



D4.3 Decision Support tools I

Document Identification			
Status	Final	Due Date	31/07/2021
Version	1.0	Submission Date	30/07/2021

Related WP	WP4	Document Reference	D4.3
Related Deliverable(s)	D2.2, D2.3, D3.1, D3.2, D3.3	Dissemination Level (*)	PU
Lead Participant	ENG	Lead Author	Francesco Iadanza
Contributors		Reviewers	Estela Carmona (i2CAT)
			Verena Stanzl (FILL)

Keywords:
DSS, SLA, guarantee, prototype, architecture, demo

This document is issued within the frame and for the purpose of the PLEDGER project. This project has received funding from the European Union's Horizon2020 Framework Programme under Grant Agreement No. 871536. The opinions expressed and arguments employed herein do not necessarily reflect the official views of the European Commission.

[The dissemination of this document reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains. This deliverable is subject to final acceptance by the European Commission.

This document and its content are the property of the PLEDGER Consortium. The content of all or parts of this document can be used and distributed provided that the PLEDGER project and the document are properly referenced.

Each PLEDGER Partner may use this document in conformity with the PLEDGER Consortium Grant Agreement provisions.

(*) Dissemination level: **PU**: Public, fully open

Document Information

List of Contributors	
Name	Partner
Gabriele Giammatteo	ENG
Francesco Iadanza	ENG
Keven Kearney	ENG

Document History			
Version	Date	Change editors	Changes
0.1	26/03/2021	Francesco Iadanza (ENG)	ToC and first data model release
0.2	26/04/2021	Francesco Iadanza (ENG)	Reviewed data model, high-level architecture
0.3	14/05/2021	Francesco Iadanza (ENG)	Finalised data model
0.4	04/06/2021	Francesco Iadanza (ENG)	Structure aligned to WP3/4 deliverables. All content in place, to be finalised.
0.5	08/07/2021	Francesco Iadanza (ENG)	Ready for the internal review.
0.6	15/07/2021	Estela Carmona (i2CAT)	Internal review I
0.7	15/07/2021	Verena Stanzl (FILL)	Internal review II
0.8	16/07/2021	Francesco Iadanza (ENG)	Ready for quality review
0.9	28/07/2021	Carmen San roman (ATOS)	Quality assurance review
1.0	30/07/2021	Lara Lopez (ATOS)	Final version to be submitted

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable leader	Francesco Iadanza (ENG)	16/07/2021
Quality manager	Carmen San roman (ATOS)	28/07/2021
Project Coordinator	Lara Lopez (ATOS)	30/07/2021

Document name:	D4.3 Decision Support tools I			Page:	2 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0
				Status:	Final

Table of Contents

Document Information	2
Table of Contents	3
List of Tables.....	4
List of Figures	5
List of Acronyms.....	6
Executive Summary	7
1 Introduction	8
1.1 Purpose of the document.....	8
1.2 Relation to the other project work.....	8
1.3 Structure of the document.....	8
2 Functional description	9
3 Technical description.....	14
3.1 Baseline technologies and dependencies	14
3.2 Components Architecture.....	14
3.3 Interfaces provided.....	15
3.4 Data models.....	16
3.5 REST API	19
4 Installation and usage guides	20
4.1 Requirements	20
4.2 Installation.....	20
4.3 Usage.....	20
4.4 Licenses.....	20
4.5 Source code repository.....	20
5 Demonstration	21
5.1 Scenario description.....	21
5.2 Validation and Verification.....	22
5.3 Demo.....	23
5.3.1 Screenshots from the Administrator user.	23
5.3.2 Screenshots from the Infrastructure Provider user.	25
5.3.3 Screenshots from the Service Provider user.....	26
5.3.4 Screenshots about custom flows and latency metrics.....	31
6 Conclusions and next steps.....	33
7 References	34

Document name:	D4.3 Decision Support tools I				Page:	3 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status: Final

List of Tables

Table 1 : Pledger Configuration subsystem functional components implemented by the DSS 10
Table 2 : Pledger Orchestrator subsystem functional components implemented by the DSS..... 10

Document name:	D4.3 Decision Support tools I				Page:	4 of 34	
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status:	Final

List of Figures

Figure 1: Pledger functional subsystems	9
Figure 2: Pledger Management functional components	9
Figure 3: DSS decision tree for the resource increase check	12
Figure 4: DSS decision tree for the resource decrease check	13
Figure 5: DSS component diagram – interfaces for configuration updates	14
Figure 6: Sequence diagram about Configuration service notifying new data to Pledger components	15
Figure 7: Sequence diagram about SLA violation and Benchmarking report consumption in the DSS and commands sent to the Orchestrator	16
Figure 8: Configuration service and DSS data model	17
Figure 9: example of Service REST resource for the Configuration service	19
Figure 10: DSS Swagger-UI accessible from the UI	19
Figure 11: Administrator manages Pledger users	23
Figure 12: Users with roles	23
Figure 13: users login audit	24
Figure 14: definition of Pledger SP and IP users	24
Figure 15: infrastructure with monitoring plugin and resources capacity	25
Figure 16: nodes with feature auto-discovery features working to identify HW availability	25
Figure 17: catalog apps	26
Figure 18: App composed by multiple services with K8S descriptor	26
Figure 19: Service with descriptor and initial configuration about resources (Kubernetes)	27
Figure 20: SLA definition	27
Figure 21: Guarantees definition	28
Figure 22: ServiceConstraints, setup by the SP, to express deployment preferences	28
Figure 23: DSS deployment options for a specific App based on the ServiceConstraints	29
Figure 24: SLA violations	29
Figure 25: CriticalService found, that might require offload	30
Figure 26: SteadyService found, that might require a scale down	30
Figure 27: Service instantiated by the DSS on a Kubernetes node	31
Figure 28: NodeRed integration for custom flows	31
Figure 29: Latency monitoring for offloading to cloud infrastructures	32

Document name:	D4.3 Decision Support tools I			Page:	5 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0
				Status:	Final

List of Acronyms

Abbreviation / acronym	Description
API	Application Programming Interface
CI/CD	Continuous Integration and Deployment
DApp	Distributed App
DB	Data Base
DSS	Decision Support System
Dx.y	Deliverable number y belonging to WP x
GUI	Graphical User Interface
IaaS	Infrastructure as a Service
IP	Infrastructure Provider
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
MVP	Minimum Viable Product
Mx	Project Month x
OS	Operating System
PaaS	Platform as a Service
QoS	Quality of Service
RDBMS	Relational Database Management System
REST	REpresentational State Transfer
SLA	Service Level Agreement
SP	Service Provider
SQL	Simple Query Language
Tx.y	Task number y belonging to WP x
UI	User Interface
WP	Work Package

Document name:	D4.3 Decision Support tools I			Page:	6 of 34		
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status:	Final

Executive Summary

This document accompanies the delivery of the Decision Support System (DSS) prototype developed in Task 4.3 “Decision Support Tools I” which includes Pledger’s Configuration and Recommender functional subsystems described in the D2.3 “Pledger Overall Architecture” [1].

The DSS is a new component developed in Pledger to support the service provider to take optimal decisions about the allocation of resources for DApps while minimizing the number of SLA violations.

The DSS acts autonomously trying to reduce the resources allocated to the edge, where they are scarce, whenever the Apps do not produce any warning, then increasing them when the opposite occurs. Eventually, the DSS offloads the Apps to the cloud if there are no more edge resources available, then back to the edge when possible.

As a basis for such decisions, the DSS uses the service providers’ preferences in order to prioritize choices and filter out others. In case of equivalent choices, the DSS chooses the best node where to allocate resources, either considering the time needed to free them, or latency in case of offloading to the cloud.

The DSS also supports the creation and triggering of custom automations using a GUI based on NodeRed which allows Service Providers (SPs), as domain experts, to easily design and execute specific rules based on the DSS data.

The design and development of the DSS has followed an agile approach, with bi-weekly calls to discuss requirements in detail, specifically with the Use Cases (UCs) in order to support scenarios that could be valuable.

During this process, the data model has been reviewed and a prototype has been produced at every iteration to validate the assumptions and make changes. For this reason, JHipster has been used for rapid prototyping and CI/CD has been configured since the very beginning in order to have all the changes available for testing. Along with CI/CD, a custom procedure has been introduced to easily switch among different DSS versions to facilitate progress analysis.

In the first development iteration, the DSS has implemented the high priority Minimal Viable Product (MVP) requirements. In the next iteration, a more robust integration with the Orchestrator and the AppProfiler will be available, as well as finer-grained optimizations based on latency metrics for cloud offloading and time metrics to free resources. This will lead to a comprehensive full end-to-end integration with all the Pledger core components and UC DApps.

Different scenarios have been prepared for this first demonstrator: a first one about the manual configuration and the automatic retrieval of all the information used by the DSS; a second one about the scaling down of steady Apps when no Service Level Agreement (SLA) warnings are received, then a final one about scaling up/offloading to the cloud when warnings or SLA violations are received.

Document name:	D4.3 Decision Support tools I				Page:	7 of 34	
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status:	Final

1 Introduction

1.1 Purpose of the document

The scope of the current deliverable (D4.3) is to provide functional and technical description of the main functionalities, functional components and services of the Pledger Decision Support tools (T4.3). This deliverable accompanies the prototype during the first iteration of WP4 and provides installation and usage guidelines of tools of the subsystem as well as demos of the main functionalities.

1.2 Relation to the other project work

The DSS and the deliverable is guided by the work performed in WP2 about the requirement analysis and the architecture (D2.2, D2.3).

The DSS tightly couples with the core Pledger components developed in the WP3 (D3.1, D3.2, D3.3): it relies on the Benchmarking and AppProfiler for the selection of the best infrastructure to host a DApp, especially when offloading to the cloud, then on the Orchestrator to actually deploy DApps on an infrastructure, finally on the SLA Manager to receive SLA Violations in case of Quality of Service (QoS) levels lower than agreed.

1.3 Structure of the document

This document is structured in 6 major chapters:

- ▶ **Chapter 2** presents the functional description of the tools in alignment with the overall architecture of Pledger.
- ▶ **Chapter 3** presents the technical description of the tools (data model, interfaces, and component architecture).
- ▶ **Chapter 4** presents the installation and usage guide of the tools.
- ▶ **Chapter 5** presents the demonstration of the tool with the description of the validation scenarios.
- ▶ **Chapter 6** presents the conclusions and next steps for this first iteration of the tool.

Document name:	D4.3 Decision Support tools I				Page:	8 of 34	
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status:	Final

2 Functional description

Pledger’s Decision Support System (DSS) is a component specifically designed and implemented for Pledger project. Its main purpose is to provide instructions to the orchestrator for the “Day-2” configuration of the DApps in order to ensure required QoS levels using SLAs warnings and violations as a feedback.

Within Pledger functional architecture represented in D2.3 [1], the DSS is part of the Management layer (Figure 1) and spans over the Orchestration and the Configuration subsystems.

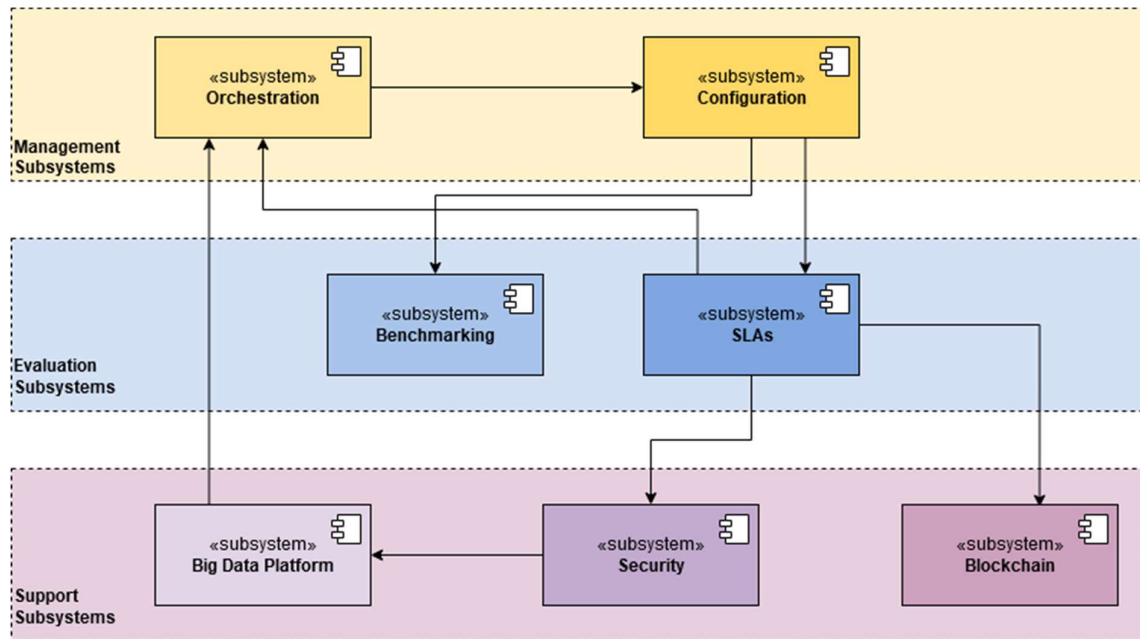


Figure 1: Pledger functional subsystems

The Figure 2, extracted from the architecture [1] shows an insight of the functional components and their relations.

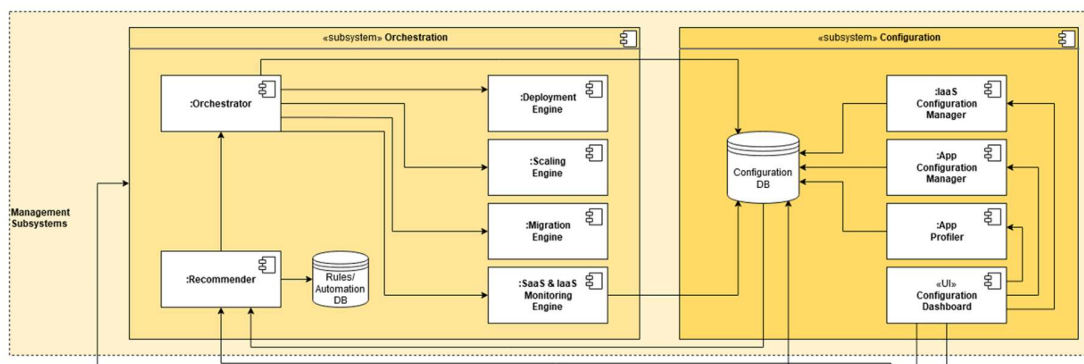


Figure 2: Pledger Management functional components

For the Configuration subsystem, the DSS implements the following functional components: IaaS and App Configuration Manager, merged into the Configuration Service backend, the Configuration

Document name:	D4.3 Decision Support tools I	Page:	9 of 34
Reference:	D4.3 Dissemination: PU	Version: 1.0	Status: Final

Dashboard, which is the Configuration Service User Interface (UI), then the Configuration Data Base (DB) used to persist data.

The list of the Configuration subsystem functional components implemented by the DSS and their roles are reported in Table 1.

Table 1 : Pledger Configuration subsystem functional components implemented by the DSS

ID	Component	Functionality
FC.1.1	IaaS Configuration Manager	The IaaS Configuration Manager is responsible for the management of infrastructure configuration, spanning from the credentials to the main topology and properties of the infrastructure (such as the servers URI, the master/worker properties for Kubernetes, GPU type, CPU model, etc.) that could impact scheduling decisions, as well as resource limits configured from the IaaS providers.
FC.1.2	App Configuration Manager	The App Configuration Manager is responsible for the management of the app configuration, which includes generic information used to match SaaS providers' preferences expressed in their profiles, along with those specific related to QoS and SLAs. QoS keys are listed for each application and SLA values/ thresholds are stored to allow the SLA Manager to check their violations and prioritize them.
FC.1.4	Configuration DB	The Configuration DB is responsible for storing the aforementioned configuration information and sharing it through specific API to the other Pledger subsystems, such as the Recommender in the Orchestration subsystem, the Benchmarking and the SLA creators' components.
FC.1.5	Configuration Dashboard	The Configuration Dashboard is the UI provided to IaaS and SaaS users to allow the proper configuration of the aforementioned data and also includes reports to allow the SaaS users to have a detailed view of the infrastructures and apps status and the recommendations for the app orchestration.

For the Orchestration subsystem, the DSS implements the following functional components: the Recommender and Rules/Automation DB. The latter connects to the ConfigurationDB to access configuration data and produce actions for the Orchestrator. The list of the Orchestrator subsystem functional components implemented by the DSS and roles are reported in Table 2

Table 2 : Pledger Orchestrator subsystem functional components implemented by the DSS

ID	Component	Functionality
FC.2.6	Recommender	The Recommender component is responsible for the provisioning of suggestions to the SaaS providers related to the app orchestration. In particular, the suggestions focus on the most efficient allocation within the available infrastructures, taking into account the infrastructure and app properties from the configuration subsystem, the infrastructure and app status from the Infrastructure as a Service (IaaS) Monitoring component and the SLA Manager, and the user preferences stored again in the configuration subsystem.
FC.2.7	Rules/ Automation DB	The Rules/ Automation DB component stores the Recommender suggestions and includes rules to decline the decision-making process (with regard to the user preferences) and possibly provide (small) automations to allow edge nodes act autonomously, depending on the resources available, whenever a disconnection from the infrastructure occurs.

A DApp is a distributed application made of multiple services that can run on different infrastructure nodes according to the DApp constraints. The SP can add her/his own preferences and prioritize and restrict the range of deployment possibilities of a DApp deployment.

A first feature of the DSS is to keep track of the infrastructures' topologies, of the DApp constraints and those configured by the SP. Within such constraints, the DSS works to ensure the right resources to each DApp service, either reducing if the amount requested is not used for some time and no SLA warnings are received or increasing if the opposite.

According to the SP preferences, which in Pledger prioritise edge over cloud, the DSS also verifies if a higher priority deployment option is available and, in case, triggers the placing of services on different nodes. Mainly, this means to offload services from edge to cloud if there is no more space on the edge, and the opposite if resources become available on the edge.

Similarly, the DSS checks if the DApp is requesting more resources than necessary, meaning the actual usage is far below the resource reserved and no SLA warning/violations have been received for a specific amount of time. In this case, the DSS gradually decrease resources, which is crucial on the edge due to usual scarcity.

At the same time, the DSS gradually increases resources allocated to each DApp if they produce any SLA warning/violation. Eventually this will lead to saturate edge resources and result to offloading to the cloud.

The aforementioned checks about increasing/decreasing resources are done in parallel by the DSS in order to find the optimal resource allocation: Figure 3 shows the decision tree used by the DSS resource increase check, while Figure 4 shows the decision tree used by the DSS resource decrease check. Both report a sample percentage for resource increment/decrement based on the SP preferences.

Document name:	D4.3 Decision Support tools I				Page:	11 of 34	
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status:	Final

"critical score" is computed on historical SLA violations weighted by time and severity

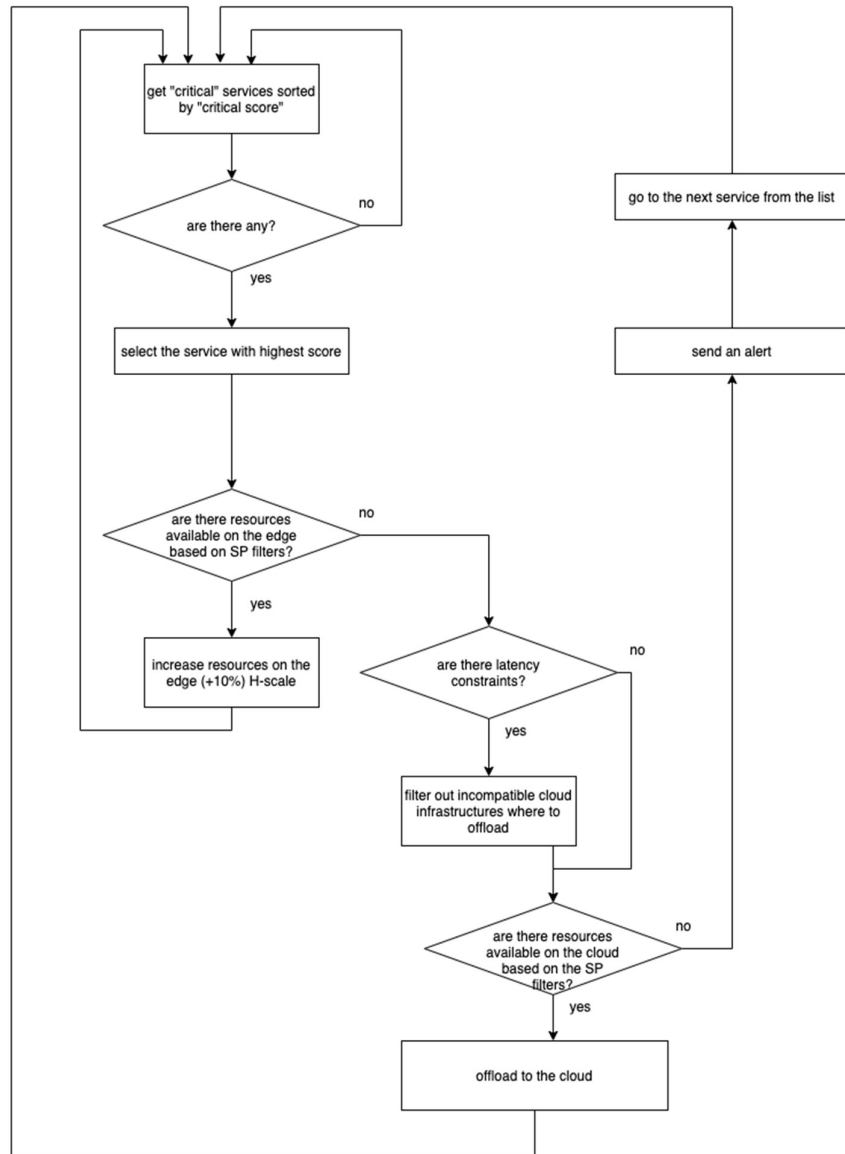


Figure 3: DSS decision tree for the resource increase check

Document name:	D4.3 Decision Support tools I			Page:	12 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0
				Status:	Final

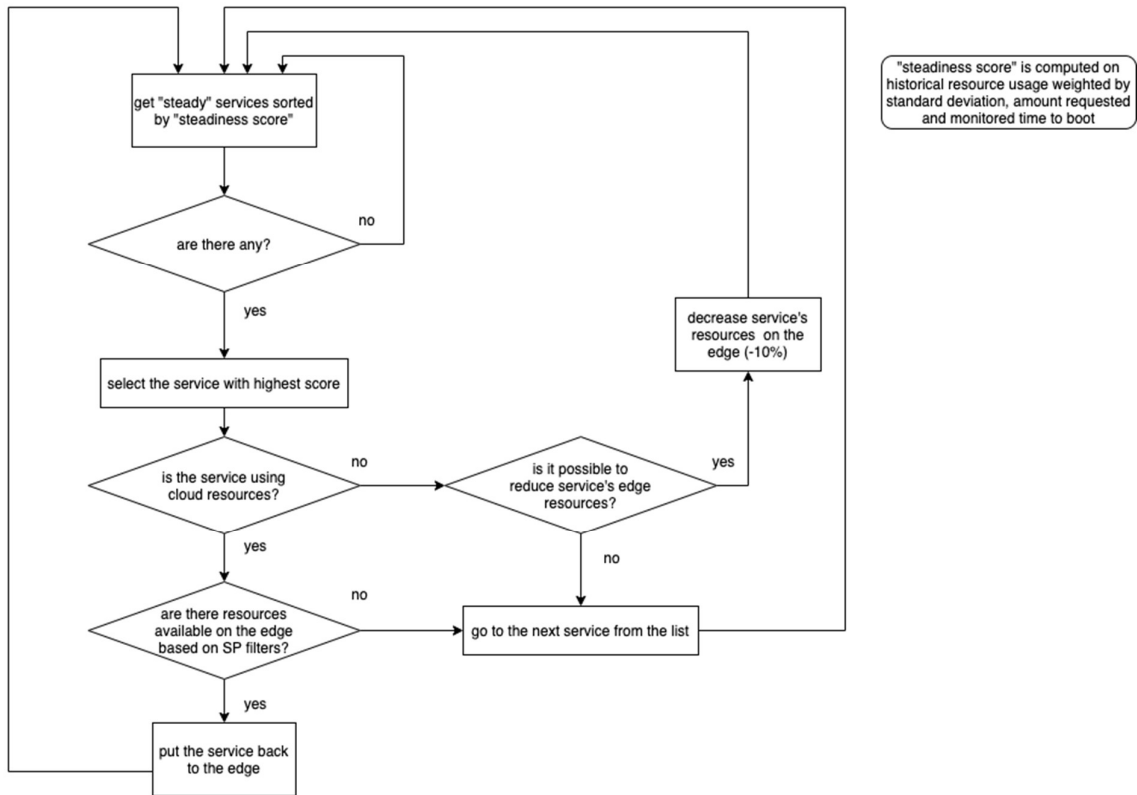


Figure 4: DSS decision tree for the resource decrease check

For this first release, the DSS gets information from IaaS and PaaS monitoring, the SLA violations and the Benchmarking and provides inputs to the Orchestrator in order to scale or place DApps on the infrastructure either on a pair node on edge or cloud according to the SP preferences, to the DApps resource requests and their availability on to the infrastructure.

For the next iteration at M33, the plan is to consolidate the integration with the Orchestrator to support different platforms, include AppProfiler into the evaluation of the best node where to offload DApps, introduce additional optimizations which includes latency when offloading to the cloud and the time needed to free resources in case of “heavy” monolithic applications, then allow the SP to express finer-grained preferences for each DApp.

Document name:	D4.3 Decision Support tools I			Page:	13 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0
				Status:	Final

3 Technical description

3.1 Baseline technologies and dependencies

The DSS is composed by a UI frontend to allow SP and Infrastructure Providers (IP) change the configuration, a Configuration Service backend with the business logic and a Relational Database Management System (RDBMS) to persist data.

The main technologies used for the DSS are NodeJS[9] v11, Angular[4] v.11 for the frontend, OpenJDK [2] v.11 and SpringBoot[3] v2.4 for the backend, MySQL[5] v5.5 for the RDBMS. It also includes Kafka[6] connectors to communicate with the StreamHandler, used for asynchronous communication with the other Pledger components as shown in the next subsection.

Both the Configuration service and the DSS services are deployed on Kubernetes using Pledger’s CI/CD pipeline to leverage on continuous integration and manage the whole development and deployment lifecycle. It also includes a custom component that allows the re-creation of table structure and initial configuration data according to the version desired. This is meant to easily restore the Configuration status to a consistent version shared on the source repository.

The DSS also reference a local instance of NodeRed[7] with some custom blocks interfacing the DSS API and the StreamHandler to facilitate the creation of custom automation by the SP.

3.2 Components Architecture

Figure 5 describes the main components of the DSS, UI, Configuration and DSS services, with detail about how they are interfaced by the main Pledger components to provide the common configuration through a REST API using a publish/subscribe communication with the StreamHandler.

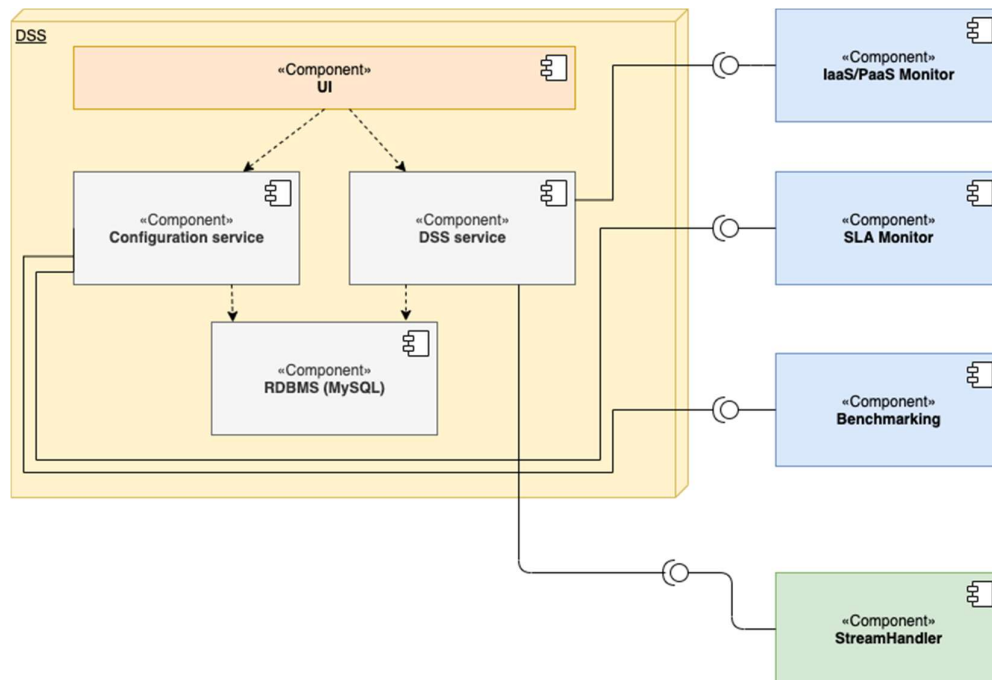


Figure 5: DSS component diagram – interfaces for configuration updates

Document name:	D4.3 Decision Support tools I	Page:	14 of 34
Reference:	D4.3 Dissemination: PU	Version: 1.0	Status: Final

In this iteration, the actual scheduling and execution of Services is done directly by the DSS using a plugin that interacts with Kubernetes and Docker API.

In the next iteration, the DSS will integrate with the Orchestrator and the AppProfiler with publish/subscribe JSON messages on the StreamHandler. This activity will be described as part of the integration task T5.2 and the corresponding deliverable D5.2 at M24.

3.3 Interfaces provided

The sequence diagram in Figure 6 shows the main flow for the resources configuration that includes three main steps: the update of the information on the RDBMS, the notification sent over the StreamHandler and the interested Pledger components that subscribe to retrieve updates using REST calls on the Configuration service.

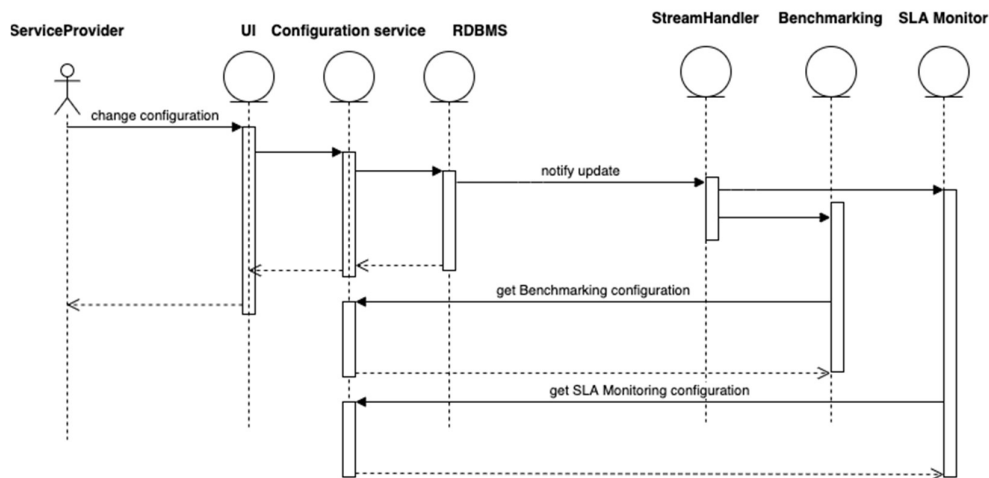


Figure 6: Sequence diagram about Configuration service notifying new data to Pledger components

To summarize, the Configurator main interfaces are:

- A. with the StreamHandler to send notifications when a configuration entity changes.
- B. with the Benchmarking, SLA Monitor, Orchestrator to provide the main configuration about Pledger setup using REST API

The interface A) is a publish/subscribe protocol that uses the topic “configuration” to advertise updates on any REST entity with a JSON message sent to Kafka in the format:

```
{'id': ID, 'entity': 'ENTITY_NAME', 'operation': 'OPERATION_NAME'}
```

The format is straightforward:

- ▶ ID is the unique identifier in the Configuration service, used to possibly filter only the interested entities.
- ▶ ENTITY_NAME is the REST resource name.
- ▶ OPERATION_NAME is the operation executed, UPDATE for creation/update or DELETE for deletion.

Document name:	D4.3 Decision Support tools I			Page:	15 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0
				Status:	Final

The interface B) is the Configuration service REST API available via Swagger[8] to facilitate integration and described in section 3.5.

The sequence diagram in Figure 7 shows the main flow for the SLA Violation and the Benchmarking report consumption by the DSS using the StreamHandler as publish/subscribe service in order to consume messages.

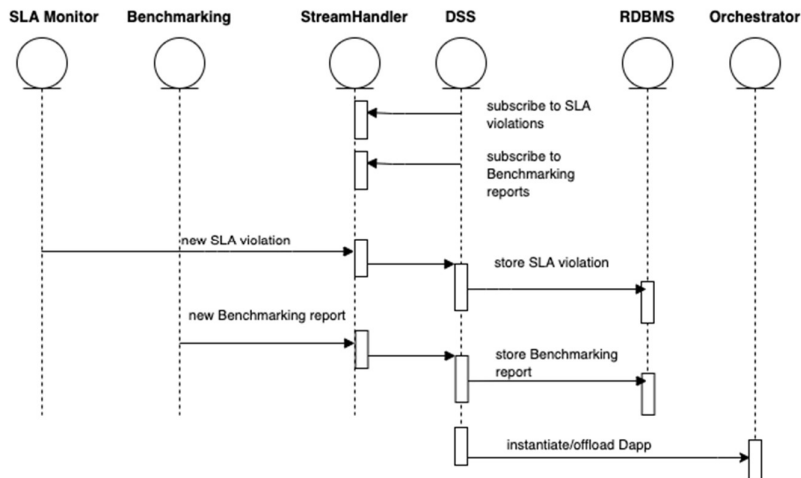


Figure 7: Sequence diagram about SLA violation and Benchmarking report consumption in the DSS and commands sent to the Orchestrator

3.4 Data models

The Configuration and the DSS components share the same data model. The main entities and relations are shown in the class diagram in Figure 8.

Document name:	D4.3 Decision Support tools I			Page:	16 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0
				Status:	Final

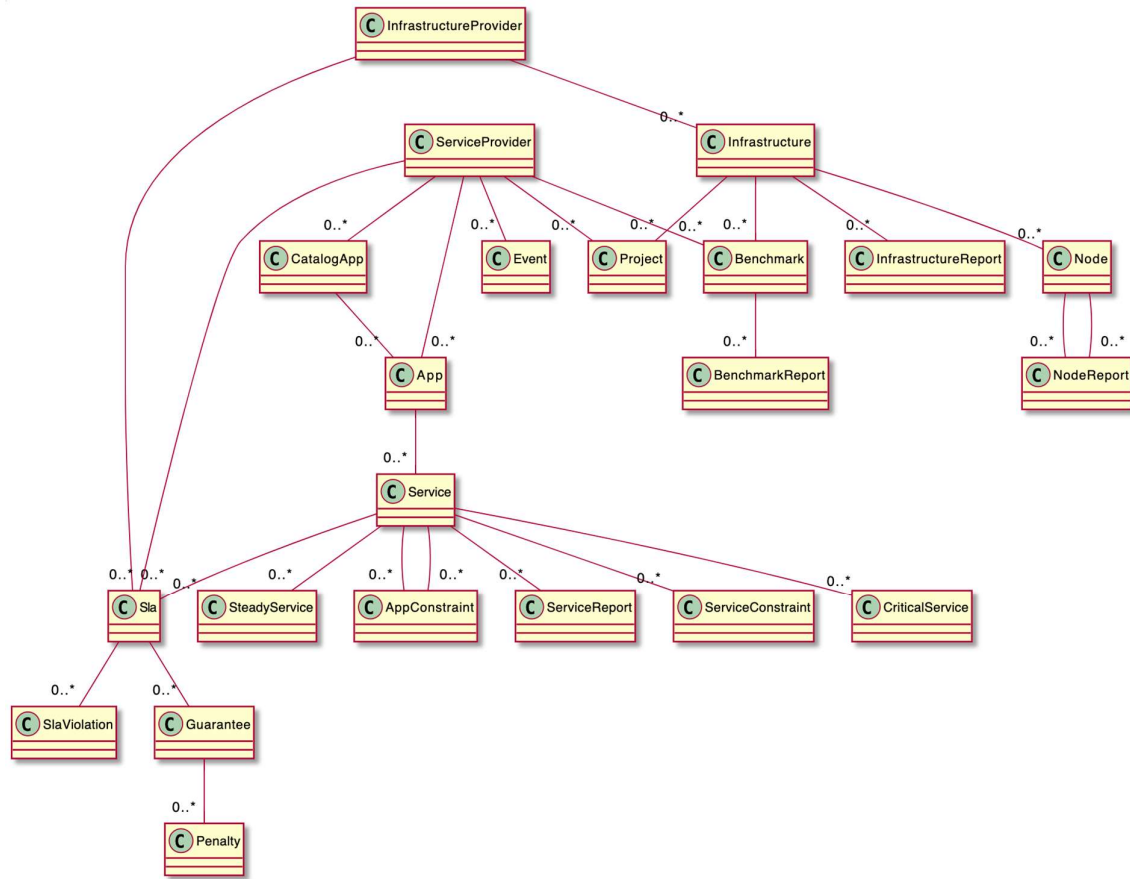


Figure 8: Configuration service and DSS data model

The entity list and their role are described below:

- ▶ App: represents a DApp, an application that is tied to an SLA and used by a SP.
- ▶ AppConstraint: represents the App constraints about infrastructure and node properties
- ▶ Benchmark: represent a benchmark used to estimate the performance of an infrastructure and its Nodes.
- ▶ BenchmarkReport: represents the historical data produced by the Benchmarks.
- ▶ CatalogApp: represents the applications available in Pledger.
- ▶ CriticalService: represents a Service that is eligible to resource increment or offloading to a lower priority Node if not available on the Node currently hosting the service.
- ▶ Guarantee: represents the list of constraints agreed within a specific SLA.
- ▶ Infrastructure: represents the infrastructure used, assuming Pledger can manage more of them, and that are owned by separate providers, with type describing whether it is Kubernetes, etc.
- ▶ InfrastructureProvider: represents the responsible for the infrastructure; such entity is introduced to identify one of the two parties involved in the definition of an SLA.
- ▶ Node: represent one element of an infrastructure which might have specific resources available so that it is possible to differentiate the orchestration (scaling and placing) of applications according to the Apps needs.
- ▶ NodeReport and InfrastructureReport represent the resource usage
- ▶ Penalty: represents the amount to pay in case of a violation for a specific Guarantee.

Document name:	D4.3 Decision Support tools I			Page:	17 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0
				Status:	Final

- ▶ **Project:** represents a high-level contract between SP and IP which allows a SP to use an infrastructure with a resource quota.
- ▶ **Service:** represents an executable component of an App.
- ▶ **ServiceConstraint:** represents a constraint on a specific Service, such as the number of resources requested.
- ▶ **ServiceProvider:** represents the provider of the services in Pledger.
- ▶ **ServiceReport:** contains report about the Service resource usage.
- ▶ **SLA Violation:** represents the violation of an agreement (Guarantee) within an SLA.
- ▶ **SLA:** represents a contract between an InfrastructureProvider and a ServiceProvider about the execution of a specific App.
- ▶ **SteadyService:** represents a Service that is eligible for resource reduction or offloading to a higher priority Node if available with respect to the Node currently hosting a service

The Configuration service supports two main roles, “Service Provider” and “Infrastructure Provider, plus an “Administrator” role that manages the users and permissions but is not allowed to access any other data. In particular:

Administrator is responsible for:

- ▶ the management of the IP and SP users.
 - ▶ the creation of Projects to bind SP with Infrastructures and define the quota of resources assigned
- This role allows to manage Pledger users and assign them resources.

Service Provider is responsible for:

- ▶ getting access to his/her own configuration data.
- ▶ configuring his/her profile with preferences used by the DSS.
- ▶ configuring CatalogApps and Apps, either public or private, and their priorities.
- ▶ configuring Benchmarks, either public or private.
- ▶ configuring SLA, their Guarantees and Penalties.
- ▶ reading the SLA violations.
- ▶ getting information about services that have their resources reduced (SteadyService) or increased (CriticalService) by the DSS.
- ▶ getting information about Service actual reserved usage and metrics about latency (ServiceReport).
- ▶ setting constraints and priorities about the preferred deployments (ServiceConstraints) and monitor which options are available due to the actual resources (DeploymentOptions).

Infrastructure Provider is responsible for:

- ▶ defining the Infrastructure and Node topology.
- ▶ configuring custom properties on Nodes about the HW availability and so on.
- ▶ reading reports about Infrastructure and Node resource availability.

Limitations are added to allow each SP to access only their own “private data” belongs to her/him, as well as, of course, public data.

By “private data” we consider any entity that is tied to a specific instance of a SP according to the relationships shown in the entity diagram in Figure 8.

Document name:	D4.3 Decision Support tools I				Page:	18 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status: Final

This means, for example, that a specific SP can access only the App and Service that are attached to that instance. The same also for indirect relations, such as Infrastructure and so on. In the latter case, a SP can access only the Infrastructures that are attached to Projects assigned to that SP instance.

By “public data” we considered any entity that is not tied to any specific instance of a SP. This for example might happen with Benchmark or CatalogApp whenever the relation with SP is not populated. Such scenario is included in section 5.1 about the main configurations SP and IP can do using the Configuration UI.

3.5 REST API

All the aforementioned entities in Section 3.4 are published as REST resources through the Configuration REST API with methods that allow to: get a complete/filtered list, get a specific instance, save, update or delete one instance. They use HTTP methods GET, POST, PUT and DELETE as by REST specification.

An example is reported in Figure 9 for the Service REST resource.



Figure 9: example of Service REST resource for the Configuration service

The only exceptions are the InfrastructureReport, NodeReport, Event, SteadyService and CriticalService which are generated by the DSS, so they are read-only and available only through GET methods.

The complete list of entities is available through Swagger-UI to facilitate the integration with third-party components and is integrated in the Configuration UI. Figure 10 shows the Swagger-UI webpage with an example of filtering the *service* resources.

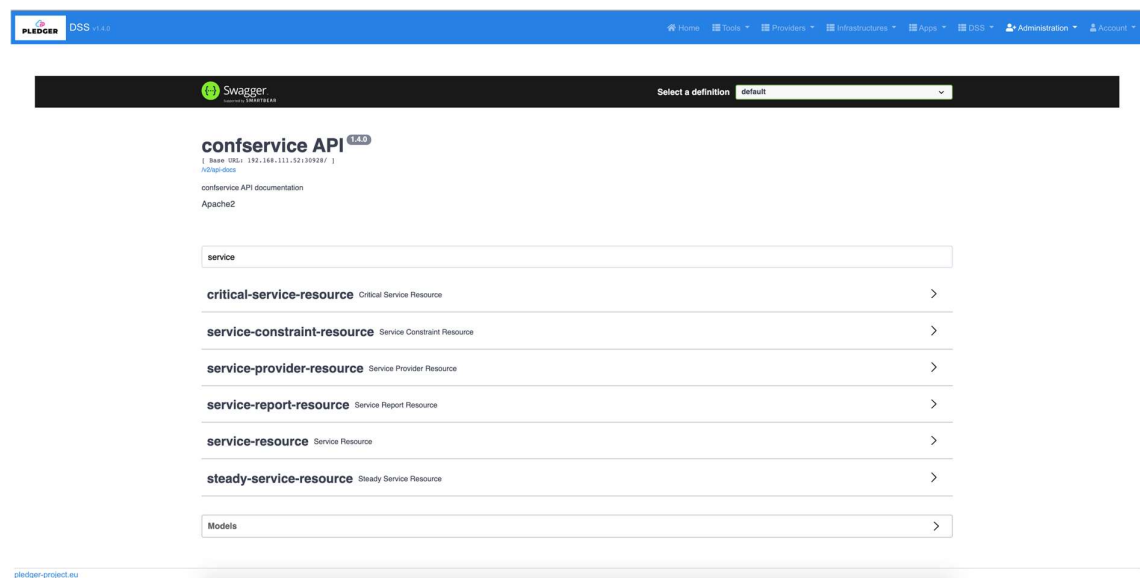


Figure 10: DSS Swagger-UI accessible from the UI

Document name:	D4.3 Decision Support tools I			Page:	19 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0
				Status:	Final

4 Installation and usage guides

4.1 Requirements

In order for the DSS to be correctly executed, it requires:

- ▶ OpenJDK [2] v11.0
- ▶ NodeJS [9] v12.0
- ▶ Npm[10] v6.0

In addition, for the correct building of the DSS application, Maven[11] v3 needs to be installed too. As shown in the next sub-section, a Jenkins[13] build file is also provided to automate the build process and YAML files for the deployment on Kubernetes.

4.2 Installation

The DSS and its components are packaged as a full-stack Java application using the CI/CD services with the image made available on the Pledger Docker registry[12].

The source code and the configuration file used for the build with Jenkins are also stored on GitLab[14] and the build is triggered automatically whenever a change into the code is committed.

Kubernetes[15] descriptors are also shared on GitLab to allow versioning and automatic deployment and they can be used to replicate the installation on any other instance of Kubernetes outside Pledger infrastructure.

4.3 Usage

Once the installation is in place, an initial SQL data is also provided to show a basic configuration and allow some tests about assignments of resources to Service Providers and so on.

For the actual usage of the DSS, an infrastructure, the Orchestrator, the Benchmarking and some DApps need to be configured according to the scenario described in section 5.

4.4 Licenses

All the code is provided with Apache 2 [16] license.

4.5 Source code repository

The DSS source code repository is <https://gitlab.com/pledger/confservice> with plan to be publicly available before the end of the project.

Document name:	D4.3 Decision Support tools I			Page:	20 of 34		
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status:	Final

5 Demonstration

5.1 Scenario description

For this iteration, three demonstration scenarios were prepared.

The first shows the main interactions with the Configuration UI. In particular:

- ▶ Administrator reading the SP and IP users defined, their roles, then creating a Project to bind SP to an Infrastructure.
- ▶ IP configuring Infrastructure and Node topology with custom properties, configuring among other things, which Nodes are edge, and which are cloud.
- ▶ SP creating a private CatalogApp made of two Services, then the corresponding App to be instantiated, adding some SLA and Guarantees, then ServiceConstraints to limit the instantiation of a subset of Infrastructures and Nodes, checking if the DeploymentOptions have been correctly parsed by the DSS, then launching the App and getting information about the Nodes where its Services have been deployed, and reading several metrics from ServiceReport.

The second shows the DSS reducing resources to Services which allocate more than necessary:

- ▶ the SP that instantiates an App made of two Services which are configured to use some resources (CPU, memory) which are overestimated.
- ▶ After the App is launched, the SP checks the SteadyService, and sees that the Services are reported to use much less resources than requested.
- ▶ Then the SP sees that the DSS resizes the Services while relying on load balancers to give no disruption to the final users.

The third shows the DSS offloading to the cloud when receiving an SLA Violation and there is no space on the edge

- ▶ The instantiation of an App made of two Services which are configured to use some resources (CPU, memory).
- ▶ For such App, the SP sets ServiceConstraint to force one Service on edge Nodes and leave the second Service again on the edge but with an option with lower priority to go on cloud Nodes.
- ▶ The SP starts the App, and the Services are allocated both on the edge.
- ▶ Then a SLAViolation (as type “warning”) is received as the resources are not enough.
- ▶ The DSS checks the DeploymentOptions with lower ranking and see that it is possible to offload only one service to the cloud.
- ▶ The DSS offloads such Service to the cloud.
- ▶ After some time, as no more SLAViolations are received and more resources are freed on the edge, the DSS offloads the Service back on the edge, so that now both are there.

Videos showing the above scenarios are available on the Pledger project [YouTube channel](#).

Document name:	D4.3 Decision Support tools I				Page:	21 of 34	
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status:	Final

5.2 Validation and Verification

The DSS addresses the MVP requirements for this first iteration of the tool. The System User Cases (SUC) enabled by the release of this version of the tool are the following:

- ▶ SUC.01 “provide App profile details” partially, with the manual insertion of DApp properties; they should be filled by the AppProfiler in the integration activities of T5.2.
- ▶ SUC.02 “deploy App components” partially, with support to Kubernetes with custom plugin based on direct Kubernetes API calls; it will rely on the Orchestrator to be integrated in T5.2.
- ▶ SUC.03 “view available IaaS resources”, completed, by the Configuration service and its UI.
- ▶ SUC.04 “get recommendations for IaaS resources”, completed, by the DSS.
- ▶ SUC.05 “select fitting IaaS resources” partially about the resources availability and options expressed by the SP; the plan for the next iteration is to include latency metrics for cloud offloading and time metrics for the selection of the resources to be freed.
- ▶ SUC.08 “get notified about unwanted events”, integration completed with the SLA Manager, both for configuration sharing and SLA Violation retrieval by the DSS.
- ▶ SUC.09 “handle unwanted events. Completed by the DSS which chooses better deployment options in case of SLA Violations and warnings.
- ▶ SUC.10 “view suggested recovery actions. The DSS takes autonomous decisions. The selection is done by using the Service Provider preferences which give different priorities to the actions.
- ▶ SUC.11 “choose/ perform recovery action(s). Same as SUC.10.
- ▶ SUC.14 “choose to migrate (parts of) components of App”. Completed by the DSS working on the App composed of multiple Services and deployment options with preferences.
- ▶ SUC.15 “choose to scale up/ down (components of) App”. Completed by the DSS as it supports resizing of App resources.
- ▶ SUC.16 “configure infrastructure”. Completed by the Configuration service with support to feature auto discovery.
- ▶ SUC.17 “make IaaS available”. Completed, as part of the Configuration service tasks.
- ▶ SUC.19 “IaaS monitoring”. Partially. The DSS extracts the IaaS metrics from a custom plugin of Kubernetes to access “metrics-server”. This will be replaced by Prometheus to handle metrics from Docker and custom infrastructure in the next iteration.
- ▶ SUC.20 “read infrastructure configurations”. Completed by the Configuration service, which shares the updates with the other core components.
- ▶ SUC.21 “get IaaS evaluation”. Completed by the DSS service, which supports offloading.

Validation and Verification of the DSS will be carried out during the project by checking the actual integration of the core components and the expected behavior while managing the Use Case applications.

Document name:	D4.3 Decision Support tools I				Page:	22 of 34	
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status:	Final

5.3 Demo

This section collects few screenshots of the DSS that illustrate some of the demonstration steps defined in the aforementioned scenarios.

5.3.1 Screenshots from the Administrator user.

The Administrator is responsible for the creation of Pledger users and for the monitoring of authentication activities. Figure 11 shows the Administrator drop-down menu while Figure 12 shows the user dashboard where the Administrator can assign roles to users and enable/disable them. Figure 13 shows audit logs and Figure 14 the list of Service provider users.

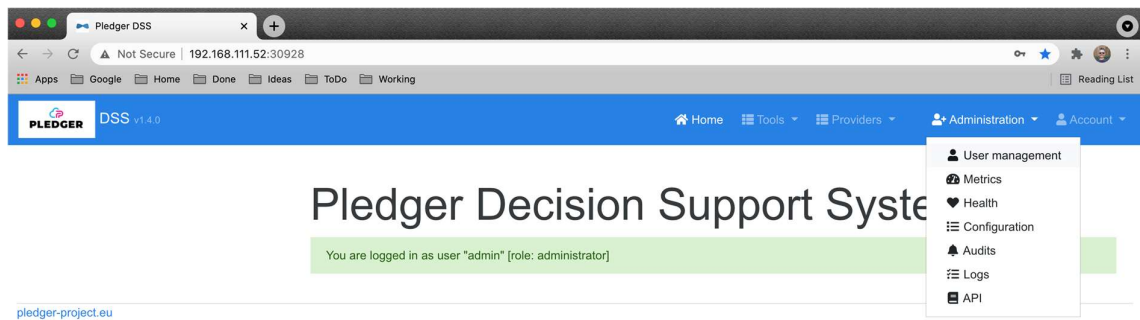


Figure 11: Administrator manages Pledger users

ID	Login	Email	Profiles	Created Date	Last Modified By	Last Modified Date	
2	francesco	francesco@example.org	Activated ROLE_IP	16/02/21 10:56	admin	16/02/21 10:56	View Edit Delete
3	admin	admin@localhost	Activated ROLE_ADMIN		system		View Edit Delete
4	root	root@localhost	Activated ROLE_ADMIN ROLE_SP ROLE_IP		system		View Edit Delete
5	api	api@localhost	Activated ROLE_ROAPI		system		View Edit Delete
7	filippo	filippo@example.org	Activated ROLE_SP	16/02/21 10:55	admin	16/02/21 10:55	View Edit Delete
8	august	august@example.org	Activated ROLE_SP	16/02/21 10:56	admin	16/02/21 10:56	View Edit Delete
9	verena	verena@example.org	Activated ROLE_SP	16/02/21 10:56	admin	16/02/21 10:56	View Edit Delete
10	estela	estela@example.org	Activated ROLE_IP	16/02/21 10:56	admin	16/02/21 10:56	View Edit Delete
11	gabriele	gabriele@example.org	Activated ROLE_SP	16/02/21 10:56	admin	16/02/21 10:56	View Edit Delete

Figure 12: Users with roles

Document name:	D4.3 Decision Support tools I				Page:	23 of 34	
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status:	Final

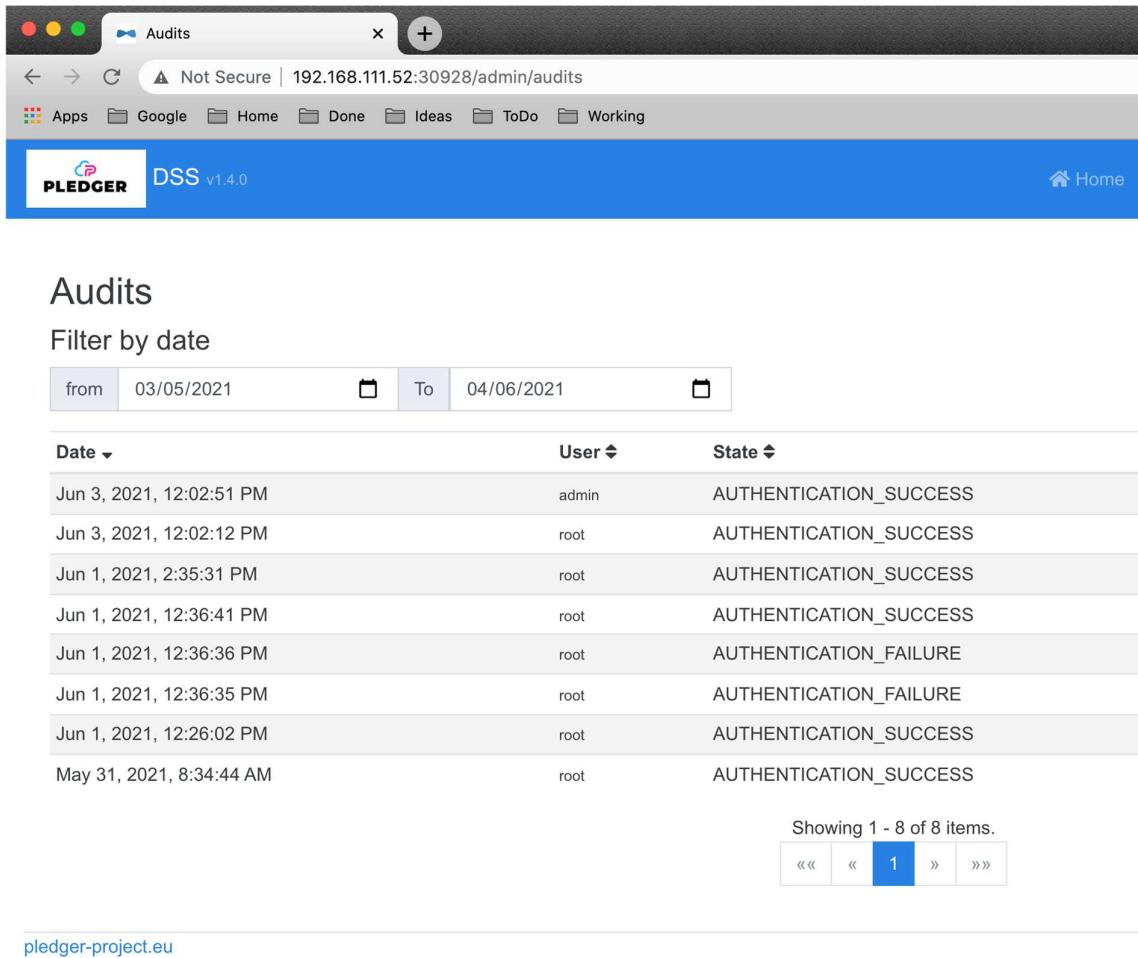


Figure 13: users login audit

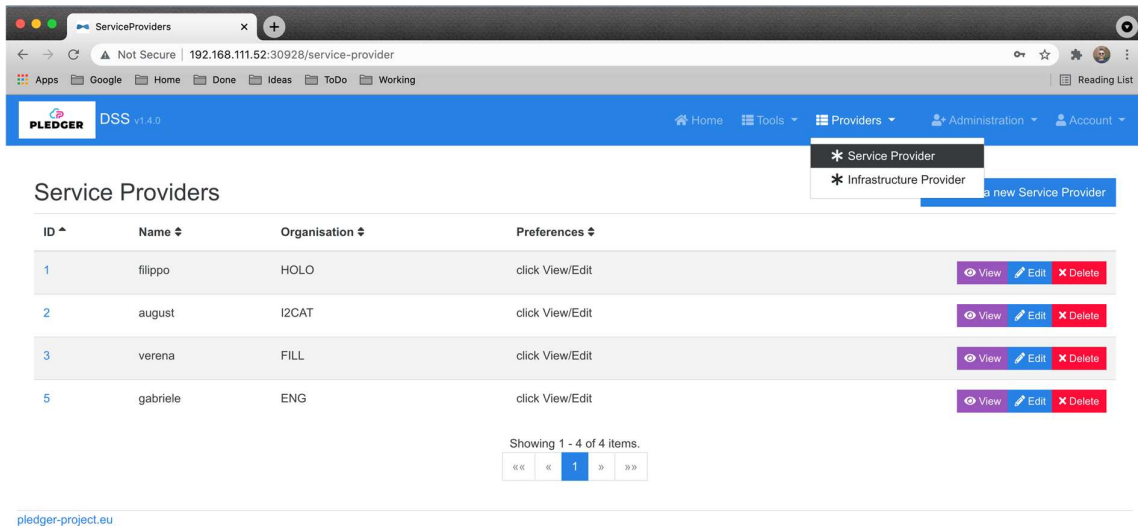


Figure 14: definition of Pledger SP and IP users

Document name:	D4.3 Decision Support tools I	Page:	24 of 34
Reference:	D4.3 Dissemination: PU	Version: 1.0	Status: Final

5.3.2 Screenshots from the Infrastructure Provider user.

The Infrastructure providers are responsible for the definition of the infrastructure and nodes in Pledger, along with the required configuration to allow its monitoring. Figure 15 shows the monitoring plugin configuration and Figure 16 shows the nodes with the hardware features automatically discovered by Pledger.

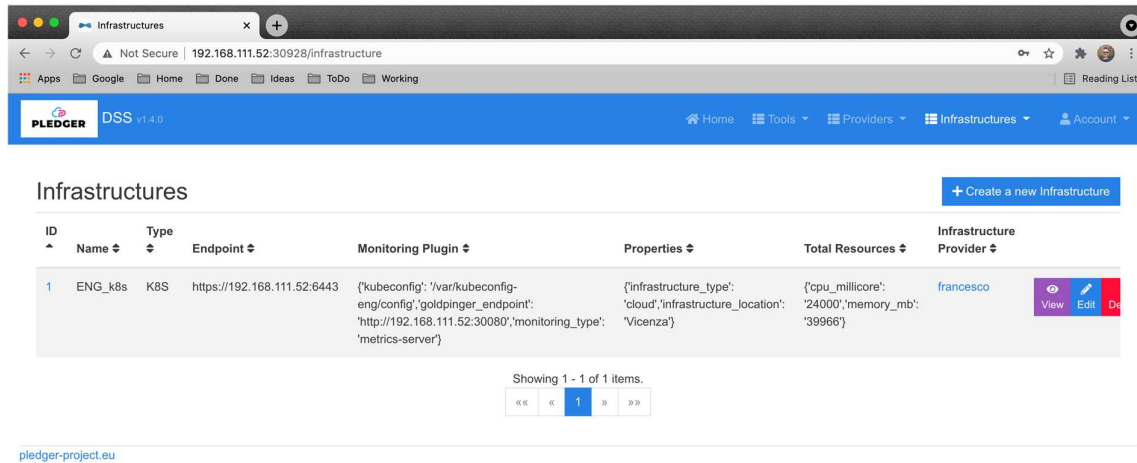


Figure 15: infrastructure with monitoring plugin and resources capacity

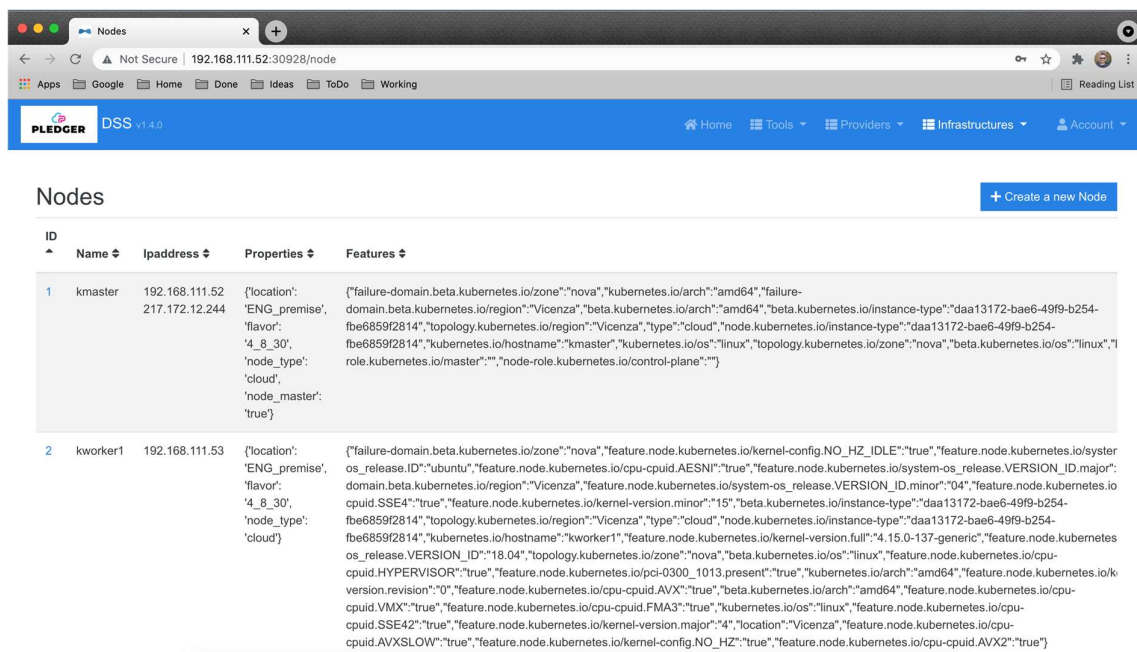


Figure 16: nodes with feature auto-discovery features working to identify HW availability

Document name:	D4.3 Decision Support tools I			Page:	25 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0
				Status:	Final

5.3.3 Screenshots from the Service Provider user.

Service providers are responsible for the configuration of the DApps, their preferences about the deployment options to filter and prioritize the DSS options. Figure 17, Figure 18, Figure 19 show the configuration of catalog apps, services. Figure 20 and Figure 21 show the configuration of SLA and Guarantees. Figure 22 shows the configuration of deployment constraints from the Service provider and Figure 23 the resulting prioritized deployment options that will be used by the DSS. Figure 24 shows the SLA violation received by the SLA manager, Figure 25 shows the “steady” and Figure 26 the “critical” service list monitored by the DSS, finally Figure 27 shows the interface to manually launch a DApp from the UI.

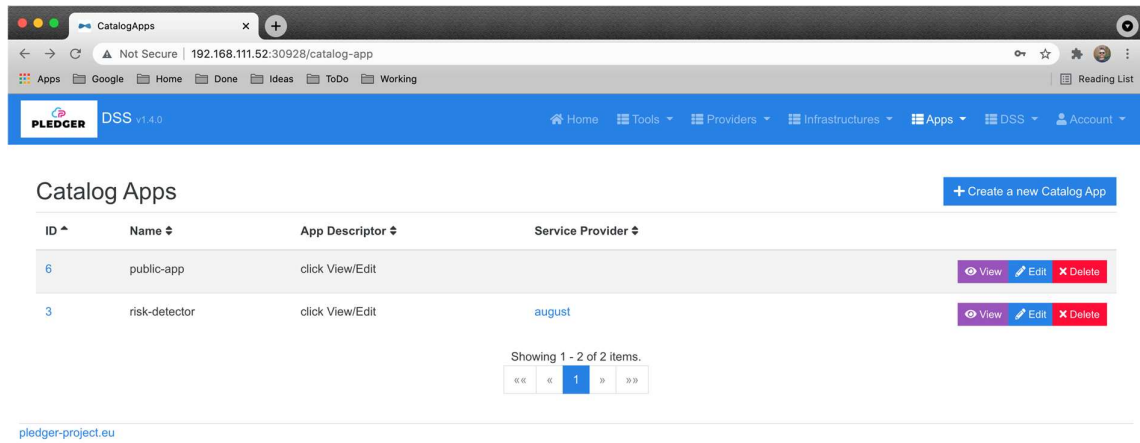


Figure 17: catalog apps

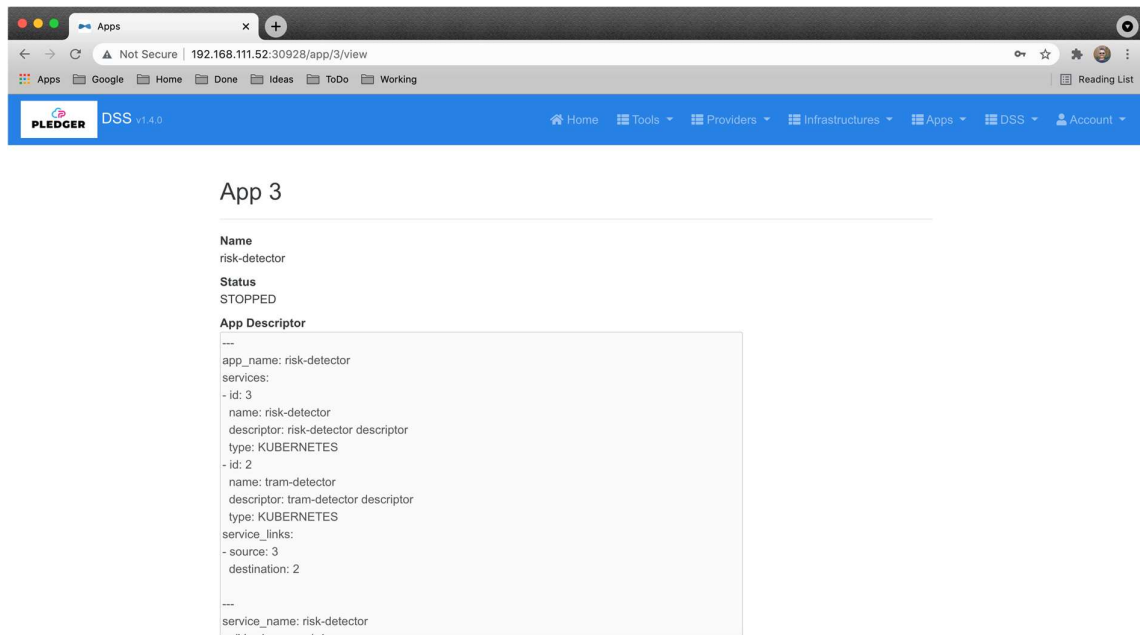


Figure 18: App composed by multiple services with K8S descriptor

Document name:	D4.3 Decision Support tools I			Page:	26 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0
				Status:	Final

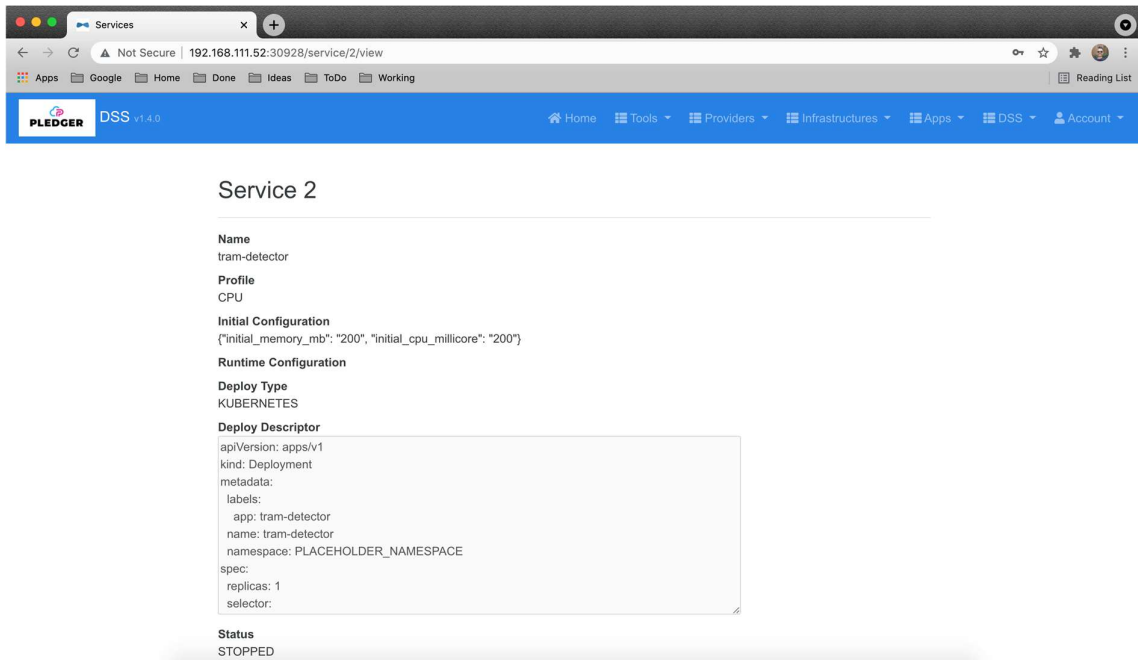


Figure 19: Service with descriptor and initial configuration about resources (Kubernetes)

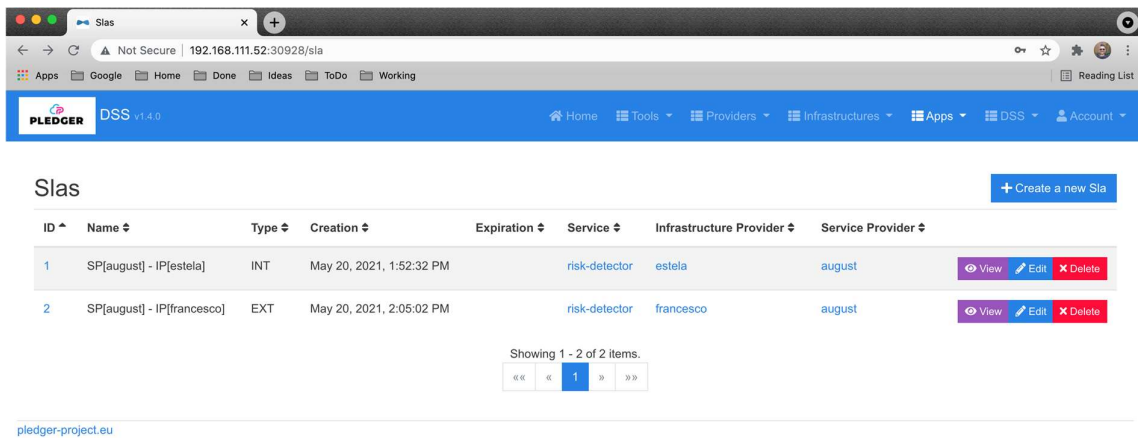


Figure 20: SLA definition

Document name:	D4.3 Decision Support tools I				Page:	27 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status: Final

Guarantees + Create a new Guarantee

ID	Name	Constraint	Threshold Warning	Threshold Mild	Threshold Serious	Threshold Severe	Threshold Catastrophic	Sla	
1	responseTime	responseTime <= 100	> 90	>100	>1000	>10000	>100000	SP[august] - IP[estela]	View Edit Delete
5	latency	latency <= 30	>27	>30	>300	>3000	>30000	SP[august] - IP[estela]	View Edit Delete
2	responseTime	responseTime <= 200	>180	>200	>2000	>20000	>200000	SP[august] - IP[francesco]	View Edit Delete
6	latency	latency <= 40	>36	>40	>400	>4000	>40000	SP[august] - IP[francesco]	View Edit Delete

Showing 1 - 4 of 4 items.

Figure 21: Guarantees definition

Service Constraints + Create a new Service Constraint

ID	Name	Category	Value	Value Type	Priority	Service	
1	node_master	FORBIDDEN	true	text	0	tram-detector	View Edit Delete
2	location	MANDATORY	I2CAT_premise	text	1	tram-detector	View Edit Delete
3	node_master	FORBIDDEN	true	text	0	risk-detector	View Edit Delete
4	location	OPTIONAL	I2CAT_premise	text	1	risk-detector	View Edit Delete
6	node_type	OPTIONAL	edge	text	1	risk-detector	View Edit Delete
5	location	OPTIONAL	I2CAT_premise	text	2	risk-detector	View Edit Delete
7	node_type	OPTIONAL	cloud	text	2	risk-detector	View Edit Delete

Showing 1 - 7 of 7 items.

Figure 22: ServiceConstraints, setup by the SP, to express deployment preferences

Document name:	D4.3 Decision Support tools I				Page:	28 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status: Final

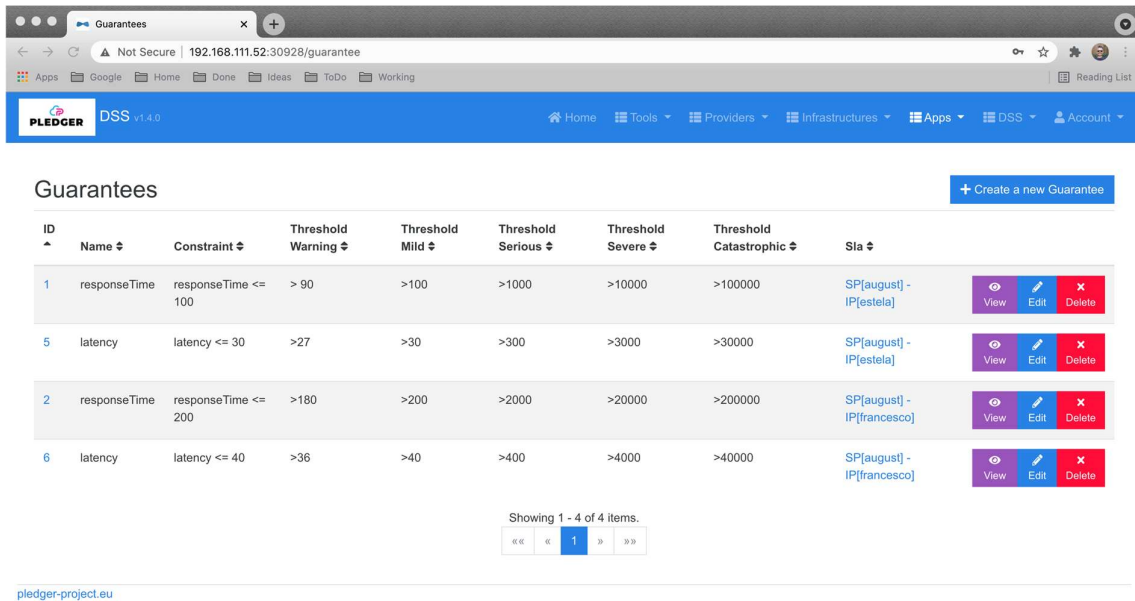


Figure 23: DSS deployment options for a specific App based on the ServiceConstraints

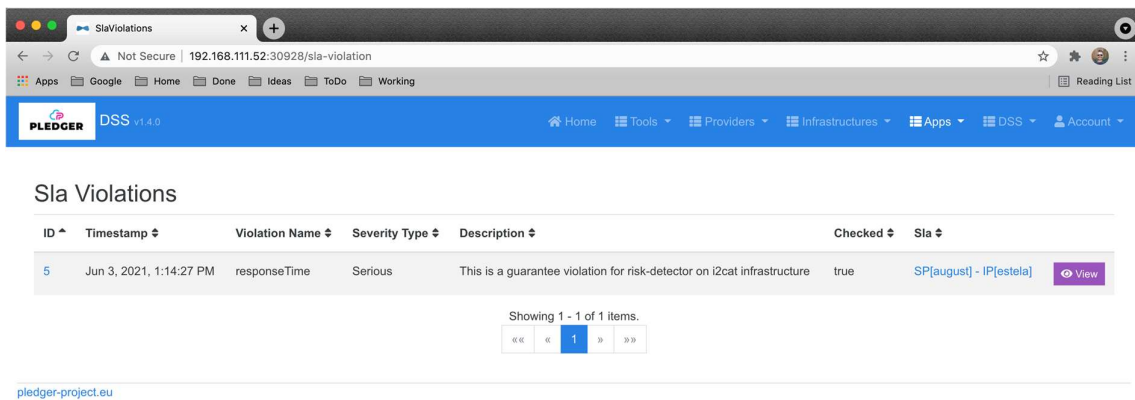


Figure 24: SLA violations

Document name:	D4.3 Decision Support tools I			Page:	29 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0
				Status:	Final

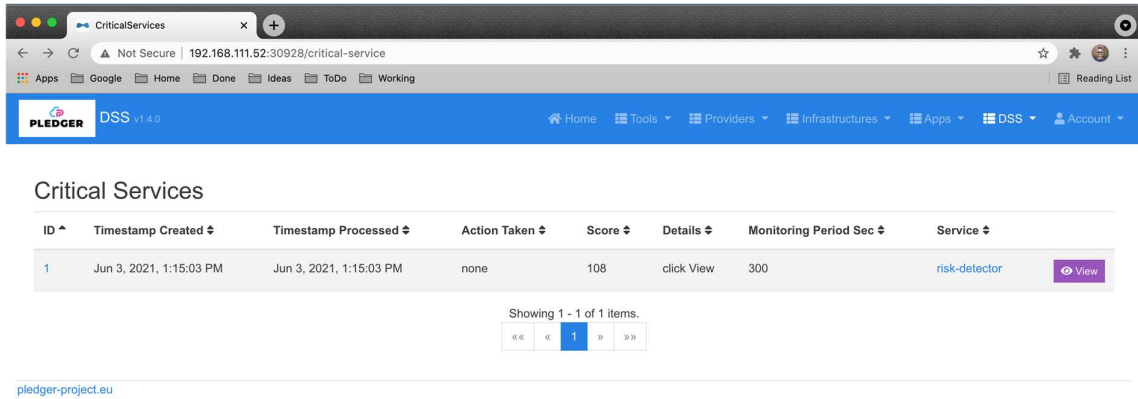


Figure 25: CriticalService found, that might require offload

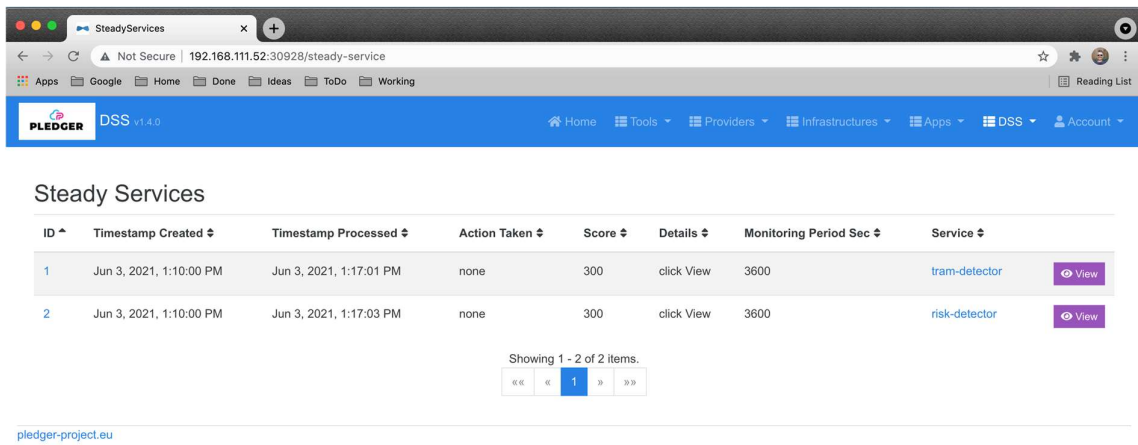


Figure 26: SteadyService found, that might require a scale down

Document name:	D4.3 Decision Support tools I				Page:	30 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status: Final

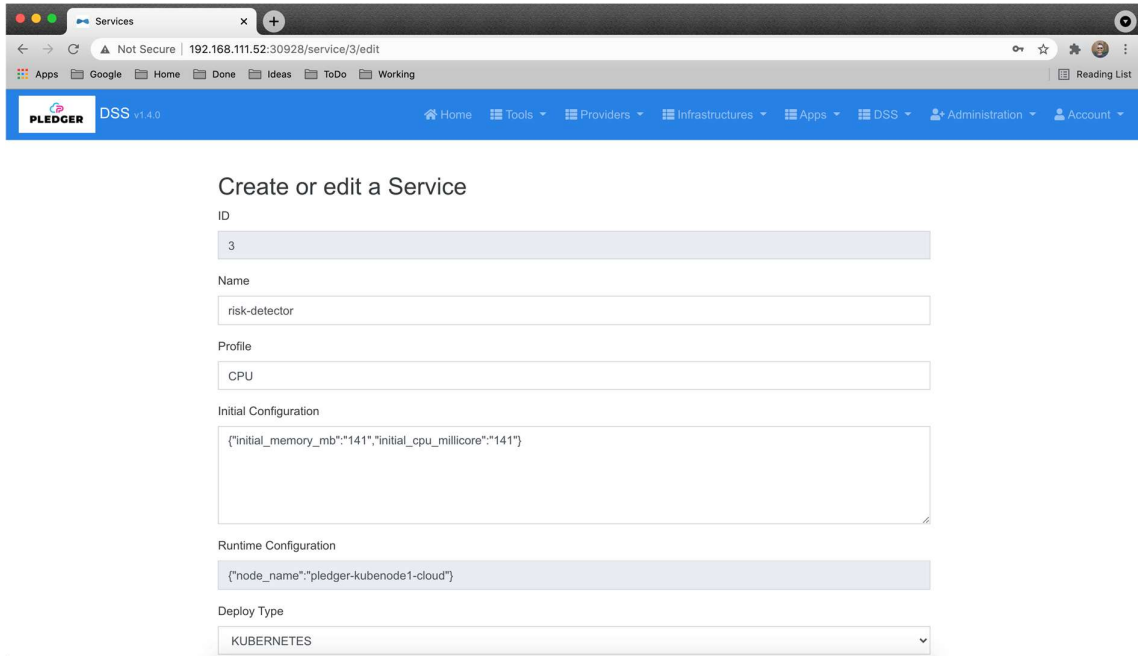


Figure 27: Service instantiated by the DSS on a Kubernetes node

5.3.4 Screenshots about custom flows and latency metrics

In this subsection, two additional tools are shown: Figure 28, with the integration with NodeRed that can be used to create custom automation using the data extracted from the Configuration service API; Figure 29 with GoldPinger, used to measure the node-to-node latency among remote infrastructures.

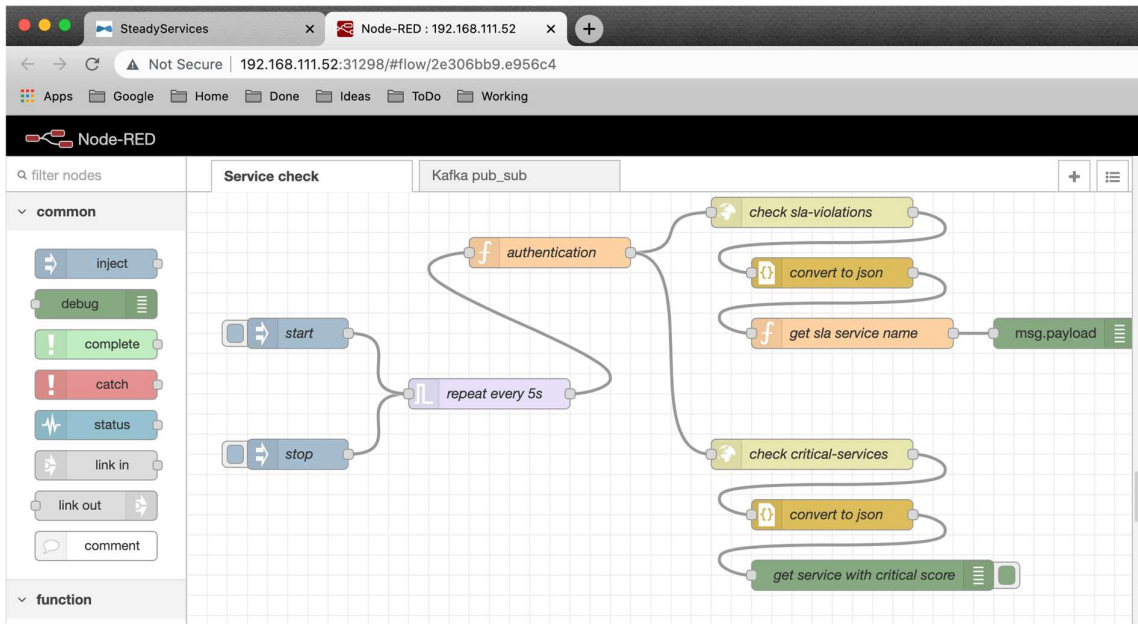


Figure 28: NodeRed integration for custom flows

Document name:	D4.3 Decision Support tools I			Page:	31 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0
				Status:	Final

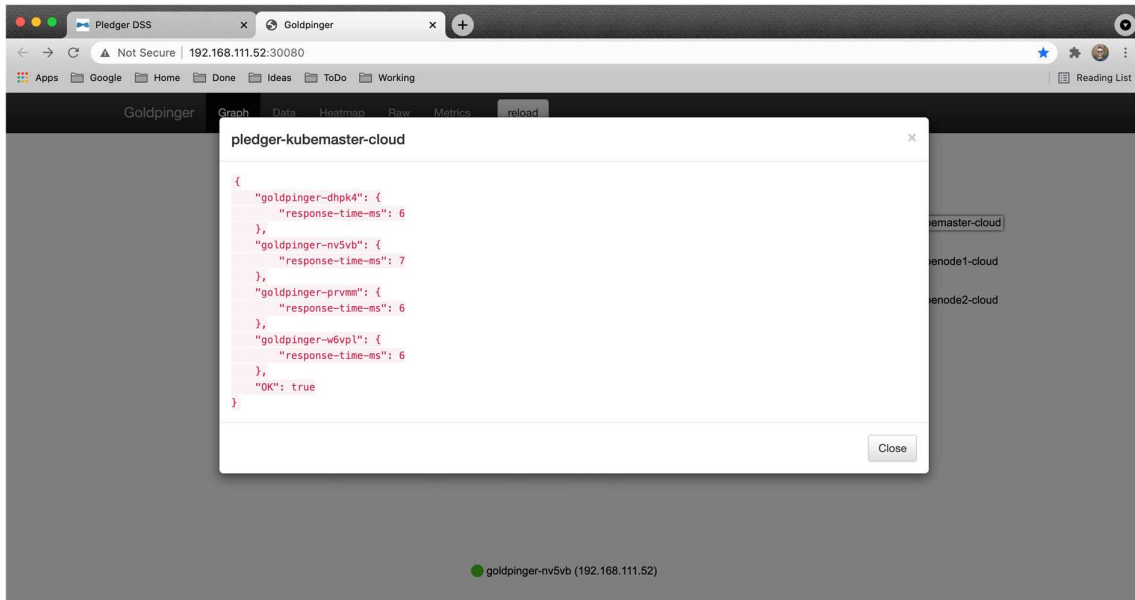


Figure 29: Latency monitoring for offloading to cloud infrastructures

Document name:	D4.3 Decision Support tools I				Page:	32 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status: Final

6 Conclusions and next steps

This document accompanies the delivery of the DSS prototype developed in the context of the project's task T4.3 "Decision Support Tools I". It presents the main functional and technical components of the prototype as well as a description of how to install and use it including references to videos demonstrations.

Such prototype has been designed and developed specifically for Pledger and released as open source with Apache 2 license.

The current version of the prototype includes a custom plugin for the orchestration of Kubernetes and Docker to support the demonstrations.

The next iteration, planned at M24 as part of the activities of task T5.2 "Pledger integrated demonstrator I", will include a more robust integration with the Orchestration and with the AppProfiler and the SLA negotiator is planned along with the actual integration of the Use Cases applications.

The final iteration, planned at M33 as part of the T4.6 "Decision Support tools II", will provide finer grained optimisations in the DSS that will integrate with the UCs and will includes latency metrics when offloading to the cloud and metrics about application boot time in order to prioritise the resources to free on the edge.

Document name:	D4.3 Decision Support tools I				Page:	33 of 34	
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status:	Final

7 References

- [1] PLEDGER. D2.3 - Pledger Overall Architecture. Voutyras, Orfefs. 2020
- [2] OpenJDK homepage, <https://openjdk.java.net/>,retrieved date 01/06/2021.
- [3] Spring homepage, <https://spring.io/projects/spring-boot>,retrieved date 01/06/2021.
- [4] Angular homepage, <https://angular.io>,retrieved date 01/06/2021.
- [5] MySQL homepage, <https://www.mysql.com/>,retrieved date 01/06/2021.
- [6] Kafka homepage, <https://kafka.apache.org/>, retrieved date 01/06/2021.
- [7] NodeRed homepage, <https://nodered.org/>, retrieved date 01/06/2021.
- [8] Swagger homepage, <https://swagger.io/>, retrieved date 01/06/2021.
- [9] NodeJS homepage, <https://nodejs.org/>, retrieved date 01/06/2021.
- [10] Npm homepage, <https://www.npmjs.com/>, retrieved date 01/06/2021.
- [11] Maven homepage, <https://maven.apache.org/>, retrieved date 01/06/2021.
- [12] Docker homepage, <https://docs.docker.com/registry/>, retrieved date 01/06/2021.
- [13] Jenkins homepage, <https://www.jenkins.io/>, retrieved date 01/06/2021.
- [14] Gitlab homepage, <https://about.gitlab.com/>, retrieved date 01/06/2021.
- [15] Kubernetes homepage, <https://kubernetes.io/>, retrieved date 01/06/2021.
- [16] Apache license, <https://www.apache.org/licenses/>, retrieved date 01/06/2021.

Document name:	D4.3 Decision Support tools I				Page:	34 of 34
Reference:	D4.3	Dissemination:	PU	Version:	1.0	Status: Final