applications, bandwidth, etc.

- **Execution log:** This group configures the execution logs of the MMT-Operator.

  - log_folder is the folder that will contain the execution logs of the MT-Operator.

  - is_in_debug_mode determines whether the MMT-Operator is running in debug mode, i. e., producing detailed log information. However, this will slow down the MMT-Operator. Its value can be true or false.

C. MMT-Security

We can modify the default thresholds of the MMT-Security module in the file /opt/mmt/security/mmt-security.conf. The file must be created if it does not already exist.

Input:

```
# Maximum size , in bytes , of a report received from the MMT -
Probe :
input . max_message_size 3000
```

Security Engine:

```
# Maximum number of reports that will be stored in a ring
buffer :
security .smp . ring_size 1000
# Maximum number of instances of a rule :
security . max_instances 100000
```

Alert:

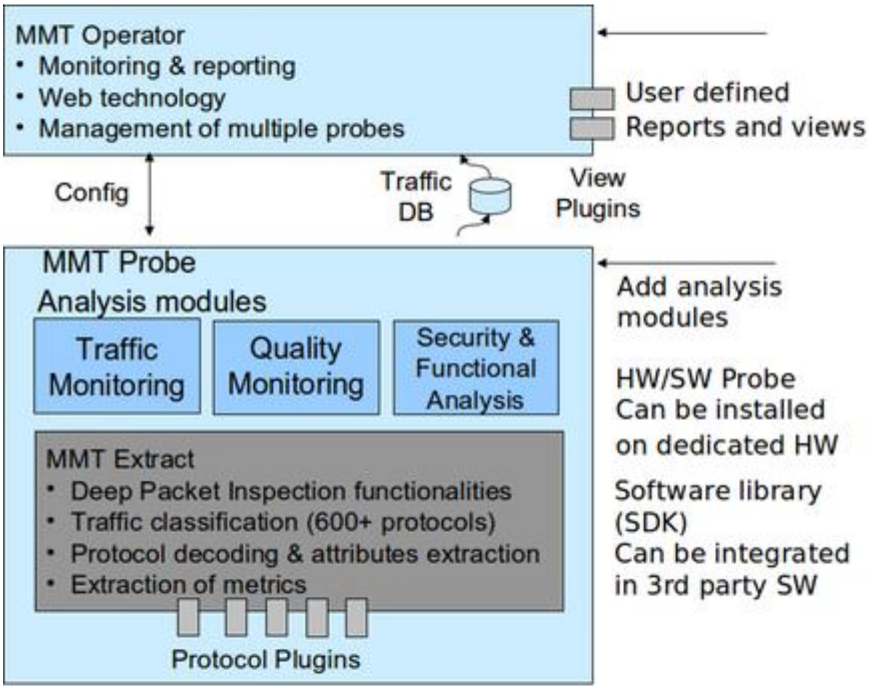
```
# Number of consecutive alerts for one rule without the full description . The first alert of a rule
always contains a description of its rule . However , to avoid huge outputs , a certain number of
consecutive alerts of that rule can be sent without the full description . For instance if value
is 20 then the first alert will contain the full description and the next 20 alerts generated by the same
rule will not .
# Set value to 0 to include the description in all alerts .
output . ignore_description 20
```

MMT User Guide

Monitoring Tool of MEASURE Platform

Montimage Monitoring Tool (MMT) is a monitoring solution that combines data capture, filtering and storage, events extraction and statistics collection, and traffic analysis and reporting, providing network, application, session, and user-level visibility. Furthermore, it is able to correlate information from different sources to detect complex events, and thanks to an advanced rule-based engine, propose counter-measures to react to detected situations (e.g., performance, security, operational incidents). MMT performs online and offline monitoring of the traces of a running system, and it allows the extraction of complex measurements from individual pieces of data. It is able to operate in a non-obstructive fashion, since the execution traces are observed without interfering with the behaviour of the system.

MMT can be easily integrated with third parties in various ways: structured data produced by other applications or systems can feed the Extract module; extracted data and detected events can be used by other tools; behaviour models, pattern matching rules, etc. can be converted to properties to correlate information; and verdicts and events can be used by external tools. All these functionalities are summarized in the MMT global view presented in Figure below.


MMT Operator
• Monitoring & reporting
• Web technology
• Management of multiple probes
User defined
Reports and views
Config
Traffic DB
View Plugins
MMT Probe
Analysis modules
Traffic Monitoring
Quality Monitoring
Security & Functional Analysis
MMT Extract
• Deep Packet Inspection functionalities
• Traffic classification (600+ protocols)
• Protocol decoding & attributes extraction
• Extraction of metrics
Protocol Plugins
Add analysis modules
HW/SW Probe Can be installed on dedicated HW
Software library (SDK) Can be integrated in 3rd party SW

MMT-Operator

This document presents Graphical User Interface of MMT-Operator.

MMT-Operator is a Web application. It has typically 2 parts: Client (front end) and Server (back end).

- **The Server** is written in NodeJS running at the server side.
- **The Client** is written in JavaScript and HTML running on Web browsers at the client sides. There may be many users using their Web browsers to connect to the Server to statistics of MMT. These statistics will be graphically represented in Web browsers of users in forms of chart elements, such as, bar, line, pie, or table. This document presents in detail of the elements.

```
 --------------------                 -------------
|                    |                |             |<=====> Database
|       Client       | <==RESTful==>  |   Server    |
| (HTML, Javascript) |                |  (NodeJS)   |<===== MMT-Probe(s)
 --------------------                 -------------
```

General Structure

The followings are some basic notations being used in MMT:

- **Protocol** is a network protocol such as, IP, HTTP, NDN, etc.

- **Application** such as BitTorrent, Skype, etc. Contact us to get the full list of protocols and applications that have been supported by MMT.

- **Profile** is a group of protocols and/or applications. MMT-Operator has currently 13 profiles: Content Delivery Network, Cloud Storage, Conversational, DataBase, Direct Download Link, File Transfer, Gaming, Mail, Network, Peer to Peer, SocialNetwork, Streaming, Web.

- **Packet** is a term used in MMT to represent a data unit of a protocol. It is not restricted only for protocols at layer 3 of OSI. A packet consists of a **header** part and **payload** part. Header part contains control information that provides data for delivering the payload, for example: source and destination network addresses, error detection codes, and sequencing information. Payload part contains user data that may be a packet of a higher protocol, e.g., payload of IP packet can be a TCP packet.

- **Micro Session** is a set of very small sessions. A session is considered as a micro one if its number of packets and data are less than some thresholds. MMT allows user to change easily these thresholds via a configuration file. Micro session will not be reported separately, rather, aggregated statistics from micro sessions will be reported together. Using micro sessions statistics reduces the report size. However, one will loose microscopic information about these micro sessions.

- Network traffic are represented through 4 metrics:

  - **Data Volume** is size of data, in Bytes or Bits, of packets.

  - **Payload Volume** is size of payload part, in Bytes or Bits, of packets.

  - **Packet Count** is number of packets.

  - **Session Count** is number of TCP/IP sessions. Each session is differed by a 4-tuple (IP source, IP destination, Port source and Port destination).

The following things are applied on GUI:

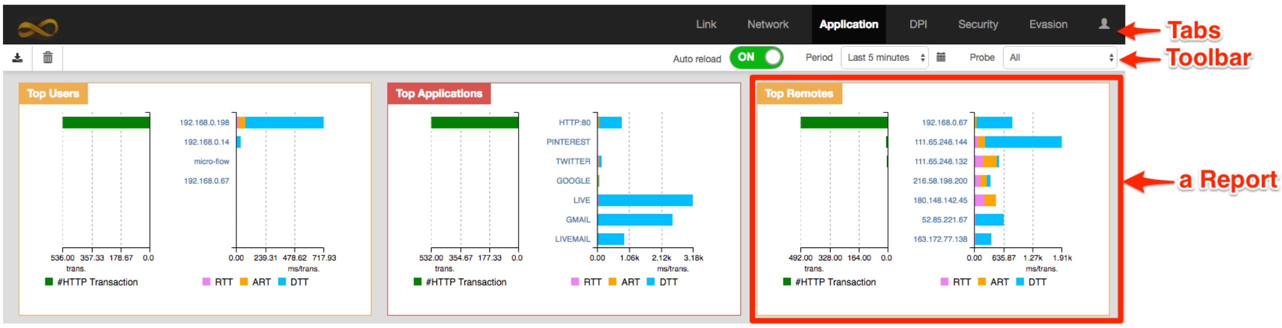
- When a button is available, the cursor will be change to a pointer when moving over the button.

- The changing of display, such as, delete/resize a Report, reorder Report, etc., is only locally. It only effects the current Web browser.

Tab

Statistics will be grouped into tabs, e.g., Link, Network, Application, DPI, Security, Evasion and Setting.

Each tab has a Toolbar and a set of Reports. The Figure below represents the Application tab having 3 reports: Top Users, Top Applications and Top Remotes.



Tabs

Toolbar

a Report

Toolbar

The toolbar often shows the following buttons, from left to right:

1. **Export Charts to Images** : When click on this button, all displayed Reports will be exported to .png files. You might allow Google Chrome to download multiple files to download several report pictures.

2. **Delete a Report** : Drag and drop a Report over this button to delete that Report.

3. **Reset View** : Click on this button to reset the view of reports to the initial state.

4. **Auto Reload** Auto reload ON : When it is enabling, the current Tab is automatically reloaded periodically.

5. **Period** Period Last 5 minutes decides a period of statistic to shown, such as, the statistic of the last 5 minutes. The available periods are: Last 5 minutes, Last hour, Last 6 hours, Last 12 hours, Last 24 hours, Last 7 days, and, Last 30 days.

One might also select a period between two dates by clicking on a small calendar button at the right of combobox.

6. **Probe** Probe All lists all running MMT-Probe in the current Period. If there is only one MMT-Probe, this combobox has only one value "All". When more than one MMT-Probe is running, one might select the combobox to see the statistics of one or all MMT-Probes.

Please note that, one of the buttons above can be hidden on some specific Tabs.

Report

A Report graphically represents a statistic of MMT. A Report consist of :

1. **A title** located on the top-left corner

2. **One or many Filters** to filter out unnecessary data. When user changes value of a Filter, the other Filters and Charts will be reloaded.

3. **One or many Charts** is the main part of a Report. A Chart might depend on another, e.g., when an element in a Chart is selected another Chart will be reloaded to show the statistic concerning to the selected element.

One can do the following actions on Report:

1. **Delete a Report**: This action is available when there are more than one Report on a Tab. In such a case, there exists a RecycleBin icon on the left of Toolbar.

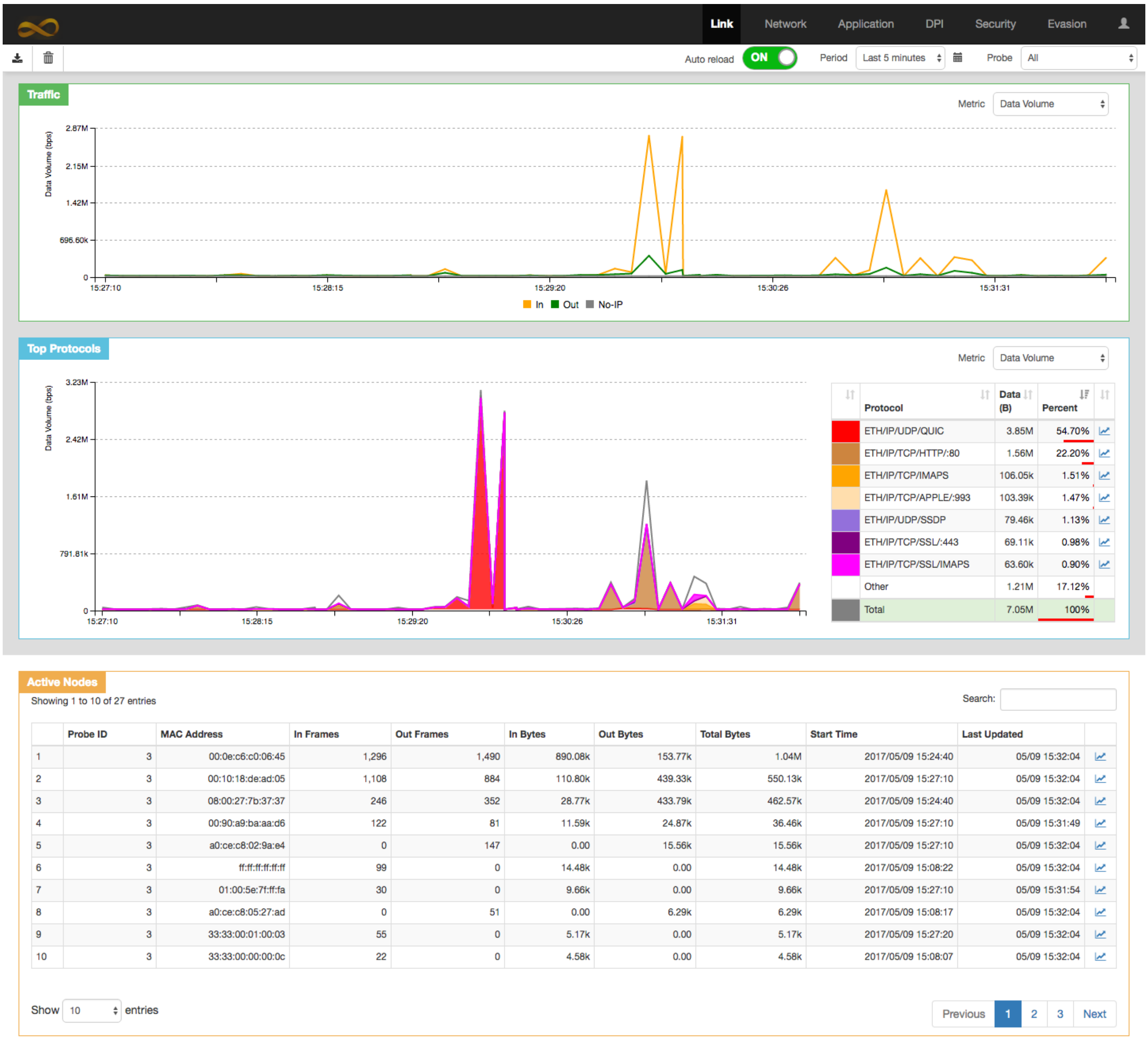
To delete a Report, click and hold on the title of the report, then drag and drop it on the RecycleBin icon.

2. **Resize a Report**: To resize a Report, move cursor to an edge of Report, then drag cursor to resize it. Some Reports cannot be resized.
3. **Reorder Reports in a Tab**: To reorder Reports, drag and drop a Report to a position by click and hold on its title.
4. **Save a Report as a Picture**: Click on the left button on the Toolbar.

Chart

Tab Link

This tab gives an overview of the network being monitored by MMT-Probes, such as, Input/output traffic, the top 7 protocols having highest traffic, list of active nodes since the last 5 minutes. On each report, one can click on detail button 📈 to view bandwidth of an individual such as a protocol or a node.



	Protocol	Data (B)	Percent	
■	ETH/IP/UDP/QUIC	3.85M	54.70%	📈
■	ETH/IP/TCP/HTTP/:80	1.56M	22.20%	📈
■	ETH/IP/TCP/IMAPS	106.05k	1.51%	📈
■	ETH/IP/TCP/APPLE/:993	103.39k	1.47%	📈
■	ETH/IP/UDP/SSDP	79.46k	1.13%	📈
■	ETH/IP/TCP/SSL/:443	69.11k	0.98%	📈
■	ETH/IP/TCP/SSL/IMAPS	63.60k	0.90%	📈
	Other	1.21M	17.12%	
■	Total	7.05M	100%	

Showing 1 to 10 of 27 entries

Search:

	Probe ID	MAC Address	In Frames	Out Frames	In Bytes	Out Bytes	Total Bytes	Start Time	Last Updated	
1	3	00:0e:c6:c0:06:45	1,296	1,490	890.08k	153.77k	1.04M	2017/05/09 15:24:40	05/09 15:32:04	📈
2	3	00:10:18:de:ad:05	1,108	884	110.80k	439.33k	550.13k	2017/05/09 15:27:10	05/09 15:32:04	📈
3	3	08:00:27:7b:37:37	246	352	28.77k	433.79k	462.57k	2017/05/09 15:24:40	05/09 15:32:04	📈
4	3	00:90:a9:ba:aa:d6	122	81	11.59k	24.87k	36.46k	2017/05/09 15:27:10	05/09 15:31:49	📈
5	3	a0:ce:c8:02:9a:e4	0	147	0.00	15.56k	15.56k	2017/05/09 15:27:10	05/09 15:32:04	📈
6	3	ff:ff:ff:ff:ff:ff	99	0	14.48k	0.00	14.48k	2017/05/09 15:08:22	05/09 15:32:04	📈
7	3	01:00:5e:7f:ff:fa	30	0	9.66k	0.00	9.66k	2017/05/09 15:27:10	05/09 15:31:54	📈
8	3	a0:ce:c8:05:27:ad	0	51	0.00	6.29k	6.29k	2017/05/09 15:08:17	05/09 15:32:04	📈
9	3	33:33:00:01:00:03	55	0	5.17k	0.00	5.17k	2017/05/09 15:27:20	05/09 15:32:04	📈
10	3	33:33:00:00:00:0c	22	0	4.58k	0.00	4.58k	2017/05/09 15:08:07	05/09 15:32:04	📈

Show 10 entries

Previous 1 2 3 Next

Tab link consists of 3 reports:

1. **Traffic** represents the total bandwidth of the network representing via 3 lines: in-bound and out-bound of IP traffic, along with the total bandwidth of other traffic that are non-IP based protocols such as ARP.

One can click on a legend item 🟧 In 🟩 Out ▨ No-IP to hide/unhide the line corresponding.

2. **Top Protocols** contains the top 7 protocols. This report consists of 2 charts: the left one represents historical bandwidth, in bit per second, of the top protocols; the right one is the list of these protocols along with their total data and percentage.

One can click on one item of the list to hide/unhide the line corresponding on the left side.

3. **Active Nodes** contains the information about the nodes in the network that are being active since the last 5 minutes. A node in a network is identified by its unique media access control address (MAC address).

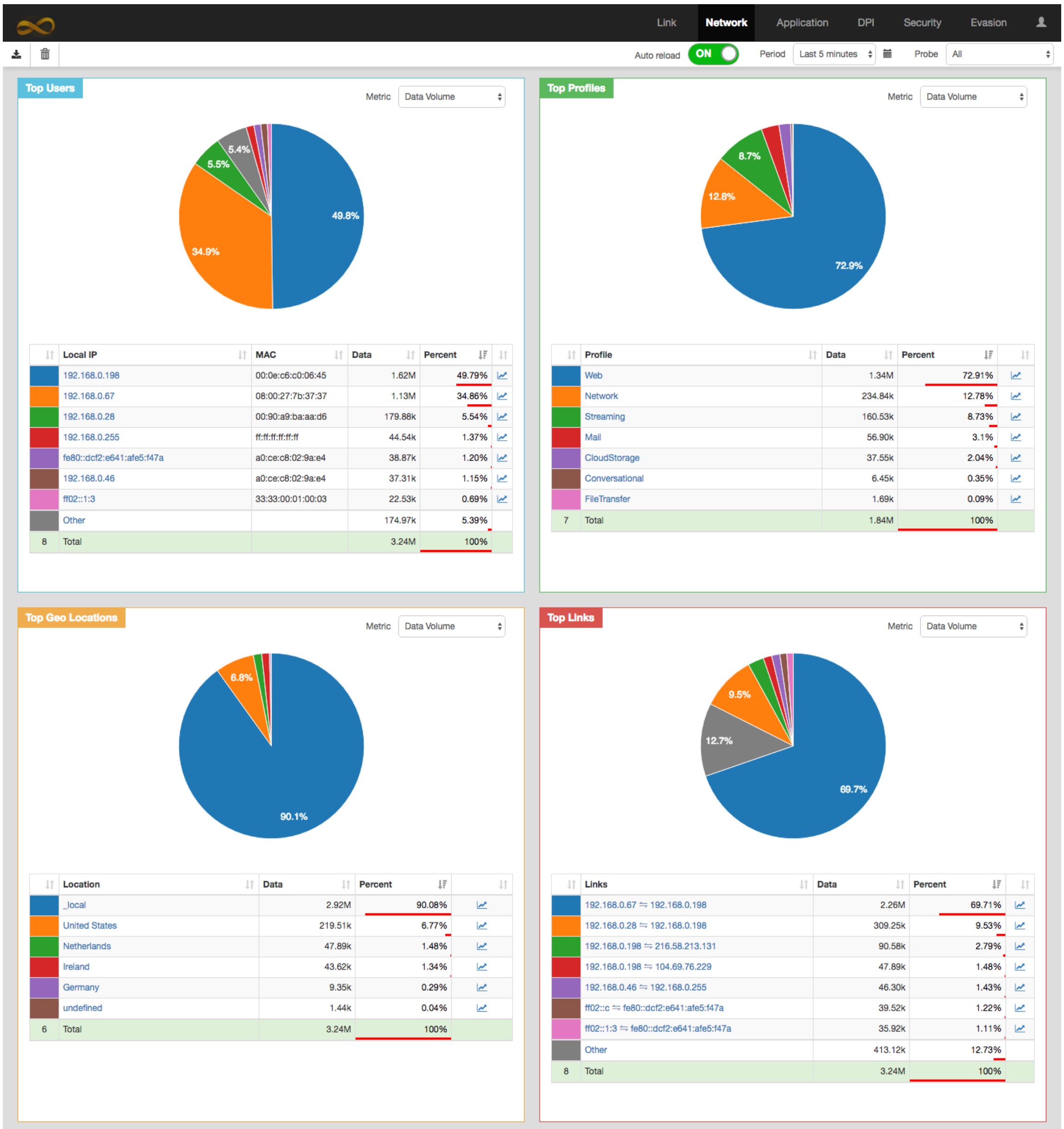
This report is not influenced by the Period filter on the toolbar. It always shows the active nodes since the last 5 minutes or the moment started MMT if MMT has been started less than 5 minutes.

Each row in the table represents a unique. Only the active nodes since the last minutes have statistical data. The statistic of the nodes, that were active since the last 5 minutes and inactive since the last minutes, are set to zero.

The start time and the last updated time are respectively the first and latest moment MMT saw a packet coming/outgoing to this node

Tab Network

This tab gives at the glance the top factors in the networks, such as, top users, top profiles, top locations, top links. These factors consume the most traffic. One can also inspect deeply one session.



	Local IP	MAC	Data	Percent	
🟦	192.168.0.198	00:0e:c6:c0:06:45	1.62M	49.79%	📈
🟧	192.168.0.67	08:00:27:7b:37:37	1.13M	34.86%	📈
🟩	192.168.0.28	00:90:a9:ba:aa:d6	179.88k	5.54%	📈
🟥	192.168.0.255	ff:ff:ff:ff:ff:ff	44.54k	1.37%	📈
🟪	fe80::dcf2:e641:afe5:f47a	a0:ce:c8:02:9a:e4	38.87k	1.20%	📈
🟫	192.168.0.46	a0:ce:c8:02:9a:e4	37.31k	1.15%	📈
■	ff02::1:3	33:33:00:01:00:03	22.53k	0.69%	📈
■	Other		174.97k	5.39%	
8	Total		3.24M	100%	

	Profile	Data	Percent	
🟦	Web	1.34M	72.91%	📈
🟧	Network	234.84k	12.78%	📈
🟩	Streaming	160.53k	8.73%	📈
🟥	Mail	56.90k	3.1%	📈
🟪	CloudStorage	37.55k	2.04%	📈
🟫	Conversational	6.45k	0.35%	📈
■	FileTransfer	1.69k	0.09%	📈
7	Total	1.84M	100%	

	Location	Data	Percent	
🟦	_local	2.92M	90.08%	📈
🟧	United States	219.51k	6.77%	📈
🟩	Netherlands	47.89k	1.48%	📈
🟥	Ireland	43.62k	1.34%	📈
🟪	Germany	9.35k	0.29%	📈
🟫	undefined	1.44k	0.04%	📈
6	Total	3.24M	100%	

	Links	Data	Percent	
🟦	192.168.0.67 ⇋ 192.168.0.198	2.26M	69.71%	📈
🟧	192.168.0.28 ⇋ 192.168.0.198	309.25k	9.53%	📈
🟩	192.168.0.198 ⇋ 216.58.213.131	90.58k	2.79%	📈
🟥	192.168.0.198 ⇋ 104.69.76.229	47.89k	1.48%	📈
🟪	192.168.0.46 ⇋ 192.168.0.255	46.30k	1.43%	📈
🟫	ff02::c ⇋ fe80::dcf2:e641:afe5:f47a	39.52k	1.22%	📈
■	ff02::1:3 ⇋ fe80::dcf2:e641:afe5:f47a	35.92k	1.11%	📈
■	Other	413.12k	12.73%	
8	Total	3.24M	100%	

Tab Network consists of first 4 reports. Each report contains the top 7 factors being represented in 2 charts:

- The pie chart represents the percentages of each factor.
- The table gives the detailed list of factors.

For each row of the table,

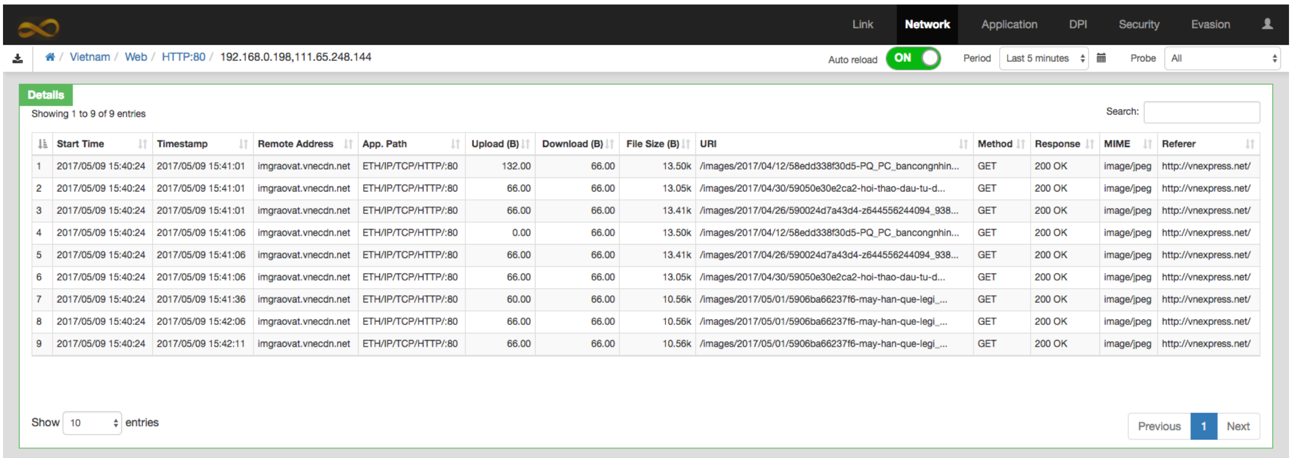
  - Click on one color item, on the left, to hide/unhide the pie corresponding
  - Click on link name to inspect the detail of its factor
  - Click on 📈 to show bandwidth used by its factor

1. **Top Users** is the top 7 users in the network. Each user is identified by a unique IP address.
2. **Top Profiles** is the top 7 Profiles in the network. When click on one profile name, one will get the top 7 applications or protocols of the profile.
3. **Top Geo Locations** is the top 7 destination countries. _local represents the traffic of 2 users in the network.
4. **Top Links** is the top 7 links. One link represents the traffic between 2 users in the network or one user with another from outside the network.

To inspect in detail of one session, one can click on name of each factor. For example, on can:

1. click on Vietnam in the Top Geo Location,
2. then Web on the Profiles,
3. then HTTP:80
4. then 192.168.0.198 <-> 111.65.248.144,

then one obtains the following list:

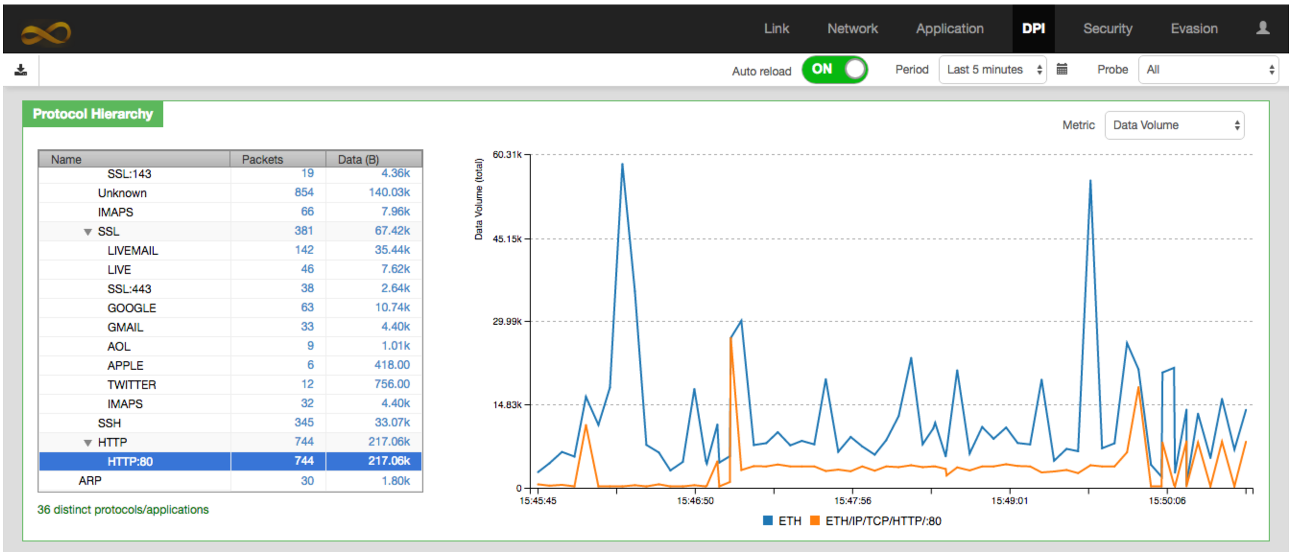


Showing 1 to 9 of 9 entries

	Start Time	Timestamp	Remote Address	App. Path	Upload (B)	Download (B)	File Size (B)	URI	Method	Response	MIME	Referer
1	2017/05/09 15:40:24	2017/05/09 15:41:01	imgraovat.vnecdn.net	ETH/IP/TCP/HTTP/:80	132.00	66.00	13.50k	/images/2017/04/12/58edd338f30d5-PQ_PC_bancongnhin...	GET	200 OK	image/jpeg	http://vnexpress.net/
2	2017/05/09 15:40:24	2017/05/09 15:41:01	imgraovat.vnecdn.net	ETH/IP/TCP/HTTP/:80	66.00	66.00	13.05k	/images/2017/04/30/59050e30e2ca2-hoi-thao-dau-tu-d...	GET	200 OK	image/jpeg	http://vnexpress.net/
3	2017/05/09 15:40:24	2017/05/09 15:41:01	imgraovat.vnecdn.net	ETH/IP/TCP/HTTP/:80	66.00	66.00	13.41k	/images/2017/04/26/590024d7a43d4-z644556244094_938...	GET	200 OK	image/jpeg	http://vnexpress.net/
4	2017/05/09 15:40:24	2017/05/09 15:41:06	imgraovat.vnecdn.net	ETH/IP/TCP/HTTP/:80	0.00	66.00	13.50k	/images/2017/04/12/58edd338f30d5-PQ_PC_bancongnhin...	GET	200 OK	image/jpeg	http://vnexpress.net/
5	2017/05/09 15:40:24	2017/05/09 15:41:06	imgraovat.vnecdn.net	ETH/IP/TCP/HTTP/:80	66.00	66.00	13.41k	/images/2017/04/26/590024d7a43d4-z644556244094_938...	GET	200 OK	image/jpeg	http://vnexpress.net/
6	2017/05/09 15:40:24	2017/05/09 15:41:06	imgraovat.vnecdn.net	ETH/IP/TCP/HTTP/:80	66.00	66.00	13.05k	/images/2017/04/30/59050e30e2ca2-hoi-thao-dau-tu-d...	GET	200 OK	image/jpeg	http://vnexpress.net/
7	2017/05/09 15:40:24	2017/05/09 15:41:36	imgraovat.vnecdn.net	ETH/IP/TCP/HTTP/:80	60.00	66.00	10.56k	/images/2017/05/01/5906ba66237f6-may-han-que-legi_...	GET	200 OK	image/jpeg	http://vnexpress.net/
8	2017/05/09 15:40:24	2017/05/09 15:42:06	imgraovat.vnecdn.net	ETH/IP/TCP/HTTP/:80	66.00	66.00	10.56k	/images/2017/05/01/5906ba66237f6-may-han-que-legi_...	GET	200 OK	image/jpeg	http://vnexpress.net/
9	2017/05/09 15:40:24	2017/05/09 15:42:11	imgraovat.vnecdn.net	ETH/IP/TCP/HTTP/:80	66.00	66.00	10.56k	/images/2017/05/01/5906ba66237f6-may-han-que-legi_...	GET	200 OK	image/jpeg	http://vnexpress.net/

Tab DPI

Tab DPI gives information about hierarchy of protocols/applications. It consists of 1 Report: Protocol Hierarchy.



Auto reload ON · Period: Last 5 minutes · Probe: All

Protocol Hierarchy

Name	Packets	Data (B)
SSL:143	19	4.36k
Unknown	854	140.03k
IMAPS	66	7.96k
▼ SSL	381	67.42k
LIVEMAIL	142	35.44k
LIVE	46	7.62k
SSL:443	38	2.64k
GOOGLE	63	10.74k
GMAIL	33	4.40k
AOL	9	1.01k
APPLE	6	418.00
TWITTER	12	756.00
IMAPS	32	4.40k
SSH	345	33.07k
▼ HTTP	744	217.06k
HTTP:80	744	217.06k
ARP	30	1.80k

36 distinct protocols/applications

Metric: Data Volume

The Protocol Hierarchy report has 2 charts: a tree chart on the left and a line chart on the right.

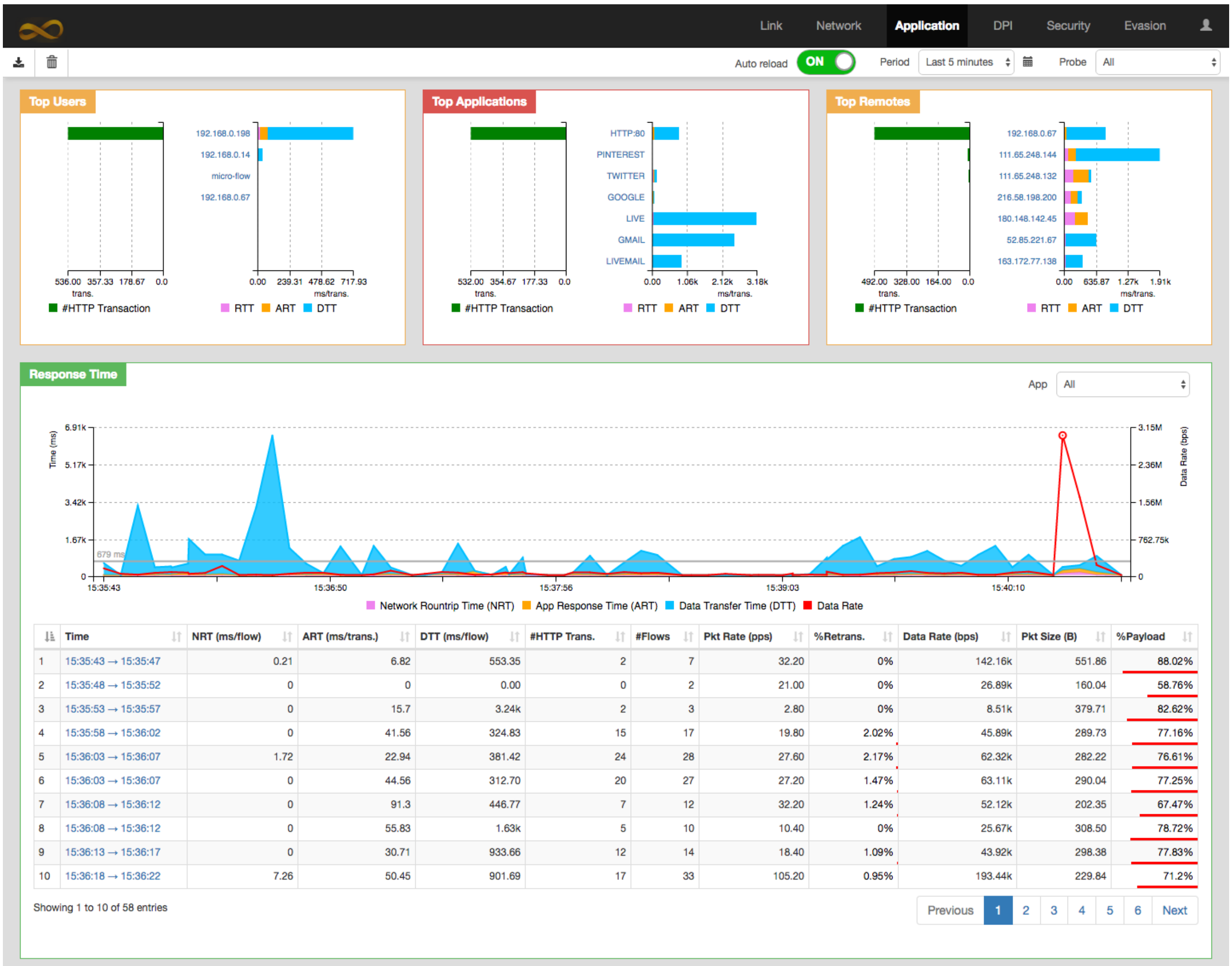
- The tree chart represents the hierarchy of protocols, e.g., there are 36 distinct protocols/application in the figure above.
  - Click on ▼ to collapse/expand the tree.
  - Click on a hyper-link to select/deselect its protocol. When a protocol is selected, its traffic will be shown on the right chart
- The line chart represents the traffic of the selected protocols of the tree chart. These lines do not represent the bandwidth of the protocols but their total traffic during a sample period that is 5 seconds by default.

Through this chart, one can easily see a consistency between protocols. For example, in the figure above, we found that the HTTP traffic vs the total traffic that is represented by ethernet.

Tab Application

Tab Application shows the information about the network's round-trip time, data transfer time, application response time and data rate for the selected application type from the App tab. Moreover, the detailed information is provided in the tables for each application every 5 seconds, that are application response time, data transfer time, server data transfer time, client transfer time, network round-trip time, Number of HTTP transaction, number of active flows, packet rate, data rate, packet size and percentage of payload.

This Tab currently supports only protocols/applications on the top of HTTP and FTP.



	Time	NRT (ms/flow)	ART (ms/trans.)	DTT (ms/flow)	#HTTP Trans.	#Flows	Pkt Rate (pps)	%Retrans.	Data Rate (bps)	Pkt Size (B)	%Payload
1	15:35:43 → 15:35:47	0.21	6.82	553.35	2	7	32.20	0%	142.16k	551.86	88.02%
2	15:35:48 → 15:35:52	0	0	0.00	0	2	21.00	0%	26.89k	160.04	58.76%
3	15:35:53 → 15:35:57	0	15.7	3.24k	2	3	2.80	0%	8.51k	379.71	82.62%
4	15:35:58 → 15:36:02	0	41.56	324.83	15	17	19.80	2.02%	45.89k	289.73	77.16%
5	15:36:03 → 15:36:07	1.72	22.94	381.42	24	28	27.60	2.17%	62.32k	282.22	76.61%
6	15:36:03 → 15:36:07	0	44.56	312.70	20	27	27.20	1.47%	63.11k	290.04	77.25%
7	15:36:08 → 15:36:12	0	91.3	446.77	7	12	32.20	1.24%	52.12k	202.35	67.47%
8	15:36:08 → 15:36:12	0	55.83	1.63k	5	10	10.40	0%	25.67k	308.50	78.72%
9	15:36:13 → 15:36:17	0	30.71	933.66	12	14	18.40	1.09%	43.92k	298.38	77.83%
10	15:36:18 → 15:36:22	7.26	50.45	901.69	17	33	105.20	0.95%	193.44k	229.84	71.2%

Showing 1 to 10 of 58 entries

Previous 1 2 3 4 5 6 Next

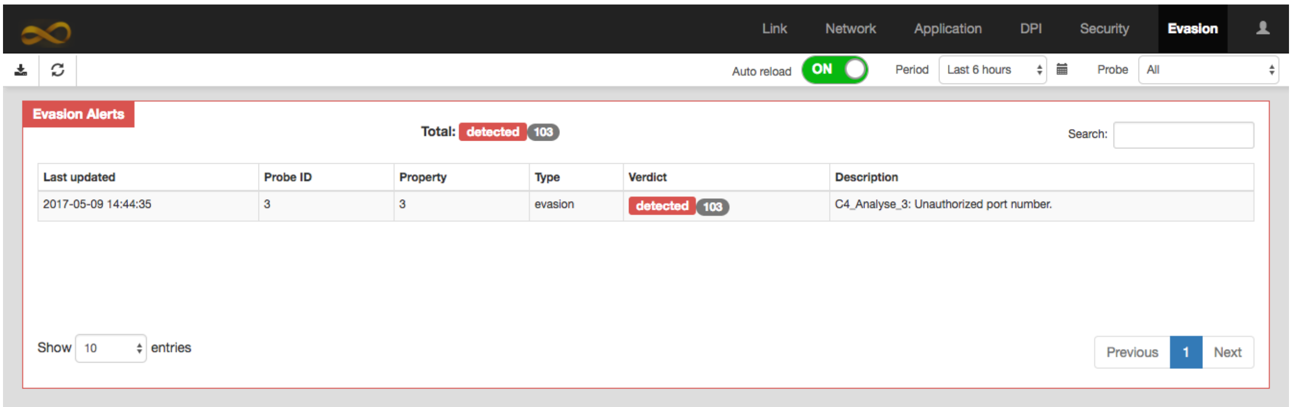
- **FTP Response Time** is the time elapsed between a client sending a request to a FTP server and receiving the response packet. The response time includes the 3 ways TCP handshake.
- **HTTP response Time** is the time elapsed between a client application sending a request (GET) to a HTTP server and receiving the response packet.

Initial TCP RTT (Handshake): Initial RTT of an application is determined by looking at the TCP Three Way Handshake. It is the time elapsed between TCP-SYN and TCP-ACK in the TCP Three Way Handshake.

- **NRT**, Network Response Time, is measured by a TCP handshake.
- **ART**, Application Response Time,
- **DTT**, Data Transfer Time,
- **#HTTP Trans** is the number of HTTP transitions. An HTTP transition is counted from starting a request to receiving completely its response. Different HTTP transitions can perform through only one TCP/IP session.
- **#Flows** indicates the number of TCP/IP sessions.
- **Pkt rate (pps)** indicates the average number of packets received per second
- **%Retrans.** is the percentage of number of packets being retransmitted
- **Data rate (bps)** indicates the average number of bits received per second
- **Packet size (B)** indicates an average packet size, in Bytes
- **%Payload** indicates the percentage of payload on the total data. When this percentage closes to 100%,

Tab Security and Evasion

Tab Security and Tab Evasion list all security alerts. The alerts are grouped by property and probe ID. These tabs list only the latest 5000 alerts.



Evasion Alerts

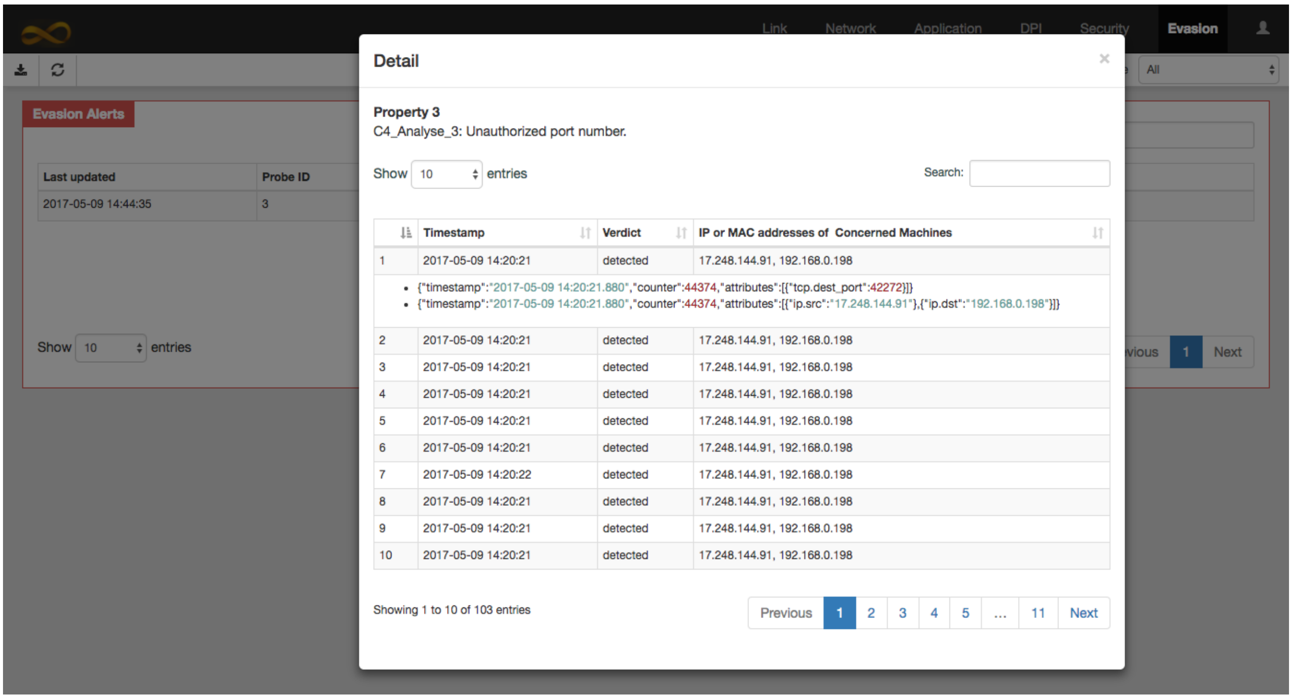
Total: detected 103

Search:

Last updated	Probe ID	Property	Type	Verdict	Description
2017-05-09 14:44:35	3	3	evasion	detected 103	C4_Analyse_3: Unauthorized port number.

Show 10 entries

Each tab has only one report consisting of one table. Each row of the table represents the alerts of one property. One can click on one row to see the alerts as in the figure below.



Detail

Property 3
C4_Analyse_3: Unauthorized port number.

Show 10 entries

Search:

	Timestamp	Verdict	IP or MAC addresses of Concerned Machines
1	2017-05-09 14:20:21	detected	17.248.144.91, 192.168.0.198
2	2017-05-09 14:20:21	detected	17.248.144.91, 192.168.0.198
3	2017-05-09 14:20:21	detected	17.248.144.91, 192.168.0.198
4	2017-05-09 14:20:21	detected	17.248.144.91, 192.168.0.198
5	2017-05-09 14:20:21	detected	17.248.144.91, 192.168.0.198
6	2017-05-09 14:20:21	detected	17.248.144.91, 192.168.0.198
7	2017-05-09 14:20:22	detected	17.248.144.91, 192.168.0.198
8	2017-05-09 14:20:21	detected	17.248.144.91, 192.168.0.198
9	2017-05-09 14:20:21	detected	17.248.144.91, 192.168.0.198
10	2017-05-09 14:20:21	detected	17.248.144.91, 192.168.0.198

Row 1 details:

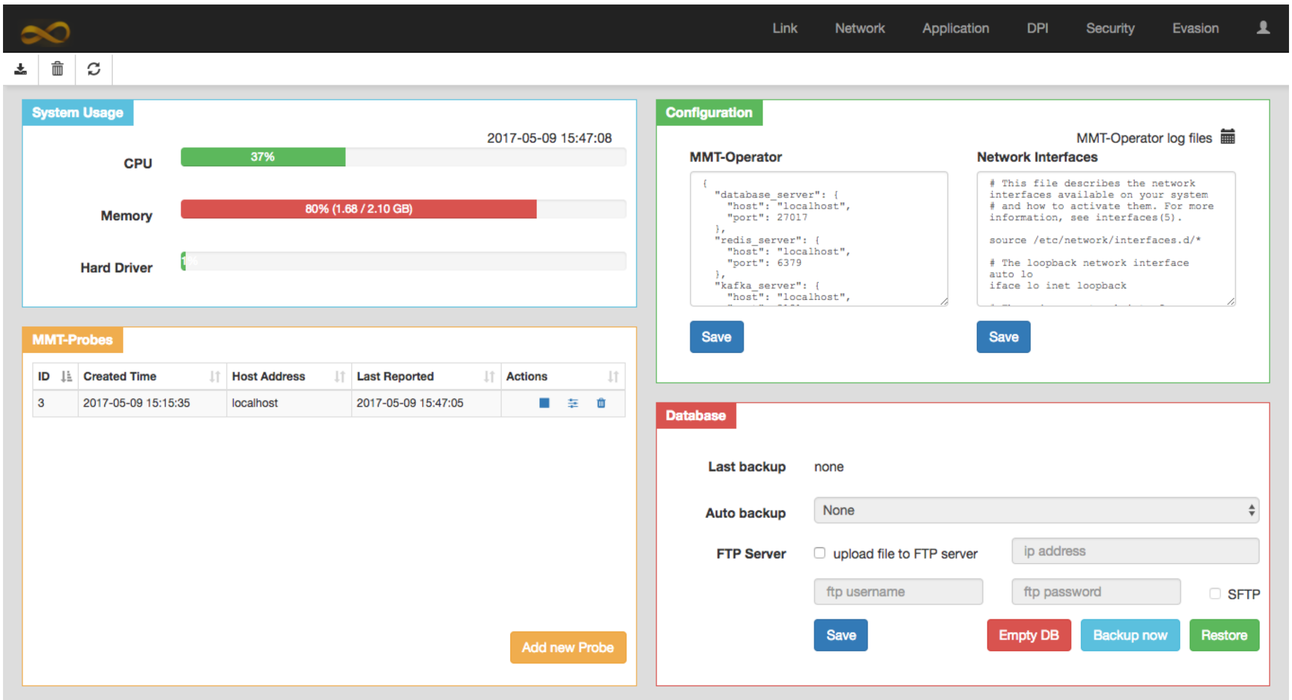
```
• {"timestamp":"2017-05-09 14:20:21.880","counter":44374,"attributes":[{"tcp.dest_port":42272}]}
• {"timestamp":"2017-05-09 14:20:21.880","counter":44374,"attributes":[{"ip.src":"17.248.144.91"},{"ip.dst":"192.168.0.198"}]}
```

Showing 1 to 10 of 103 entries

Previous 1 2 3 4 5 … 11 Next

Tab Setting

Tab setting gives some statistic of server hosting MMT-Operator such as CPU usage, memory and hard driver free space. It also allows user to update setting of MMT-Operator, backup database.



System Usage

2017-05-09 15:47:08

CPU 37%

Memory 80% (1.68 / 2.10 GB)

Hard Driver 1%

MMT-Probes

ID	Created Time	Host Address	Last Reported	Actions
3	2017-05-09 15:15:35	localhost	2017-05-09 15:47:05	■ ☰ 🗑

Add new Probe

Configuration

MMT-Operator log files

MMT-Operator

```
{
  "database_server": {
    "host": "localhost",
    "port": 27017
  },
  "redis_server": {
    "host": "localhost",
    "port": 6379
  },
  "kafka_server": {
    "host": "localhost",
```

Save

Network Interfaces

```
# This file describes the network
interfaces available on your system
# and how to activate them. For more
information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback
```

Save

Database

Last backup none

Auto backup None

FTP Server □ upload file to FTP server ip address

ftp username ftp password □ SFTP

Save Empty DB Backup now Restore

Tab Setting consists of 4 reports.

1. **System Usage** gives a statistic of usage of the server on that MMT-Operator is running.
2. **Configuration** allows to:
    - update file config.json of MMT-Operator
    - update file /etc/network/interfaces of the server
    - view execution logs of MMT-Operator
3. **MMT-Probes** allows to manager MMT-Probe. One can install MMT-Probe on a remote server by giving permission to MMT-Operator to log on that server via SSH.

When clicking on **Add new Probe** button, one is led to another window to enter SSH information of the remote server. After entering successfully, MMT-Operator will install a new MMT-Probe on the server and add it to the list of management.

For the existing MMT-Probe, one can:

- stop/start a probe
- update config file of a probe
- uninstall a probe

4. **DataBase**
    - **Save** button saves information in the form: Auto backup, FTP Server
    - **Empty DB** button empties database that contains MMT statistic. This does not change user information such as password, license, and information in this tab.

After clicking on the button, one need to confirm in another windows before MMT-Operator can empty its database.

- **Backup now** button backups immediately database using the current setting. After clicking on the button, one need to confirm the action.
- **Restore** button leads user to another window to select a backup image from a list of available ones to restore.

Others

1. **Login**: Default login information are admin/ mmt2nm for username/password respectively.
2. **Change Password**: One can change the current password by clicking on 👤 button, then "Change password".

Update Licence: One can update the licence by clicking on 👤 button, then "Profile".

Annexe H. EMIT User Guide (Monitoring Tool)

EMIT User Guide

Monitoring Tool of MEASURE Platform

EMIT Overview

EMIT is a set of web services that make possible to manage MQTT clients. In fact, EMIT allows users to define some MQTT broker connections, to create some MQTT clients and to specify some MQTT callbacks i.e. the MQTT message processes attached to MQTT clients and launched on MQTT message reception. Moreover, EMIT makes possible to update MQTT client states i.e. to connect/disconnect from MQTT brokers, to subscribe/unsubscribe to given topics and to publish messages on given topics. Finally, EMIT makes possible to view the different MQTT client state updates and MQTT messages received and stored by some MQTT client callbacks.

EMIT is provided as a web application i.e. with a HTML frontend to these web services that embeds a JavaScript library that correspond to the HTTP clients of these web services. EMIT is also provided such HTTP clients as a Java library. The usage of such HTTP client is illustrated thanks to the web interface use cases below.

EMIT Installation

EMIT code base is available on GitHub at https://github.com/jeromerocheteau/emit. It consists of a set of Java projects managed by the means of Maven. It can be compiled, packaged and deployed thanks to the following commands:

- git clone https://github.com/jeromerocheteau/emit.git
- cd emit/emit-monitoring/
- mvn clean compile package install tomcat7:redeploy

EMIT runs on a Java application container such as Apache Jetty, Apache Tomcat, Apache TomEE, IBM Webspehere, RedHat Jboss, Oracle Glassfish, etc.

The deployment settings can be modified into the Maven project description (pom.xml file) in order to specify the application server onto the EMIT instance will be depoyed to and its credentials

Dependencies

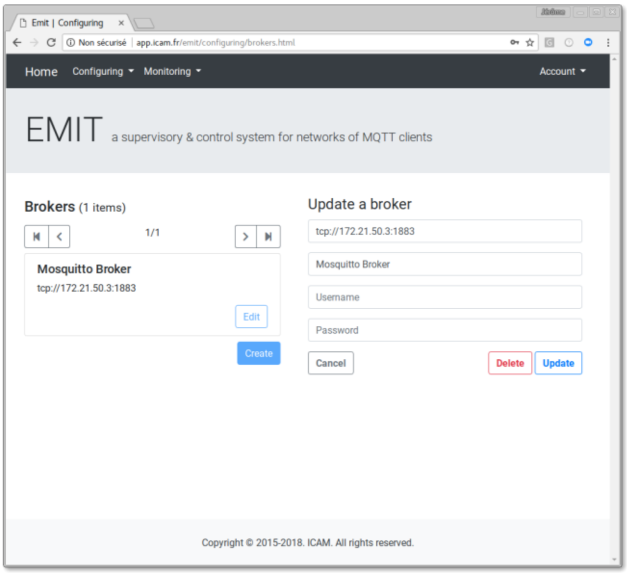
EMIT requires that the following dependencies are already installed as shared libraries within the Java application container:

- com.google.code.gson:gson:2.4
- org.mongodb:mongodb-driver:3.4.2
- org.mongodb:mongodb-driver-core:3.4.2
- org.mongodb:bson:3.4.2
- org.eclipse.paho:org.eclipse.paho.client.mqttv3:1.2.0
- com.github.jeromerocheteau:jdbc-servlet-api:1.0

An instance of EMIT is running at the URL http://app.icam.fr/emit and is accessible according to the following credentials username: measure@emit.icam.fr and password: m3@suR

Editing MQTT Clients

EMIT provides 2 main use cases in order to configure MQTT client networks: the first one consists in defining the connection settings to a given MQTT broker by specifying its URI and eventually its username/password credentials (see Illustration). EMIT also provides a use case to define MQTT clients merely by specifying its already registered MQTT broker (see Illustration).



EMIT a supervisory & control system for networks of MQTT clients

Brokers (1 items) 1/1

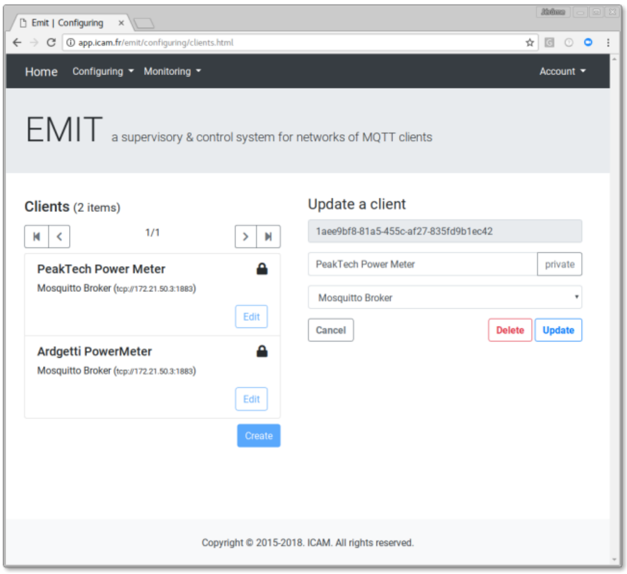
Mosquitto Broker
tcp://172.21.50.3:1883

Update a broker

tcp://172.21.50.3:1883
Mosquitto Broker
Username
Password

Copyright © 2015-2018. ICAM. All rights reserved.

Illustration 1: Editing MQTT Broker



EMIT a supervisory & control system for networks of MQTT clients

Clients (2 items) 1/1

PeakTech Power Meter
Mosquitto Broker (tcp://172.21.50.3:1883)

Ardgetti PowerMeter
Mosquitto Broker (tcp://172.21.50.3:1883)

Update a client

1aee9bf8-81a5-455c-af27-835fd9b1ec42
PeakTech Power Meter private
Mosquitto Broker

Copyright © 2015-2018. ICAM. All rights reserved.

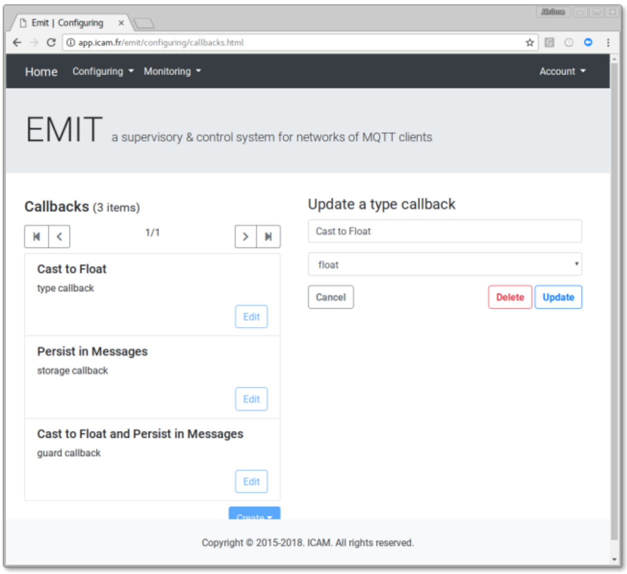
Illustration 2: Editing MQTT Client

Updating MQTT Client Callbacks

EMIT makes possible to define processes of MQTT message that are received by some MQTT clients. Such processes are called MQTT callbacks and EMIT provides several built-in MQTT callback edition use cases. Every MQTT callbacks returns a Boolean value according to the fact that the message process ends successfully or not. The 5 main MQTT callbacks are described below.

The 1st MQTT callback consists in specifying the data type of the MQTT message payload: users specify the data type from a built-in type selection list (see Illustration).

The 2nd MQTT callback consists in a MQTT topic filter or pattern matcher: user defines such pattern (see Illustration).



EMIT a supervisory & control system for networks of MQTT clients

Callbacks (3 items) 1/1

Cast to Float
type callback

Persist in Messages
storage callback

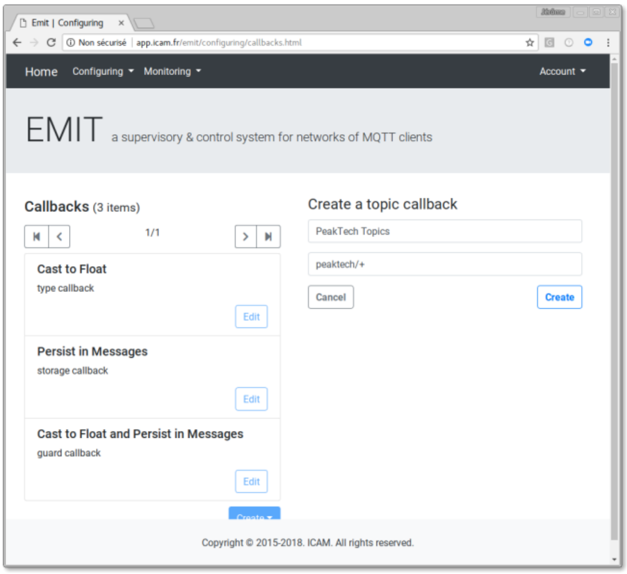
Cast to Float and Persist in Messages
guard callback

Update a type callback

Cast to Float
float

Copyright © 2015-2018. ICAM. All rights reserved.

Illustration 3: Editing MQTT Type Callback



EMIT a supervisory & control system for networks of MQTT clients

Callbacks (3 items) 1/1

Cast to Float
type callback

Persist in Messages
storage callback

Cast to Float and Persist in Messages
guard callback

Create a topic callback

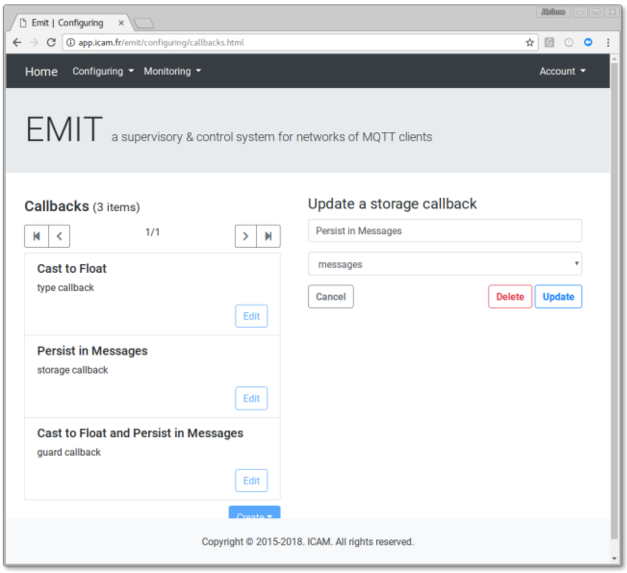
PeakTech Topics
peaktech/+

Copyright © 2015-2018. ICAM. All rights reserved.

Illustration 4: Editing MQTT Topic Callback

The 3rd MQTT callack consists in persisting messages within a database: users speficy the collection that messages will be stored into by selecting this collection between the *messages* collection and the *failures* one (see Illustration).

The 4th MQTT callback consists in verifying a condition over MQTT message payloads: users define the value, its type and the comparison operator among this operator set { =, ≠, < , >, ≤, ≥ } (see Illustration). The MQTT feature callback returns true if and only if the condition is satisfied by the MQTT message payload.



EMIT a supervisory & control system for networks of MQTT clients

Callbacks (3 items) 1/1

Cast to Float
type callback

Persist in Messages
storage callback

Cast to Float and Persist in Messages
guard callback

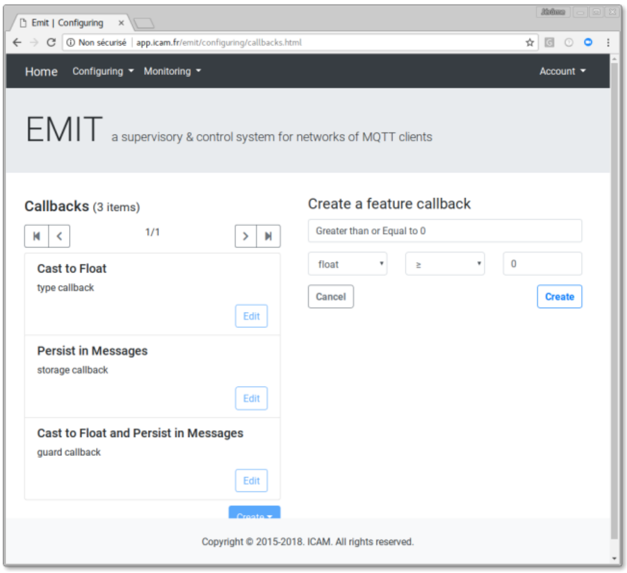
Update a storage callback

Persist in Messages

messages

Copyright © 2015-2018. ICAM. All rights reserved.

Illustration 5: Editing MQTT Storage Callback



EMIT a supervisory & control system for networks of MQTT clients

Callbacks (3 items) 1/1

Cast to Float
type callback

Persist in Messages
storage callback

Cast to Float and Persist in Messages
guard callback

Create a feature callback

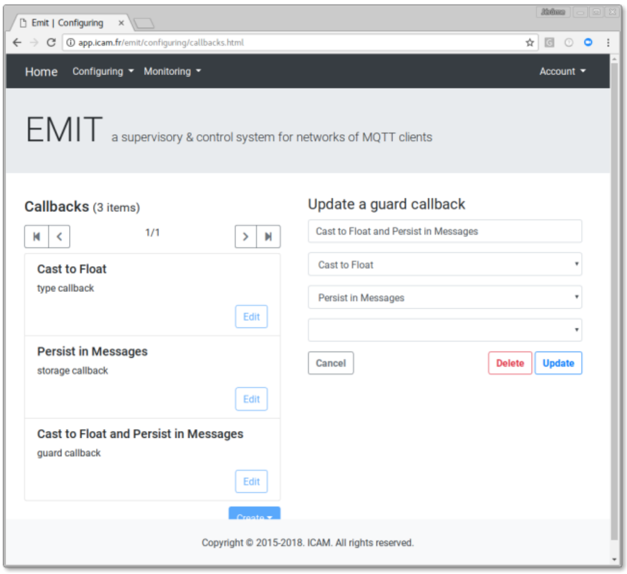
Greater than or Equal to 0

float ≥ 0

Copyright © 2015-2018. ICAM. All rights reserved.

Illustration 6: Editing MQTT Feature Callback

The 5th and last MQTT callback consists in a composite callback as it makes possible to test a first MQTT callback and to dispatch the process flow to a second callback if the first callback ended successfully and, eventually, to a third callback otherwise. Users have then to select the *test*, *success* and *failure* callbacks from the selection list of the already defined MQTT callbacks (see Illustration).



EMIT a supervisory & control system for networks of MQTT clients

Callbacks (3 items) 1/1

Cast to Float
type callback

Persist in Messages
storage callback

Cast to Float and Persist in Messages
guard callback

Update a guard callback

Cast to Float and Persist in Messages

Cast to Float

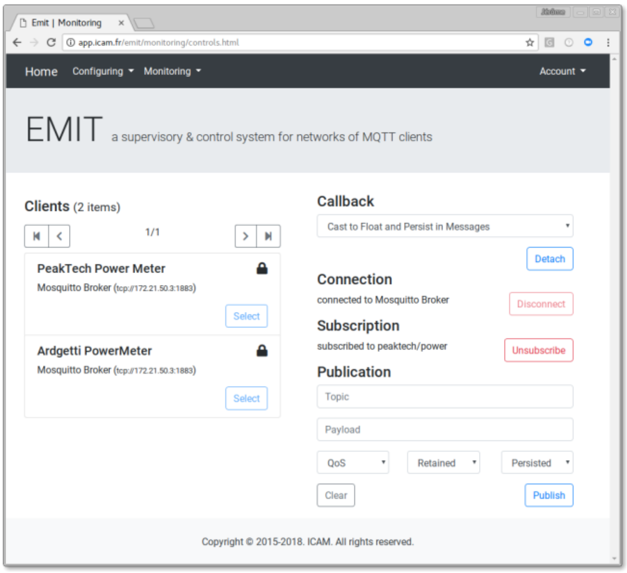
Persist in Messages

Copyright © 2015-2018. ICAM. All rights reserved.

Illustration 7: Editing MQTT Guard Callback

Updating MQTT Client States

The main EMIT use case consists in updating the different MQTT client states. In fact, EMIT makes possible to connect or disconnect a MQTT client from its MQTT broker. EMIT makes possible to subscribe or unsubscribe to a given topics from its related MQTT broker. EMIT makes also possible to publish a message to a given topic to its related broker. In addition, EMIT makes possible to attach or detach a MQTT callback to or from a MQTT client (see Illustration).



EMIT a supervisory & control system for networks of MQTT clients

Clients (2 items)

PeakTech Power Meter
Mosquitto Broker (tcp://172.21.50.3:1883)

Ardgetti PowerMeter
Mosquitto Broker (tcp://172.21.50.3:1883)

Callback
Cast to Float and Persist in Messages

Connection
connected to Mosquitto Broker

Subscription
subscribed to peaktech/power

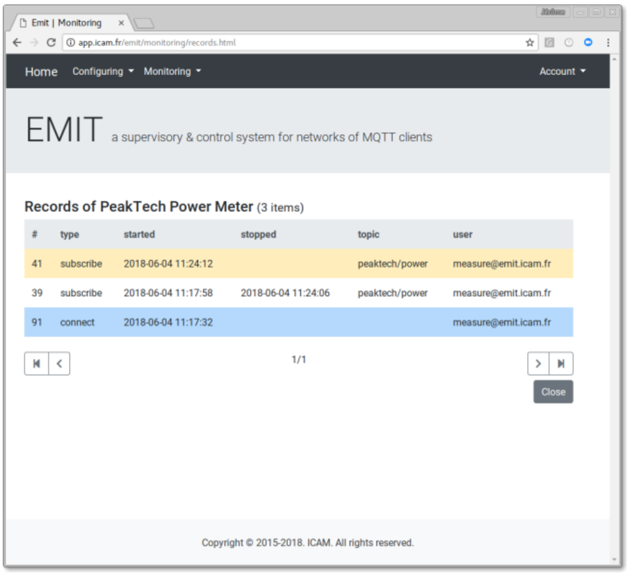
Publication

Copyright © 2015-2018. ICAM. All rights reserved.

Illustration 8: Updating MQTT Client States

Viewing MQTT Client State Updates & Messages

EMIT provides 2 other use cases: EMIT makes possible to retrieve the different MQTT client state updates (see Illustration) and EMIT makes possible to retrieve the MQTT messages persisted into the embedded database engine by the means of MQTT storage callbacks that have been attached to MQTT clients (see Illustration).



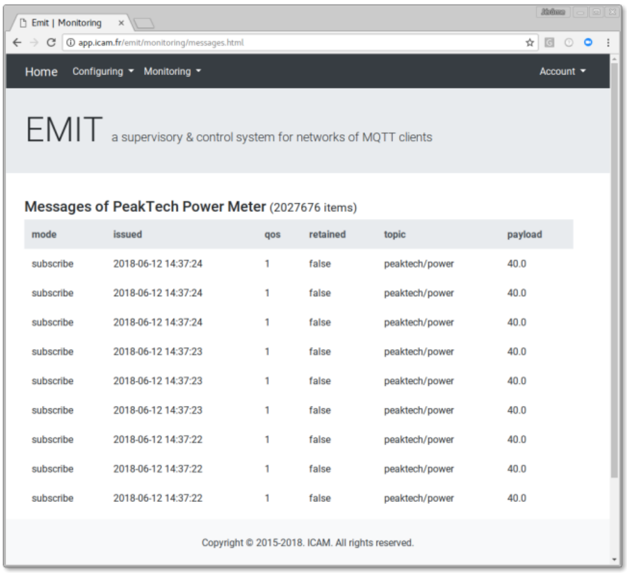
EMIT a supervisory & control system for networks of MQTT clients

Records of PeakTech Power Meter (3 items)

#	type	started	stopped	topic	user
41	subscribe	2018-06-04 11:24:12		peaktech/power	measure@emit.icam.fr
39	subscribe	2018-06-04 11:17:58	2018-06-04 11:24:06	peaktech/power	measure@emit.icam.fr
91	connect	2018-06-04 11:17:32			measure@emit.icam.fr

Copyright © 2015-2018. ICAM. All rights reserved.

Illustration 9: Viewing MQTT Client State Updates



EMIT a supervisory & control system for networks of MQTT clients

Messages of PeakTech Power Meter (2027676 items)

mode	issued	qos	retained	topic	payload
subscribe	2018-06-12 14:37:24	1	false	peaktech/power	40.0
subscribe	2018-06-12 14:37:24	1	false	peaktech/power	40.0
subscribe	2018-06-12 14:37:24	1	false	peaktech/power	40.0
subscribe	2018-06-12 14:37:23	1	false	peaktech/power	40.0
subscribe	2018-06-12 14:37:23	1	false	peaktech/power	40.0
subscribe	2018-06-12 14:37:23	1	false	peaktech/power	40.0
subscribe	2018-06-12 14:37:22	1	false	peaktech/power	40.0
subscribe	2018-06-12 14:37:22	1	false	peaktech/power	40.0
subscribe	2018-06-12 14:37:22	1	false	peaktech/power	40.0

Copyright © 2015-2018. ICAM. All rights reserved.

Illustration 10: Viewing MQTT Client Messages

Annexe I. MINT Installation Guide (Analysis Tool)

MINT Installation Guide

Analysis Tool of MEASURE Platform

Environment Installation

Prerequisite

Node.js is cross-platform meaning that Mint can work on Windows, OSX and Linux. For that, the tool requires the installation of: Node.js, MySQL and Yarn. Mint requires Node.js 8.9.0 or above, the installation of a MySQL database, Redis and Yarn.

NodeJs Installation

- Download Node.js 8.9.0 or above (includes npm) : *https://nodejs.org/en/download/*
- Install Node.js

MySQL Installation

- Download MySQL Community Server 5.7 or above : *https://dev.mysql.com/downloads/mysql/*
- Install MySQL using the following instruction : *https://dev.mysql.com/doc/refman/5.7/en/installing.html*
- Create a new database named "mint_db".

Yarn Installation

- Download Yarn 1.7.0 or above : *https://yarnpkg.com/lang/en/docs/install/*
- Install following the instructions according to the OS chosen.

Redis Installation

- Download the last Redis stable version (4.0) : *https://redis.io/download*
- Install following the instructions according to the OS chosen.

Retrieve Mint Source Code

Mint source code is hosted on GitHub. To retrive it, you can:

- Download it as zip file : *https://github.com/ersilva/Mint/archive/master.zip*
- Clone the Git repository
  - Install git: *https://git-scm.com/downloads*
  - Clone the repository: git clone *https://github.com/ersilva/Mint.git*

Configure the tool

Edit the config.js file which is in the config folder of the project.

Property	Description	Default Value
db.host	MySQL database host	localhost
db.port	MySQL database port	3306

Property	Description	Default Value
db.port	MySQL database name	mint_db
db.port	MySQL database login	root
db.port	MySQL database password	root
measure.host	MEASURE Platform host	localhost
measure.host	MEASURE Platform port	8085

Start the Application

1. Start MySQL
2. install the packages required using yarn:
   `$ yarn install`
3. Populate database. Just for the first time is indispensable to populate the database with the machines description, for this run the command:
   `yarn run seeds:up`
4. Set enviroment variable (development, production or test):
   In Windows :
   `SET NODE_ENV=production`
   In Linux :
   `export NODE_ENV=production`
5. Start the tool:
   `yarn run start`
6. Registration.
   Once the tool is running it can be registered into the Measure platform.

Recommendation Rules Description (EFSMs)

Software Modularity

The assessment of the software modularity relies on two metrics provided by the SonarQube tool that are the class complexity and the maintainability rating. The class complexity measure (also called cognitive complexity) computes the cognitive weight of a Java Architecture. The cognitive weight represents the complexity of a code architecture in terms of maintainability and code understanding. The maintainability rating is the ratio of time (according to the total time to develop the software) needed to update or modify the software. Based on these definitions, and considering that a modular code can be more understandable and maintainable, we can correlate the two metrics and compute the ratio {\textit{R = class}} complexity/maintainability rating. If this ratio is more than a specific threshold set by an expert, the recommendation "Reinforce the modular design of your development" will be provided to the software architect and developers.

Metrics

- **ClassComplexityBySonarQube**

Average complexity by class.

2 data types :

- value : u_double
- postDate : u_date

- **MaintainabilityRatingBySonarQube**

Rating given to your project related to the value of your Technical Debt Ratio. The default Maintainability Rating grid is: A=0-0.05, B=0.06-0.1, C=0.11-0.20, D=0.21-0.5, E=0.51-1 The Maintainability Rating scale can be alternately stated by saying that if the outstanding remediation cost is: <=5% of the time that has already gone into the application, the rating is A between 6 to 10% the rating is a B between 11 to 20% the rating is a C between 21 to 50% the rating is a D anything over 50% is an E
2 data types :

- value : u_double
- postDate : u_date

Formula

ClassComplexityBySonarQube/MaintainabilityRatingBySonarQube > threshold

Requirements quality

The assessment of the requirements quality can rely on two metrics provided by the SonarQube tool that are the total number of issues and the total number of reopened issues. These numbers are collected during the implementation phase and we can consider that the fact that we reopen an issue many times during the development process can be related to an ambiguous definition of the requirement that needs to be implemented. If we have a ratio R = number of reopened issues/number of issues that is more than a specific threshold, we can consider that the requirements are not well defined and that the development needs more refinement about them. The recommendation "Refine requirement definitions or provide more details" will be provided to the requirements analyst.

Metrics

- **IssuesBySonarQube**

Number of issues.

2 data types :

- value : u_double
- postDate : u_date

- **ReopenedIssuesBySonarQube**

Number of issues whose status is Reopened.

2 data types :

- value : u_double
- postDate : u_date

Software Performance

The assessment of the software performance relies on two metrics provided by the MMT tool that are the response time and the bandwidth usage. The response time denotes the delay that can be caused by the software, hardware or networking part that is computed during operation. This delay is in general the same for a constant bandwidth (an equivalent number of users and concurrent sessions). Based on this finding, we can correlate the two metrics and compute that the response time is not increasing for during time for the same bandwidth usage. If this response time is increasing, the recommendation "Optimize the code to improve performance and minimize delays" will be provided.% to the software developers and deployers.

Metrics

- **MMT-AppRespTime**
- **MMT-Bandwidth**

Software security

The assessment of the software security relies on two metrics, one provided by the SonarQube tool that is the security rating and the other is provided by MMT that is the number of security incidents. The security rating in SonarQube provides an insight of the detected vulnerabilities in the code and are presented with severity being blocker, critical, major, minor or no vulnerability. The number of the security incidents provided by MMT reports on successful attacks during operation. The evaluation of security demonstrates that if an attack is successful this means that the vulnerability in the code was at least major because an attacker was able to exploit it to perform its malicious activity. Based on these definitions and considering that a reliable code should be at last free of major vulnerabilities, we can check if there is a major vulnerability and that the number of attacks at runtime are more than a threshold. If this condition is satisfied, the recommendation "Check code to eliminate exploitable vulnerabilities" will be provided to the software developers and security experts.

Metrics

- **SecurityRatingBySonarCube**

A = 0 Vulnerability
B = at least 1 Minor Vulnerability
C = at least 1 Major Vulnerability
D = at least 1 Critical Vulnerability
E = at least 1 Blocker Vulnerability
2 data types :

- value : u_double
- postDate : u_date

- **MMT-SecurityIncidents**

Code reliability

The assessment of the code reliability relies on two metrics provided by the SonarQube tool that are the number of issues categorized by severity and the reliability rating. The issues in SonarQube are presented with severity being blocker, critical, major, minor or info and the reliability rating are from A to E: A is to say that the software is 100% reliable and E is to say that there is at least a blocker bug that needs to be fixed. Based on these definitions and considering that a reliable code should be at last free of major or critical issues, we can check that there is no major, critical nor blocker issues and the reliability rating is $<$ C corresponding to 1 major bug. If this condition is not satisfied, the recommendation "There is unsolved major issues in the code, make a code review and check untested scenarios" will be provided to the software developers and testers.

Metrics

- **IssuesBySeverityBySonarCube**

Number of issues with severity being blocker, critical, major, minor or info.

2 data types :

- value : u_double
- postDate : u_date

- **ReliabilityRatingBySonarCube**

A = 0 Bug

B = at least 1 Minor Bug

C = at least 1 Major Bug

D = at least 1 Critical Bug

E = at least 1 Blocker Bug

Annexe J. MINT User Guide (Analysis Tool)

MINT User Guide

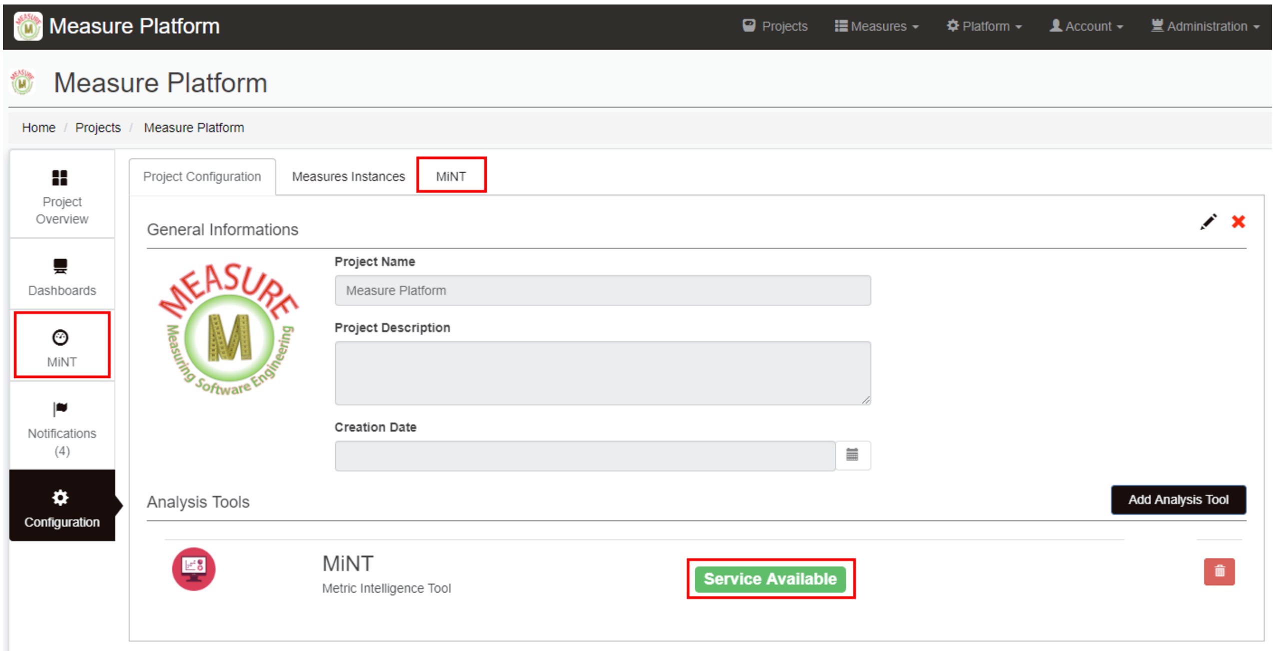
Analysis Tool of MEASURE Platform

Registration

The first step to start using Mint is add it into the MEASURE web application, allowing the tool to run embedded into MEASURE and retrieve the measurements data for the analysis.

Once the tool is running it can be registered into the Measure platform.

- Access to the project where you want to use Mint.
- Access to the configuration tab of the project page configuration and click on *Add Analysis tool*



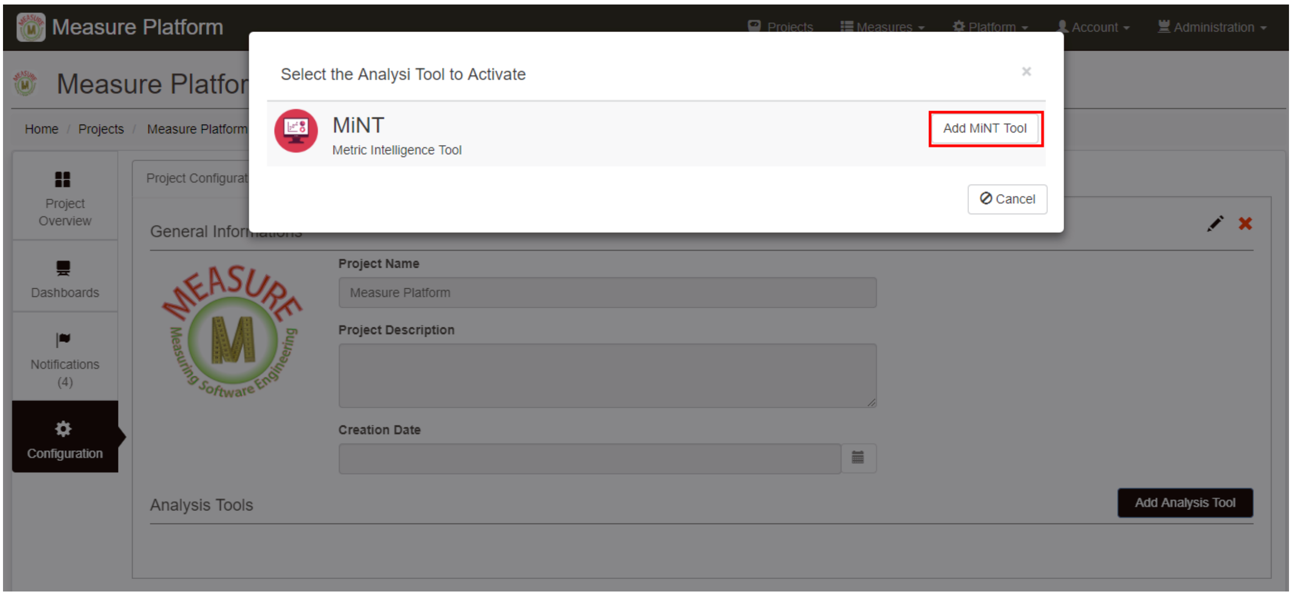
Project Name: Measure Platform

Project Description

Creation Date

MiNT
Metric Intelligence Tool

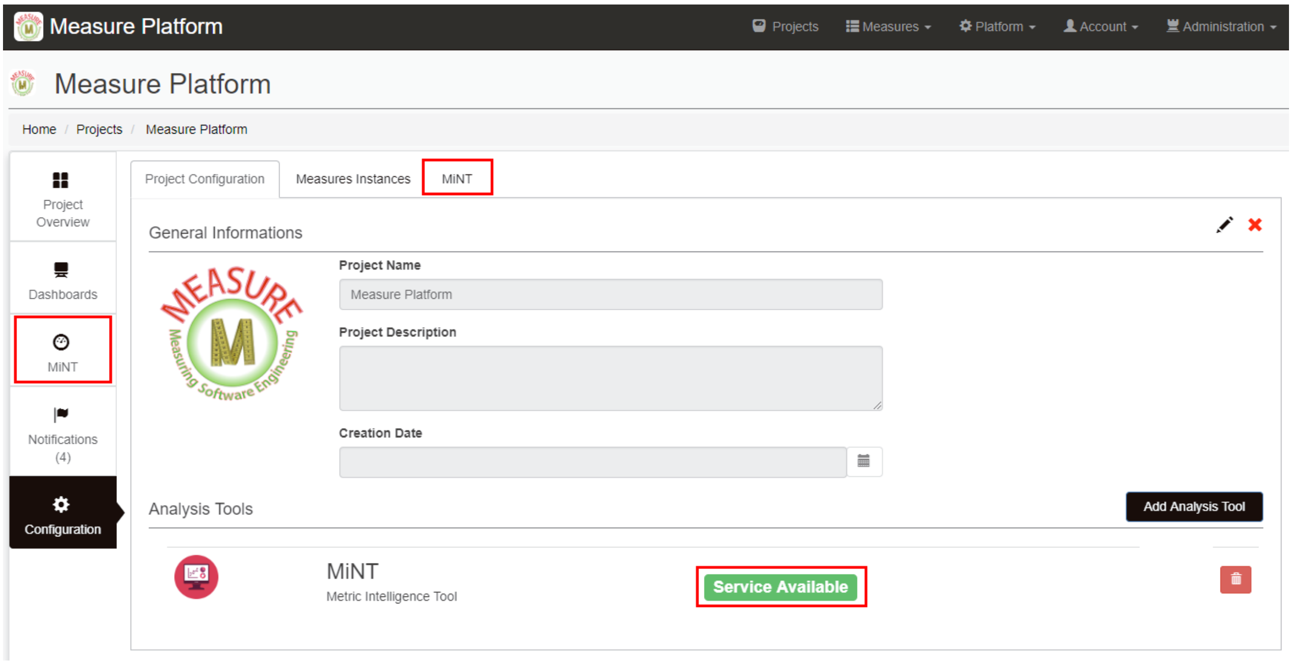
- Mint should be visible in the list of analysis tools to activate.
  Click on Add Mint Tool.



MiNT
Metric Intelligence Tool

Project Name: Measure Platform

- If Mint was added successfully you should see it in the list of available analysis tools with the status Service Available in green. A tab should also appear in the configuration menu and an option in the Project Menu with the name Mint.



General Informations

Project Name: Measure Platform

Project Description:

Creation Date:

Analysis Tools

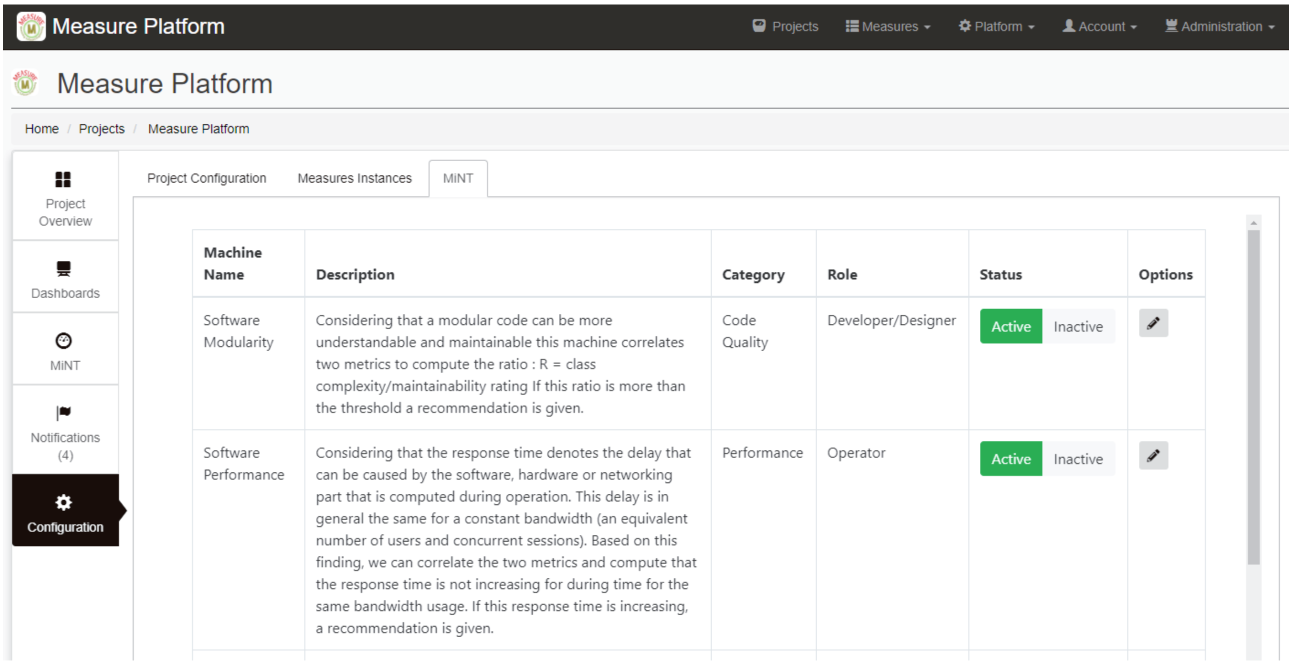
MiNT
Metric Intelligence Tool

Now the tools are ready to be configured and used.

Configuration

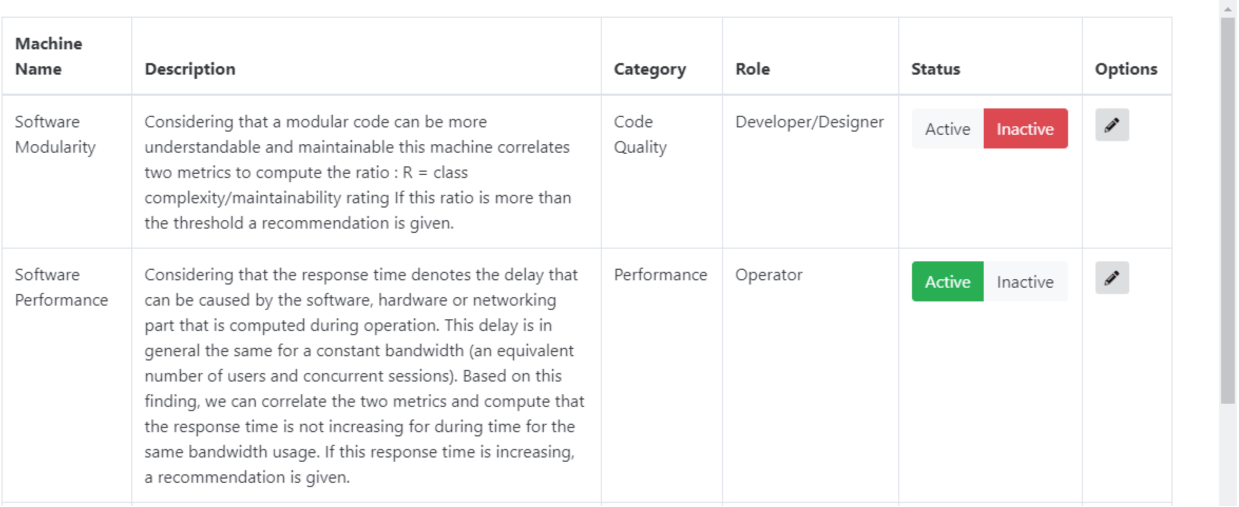
Access to the Mint tab of the project page configuration.

There is a table with the available EFSMs (Extended Finite State Machines) displaying name, description, category, role to which the recommendation is guided, status of the machine (Active or Inactive) and options.



Machine Name	Description	Category	Role	Status	Options
Software Modularity	Considering that a modular code can be more understandable and maintainable this machine correlates two metrics to compute the ratio : R = class complexity/maintainability rating If this ratio is more than the threshold a recommendation is given.	Code Quality	Developer/Designer	Active Inactive	
Software Performance	Considering that the response time denotes the delay that can be caused by the software, hardware or networking part that is computed during operation. This delay is in general the same for a constant bandwidth (an equivalent number of users and concurrent sessions). Based on this finding, we can correlate the two metrics and compute that the response time is not increasing for during time for the same bandwidth usage. If this response time is increasing, a recommendation is given.	Performance	Operator	Active Inactive	

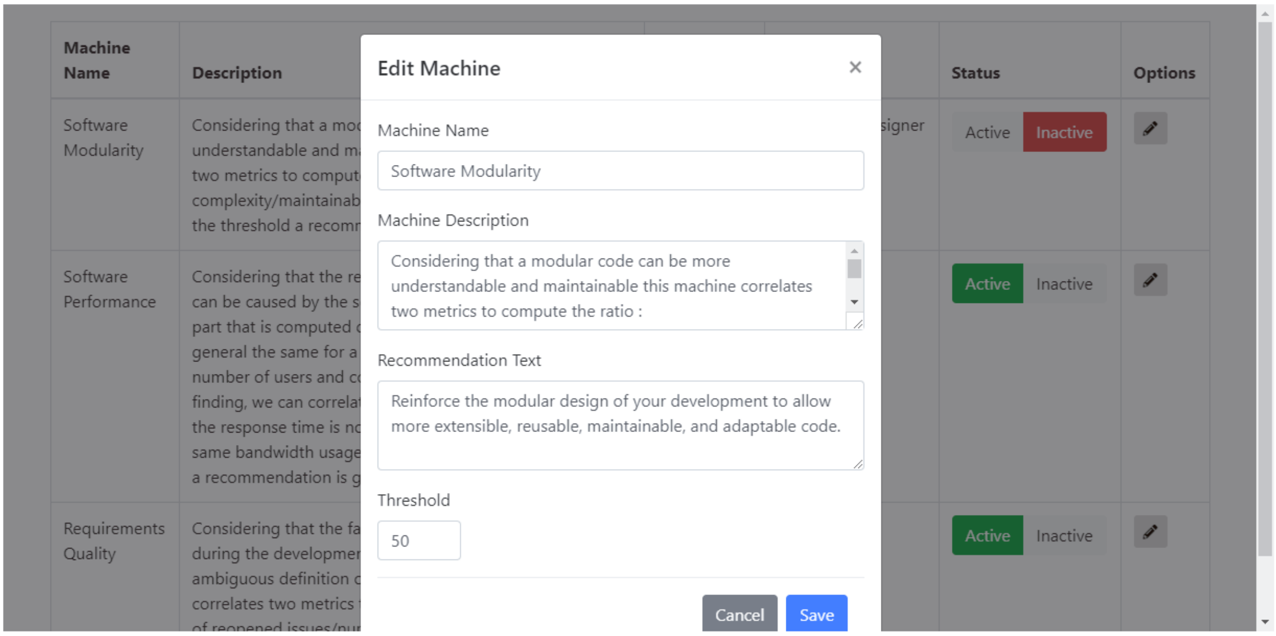
The state of each of the machines can be changed, this determines if the analysis is performed and the corresponding recommendations are received.



Machine Name	Description	Category	Role	Status	Options
Software Modularity	Considering that a modular code can be more understandable and maintainable this machine correlates two metrics to compute the ratio : R = class complexity/maintainability rating If this ratio is more than the threshold a recommendation is given.	Code Quality	Developer/Designer	Active Inactive	✎
Software Performance	Considering that the response time denotes the delay that can be caused by the software, hardware or networking part that is computed during operation. This delay is in general the same for a constant bandwidth (an equivalent number of users and concurrent sessions). Based on this finding, we can correlate the two metrics and compute that the response time is not increasing for during time for the same bandwidth usage. If this response time is increasing, a recommendation is given.	Performance	Operator	Active Inactive	✎

The name, description and text of the recommendation can also be modified, as well as the threshold value (if applicable) as required.

These changes are only applied to the project where the modifications are being made and does not affect the rest.



Edit Machine

Machine Name

Software Modularity

Machine Description

Considering that a modular code can be more understandable and maintainable this machine correlates two metrics to compute the ratio :

Recommendation Text

Reinforce the modular design of your development to allow more extensible, reusable, maintainable, and adaptable code.

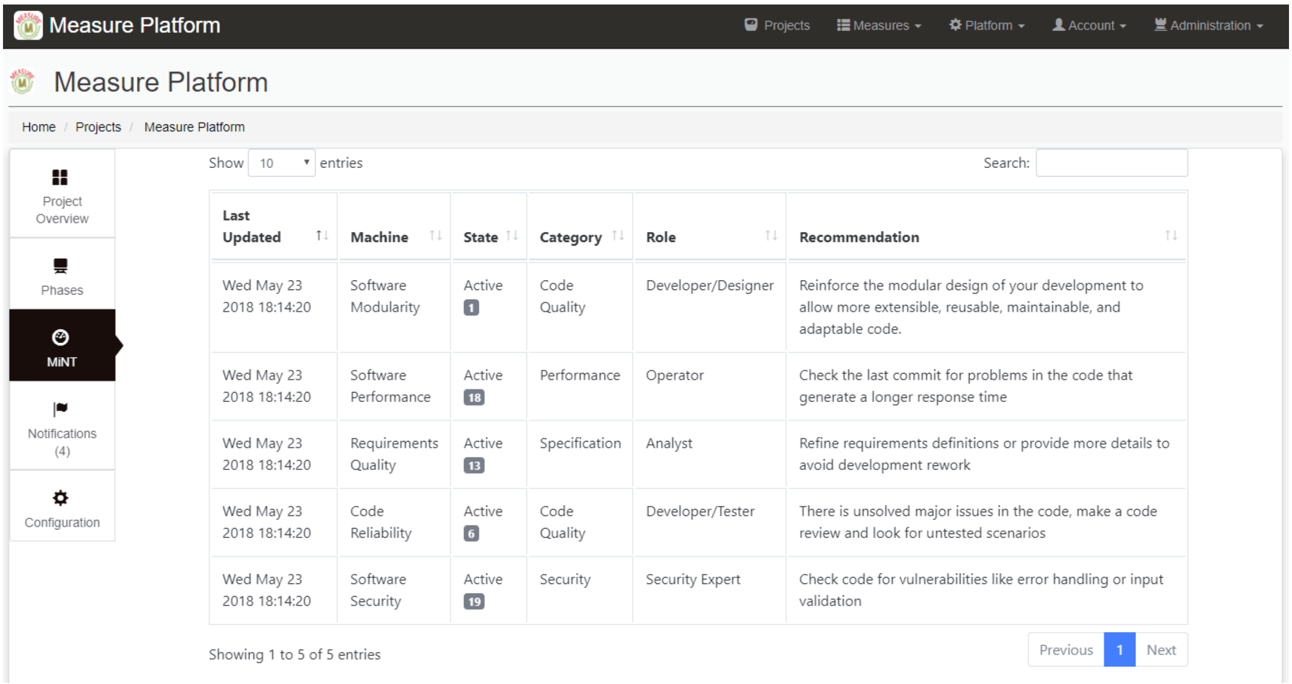
Threshold

50

Visualization

The list of recommendations can be accessed from the Mint page within the project.

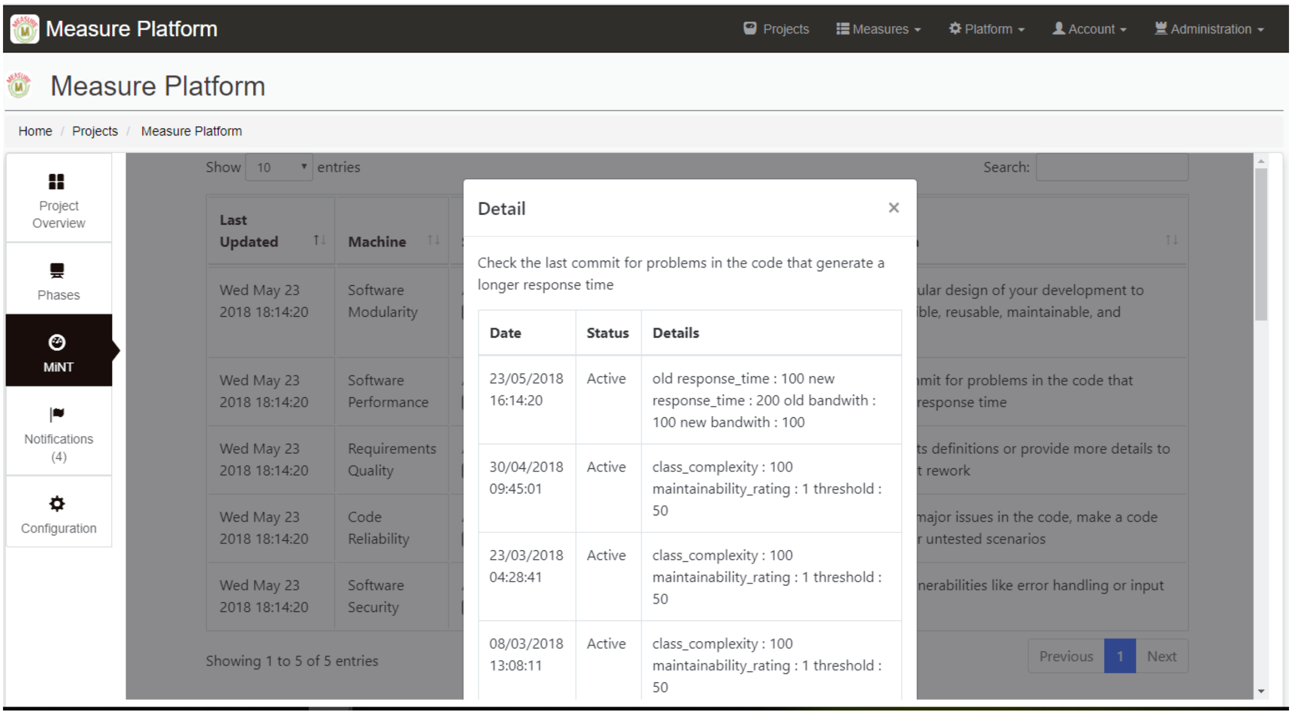
The recommendations are found in a table sorted by date, with the columns last updated, machine name, status of the recommendation (and number of recommendations made), category, role to which the recommendation is directed and recommendation text.



Last Updated	Machine	State	Category	Role	Recommendation
Wed May 23 2018 18:14:20	Software Modularity	Active 1	Code Quality	Developer/Designer	Reinforce the modular design of your development to allow more extensible, reusable, maintainable, and adaptable code.
Wed May 23 2018 18:14:20	Software Performance	Active 18	Performance	Operator	Check the last commit for problems in the code that generate a longer response time
Wed May 23 2018 18:14:20	Requirements Quality	Active 13	Specification	Analyst	Refine requirements definitions or provide more details to avoid development rework
Wed May 23 2018 18:14:20	Code Reliability	Active 6	Code Quality	Developer/Tester	There is unsolved major issues in the code, make a code review and look for untested scenarios
Wed May 23 2018 18:14:20	Software Security	Active 19	Security	Security Expert	Check code for vulnerabilities like error handling or input validation

Showing 1 to 5 of 5 entries

By clicking on any of the recommendations, a model is displayed with the details of each of the recommendations made: date and time, status and details.

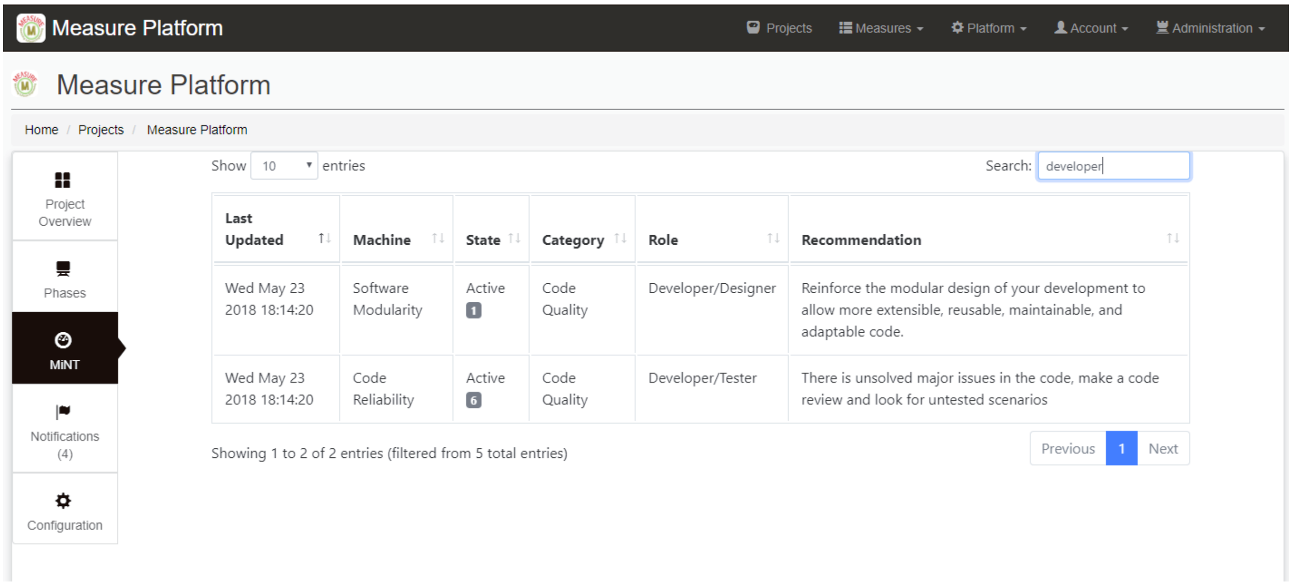


Detail

Check the last commit for problems in the code that generate a longer response time

Date	Status	Details
23/05/2018 16:14:20	Active	old response_time : 100 new response_time : 200 old bandwith : 100 new bandwith : 100
30/04/2018 09:45:01	Active	class_complexity : 100 maintainability_rating : 1 threshold : 50
23/03/2018 04:28:41	Active	class_complexity : 100 maintainability_rating : 1 threshold : 50
08/03/2018 13:08:11	Active	class_complexity : 100 maintainability_rating : 1 threshold : 50

The recommendations table can be ordered by any of its columns and the results can also be filtered by searching for some text.



Last Updated	Machine	State	Category	Role	Recommendation
Wed May 23 2018 18:14:20	Software Modularity	Active 1	Code Quality	Developer/Designer	Reinforce the modular design of your development to allow more extensible, reusable, maintainable, and adaptable code.
Wed May 23 2018 18:14:20	Code Reliability	Active 6	Code Quality	Developer/Tester	There is unsolved major issues in the code, make a code review and look for untested scenarios

Showing 1 to 2 of 2 entries (filtered from 5 total entries)

Annexe K. Quality Guard Installation Guide (Analysis Tool)

Quality Guard Installation Guide

Analysis Tool of MEASURE Platform

Hardware requirements

Below a list of the minimum Hardware requirements to get started with the Quality Guard analysis tool:

- Processor (CPU) with 2 gigahertz (GHz) frequency or above.
- A minimum of 8 GB of RAM.
- A minimum of 20 GB of available space on the hard disk.

Prerequisites

The Quality Guard Analysis tool can be executed on Windows, Linux or Mac OSX systems. To be executed, the tool requires the installation of the following tools:

- MySQL Installation
  - Download MySQL Community Server ver. 5.7 or above: *https://dev.mysql.com/downloads/mysql/*
  - Install MySQL using these instructions: *https://dev.mysql.com/doc/refman/5.7/en/installing.html*
  - Create a new database named "qualityguardanalysis".
- Java 1.8 Installation
  - Download and install the jdk8 in your environment: *http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html*

N.B: The Measure platform is required to start executing the Quality Guard tool.

Quality Guard Tool Installation

To install the Quality Guard tool, you have to:

- Download the last released version of the Quality Guard Analysis tool: *https://github.com/ITEA3-Measure/QualityGuardAnalysis/releases*
- Unzip the project in your tool directory.

Configuration

The tool is parametrized using a property file. This property file has to be put in the same folder of the quality-guard-analysis-{version}.war binary application.

- **General Properties:**

Property	Description	Default value
measure-platform.url	URL of the measure platform	localhost/
analysis-tool.url	URL of the quality guard analysis tool	localhost:8585/#/
analysis-tool.elasticsearch.url	Url of Elasticsearch search engine	127.0.0.1
analysis-tool.elasticsearch.port	Port of Elasticsearch nodes communication	9300
analysis-tool.elasticsearch.cluster-key	CLuster key of Elasticsearch search engine	cluster.name
analysis-tool.elasticsearch.cluster-name	CLuster name of Elasticsearch search engine	elasticsearch

spring.datasource.url	JDBC URL of the database ex: "jdbc:mysql://"+ ip of computer in which is installed MySQL database name.	jdbc:mysql://localhost:3306/qualityguardanalysis
spring.datasource.username	Loign MySQL.	root
spring.datasource.password	Password MySQL.	root
spring.datasource.driver-class-name	Driver JDBC for MySQL	com.mysql.jdbc.Driver
server.port	Port of the Quality Guard Analysis tool web application	8585

Start the Quality Guard tool

- Start MySQL
- The Measure platform must be running
- Start the Quality Guard Analysis tool
    - java -jar quality-guard-analysis-{version}.war
- Check the availability of the Analysis Tool in the Measure Platform

Annexe L. Quality Guard User Guide (Analysis Tool)

Quality Guard User Guide

Analysis Tool of MEASURE Platform

●●

The Quality Guard Analysis overview

The Quality Guard analysis tool is an external extension which is integrated as a component to the Measure Platform through Rest APIs. It provides mechanisms for the advance data analysis functionalities apply to data produced by the continuous measurement applied by the platform.

The Quality Guard Analysis tool provides services for analysing the big data produced by the continuous measurement to enable continuous improvements of software engineering activities and artefacts. In this context, both the QA engineers and the project managers are involved in the quality assurance process of the entire software development activities and artefacts.

The QA engineers would like to define set of constraints and conditions based on measure thresholds to evaluate the data produced by the continuous measurement to ensure an overall product quality. On the other hand, the project manager hopes to be notified when a specific event is occurring on platform side, either by getting the incidents history of all constraints defined by the Quality Guard tool or by monitoring the state of each constraint periodically.

The Quality Guard Analysis features

In this section, we will discuss the main features provided by the Quality Guard tool.

Configuring a quality guard rule

First of all, the Quality Gate tool should be registered into the Measure platform in order to manage and configure the quality guard rules (*more details*).

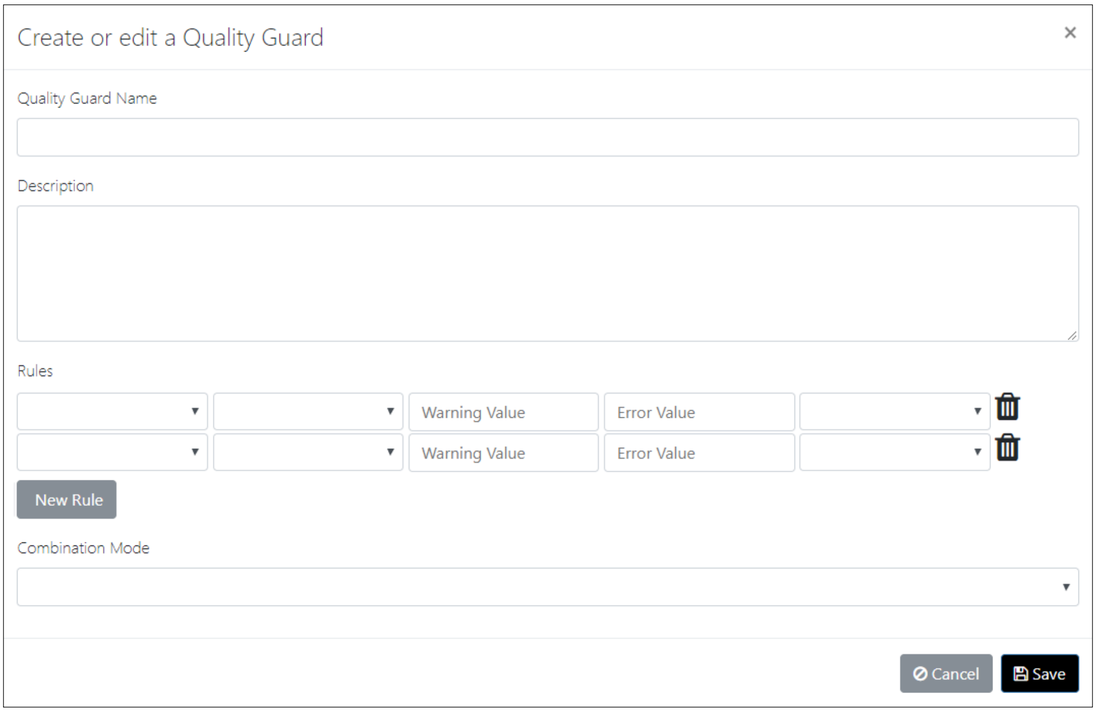
The Project Quality Guard Configuration provides services to manage and configure the quality guard rules set on measures project by allowing the following sub-features:

- Adding new quality guard

A quality guard is an expression with a regular syntax define on measures project to check that a numeric measure stays on delimited range.

Each quality guard can be composed of either one or more guard conditions. By the way, this last constitute of different elements including:

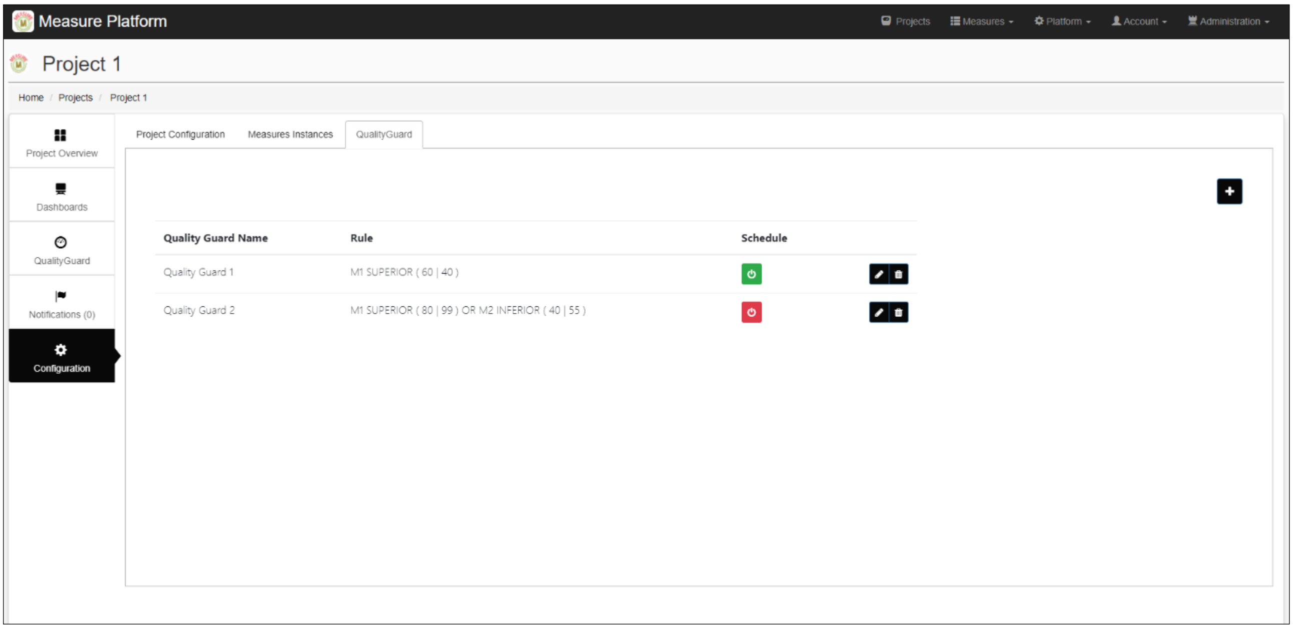
- Measure instance type: which includes a measure instance and its corresponding measure field.
- Guard operators: which include comparison operators like “SUPERIOR” and “INFERIOR” allowing the evaluation of each constraints according to the certain thresholds.
- Thresholds value: which include a warning value field and an error value field.
- Interval aggregation: defined a period used to detect a violation. The value used in guard condition is the average value of all measure value collected during the period.
- Combination mode: which includes logic operators like “AND”, “OR” to support conditions and constraints based on cross measures expressions.



Once the numeric measure evaluated on the guard condition is not on delimited range, which means that the quality guard is not respected. Consequently, a new violation is opened related to this constraint. A Condition violation is the value on one of guard conditions during an incident.

- *Scheduling a registered quality guard*

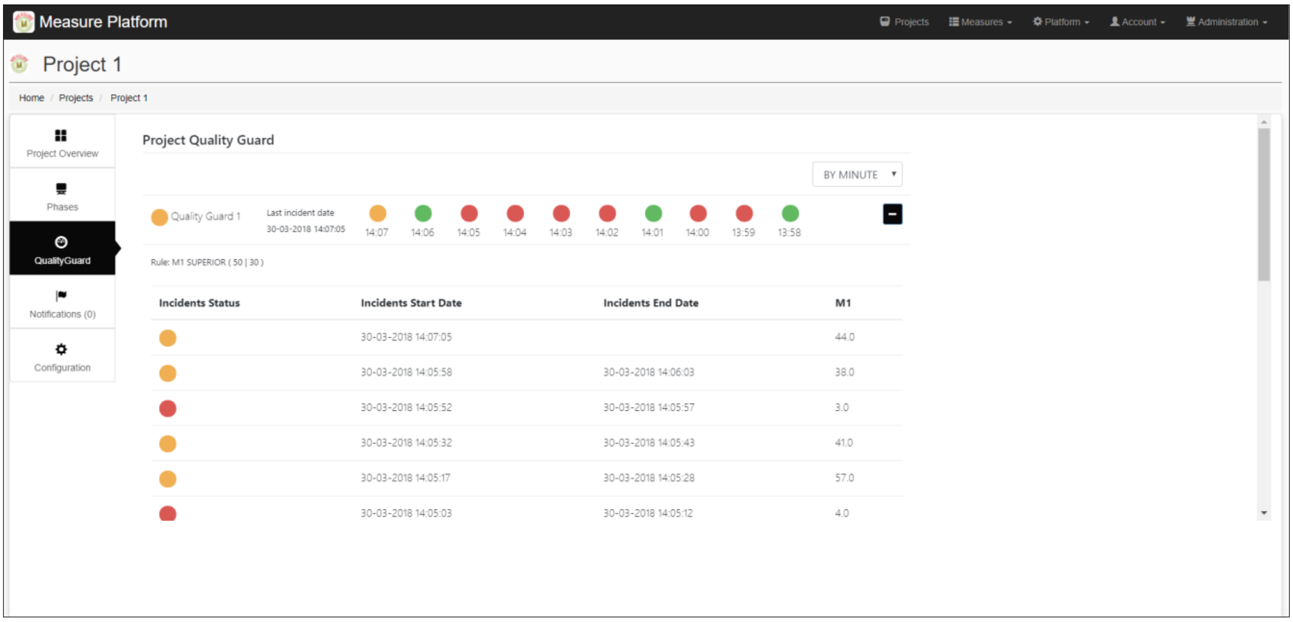
The scheduling mechanism is a way used by activate and deactivate the evaluation of a quality guard rule in order to detect a violation.



Quality Guard Name	Rule	Schedule
Quality Guard 1	M1 SUPERIOR (60 | 40)	🟩
Quality Guard 2	M1 SUPERIOR (80 | 99) OR M2 INFERIOR (40 | 55)	🟥

Visualizing Quality Guards

The main view provided by the Quality Guard Analysis tool allow to visualise the state of each constraints defined by the tool. For each constraint, a history of the last incidents can be visualizing.



Project 1

Home / Projects / Project 1

Project Quality Guard

Quality Guard 1 Last incident date 30-03-2018 14:07:05

Rule: M1 SUPERIOR (50 | 30)

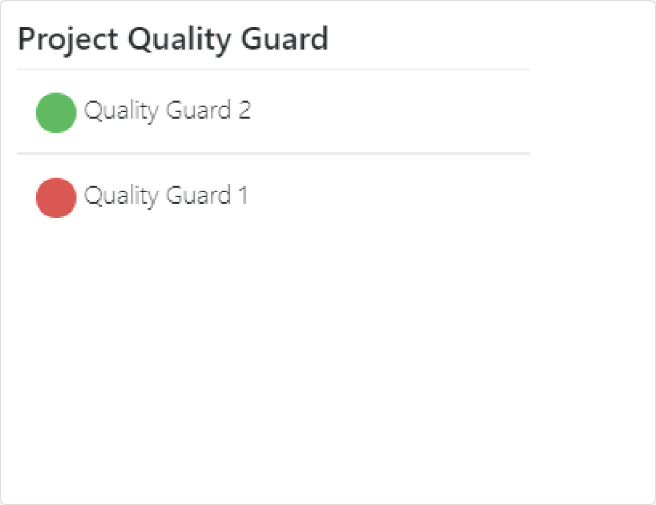
Incidents Status	Incidents Start Date	Incidents End Date	M1
●	30-03-2018 14:07:05		44.0
●	30-03-2018 14:05:58	30-03-2018 14:06:03	38.0
●	30-03-2018 14:05:52	30-03-2018 14:05:57	3.0
●	30-03-2018 14:05:32	30-03-2018 14:05:43	41.0
●	30-03-2018 14:05:17	30-03-2018 14:05:28	57.0
●	30-03-2018 14:05:03	30-03-2018 14:05:12	4.0

Visualizing Dashboard Cards

The dashboard cards provided by the Quality Guard Analysis tool allow getting an overview of the state of either the quality guards or the constraints violations occurred in each project. It composes of the following parts:

- Quality Guard dashboard

The dashboard card presents an overview of the Quality Guards defined in the project.



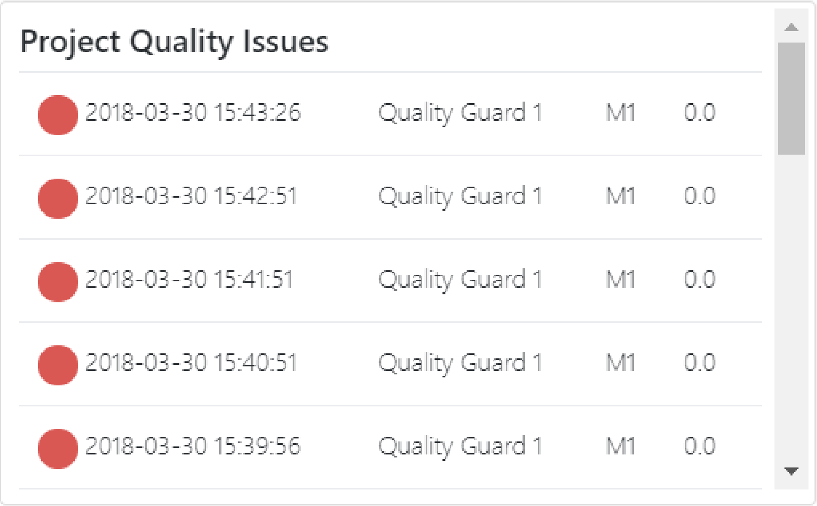
Project Quality Guard

● Quality Guard 2

● Quality Guard 1

- Incident history dashboard

The dashboard card exposes the last constraints violations which occurred in the project.



Project Quality Issues

●	2018-03-30 15:43:26	Quality Guard 1	M1	0.0
●	2018-03-30 15:42:51	Quality Guard 1	M1	0.0
●	2018-03-30 15:41:51	Quality Guard 1	M1	0.0
●	2018-03-30 15:40:51	Quality Guard 1	M1	0.0
●	2018-03-30 15:39:56	Quality Guard 1	M1	0.0

Annexe M. Metric Suggester Installation Guide (Analysis Tool)

Metric Suggester Installation Guide

Analysis Tool of MEASURE Platform

Hardware and System Requirements

The Metric Suggester Analysis Tool for Measure platform must be deployed on an Ubuntu Server 16 or greater.

Prerequisites

Installation of redis-server

```
> sudo apt-get install redis-server
```

Installation of PostgreSQL Database

- Installation of PostgreSQL : sudo apt-get install libpq-dev postgresql postgresql-contrib
- Configuration of PostgreSQL :

Creation of a progress User

```
> sudo -i -u postgres

> psql

> CREATE USER suggesterprojectuser;

> ALTER ROLE suggesterprojectuser WITH CREATEDB;

> ALTER USER suggesterprojectuser WITH ENCRYPTED PASSWORD '#aMY@aj870';
```

Configuration of the Database

```
CREATE DATABASE suggesterproject OWNER suggesterprojectuser;
```

Installation of Python

```
> sudo apt-get install python-dev python-pip nginx

> sudo pip install virtualenv :

> sudo apt-get install python-requests
```

Installation of Git

```
> sudo apt-get install git
```

Installation of Metric Suggester

1. Clone the repository of the tool

```
> git clone https://github.com/sasdahab/Suggester_tool.git
```

2. Set Up Python Virtualenv

```
> cd Suggester_tool/

> virtualenv suggesterenv
```

3. Activate the Virtual Environnement

```
> source suggesterenv/bin/activate
```

4. Installer Gunicorn, Django, Celery, Redis and other dependencies

```
> pip install requests

> pip install -r requirements.txt

> add the public IP address in the variable ALLOWED_HOSTS in the file
suggester_tool/suggesterproject/setting.py
```

5. Configure the Database

```
> ./manage.py makemigrations

> ./manage.py migrate

> ./manage.py

> ./manage.py createcachetable
```

6. Configure Gunicorne

```
> gunicorn --bind 127.0.0.1:8000 suggesterproject.wsgi:application
```

Start the Metric Suggester Tool

- Start Redis :

  ```
  > sudo service redis start
  ```

- Start postgresql :

  ```
  > sudo service postgresql start
  ```

- Start Metric Suggester

```
> cd suggester_tool

> source suggesterenv/bin/activate

> ./manage.py runserver localhost:8000 (pour une utilisation local)

> ./manage.py runserver 0:8000 (pour une utilisation non local)
```

The Metrics Suggester tool overview

The Metrics Suggester tool is an external analysis tool which is integrated as a component to the Measure Platform through Rest APIs. It provides mechanisms for the advance data analysis functionalities apply to data produced by the continuous measurement applied by the platform.

This tool train a classifier according to a defined measurement plan which described the executed metrics, the software properties observed by this metrics and the link between both latter. Then, it analyses the measurement data through a trained classifier to suggest a new measurement plan.

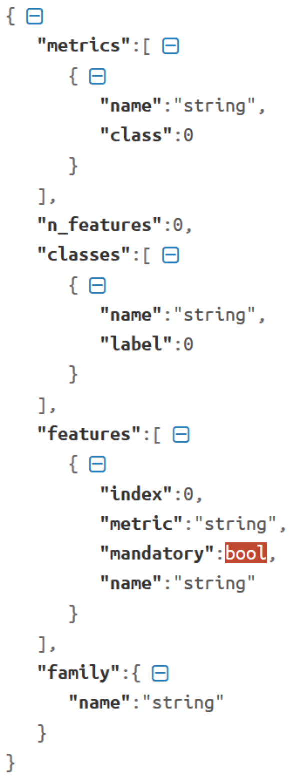
The Metrics Suggester tool features

In this section, we will present the different steps to generate a suggestion.

Measurement Plan configuration

First of all, we should initialize the measurement plan which defines the classes (software properties evaluated), the metrics and the link between the classes and the metrics. The mandatory metrics if there is by a Boolean value and the place of each metric in the training file through the “index” field. This configuration should be done through json file as the model below cons and uploaded into the tool according to the detailed steps below

- Click on Settings tab >> plan form
- copy past the contained of the mp in json format >> create
- Click on Measurement Plans tab to display the MP details



```
{ ⊟
   "metrics":[ ⊟
      { ⊟
         "name":"string",
         "class":0
      }
   ],
   "n_features":0,
   "classes":[ ⊟
      { ⊟
         "name":"string",
         "label":0
      }
   ],
   "features":[ ⊟
      { ⊟
         "index":0,
         "metric":"string",
         "mandatory":bool,
         "name":"string"
      }
   ],
   "family":{ ⊟
      "name":"string"
   }
}
```

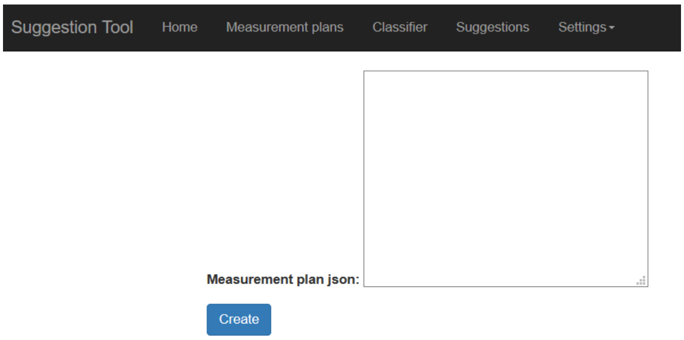
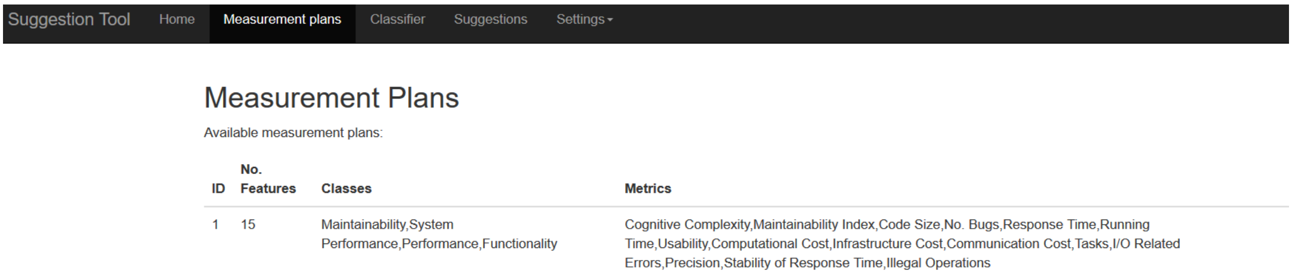


Figure 7 Section to add the measurement plan configuration

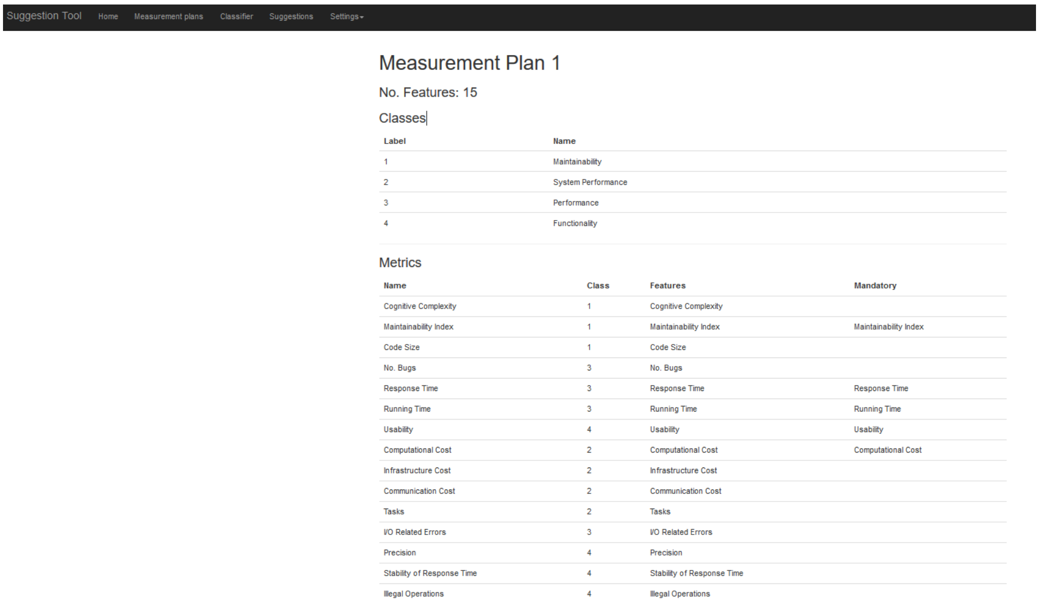


Measurement Plans

Available measurement plans:

ID	No. Features	Classes	Metrics
1	15	Maintainability,System Performance,Performance,Functionality	Cognitive Complexity,Maintainability Index,Code Size,No. Bugs,Response Time,Running Time,Usability,Computational Cost,Infrastructure Cost,Communication Cost,Tasks,I/O Related Errors,Precision,Stability of Response Time,Illegal Operations

Figure 8 Measurement plan



Measurement Plan 1

No. Features: 15

Classes

Label	Name
1	Maintainability
2	System Performance
3	Performance
4	Functionality

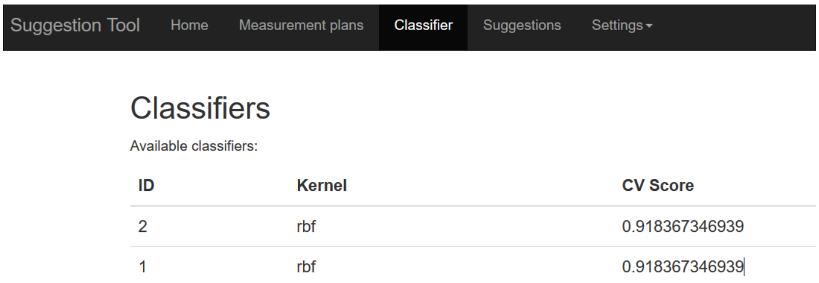
Metrics

Name	Class	Features	Mandatory
Cognitive Complexity	1	Cognitive Complexity	
Maintainability Index	1	Maintainability Index	Maintainability Index
Code Size	1	Code Size	
No. Bugs	3	No. Bugs	
Response Time	3	Response Time	Response Time
Running Time	3	Running Time	Running Time
Usability	4	Usability	Usability
Computational Cost	2	Computational Cost	Computational Cost
Infrastructure Cost	2	Infrastructure Cost	
Communication Cost	2	Communication Cost	
Tasks	2	Tasks	
I/O Related Errors	3	I/O Related Errors	
Precision	4	Precision	
Stability of Response Time	4	Stability of Response Time	
Illegal Operations	4	Illegal Operations	

Figure 9 Measurement plan detailed

Classifier training from the initial MP

- Click on Settings tab >> classifier form
- insert the training file >> upload
- Click on Classifier tab to display the classifiers list
- Click on one classifier to display the training details

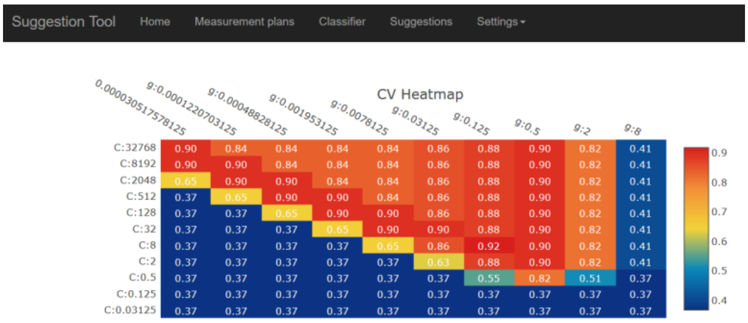


Classifiers

Available classifiers:

ID	Kernel	CV Score
2	rbf	0.918367346939
1	rbf	0.918367346939

Figure 10 Classifiers list



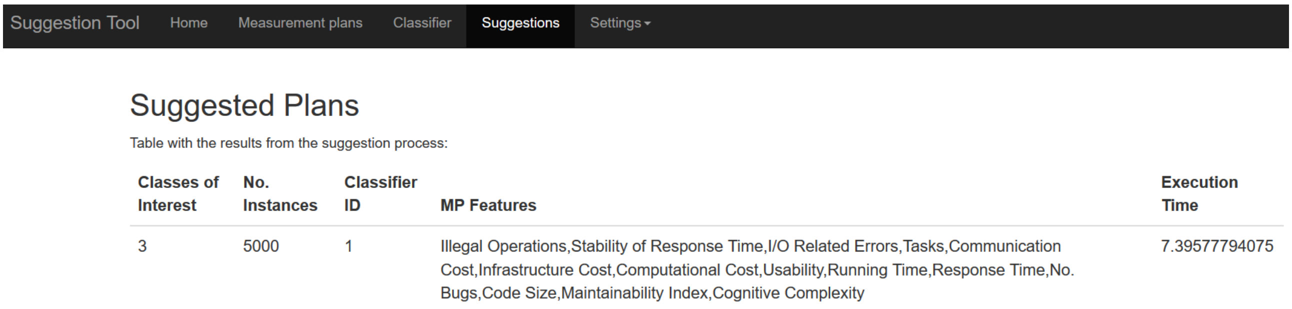
CV Heatmap

	0.000030517578125	g:0.0001220703125	g:0.00048828125	g:0.001953125	g:0.0078125	g:0.03125	g:0.125	g:0.5	g:2	g:8
C:32768	0.90	0.84	0.84	0.84	0.84	0.86	0.88	0.90	0.82	0.41
C:8192	0.90	0.90	0.84	0.84	0.84	0.86	0.88	0.90	0.82	0.41
C:2048	0.65	0.90	0.90	0.84	0.84	0.86	0.88	0.90	0.82	0.41
C:512	0.37	0.65	0.90	0.90	0.84	0.86	0.88	0.90	0.82	0.41
C:128	0.37	0.37	0.65	0.90	0.90	0.86	0.88	0.90	0.82	0.41
C:32	0.37	0.37	0.37	0.65	0.90	0.90	0.88	0.90	0.82	0.41
C:8	0.37	0.37	0.37	0.37	0.65	0.86	0.92	0.90	0.82	0.41
C:2	0.37	0.37	0.37	0.37	0.37	0.63	0.88	0.90	0.82	0.41
C:0.5	0.37	0.37	0.37	0.37	0.37	0.37	0.55	0.82	0.51	0.37
C:0.125	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37
C:0.03125	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37	0.37

Figure 11 Classifier details

Suggestions

- Click on Settings tab >> Suggestion form
- select the last MP in the list
- Select the first classifier in the list
- insert the file with the measurement to be analyzed >> ok
- Click on Suggestions tab to display the suggestion result



Suggested Plans

Table with the results from the suggestion process:

Classes of Interest	No. Instances	Classifier ID	MP Features	Execution Time
3	5000	1	Illegal Operations,Stability of Response Time,I/O Related Errors,Tasks,Communication Cost,Infrastructure Cost,Computational Cost,Usability,Running Time,Response Time,No. Bugs,Code Size,Maintainability Index,Cognitive Complexity	7.39577794075

Annexe N. M·ELKI User Guide (Analysis Tool)

M·ELKI User Guide

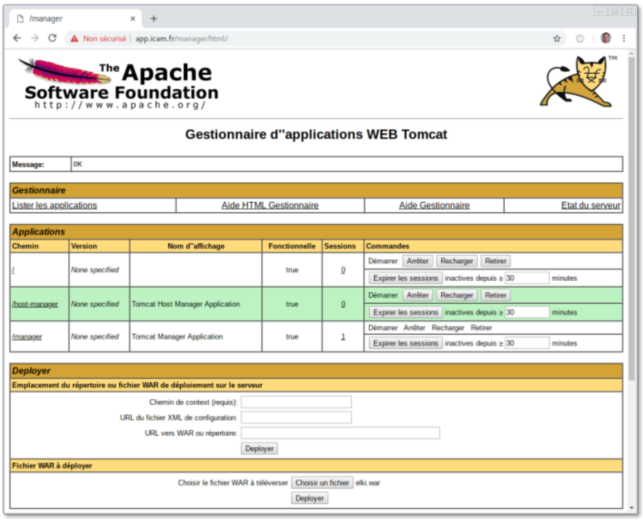
Analysis Tool of MEASURE Platform

M·ELKI is a set of web services that make possible to run clustering algorithms from projects hosted on the MEASURE Platform.

Installation

M·ELKI installation is very easy. It merely consists in uploading a web application archive (WAR file) on a Java Servlet Container such as Apache Tomcat, Apache TomEE, Apache Jetty, Oracle GlassFish, RedHat Jboss, IBM Websphere.

The following figures show such a deployment of the M·ELKI web app archive:


/manager
Non sécurisé | app.icam.fr/manager/html/
The Apache Software Foundation
http://www.apache.org/

Gestionnaire d''applications WEB Tomcat

Message:	OK

Gestionnaire

Lister les applications	Aide HTML Gestionnaire	Aide Gestionnaire	Etat du serveur

Applications

Chemin	Version	Nom d''affichage	Fonctionnelle	Sessions	Commandes
/	*None specified*		true	0	Démarrer Arrêter Recharger Retirer Expirer les sessions inactives depuis ≥ 30 minutes
/host-manager	*None specified*	Tomcat Host Manager Application	true	0	Démarrer Arrêter Recharger Retirer Expirer les sessions inactives depuis ≥ 30 minutes
/manager	*None specified*	Tomcat Manager Application	true	1	Démarrer Arrêter Recharger Retirer Expirer les sessions inactives depuis ≥ 30 minutes

Deployer

Emplacement du répertoire ou fichier WAR de déploiement sur le serveur

Chemin de context (requis):

URL du fichier XML de configuration:

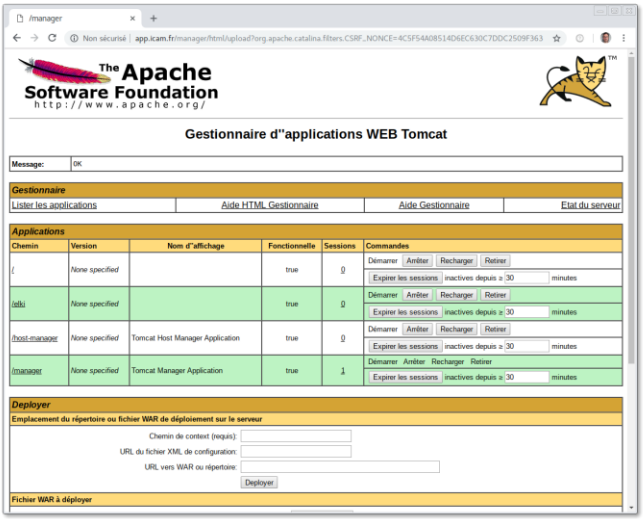
URL vers WAR ou répertoire:

Deployer

Fichier WAR à déployer

Choisir le fichier WAR à téléverser Choisir un fichier elki.war

Deployer


/manager
Non sécurisé | app.icam.fr/manager/html/upload?org.apache.catalina.filters.CSRF_NONCE=4C5F54A08514D6EC630C7DDC2509F363
The Apache Software Foundation
http://www.apache.org/

Gestionnaire d''applications WEB Tomcat

Message:	OK

Gestionnaire

Lister les applications	Aide HTML Gestionnaire	Aide Gestionnaire	Etat du serveur

Applications

Chemin	Version	Nom d''affichage	Fonctionnelle	Sessions	Commandes
/	*None specified*		true	0	Démarrer Arrêter Recharger Retirer Expirer les sessions inactives depuis ≥ 30 minutes
/elki	*None specified*		true	0	Démarrer Arrêter Recharger Retirer Expirer les sessions inactives depuis ≥ 30 minutes
/host-manager	*None specified*	Tomcat Host Manager Application	true	0	Démarrer Arrêter Recharger Retirer Expirer les sessions inactives depuis ≥ 30 minutes
/manager	*None specified*	Tomcat Manager Application	true	1	Démarrer Arrêter Recharger Retirer Expirer les sessions inactives depuis ≥ 30 minutes

Deployer

Emplacement du répertoire ou fichier WAR de déploiement sur le serveur

Chemin de context (requis):

URL du fichier XML de configuration:

URL vers WAR ou répertoire:

Deployer

Fichier WAR à déployer

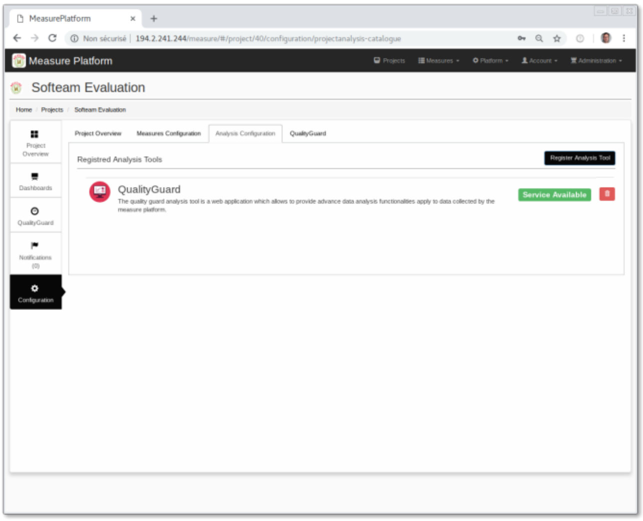
There exists a second installation procedure of M·ELKI that consists in retrieving its source from its GitHub repository and in deploying it programmatically thanks to Maven by the means of the following commands:

- git clone https://github.com/ITEA3-Measure/M-ELKI.git
- mvn clean compile package install tomcat7:redeploy

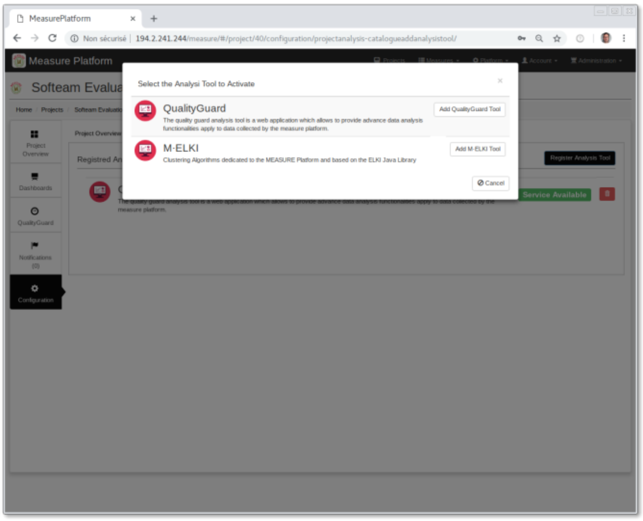
That's it! You merely have to update the correct settings into the different Maven project configuration descriptions (pom.xml files) in order to host M·ELKI onto a custom and dedictaed server. However, a default M·ELKI instance already runs on Icam servers.

Registration

Once deployed, the M·ELKI analysis tool automatically registered itself on the MEASURE Platform. However, it remains to the users to apply M·ELKI to their targeted projects. The following pictures show how to do such a registration: it is straightforward as users only have to select the M·ELKI tool.

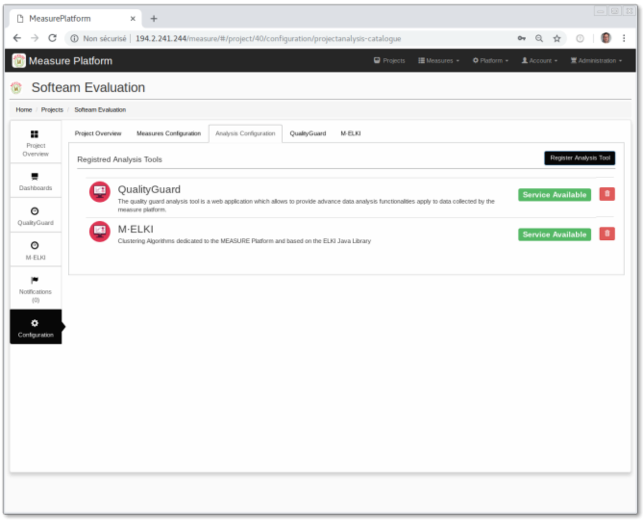


The quality guard analysis tool is a web application which allows to provide advance data analysis functionalities apply to data collected by the measure platform.


MeasurePlatform
Non sécurisé | 194.2.241.244/measure/#/project/40/configuration/projectanalysis-catalogueaddanalysistool/
Measure Platform
Select the Analysi Tool to Activate
QualityGuard
Add QualityGuard Tool
M·ELKI
Add M·ELKI Tool
Cancel

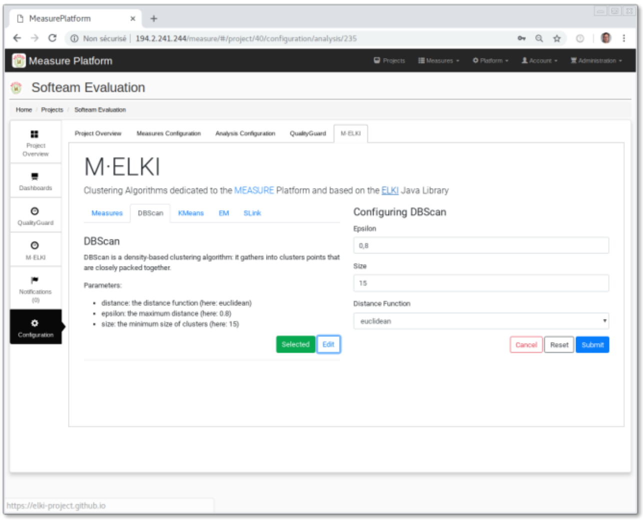
The quality guard analysis tool is a web application which allows to provide advance data analysis functionalities apply to data collected by the measure platform.

Clustering Algorithms dedicated to the MEASURE Platform and based on the ELKI Java Library



The quality guard analysis tool is a web application which allows to provide advance data analysis functionalities apply to data collected by the measure platform.

Clustering Algorithms dedicated to the MEASURE Platform and based on the ELKI Java Library



Clustering Algorithms dedicated to the MEASURE Platform and based on the ELKI Java Library

DBScan

DBScan is a density-based clustering algorithm: it gathers into clusters points that are closely packed together.

Parameters:

- distance: the distance function (here: euclidean)
- epsilon: the maximum distance (here: 0.8)
- size: the minimum size of clusters (here: 15)

Configuring DBScan

Epsilon: 0,8

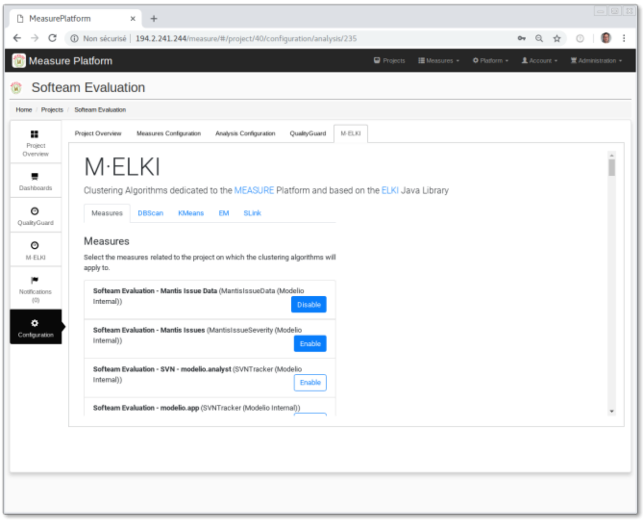
Size: 15

Distance Function: euclidean

Configuration

The M·ELKI project instance is also asily configurable. There is two kinds of settings:

1. the selection and parametrization of a clustering algorithm among 4 algorithms (DB SCAN, K MEANS, EM and SLINK, see picture above);
2. the selection of the project-related measures whose measurements will be processed by the select clustring algorithm (see picture below).



Clustering Algorithms dedicated to the MEASURE Platform and based on the ELKI Java Library

Measures

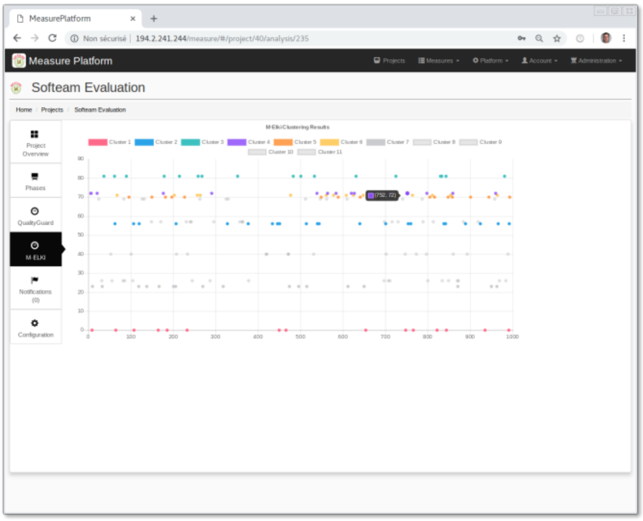
Select the measures related to the project on which the clustering algorithms will apply to.

Softeam Evaluation - Mantis Issue Data (MantisIssueData (Modelio Internal))

Softeam Evaluation - Mantis Issues (MantisIssueSeverity (Modelio Internal))

Softeam Evaluation - SVN - modelio.analyst (SVNTracker (Modelio Internal))

Softeam Evaluation - modelio.app (SVNTracker (Modelio Internal))



Visualization

The last picture show how to visualize M·ELKI clustering analysis results. The latter are also backed into the MEASURE Platform ElasticSearch database.

Annexe O. Stracker Installation Guide (Analysis Tool)

Stracker Installation Guide

Analysis Tool of MEASURE Platform

Prerequisites

In order to install Stracker locally on a machine (i.e. independent of the MEASURE Platform), the following must be installed:

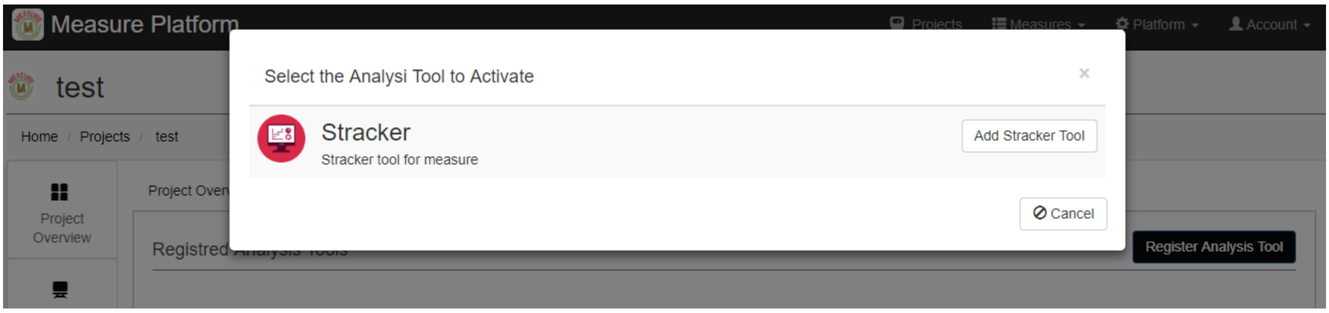
- **Python3** (lhttps://www.python.org/downloads)
- **Python libraries:** flask, pygal, numpy, pandas, matplotlib, sklearn, statsmodels
- **Elasticsearch** (https://www.elastic.co/downloads/elasticsearch)
- **Sonarqube** (https://www.sonarqube.org)

Then, download **Stracke**r from git: https://github.com/CostiCTI/Stracker

Start the program:

- Start Elasticsearch
- Start Sonarqube
- Start Stracker: in the folder *Stracker*, start app.py (*python app.py*)
- The application will run on http://localhost:5000

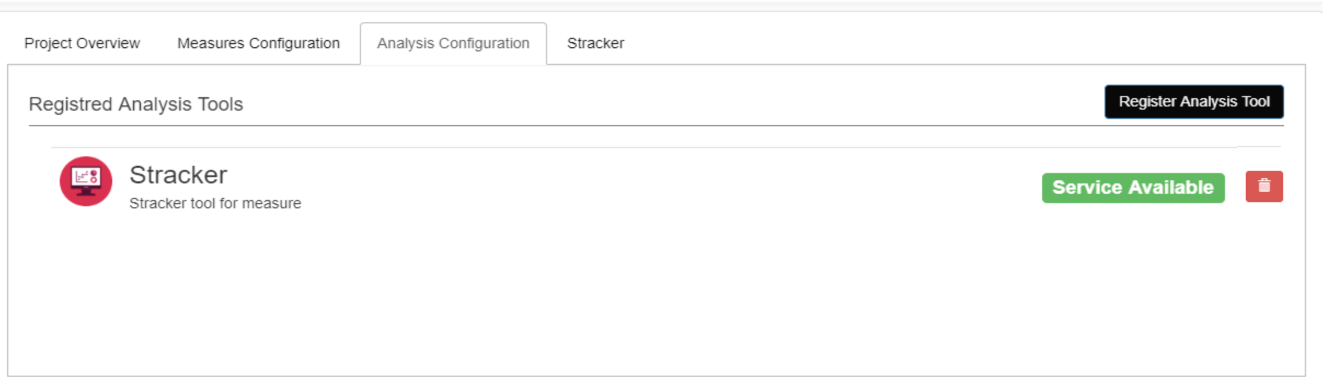
Finally, if the user wants to use the integrated version into MEASURE platform, the process is straightforward, as depicted in the following screenshots:



Select the Analysi Tool to Activate

Stracker

Stracker tool for measure



Registred Analysis Tools

Stracker

Stracker tool for measure

Annexe P. Stracker User Guide (Analysis Tool)

Stracker User Guide

Analysis Tool of MEASURE Platform

It is important to note upfront that this is the description of the first version of the tool. There will be a new major version released in May 2019, that will contain many improvements both from front- and back-end point of view.

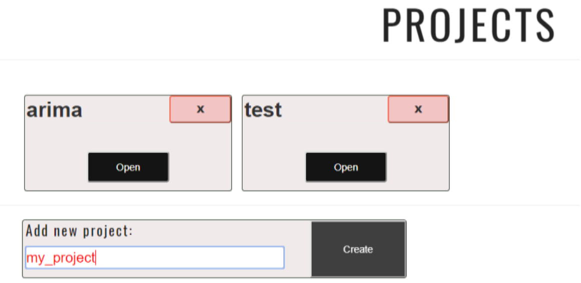
Stracker can be used on a stand-alone basis, but also integrated into the MEASURE platform, which can analyze stored metrics by communicating through APIs with the other components of the platform.

Stracker is a web application whose goal is to increase the quality of software development by tracking and suggesting values for numerous software metrics during the development process. Specifically, it helps to verify metric values using various graphical representations and provides scores for each new record. It also includes a module predicting future metrics based on the historical values recorded so far.

Create a new project

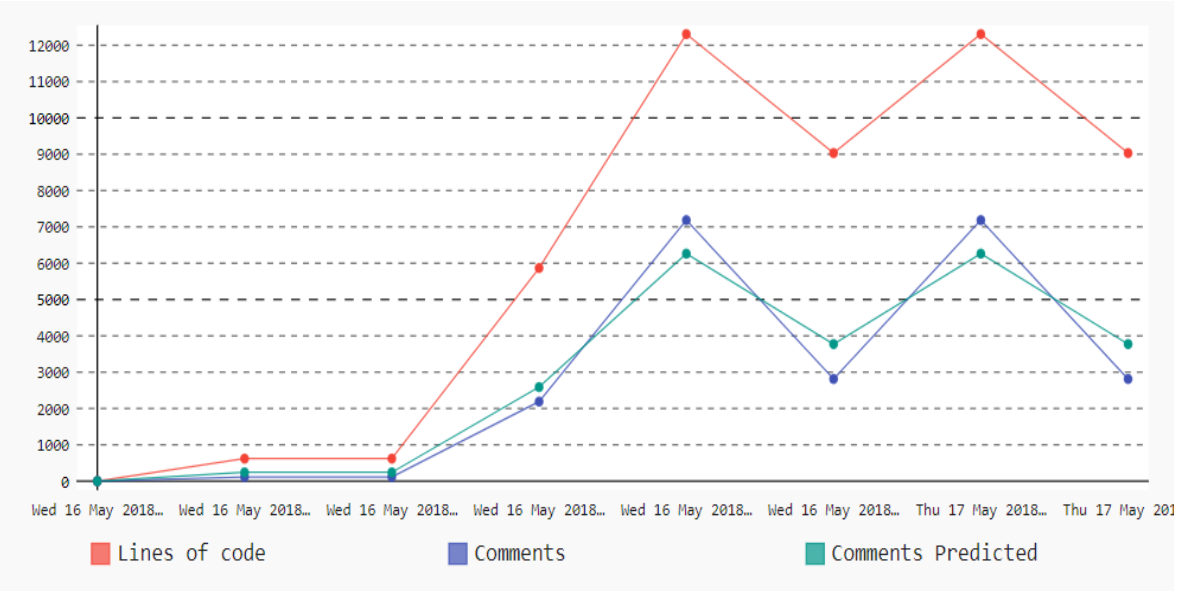
To create a new project or to enter an existing project in the menu, click on the Projects tag.

On the *new* page, go down to *the Add new project* box and write the name of the project that we want to start with. The name must be the same as the name of the project that we import from *SonarQube.*



Add new project: my_project

For **prediction,** we give two examples:

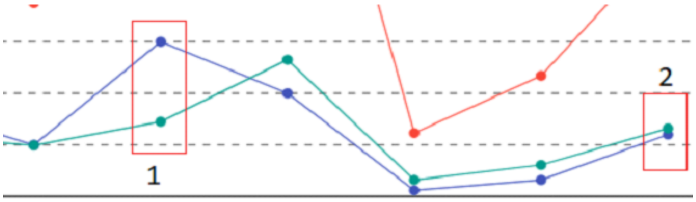


- The red line shows the total number of project code lines

- The blue line shows the total number of comment lines of the project

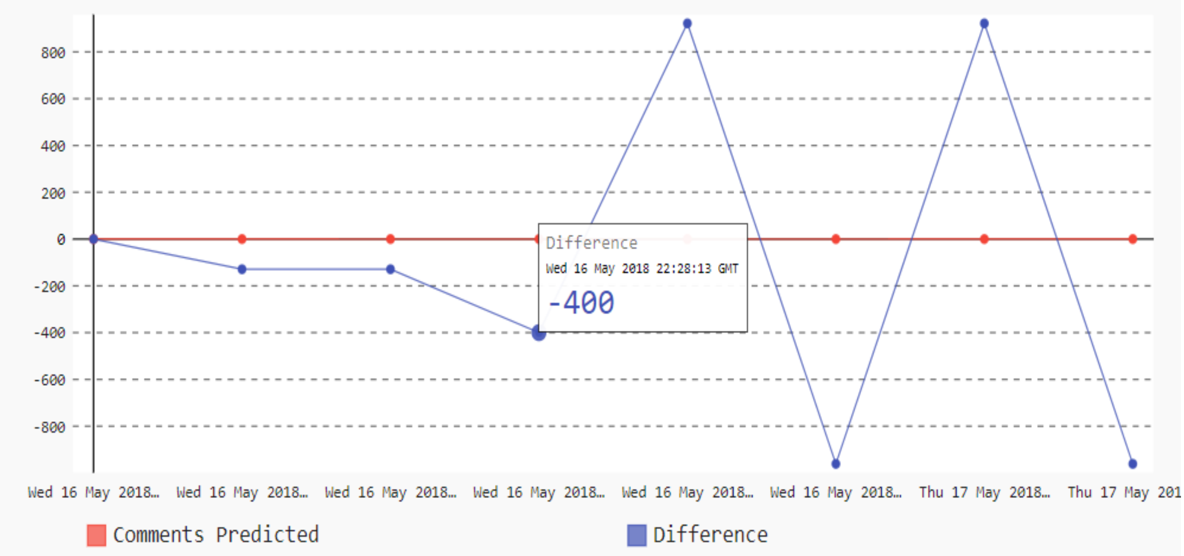
- The green line shows the prediction of the number of draft comments lines

For this metric, the goal is to make the difference between the blue line and the green line as small as possible, i.e., the number of comment lines of the project being as close as possible to the number of predicted comment lines using regression algorithms.



In the first case (1), the metric value is smaller than in (2) where the metric value and the predicted value are almost similar.

Another type of output graphics:



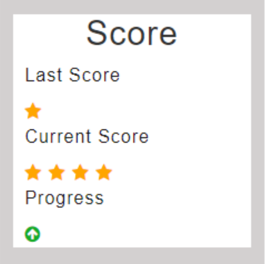
Here you can see the difference between the number of comment lines of the project and the number of predicted comment lines.

If we hover over a point, we can see the exact metric at that point. After each new record we can see if the new metric values are better than the values of the previous record, checking the score. To see the score, click the Track tag and go to the Score section.

Current score is the current record score, calculated based on the difference between our metric value and the predicted value. Last Score is the penultimate recording added (the one before the current record)

We say that progress is positive, if the current score is better than last score and negative otherwise.

Ideally, each measurement should be better than the previous one. If we get a small score for a certain metric, we can track the metrics that led to that score in the two graphs and try to bring those values as close as possible to the predictions made by the app.



Score

Last Score

★

Current Score

★ ★ ★ ★

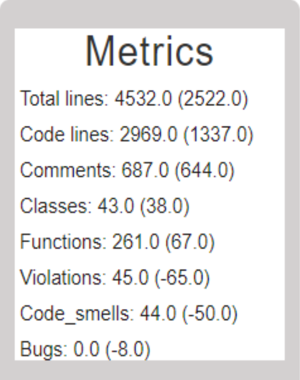
Progress

⬆

For each metric suggestion, the score is calculated differently. For example, if for no. of lines metric, the score is better as the value approaches the predicted value, for the (violations - minor violations) metric, the number of minor violations must exceed the predicted limit as much as possible because a large number of them reduce the number of major, critical, or blocking violations.

In the Metrics section, you can see the numeric values of the metrics as well as the difference from the last record.

For example, the picture shows that the project now contains 4532 lines, with 2522 more than the previous measured record. In the case of the code metrics the code smells metric, the number decreased by 50, reaching 44.

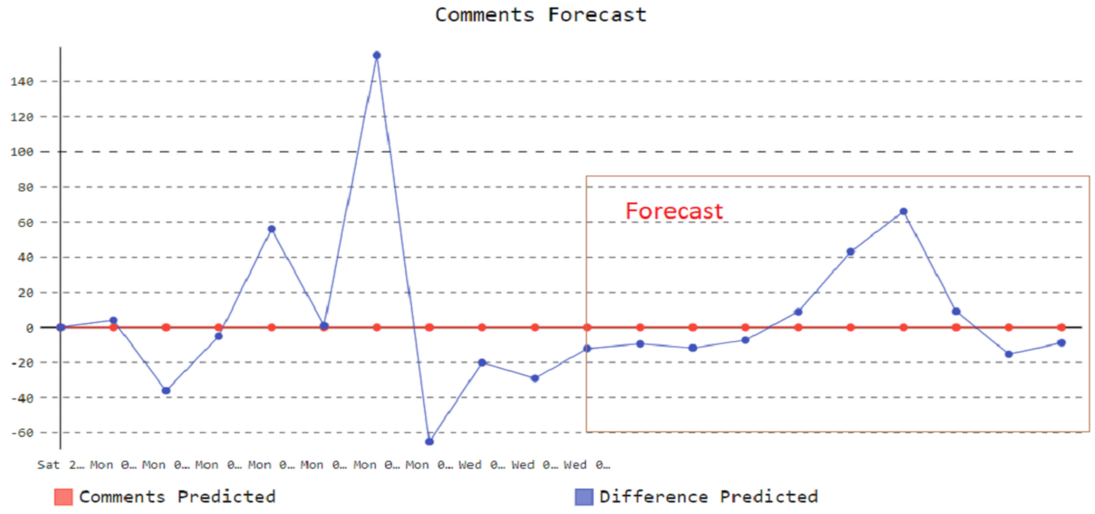


Metrics

Total lines: 4532.0 (2522.0)
Code lines: 2969.0 (1337.0)
Comments: 687.0 (644.0)
Classes: 43.0 (38.0)
Functions: 261.0 (67.0)
Violations: 45.0 (-65.0)
Code_smells: 44.0 (-50.0)
Bugs: 0.0 (-8.0)

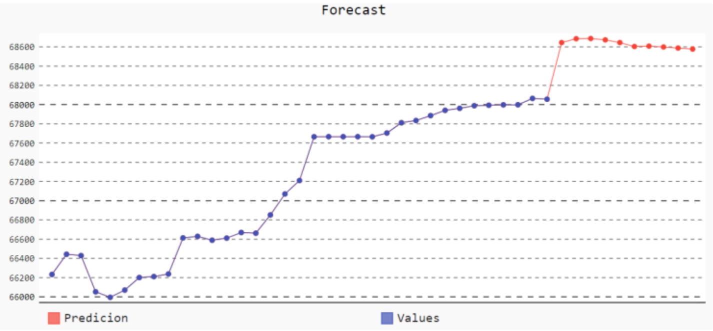
Forecasting metrics

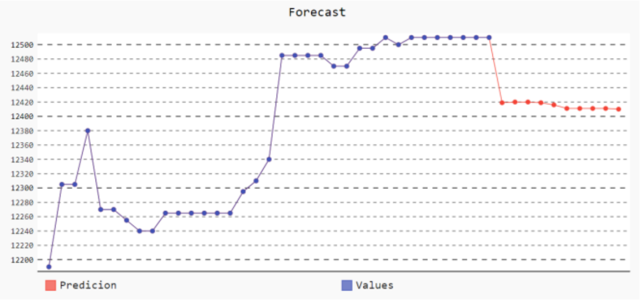
If we press the Forecasting tag, we can see a graph of the predicted values for our metrics. The classic algorithm ARIMA was used for forecasting.



For an efficient prediction, the number of historical data on which to calculate future data values should be as large as possible.

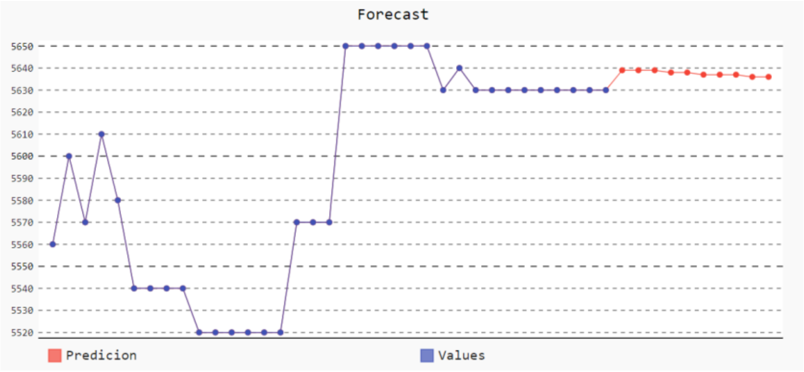
Last but not least, Stracker awakened the interest of the other partners such as Softeam and Bitdefender. For example, Softeam wanted to get certain metric forecasting. The figures below show the data forecasted by Stracker based on production data for Softeam.





Forecasting for an architectural metric

Forecasting for a bug remediation effort metric



Forecasting for a security metric