Title:

MELODIC final release

Multi-cloud Execution-ware for Large-scale Optimized Data-Intensive Computing

H2020-ICT-2016-2017
Leadership in Enabling and Industrial Technologies; Information and Communication Technologies

Grant Agreement No.:
731664

Duration:
1 December 2016
30 November 2019

www.melodic.cloud

Deliverable reference:
D5.09

Date:
30 November 2019

Responsible partner:
7bulls

Editor(s):
Anna Wyszomirska

Author(s):
Anna Wyszomirska

Approved by:
Jörg Domaschka

Document URL:
https://melodic.cloud/deliverables/

Executive summary:

This document presents the final release of the MELODIC project including initial testing environments, the processes of testing, reported bugs and created test cases. The work has been executed during the last four months of the project. This deliverable shows the steps of the test process in the project.

The test cases comprise initial deployment testing, global reconfiguration and local reconfiguration testing, metric management testing, and reasoning-related testing. Apart from functional requirements, testing non-functional requirements are also covered in order to verify the proper implementation of the non-functional features of the MELODIC system. The non-functional testing test cases include testing of fault handling, performance testing, security testing, and other tests related to non-functional requirements.

The final release concludes the last development phase of the MELODIC in the scope of the H2020 research and innovation project.

This project has received funding from
the European Union's Horizon 2020 research
and innovation programme under grant agreement No 731664

Document	
Period Covered	M18-36
Deliverable No.	D5.09
Deliverable Title	MELODIC final release
Editor(s)	Anna Wyszomirska
Author(s)	Anna Wyszomirska
Reviewer(s)	Amir Taherkordi, Jörg Domaschka
Work Package No.	5
Work Package Title	Integration and security
Lead Beneficiary	7bulls
Distribution	PU
Version	1.0
Draft/Final	Final
Total No. of Pages	60

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

Table of Contents

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

Index of Tables

Index of Figures

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

1. Introduction

This document is the companion document for Release 3.0, the final release, of the MELODIC platform. The information presented here contains a set of instructions about environment requirements for the platform, the installation guide, descriptions of the performed test cases, a list of identified and fixed bugs, occurred during the testing process, and overview of the MELODIC software components. This work has been done during a four-months-period of the project and summarises the steps of the test process in the project.

The test cases include initial deployment, testing, global reconfiguration and local reconfiguration, testing, metric management testing and reasoning related testing. Apart from functional requirements, testing non-functional requirements are also covered in order to verify the proper implementation of the non-functional features of MELODIC. The non-functional test cases include testing of fault handling, performance testing, security testing, and other tests related to non-functional requirements.

1.1.Final release

For the MELODIC project, there are three main releases were planned:

- Release 1 – planned for 30 November 2017
- Release 2 – planned for 30 November 2018
- Release 3 – planned for 30 November 2019

The last release presented in this document mostly focuses on stability of the platform and maturing the implemented features. Detailed tests of the existing features detected many bugs, fixed by developers and enabled to stabilize project. The actions targeting release 3.0 were concentrated on simplifying the usability of the platform by creating the MELODIC User Interface for deploying customers applications, implementing visualization and metric measurements in Grafana, displaying logs in Kibana, and enhancement of the security. To achieve high quality of the software products, ensure security and privacy, enable transparent deployment of software, and minimize the risk of failure in the real world, appropriate software testing tools and techniques have been used for this release.

1.2.Structure of the document

This deliverable describes the final release and the final test environments, and contains the following information as structured within the following chapters:

- Chapter 2 – "Components – The MELODIC Architecture": A brief summary of the MELODIC architecture
- Chapter 3 – "Testing Environments": Information about the testing environments currently used in the project
- Chapter 4 – "The MELODIC Platform Installation guide": Requirements and steps on how to install the MELODIC platform
- Chapter 5 – "Testing Guide": Detailed instructions on how to execute test cases
- Chapter 6 – "Test Cases": A comparison of all executed test cases, further detailed in Appendix A
- Chapter 7 – "Bugs": Information about all bugs which have been detected during the testing of release 3.0

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

The deliverable ends with a short summary in Chapter 8.

2. Components – The MELODIC architecture

Figure 1: Overview of MELODIC architecture presents an overview of the MELODIC architecture. The figure also covers the high-level interaction between the MELODIC components, while a detailed architecture is presented in D2.2 "Architecture and initial feature definitions" (al. F. Z., 2018). The main MELODIC sub-systems are:

- Upperware: Applications and data models created through the modelling interfaces, in the form of CAMEL, are given as input to the MELODIC Upperware. The job of the Upperware is to calculate optimal data and application deployments on dynamically acquired cross-cloud resources in accordance with the specified applications and data models in CAMEL, as well as in consideration of the current Cloud performance, workload situation, and costs. An overview of the Upperware components is shown in Figure 2, further details are available in (al. F. Z., 2018)
- Executionware: The Executionware is responsible for the actual deployment of the Cloud application and its monitoring infrastructure, as well as the corresponding publishing of measurement information to the Upperware (Figure 3).
- Integration layer: The components in the MELODIC platform are integrated through two separate integration layers, the Control Plane and the Monitoring Plane (blue boxes in Figure 1: Overview of MELODIC architecture), where each brings its own set of unique requirements.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

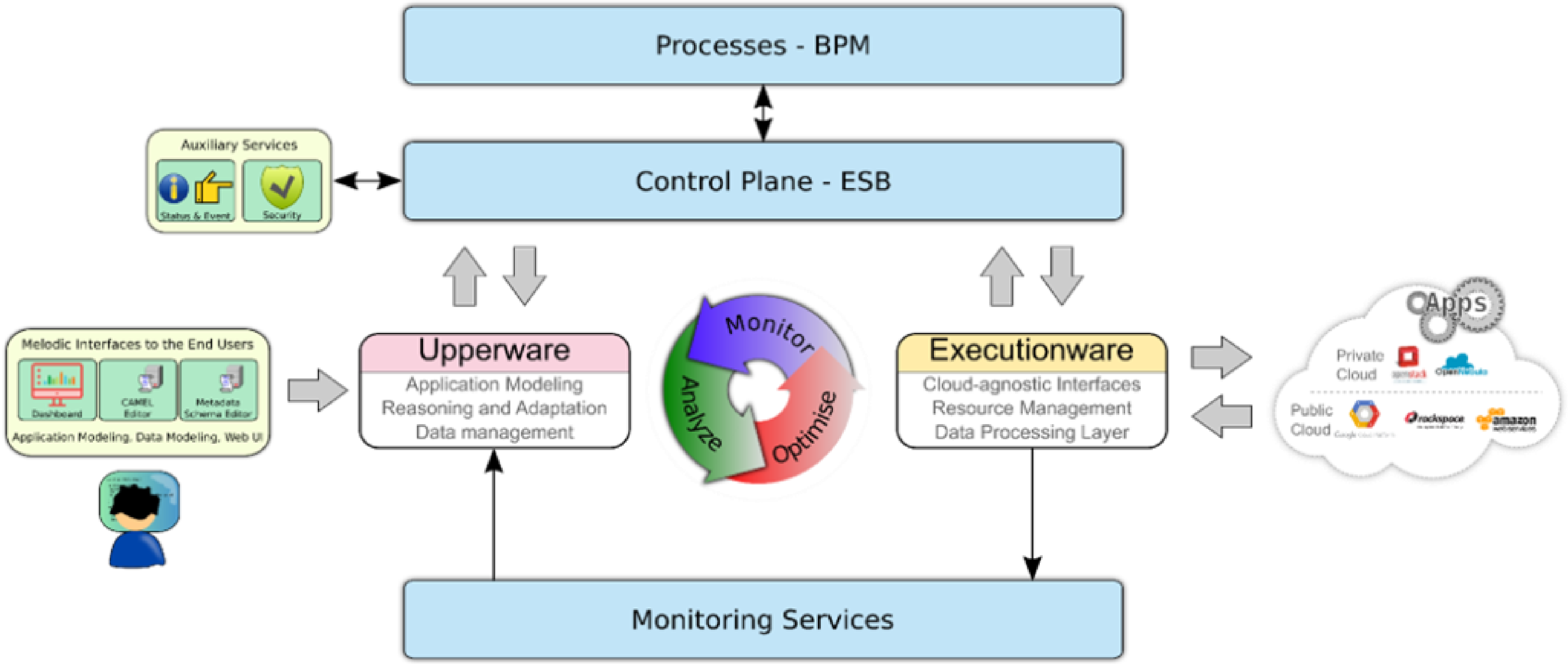


Figure 1: Overview of MELODIC architecture

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

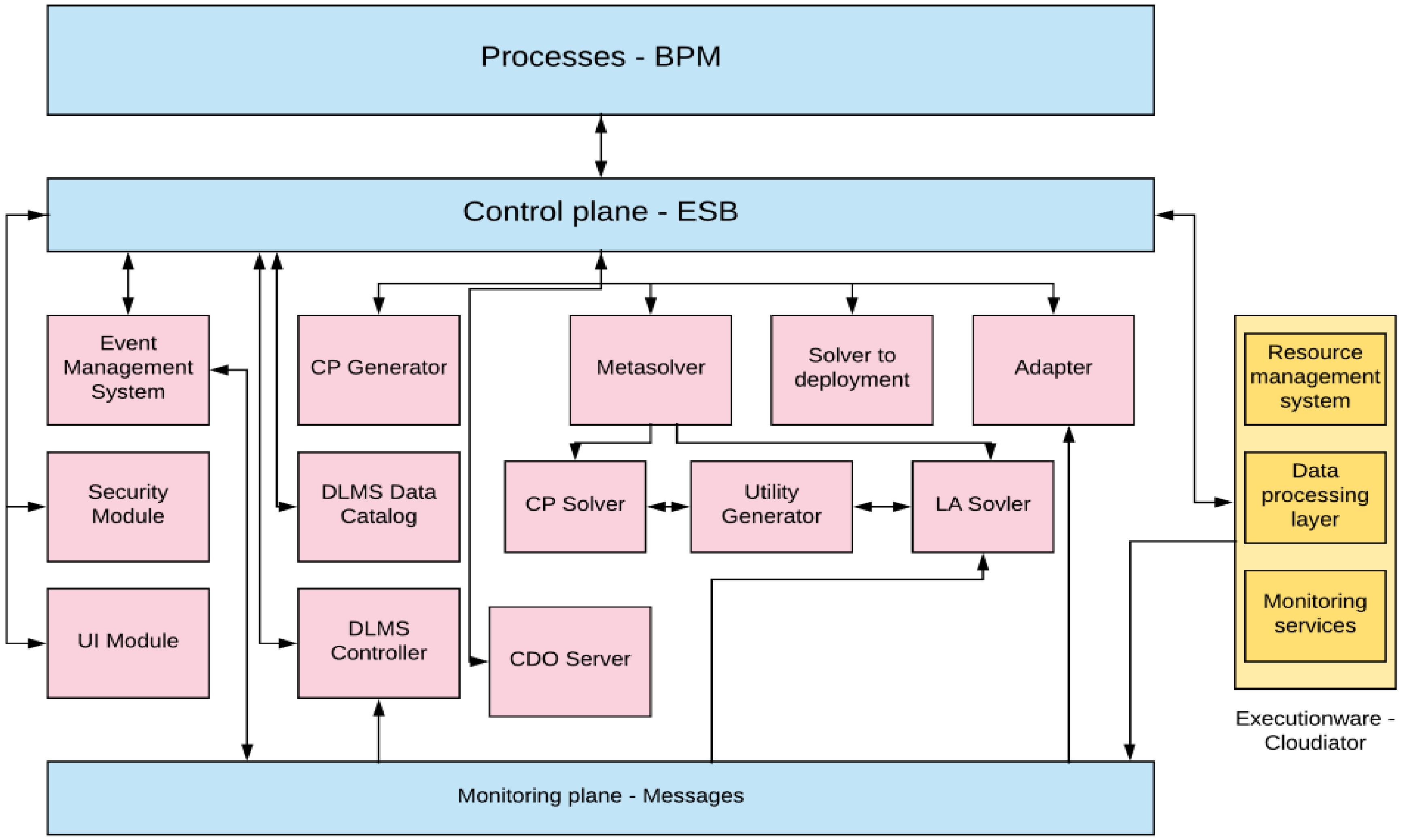


Figure 2 Overview of the Upperware Components

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

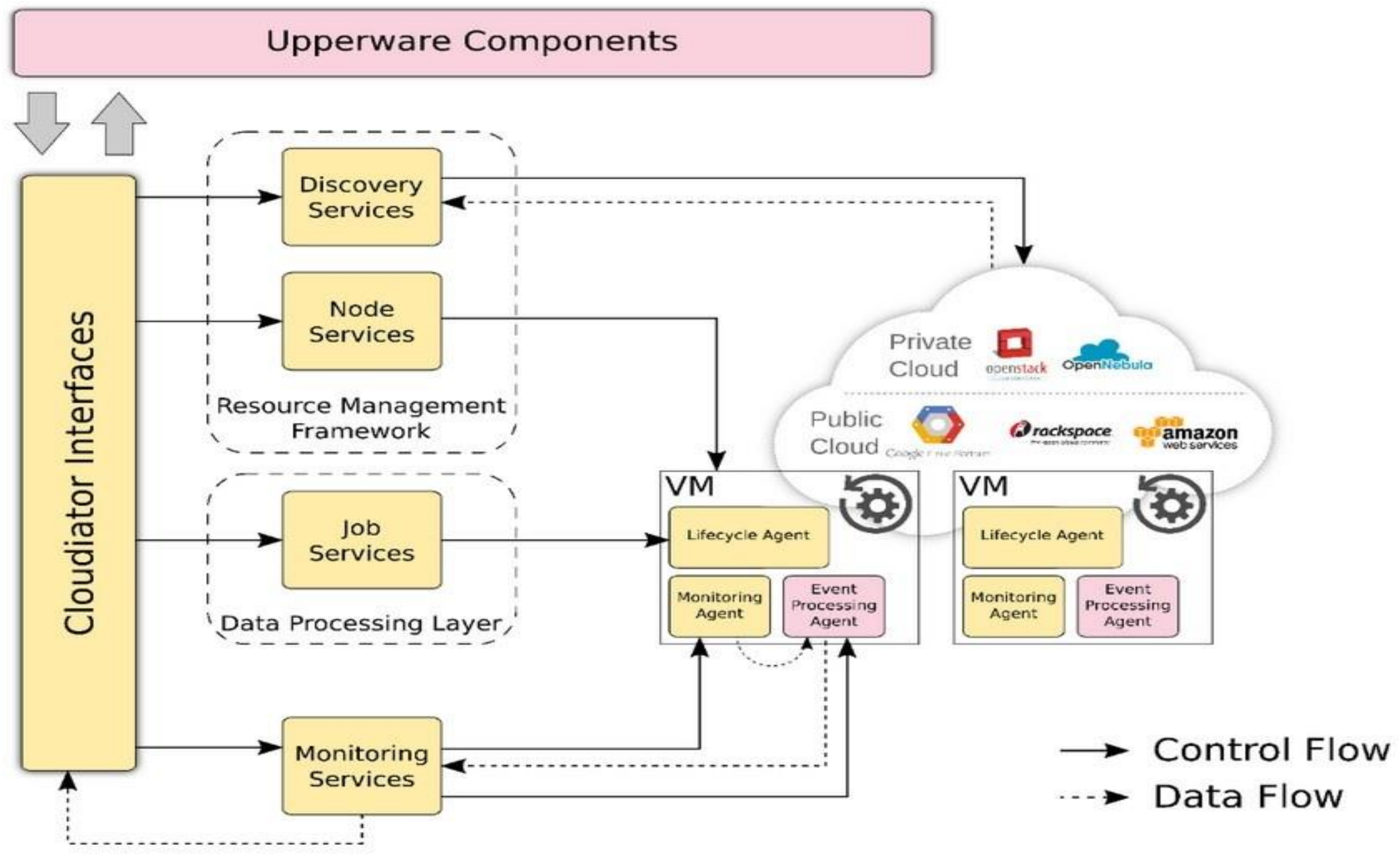


Figure 3: High-level Executionware Architecture

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

2.1. Software components

Table 1: Main components used in the MELODIC framework, with particular relevance to testing lists central components in MELODIC, including their key features and the sub-systems they belong to. D2.2 provides further details (al. F. Z., 2018).

Table 1: Main components used in the MELODIC framework, with particular relevance to testing

Component	Description, key feature		Sub-system
CP Generator	Profiling of the application and preparation of the constraint programming (CP) model		Upperware
CP Solver	Solving all types of problems encoded in the CP model using gradient descent approach.		Upperware
Solver-To-Deployment	Transforming CP models encompassing the solution produced by solvers to a provider-specific deployment model.		Upperware
Adapter	Deployment control and adaptation of multi-cloud applications		Upperware
Cloudiator[2]	Cloudiator (server part)	Deploying an application and infrastructure to the Cloud Providers.	Executionware
	Cloudiator (Cloud part)	Components deployed on the created Virtual Machines (VMs)	
Camunda[3] with status/event service	Camunda is an open-source workflow engine written in Java that can execute business processes		Integration layer
ESB	Allowing communication between all components (The exception is CDO [1])		Integration layer
ESPER	Gathering metrics		Upperware

3. Testing Environments

The testing environment is used by testers, QA analysts or other testing professionals to perform many forms of functional and non-functional testing, such as end-to-end testing, load testing and integration testing, which helps in ensuring the reliability of our code, infrastructure and overall product. It is a setup of software and hardware for the testing teams to execute test cases. In other words, the environment supports test execution with hardware, software and network configured. Such environments may vary significantly in size: the development environment is typically an individual developer's workstation, while the production environment may be a network of many geographically distributed machines in data centres, or virtual machines in cloud computing. The code, data, and configuration may be deployed in parallel, and need not to be connected to the corresponding tier. For example, pre-production code may connect to a production database.

We used testing environments for release 3.0 at 7bulls, UULM, and Simula. The corresponding parameters of the testing environments are provided in Section 3.1 to Section 3.3 . Non-functional and functional tests were executed on the same, common set of environments at 7bulls, UULM and Simula, to check if the platform works correctly in every infrastructure.

3.1. Environment at Ulm University

We used one virtual machine on which we installed the MELODIC platform, using the OpenStack-based UULM datacentre "Omistack". The specification for the UULM environment is provided in Table 2.

Table 2: Specification of the Ulm environment

environment name	machine name	IP	description
mc5	mc5	134.60.64.70	MELODIC/Cloudiator, 8CPU, 32GB of RAM, UULM Openstack

3.2. Environment in 7bulls

All tests were executed in the 7bulls environment, hosted on Amazon Web Services platform, requirements for the instance is described in Table 3.

Table 3: Specification of the 7bulls environment

environment name	machine name	IP	description
mc1	mc1	changes after each restart	MELODIC/Cloudiator, 8CPU, 64GB of RAM, AWS

3.3. Environment in Simula

During release 3.0, we used two virtual machines in the Simula environment. The specification of the instances is listed in Table 4.

Table 4: Specification of the Simula environment

environment name	machine name	IP	description
mc2	mc2	158.39.75.140	MELODIC/Cloudiator, 16CPU, 64GB of RAM, UiO Openstack
mc3	mc3	158.39.75.33	MELODIC/Cloudiator, 16CPU, 64GB of RAM, UiO Openstack

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

4. The MELODIC Platform Installation Guide

This section describes how to install the MELODIC platform from scratch. First step in for installing is to adjust a hardware, then download a public repository and execute installation script. The full guide is also available online on MELODIC's website www.MELODIC.cloud in bookmark education.

4.1. Requirements for MELODIC's machine

The requirements of hardware and operating system for the final MELODIC software:

- RAM: min 16GB, optimally 32GB
- Storage: 100GB+
- OS: Ubuntu 16.04/18.04

Table 5: List of open ports required by MELODIC specifies the incoming port numbers used by MELODIC, which are required for using the platform.

Table 5: List of open ports required by MELODIC

Port	Protocol	Component	Purpose
22	TCP	ssh	Console
80	TCP	UI frontend	MELODIC UI frontend
443	TCP	UI frontend	MELODIC UI frontend SSL
8088	TCP	ESB	REST API
8095	TCP	Camunda UI	Process UI
8181, 8998, 7077, 38000, 38100, 38200, 38300, 38400, 38500	TCP	Spark	Spark components
8080	TCP	UI	Cloudiator's webinterface
4001	TCP	Lance	etc registry
9000	TCP	Cloudiator	Cloudiator's REST API
33034	TCP	Lance	Cloudiator rmi registry
61610-61619	TCP	EMS	ActiveMQ event broker ports
2222	TCP	EMS	Baguette server port
1099	TCP	EMS	ActiveMQ JMX connector port
8111	TCP	EMS	REST API of EMS
8078	TCP	UI backend	MELODIC UI backend
2036	TCP	CDO Server	CDO Server
3077	TCP	JWT	JWT
2121	TCP	webssh	webssh
3000	TCP	Grafana	

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

8123	TCP	mq-http-adapter/UI	(optional, if diagnosis endpoint is used)
5601	TCP	Kibana	

4.2. Installation steps

Full installation includes installing of both: Upperware and the Executionware components of the MELODIC platform on one instance of virtual machine.

1. SSH login into machine (ubuntu 16.04/18.04)
2. Run the following commands (this will download installation files):

```
git clone https://bitbucket.7bulls.eu/scm/mel/utils.git
```

3. Run the MELODIC installation script:

```
sudo ~/utils/MELODIC_installation/installMELODIC.sh
```

4. After installation, a new ".profile" file has been created in home dir of the user. Load it by executing the following:

```
cd ~/
. .profile
```

5. Now the machine is ready to download and run the latest Docker images from MELODIC and the Cloudiator artefact repository. To download and start the components, use the following command:

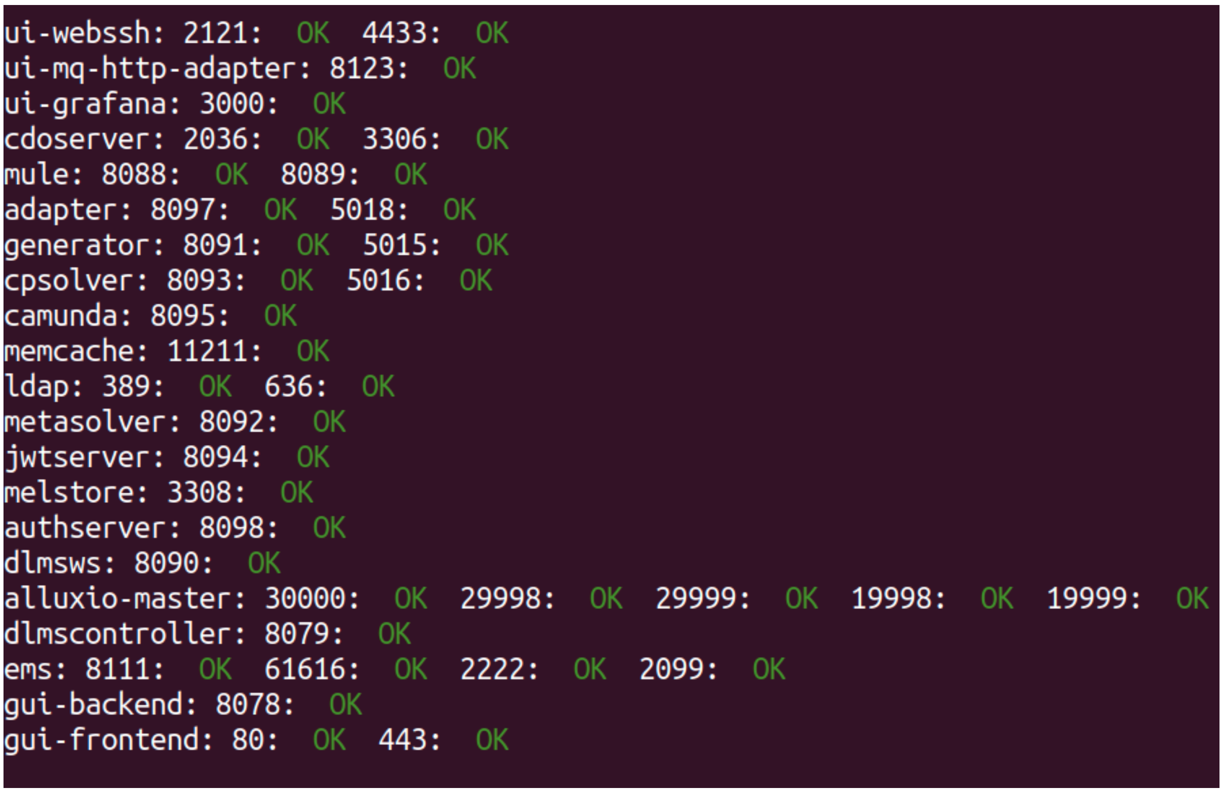
```
drestart
```

6. Running this for the first time can take more time as Docker swarm is being initialised. After running the above command, components should be started. You can check the status by running the following two commands:

```
dps
mping
```

The screenshot below shows part of the output after executing the *mping* command:

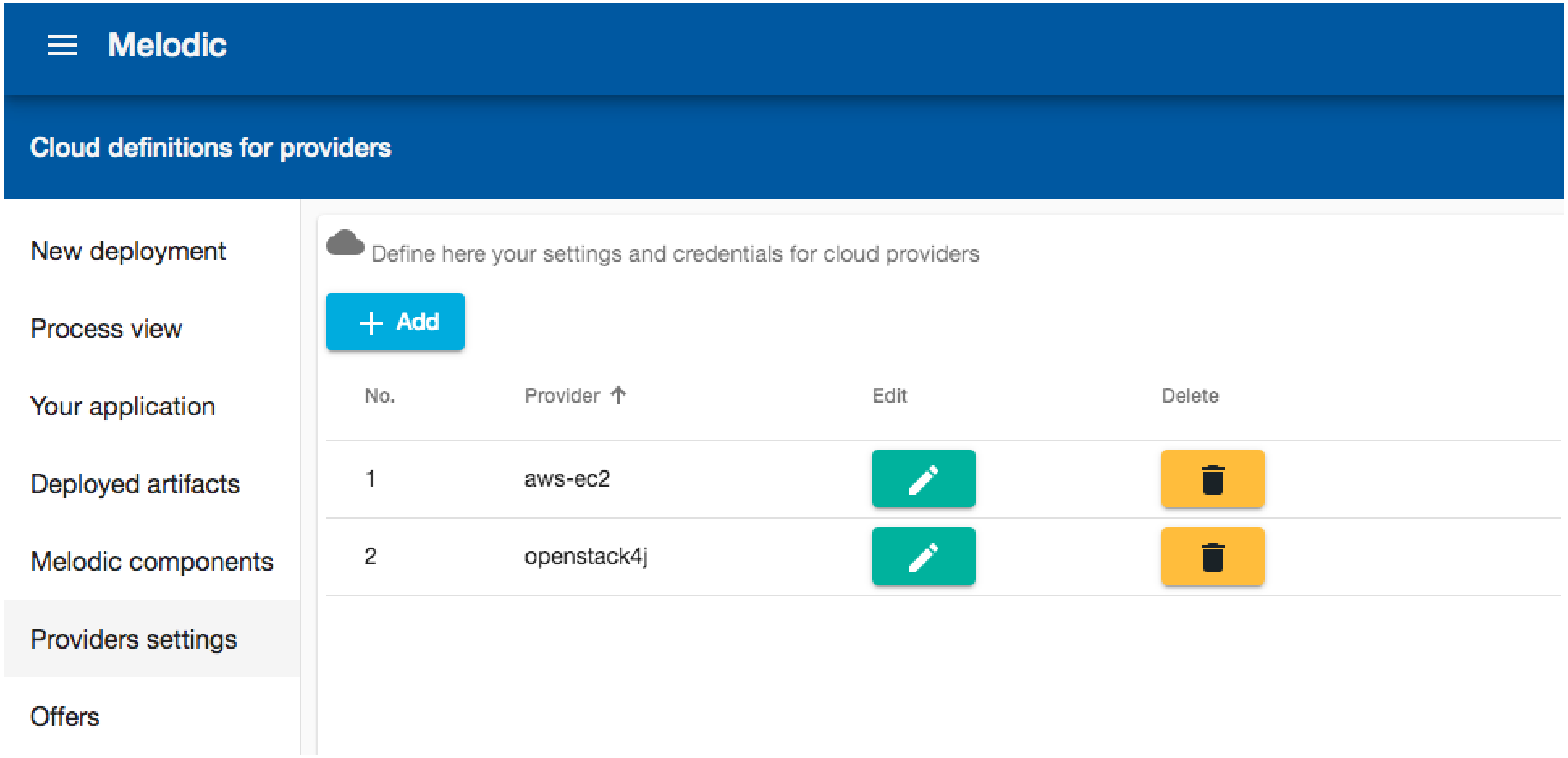
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664


```
ui-webssh: 2121:  OK  4433:  OK
ui-mq-http-adapter: 8123:  OK
ui-grafana: 3000:  OK
cdoserver: 2036:  OK  3306:  OK
mule: 8088:  OK  8089:  OK
adapter: 8097:  OK  5018:  OK
generator: 8091:  OK  5015:  OK
cpsolver: 8093:  OK  5016:  OK
camunda: 8095:  OK
memcache: 11211:  OK
ldap: 389:  OK  636:  OK
metasolver: 8092:  OK
jwtserver: 8094:  OK
melstore: 3308:  OK
authserver: 8098:  OK
dlmsws: 8090:  OK
alluxio-master: 30000:  OK  29998:  OK  29999:  OK  19998:  OK  19999:  OK
dlmscontroller: 8079:  OK
ems: 8111:  OK  61616:  OK  2222:  OK  2099:  OK
gui-backend: 8078:  OK
gui-frontend: 80:  OK  443:  OK
```
Figure 4: Status of the components

7. In order to manage MELODIC's users, you need to configure an LDAP policy and create a new LDAP users. For convenience, there is simple script packaged with MELODIC allowing to configure LDAP and add a user with admin permissions using the following commands:

```
cd ~/utils/MELODIC_installation/
./configureLdap.sh
```

8. You need to add gui-backend self-signed certificate to trust certificates in your browser. (**ATTENTION**: Execution of this point is required after each change of MELODIC IP, e.g. after each running of the 'ipupdate' command). The easiest way is to:
    a. open https://{PUBLIC_MELODIC_IP}:8078 in your browser
    b. confirm the security exception
9. Login to GUI on https://{{PUBLIC_MELODIC_IP}}
10. Go to Provider settings and update your Cloud credentials



No.	Provider ↑	Edit	Delete
1	aws-ec2		
2	openstack4j		

Figure 5: Cloud definitions for providers view in GUI

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

11. Now this machine should be ready for deploying applications.
    - GUI should be available under: {PUBLIC_MELODIC_IP} (log in with your ldap credentials)
    - Process GUI is present in UI and http://{PUBLIC_MELODIC_IP}:8095 (log in with your ldap credentials)

4.3.Useful aliases

Below you can find useful commands to manage MELODIC components:

Table 6: List of aliases

Commands
dps - displays running Docker containers (alias for sudo docker images)
mping - tests connection to each of the components
drestart - stops and then starts all the MELODIC's components
dundeploy - stops all components
ddeploy - starts all components
ipupdate - updates Cloudiator's and Upperware's env files with current IP of the machine - useful when the IP of the machine changes

5. Testing Guide

This section refers to the test procedures and provides a guideline for performing all activities within the testing processes. Each test case has an attached CAMEL model. Every created and included model specifies the testing application, including requirements, topology, metrics and credentials. During release 3.0, the process of testing was much simplified, because of the MELODIC User Interface implementation in release 2.5. Currently, the execution of the test contains seven straightforward steps, described in the following paragraph.

1. Install the MELODIC platform in detail described in point "4.2 Installation Steps".
2. Go to https://{PUBLIC_MELODIC_IP} and log in with your ldap credentials.
3. Select providers settings and add cloud definition.
4. Upload xmi file to the MELODIC User Interface, the file is available in the repository: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases?at=refs%2Fheads%2Frc3.0
5. Add required information such as application id and select cloud configuration.
6. Start application choosing green button.
7. Wait for statement "Your application successfully started".

6. Test Cases

A test case describes how to perform a specific test. The test case includes a set of test data, pre-conditions, expected results and post-conditions targeting a certain implementation, developed for a specific purpose or for the condition mapping to the test such as the execution of a program path, or to verify compliance with a specific requirement. Test cases are created by the test team; either its members or the test leader.

The test plan should also explicate the dependencies between the test cases (if needed), by clarifying which test cases should be executed before others. This whole exercise has three main benefits:

- It allows the test team to understand the system to be developed
- It serves as a review of the system specifications and requirements
- It eases solving issues, as all parties (the test team and the development teams) have the same base data (the test data; input parameters for test cases, necessary to execute test cases and to reproduce bugs, if occurring during test case execution).

The test process starts at the very beginning of a project life cycle. During the analysis and design phase, the test team starts producing a test plan, as this should be prepared as early as possible. A test plan contains test cases, which are described in detail in the D5.10 "Quality Assurance Guide" deliverable (al. M. J., 2017).

6.1.Status of test cases executed during release 3.0

For the release 3.0, we mostly tested using regression tests, which were executed for those components of the MELODIC platform in which the most substantial changes have been implemented. Parts of the old test cases were archived due to obsolete features, and new test cases referring to new functionality have been created.

Every test case was categorized into a particular non-functional or functional group according to the component's functionality, as shown in Table 7: Categories of test cases in release 3.0. The groups not specific to a use case, are further described below the table. The use cases are described in "D6.4 Use Cases Deployment and Operation" (al P. e., 2019).

Table 7: Categories of test cases in release 3.0

Functional Test Cases groups	Group ID	Test Cases executed during release 3.0 (summary)
Cloudiator/Spark	CLDTR	4
CAS - use case application	CAS	9
Initial deployment	INIT	3
CE-Traffic – use case application	CET	34
Global reconfiguration	GLOB	4
Total		**54**
Non-functional Test Cases groups		
Security Related	SEC	17

- Cloudiator/Spark: This group contains all test cases related to Cloudiator V2 and Spark, including node candidates fetching, creation of VMs, Spark framework creation, Docker Container deployment, and BYON node creation.
- Initial deployment: This group contains all test cases related to the initial deployment of an application in the MELODIC platform.
- Security (Access control rules verification): Test cases referring to selective restrictions of access to a place or resource, as well as verification of authentication and authorization.
- Global reconfiguration: Global reconfiguration refers to the reconfiguration of the application at a global scope, where a new solution is applied globally for the whole application, and not only on its specific parts (in contrast to local reconfiguration).
- Global reconfiguration test cases also verify the functionality of the EMS component. The role of EMS is to properly deliver events (produced according to CAMEL model specifications) to Metasolver.
- Security Related: This section presents test cases related to testing security in the MELODIC system and in its communication with external systems. Security, for the purpose of these tests, means authentication as well as authorization of methods invocation between components, especially methods exposed by ESB, and the usage of secure, cryptographically protected communications protocols. The security scope, the security mechanism requirements and design, as well as the security testing cases related to advanced security scenarios, will be covered in more detail in the "D5.03 Security requirements and design" deliverable (al P. S., 2018) .

The whole content of the executed test cases during release 3.0 can be found in Appendix A – Test Cases Release 3.0

Table 8: List of all executed Test Cases in release 3.0

ID	Summary	Type	Group	Status
UC-CAS-9	[T] Reconfiguration Correctly Handles Load Balancer Configuration When Removing Instance(s)	Non-Functional	CAS	Passed
UC-CAS-8	[F] Deployment According To Security Rule/s	Non-Functional	CAS	Passed
UC-CAS-7	[T] Application Is Deployed Cross-Cloud	Functional	CAS	Passed
UC-CAS-6	[T] Application Is Deployed on 1&1 IONOS	Functional	CAS	Passed
UC-CAS-5	[T] Application Is Deployed on NordicStack	Functional	CAS	Passed
UC-CAS-4	[T] Application Is Deployed on OMI	Functional	CAS	Passed

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

UC-CAS-3	[T] Application Is Deployed on AWS	Functional	CAS	Passed
UC-CAS-2	[T] Reconfiguration Happens Within Bounds Based On SLOs and UF	Functional	CAS	Passed
UC-CAS-1	[T] Multi-Component App Is Initially Correctly Deployed	Functional	CAS	Passed
UC-CET-34	[T] Multi-Cloud deployment of 3-component application on AWS, Azure and OpenStack	Functional	CET	Passed
UC-CET-33	F/T] Multi-Cloud deployment of docker app - choosing the best solution from 4 cloud providers	Functional	CET	Passed
UC-CET-32	[F/T] Multi-Cloud deployment - choosing the best solution from 4 cloud providers	Functional	CET	Passed
UC-CET-31	[F/T] Multi-Cloud deployment of dockerized application on 4 cloud providers	Functional	CET	Passed
UC-CET-30	[F/T] Scale 4-component dockerized application on Azure	Functional	CET	Passed
UC-CET-29	[F/T] Deploy 4-component dockerized application on Azure	Functional	CET	Passed
UC-CET-28	[F] Scale in 1 of 3 component on Azure	Functional	CET	Failed
UC-CET-27	[T] Scale out 1 of 3 components on Azure	Functional	CET	Passed
UC-CET-26	[T] Deploy 3-component app on Azure	Functional	CET	Passed
UC-CET-25	[T] Deploy 4-component dockerized application on OpenStack	Functional	CET	Passed
UC-CET-24	[T] Multi-Cloud deployment of 3-component application on AWS, GCP and OpenStack	Functional	CET	Passed
UC-CET-23	[T] Scale out 4-component dockerized application on GCP	Functional	CET	Passed
UC-CET-22	[T] Deploy 4-component dockerized application on GCP	Functional	CET	Passed
UC-CET-21	[T]Scale in 1 of 3 component on GCP	Functional	CET	Passed
UC-CET-20	[T] Scale out 1 of 3 components on GCP	Functional	CET	Passed
UC-CET-19	[T] Deploy 3-component app on GCP	Functional	CET	Passed

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

UC-CET-18	[T] Deploy 3-component application in 2 different locations	Functional	CET	Passed
UC-CET-17	[T] Deploy and scale dockerized application on Two Cloud Providers	Functional	CET	Passed
UC-CET-16	[T] Deploy 3-component application on Two Cloud Providers	Functional	CET	Passed
UC-CET-15	[T] Scale out 4-component dockerized application	Functional	CET	Passed
UC-CET-14	[T] Deploy 4-component dockerized application on AWS	Functional	CET	Passed
UC-CET-13	[T] Scale in 1 of 3 component	Functional	CET	Passed
UC-CET-12	[T] Scale out 1 of 3 component	Functional	CET	Passed
UC-CET-11	[T] Deploy 3-component app on Openstack when one component starts from 0 instaces	Functional	CET	Passed
UC-CET-10	[T] Deploy 3-component app when one component starts from 0 instaces	Functional	CET	Passed
UC-CET-9	[T] Deploy 3-component app on AWS	Functional	CET	Passed
UC-CET-3	[T] Deploy instance with GB RAM >= 8	Functional	CET	Passed
UC-CET-2	[T] Deploy app on AWS in United Kingdom	Functional	CET	Passed
UC-CET-1	[F] Request VM with exactly 3 cores at AWS	Functional	CET	Passed
AC-13	[F] At least one node of Deployment Instance Model (given to Adapter) has less than 16 GB RAM	Non-functional	SEC	Passed
AC-12	[T] The nodes of Deployment Instance Model (given to Adapter) have 16 GB RAM or more (each)	Non-functional	SEC	Passed
AC-11	[F] Deployment Instance Model (given to Adapter) has a total number of GB RAM (across all nodes) higher than 64	Non-functional	SEC	Passed
AC-10	[T] Deployment Instance Model (given to Adapter) has a total number of GB RAM (across all nodes) lower or equal than 64	Non-functional	SEC	Passed
AC-9	[F] Deployment Instance Model (given to Adapter) contains at least one node with NO location information	Non-functional	SEC	Passed

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

AC-8	[F] Deployment Instance Model (given to Adapter) contains at least one node NOT located in DE	Non-functional	SEC	Passed
AC-7	[T] Deployment Instance Model (given to Adapter) contains node located in DE (all of them)	Non-functional	SEC	Passed
AC-6	[F] At least one node of Deployment Instance Model (given to Adapter) has exactly 1 core	Non-functional	SEC	Passed
AC-5	[T] The nodes of Deployment Instance Model (given to Adapter) have more than 1 core (each)	Non-functional	SEC	Passed
AC-4	[F] Deployment Instance Model (given to Adapter) has a total number of cores (across all nodes) higher than 4	Non-functional	SEC	Passed
AC-3	[T] Deployment Instance Model (given to Adapter) has a total number of cores (across all nodes) lower or equal than 4	Non-functional	SEC	Passed
AC-2	[F] Successfully connect to Authorization Server and get a Negative response	Non-functional	SEC	Passed
AC-1	[T] Successfully connect to Authorization Server and get a Positive response	Non-functional	SEC	Passed
T7.4	[F] Deployment preauthorisation for FCRnew application	Non-functional	SEC	Passed
T7.3	[T] New plan deployment authorisation	Non-functional	SEC	Passed
T7.2.6	[T] Custom raw metric - FCR	Non-functional	SEC	Passed
T7.2	[F] Rejection ESB Authentication with invalid token	Non-functional	SEC	Passed
T1.5b	[T] Installation and deployment of FCR application, where one component is installed in a Docker container and another on a normal VM on two different Cloud Providers	Functional	INIT	Passed
T1.5a	[T] Installation and deployment of FCR application in Docker containers on two different Cloud Providers	Functional	INIT	Passed
T1.4	[T] Installation and deployment of FCR application in Docker container on one Cloud Provider	Functional	INIT	Passed
T4.7	[T]Global reconfiguration - GenomWithSpark, time = 7200s	Functional	GLOB	Passed

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

T4.6	[T]Global reconfiguration - GenomWithSpark, time = 3600s	Functional	GLOB	Passed
T4.5	[T]Global reconfiguration - GenomWithSpark deployed on OpenStack Cloud Provider	Functional	GLOB	Passed
T4.4b	[T]Global reconfiguration - GenomWithSpark deployed on OpenStack Cloud Provider	Functional	GLOB	Passed
T4.4	[T] Global reconfiguration - GenomWithSpark deployed on AWS Cloud Provider	Functional	GLOB	Passed
DEP4	[T] Deploy dockerized application from private registry	Functional	CLDTR	Partially Passed
DEP3	[T] Deploy application with native and dockerized component	Functional	CLDTR	Passed
DEP2	[T] Deploy dockerized application	Functional	CLDTR	Passed
DEP1	[T] Deploy native application	Functional	CLDTR	Passed

7. Bugs

During developing and testing of the MELODIC platform, bugs were found and registered in Jira. A bug is a defect in a component or system that can cause the component or system not to perform its required function. Bugs are reported to support their correction, which can then enable the system to subsequently work as planned and assumed. The software development team fix the bugs and the testers checks if its works as expected.

7.1. Reported bugs

During testing for release 3.0, 37 new bugs were reported. Table 9 shows the latest status of the corresponding bugs. The methodology of bugs management is described in detail in "D5.10 Quality Assurance Guide" deliverable (al. M. J., 2017). The status "Won't fix" means that the developers have closed this because they are not able to fix this issue or because the bug turned out to be caused by external factors.

Table 9: List of bugs reported in release 3.0

Component	Summary	Priority	Status
Adapter	Incorrect reconfiguration solution	Highest	Closed
Adapter	instance name for GCP must be a match of regex '(?:[a-z](?:[-a-z0-9]{0,61}[a-z0-9])?)	Medium	Closed
Adapter	BYON - Same NC is picked several times	Medium	Won't fix

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

Adapter	Rc3.0 build fails due to lack of tests in adapter	Low	Closed
CDO server	Not active: PackageRegistry	Lowest	New
Cloudiator	VMs on Azure are not terminated during scaling in reconfiguration	Medium	Closed
Cloudiator	Node is affected by failure of vm on GCP deployment	Highest	Closed
Cloudiator	Unexpected Timeout error message while processing request	Medium	Won't fix
Cloudiator	All processes in ERROR state: Unable to connect to lance agent	Medium	Closed
Cloudiator	Offer discovery runs almost endlessly	Medium	Won't fix
Cloudiator	Error by MELODIC installation - etcd-browser failed	Highest	Closed
Cloudiator	Cannot execute procedures by lance	Highest	Closed
Cloudiator	Metrics are not sent	High	Done
Cloudiator	Could not establish JMS communication	High	Closed
Cloudiator	env-template file changes	Medium	Closed
Cloudiator	not correct value of price	Low	New
Cloudiator	Error during installation of Lance	Highest	Done
Cloudiator	cherry pick commit to master	Medium	Closed
Cloudiator	Error during lance deployment Container reached illegal state UNKNOWN while waiting for state READY	Medium	Closed
Cloudiator	could not get node candidates	Highest	Closed
DLMS	DLMS - MySQL metrics not gathered	Highest	Won't fix
DLMS	DLMS agent fails to start on deployed VM	Highest	Closed
DLMS	Merge of DLMS WS and DLMS Controller	Highest	Won't fix

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

DLMS	DataSource definition in CAMEL should contain a possibility to provide credentials	Highest	Closed
DLMS	installation of DLMS tools only when it is specify in CAMEL model	Highest	Won't fix
EMS	EMS Client installation failed	Medium	Closed
EMS	Ems failed with SecurityException	Lowest	Closed
EMS	Multideployment of FCR and Genom applications	Lowest	Closed
Environment	unable to install MELODIC on stable branch	Lowest	Won't fix
Grafana	Missing credentials by receving metrics in ui-mq-http-adapter	High	Done
Grafana	Mq-adapter: Error while using BrokerCLient	Medium	Closed
Meta solver	Metasolver connectivity issue to the port 61616	Medium	Closed
Penalty_function	high value of penalty	High	Closed
Spark	Spark app does not work, because it is needed to be excludes the parameters from the Genome app	Medium	Closed
Spark	Spark sensors value of currentMeasurementValues.getNumCompletedTasks() is to high	Lowest	Closed
UI	WEBSSH upgrade in order to support user/password connection	Medium	Closed
UI	Bookmark "Your Application" does not show components, but deployment is successfully finished	High	Closed

8. Summary

The deliverable presented the final release of the MELODIC project and the corresponding new test cases for this release. The following information has been described in the document:

1. Introduction to the final release
2. An overview of the MELODIC architecture
3. The testing environments
4. The platform installation guide
5. Testing guides on how to execute test cases with attached CAMEL Models
6. Statuses of test cases executed during the final release
7. List of the bugs found during development and testing MELODIC release 3.0

9. References

[1] Feroz Zahid et al., D2.2 "Architecture and initial feature definitions" 2018.

[2] Małgorzata Jakubczyk et al., D5.10 "Quality Assurance Guide" 2017.

[3] Michał Semczuk et al, D6.4 "Use Cases Deployment and Operation" 2019.

[4] Paweł Skrzypek et al, D5.03 "Security requirements & design" 2018.

10. Appendix A – Test Cases Release 3.0

UC-CAS-9 [T] Reconfiguration Correctly Handles Load Balancer Configuration When Removing Instance(s)
Input Conditions:
1. Deployed and functional application 'SmartWe'.
Steps to Complete:
1. Login to both components of 'SmartDesign' via ssh 2. Increase RAM load to 80% with 'createRamLoad.sh' on both instances of component 'SmartDesign'

```
total=$(free | awk '{print $2}'| head -2| tail -1); echo "Free:"$total;
used=$(free | awk '{print $3}'| head -2| tail -1); echo "Used:"$used;
avail=$(free | awk '{print $7}'| head -2| tail -1); echo "Avail"$avail;
target=$((total / 100 * $1));
echo "Targeted:"$target;
target_n=$((target - used));
target_m=$((target_n/1000))
echo "Targeted_MB:"$target_m

if [[ $target_m =~ ^[\-0-9]+$ ]] && (( $target_m > 0)); then
stress --vm-bytes "$target_m"M --vm-keep -m 1
else
```

```
    echo "Memory load already higher than "$1"%"
    fi
```

3. Wait until third or more instances were added by reconfiguration mechanism of MELODIC
4. Assure newly added instance is available
- direct access to VM IP on SmartDesign port
- configuration of HA Proxy contains newly added machine
5. Stop script 'createRamLoad.sh'

Expected result:

1. Newly added instance is removed
- direct access to VM IP on SmartDesign port is not working any longer

- configuration of HA Proxy does NOT CONTAIN IP anymore

UC-CAS-8 [F] Deployment According To Scurity Rule/s

Input Conditions:

1. Installed and configured MELODIC platform without any application related artefacts.

Steps to Complete:

1. Have application's CAMEL model file 'smartwe-docker.xmi' ready
1. change model in a way that causes deployment of one or more components outside of Europe
1. Upload CAMEL model file to CDO MELODIC (CDO Server)
1. Initiate deployment by POST to deploymentProcess endpoint with valid credentials for AWS provider

Expected result:

- Deployment fails due to violation of security rule.

UC-CAS-7 [T] Application Is Deployed Cross-Cloud

Input Conditions:

- MELODIC platform is installed and available without any application artifacts

Steps to Complete:

1. Have application's CAMEL model file 'smartwe-docker.xmi' ready
2. change model in a way that requirements of two different components match closely offerings different between OMI and AWS
3. Upload CAMEL model file to CDO MELODIC (CDO Server)
4. Initiate deployment by POST to deploymentProcess endpoint with valid credentials for two different providers OMI and AWS

```
{
```

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

```
  "applicationId": "App-name",
  "username": "user1",
  "password": "MELODIC",
  "cloudDefinitions": [
   {
     "endpoint": "https://omistack.e-technik.uni-ulm.de:5000/v3/",
     "cloudType": "PRIVATE",
     "api": {
       "providerName": "openstack4j"
     },
     "credential": {
       "user": "openstack-login",
       "secret": "openstack-password"
     },
     "cloudConfiguration": {
       "nodeGroup": "user",
       "properties": {
         "sword.openstack4j.defaultNetwork": "e7857216-8b0f-487f-bf40-62fc90edb126",
         "sword.providerId.blacklist": "beta,cindertest,db-docker-vm-
evaluation,hdd,internal,nova,ssd,noisy-neighbour,noisy-neighbour-ctrl"
       }
     },
     "id": "7697dce7103d5926a4027d17ad975446"
   },
   {
     "endpoint": "",
     "cloudType": "PUBLIC",
     "api": {
       "providerName": "aws-ec2"
     },
     "credential": {
       "user": "....",
       "secret": "......"
     },
     "cloudConfiguration": {
       "nodeGroup": "test",
       "properties": {
         "sword.ec2.ami.cc.query": "image-id=ami-08c81eda0912baa69",
         "sword.ec2.ami.query": "image-id=ami-08c81eda0912baa69"
       }
     }
```

```
    }
  ],
  "watermark": {
    "user": "edyta",
    "system": "UI",
    "date": "2016-02-28T16:41:41+0000",
    "uuid": "fb6280ec-1ab8-11e7-93ae-92361f002AAA"
  }
}
```

Expected result:

1. Application and all components are available and functional.

UC-CAS-6 [T] Application Is Deployed On 1&1 IONOS
Input Conditions:
1. Available MELODIC platform and valid credentials for 1&1 IONOS (formerly ProfitBricks)
Steps to Complete:
1. Upload application's CAMEL model 'smartwe-docker.xmi' to MELODIC 2. Issue deployment on endpoint by providing application name and credentials for provider
Expected result:
1. Application and all components are available and functional.

UC-CAS-5 [T] Application Is Deployed On NordicStack
Input Conditions:
1. Available MELODIC platform and valid credentials for NordicStack
Steps to Complete:
1. Upload application's CAMEL model 'smartwe-docker.xmi' to MELODIC 2. Issue deployment on endpoint by providing application name and credentials for provider
Expected result:
1. Application and all components are available and functional.

UC-CAS-4 [T] Application Is Deployed On OMI
Input Conditions:
1. Available MELODIC platform and valid credentials for OMI
Steps to Complete:
1. Upload application's CAMEL model 'smartwe-docker.xmi' to MELODIC 2. Issue deployment on endpoint by providing application name and credentials for provider
Expected result:
1. Application and all components are available and functional.

UC-CAS-3 [T] Application Is Deployed On AWS
Input Conditions:
1. Available MELODIC platform and valid credentials for Amazon AWS
Steps to Complete:
1. Upload application's CAMEL model 'smartwe-docker.xmi' to MELODIC 2. Issue deployment on endpoint by providing application name and credentials for provider
Expected result:
1. Application and all components are available and functional.

UC-CAS-2 [T] Reconfiguration Happens Within Bounds Based On SLOs and UF
Input Conditions:
1. Deployed and functional application 'SmartWe'.
Steps to Complete:
1. Login to both components of 'SmartDesign' via ssh 2. Increase RAM load to 80% with 'createRamLoad.sh' on both instances of component 'SmartDesign' Sample body

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

```
total=$(free | awk '{print $2}'| head -2| tail -1); echo "Free:"$total;
used=$(free | awk '{print $3}'| head -2| tail -1); echo "Used:"$used;
avail=$(free | awk '{print $7}'| head -2| tail -1); echo "Avail"$avail;

target=$((total / 100 * $1));
echo "Targeted:"$target;

target_n=$((target - used));
target_m=$((target_n/1000))
echo "Targeted_MB:"$target_m

if [[ $target_m =~ ^[\-0-9]+$ ]] && (( $target_m > 0)); then
  stress --vm-bytes "$target_m"M --vm-keep -m 1
else
  echo "Memory load already higher than "$1"%"
```

Expected result:

1. MELODIC performed a reconfiguration and added one or more instances of the component 'SmartDesign'
2. The new instance of 'SmartDesign' was made available in the LoadBalancer component and is used for new sessions (configuration file contains IP ad new node)

UC-CAS-1 [T] Multi-Component App Is Initially Correctly Deployed on OMI Stack

Input Conditions:

1. Installed and configured MELODIC platform, without any application related artefacts.

Steps to Complete:

1. Have application's CAMEL model file 'smartwe-docker.xmi' ready
2. Upload CAMEL model file to CDO MELODIC (CDO Server)
3. Initiate deployment by POST to deploymentProcess endpoint with valid credentials for the OMI stack provider

Expected result:

1. All components are deployed and functional and within the range of their initial instance count.

UC-CET-34[T] Multi-Cloud deployment of 3-component application on AWS, Azure and OpenStack

Input Conditions:

- Available and tested MELODIC platform

- Valid credentials for AWS, Azure and OpenStack
- Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication)

CAMEL model attached.

Steps to Complete:

- Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials
- Upload xmi file with Model to GUI
- Select application id and cloud configuration
- Start application deployment choosing green button

Expected result:

Application and all components are available and functional.

UC-CET-33[F/T] Multi-Cloud deployment of docker app - choosing the best solution from 4 cloud providers

Input Conditions:

- Available and tested MELODIC platform
- Valid credentials for AWS, GCP, Azure and OpenStack
- Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication)

CAMEL model attached.

Steps to Complete:

- Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials
- Upload xmi file with Model to GUI
- Select application id and cloud configuration
- Start application deployment choosing green button

Expected result:

Application is deployed on the cheapest cloud and all components are available and functional.

UC-CET-32[F/T] Multi-Cloud deployment - choosing the best solution from 4 cloud providers

Input Conditions:

- Available and tested MELODIC platform
- Valid credentials for AWS, GCP, Azure and OpenStack
- Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication)

CAMEL model attached.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials • Upload xmi file with Model to GUI • Select application id and cloud configuration • Start application deployment choosing green button
Expected result:
Application is deployed on the cheapest cloud and all components are available and functional.

UC-CET-31[F/T] Multi-Cloud deployment of dockerized application on 4 cloud providers
Input Conditions:
• Available and tested MELODIC platform • Valid credentials for AWS, GCP, Azure and OpenStack • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials • Upload xmi file with Model to GUI • Select application id and cloud configuration • Start application deployment choosing green button
Expected result:
Application is deployed on the cheapest cloud and all components are available and functional.

UC-CET-30[F/T] Scale 4-component dockerized application on Azure
Input Conditions:
• Available and tested MELODIC platform • Valid credentials for AWS, GCP, Azure and OpenStack • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials • Upload xmi file with Model to GUI • Select application id and cloud configuration • Start application deployment choosing green button
Expected result:

- Worker component scales out - new VMs are launched and take over the execution of part of the simulations and Worker component scales in - old VMs are removed

UC-CET-29[F/T] Deploy 4-component dockerized application on Azure

Input Conditions:

- Available and tested MELODIC platform
- Valid credentials for AWS, GCP, Azure and OpenStack
- Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication)

CAMEL model attached.

Steps to Complete:

- Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials
- Upload xmi file with Model to GUI
- Select application id and cloud configuration
- Start application deployment choosing green button

Expected result:

Application is deployed on the cheapest cloud and all components are available and functional

UC-CET-28[F] Scale in 1 of 3 component on Azure

Input Conditions:

- Available and tested MELODIC platform
- Valid credentials for AWS, GCP, Azure and OpenStack
- Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication)

CAMEL model attached.

Steps to Complete:

- Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials
- Upload xmi file with Model to GUI
- Select application id and cloud configuration
- Start application deployment choosing green button

Expected result:

Worker component scales in - worker VMs are deleted when all calculations are finished

UC-CET-27[T] Scale out 1 of 3 components on Azure

Input Conditions:

- Available and tested MELODIC platform

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

• Valid credentials for AWS, GCP, Azure and OpenStack • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials • Upload xmi file with Model to GUI • Select application id and cloud configuration • Start application deployment choosing green button
Expected result:
Worker component scales in - worker VMs are deleted when all calculations are finished

UC-CET-26[T] Deploy 3-component app on Azure
Input Conditions:
• Available and tested MELODIC platform • Valid credentials for AWS, GCP, Azure and OpenStack • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials • Upload xmi file with Model to GUI • Select application id and cloud configuration • Start application deployment choosing green button
Expected result:
• Application is deployed on the cheapest cloud and all components are available and functional

UC-CET-25[T] Deploy 4-component dockerized application on OpenStack
Input Conditions:
• Available and tested MELODIC platform • Valid credentials for AWS, GCP, Azure and OpenStack • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials

- Upload xmi file with Model to GUI
- Select application id and cloud configuration
- Start application deployment choosing green button

Expected result:

- Application is deployed on the cheapest cloud and all components are available and functional

UC-CET-24[T] Multi-Cloud deployment of 3-component application on AWS, GCP and OpenStack

Input Conditions:

- Available and tested MELODIC platform
- Valid credentials for AWS, GCP, Azure and OpenStack
- Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication)

CAMEL model attached.

Steps to Complete:

- Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials
- Upload xmi file with Model to GUI
- Select application id and cloud configuration
- Start application deployment choosing green button

Expected result:

- Application is deployed on the cheapest cloud and all components are available and functional

UC-CET-23[T] Scale out 4-component dockerized application on GCP

Input Conditions:

- Available and tested MELODIC platform
- Valid credentials for AWS, GCP, Azure and OpenStack
- Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication)

CAMEL model attached.

Steps to Complete:

- Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials
- Upload xmi file with Model to GUI
- Select application id and cloud configuration
- Start application deployment choosing green button

Expected result:

Worker component scales out - new VMs are launched and take over the execution of part of the simulations

UC-CET-22[T] Deploy 4-component dockerized application on GCP
Input Conditions:
• Available and tested MELODIC platform • Valid credentials for AWS, GCP, Azure and OpenStack • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials • Upload xmi file with Model to GUI • Select application id and cloud configuration • Start application deployment choosing green button
Expected result:
• Application is deployed on the cheapest cloud and all components are available and functional

UC-CET-21[T] Scale in 1 of 3 component on GCP
Input Conditions:
• Available and tested MELODIC platform • Valid credentials for AWS, GCP, Azure and OpenStack • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials • Upload xmi file with Model to GUI • Select application id and cloud configuration • Start application deployment choosing green button
Expected result:
• Worker component scales out - new VMs are launched and take over the execution of part of the simulations

UC-CET-20[T] Scale out 1 of 3 components on GCP
Input Conditions:
• Available and tested MELODIC platform • Valid credentials for AWS, GCP, Azure and OpenStack

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

• Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials • Upload xmi file with Model to GUI • Select application id and cloud configuration • Start application deployment choosing green button
Expected result:
• Worker component scales out - new VMs are launched and take over the execution of part of the simulations

UC-CET-19[T] Deploy 3-component app on GCP
Input Conditions:
• Available and tested MELODIC platform • Valid credentials for AWS, GCP, Azure and OpenStack • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials • Upload xmi file with Model to GUI • Select application id and cloud configuration • Start application deployment choosing green button
Expected result:
• Application is deployed on the cheapest cloud and all components are available and functional

Regression (release 3.0) - UC-CET-18[T] Deploy 3-component application in 2 different locations
Input Conditions:
• Available and tested MELODIC platform • Valid credentials for AWS • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

- Upload xmi file with Model to GUI
- Select application id and cloud configuration
- Start application deployment choosing green button

Expected result:

- The components of application are launched in different regions as defined in CAMEL model. Application and all components are available and functional.

Regression (release 3.0) - UC-CET-17[T] Deploy and scale dockerized application on Two Cloud Providers

Input Conditions:

- Available and tested MELODIC platform
- Valid credentials for AWS
- Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication)

CAMEL model attached.

Steps to Complete:

- Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials
- Upload xmi file with Model to GUI
- Select application id and cloud configuration
- Start application deployment choosing green button

Expected result:

- Application is deployed on the cheapest cloud and all components are available and functional

Regression (release 3.0) - UC-CET-16[T] Deploy 3-component application on Two Cloud Providers

Input Conditions:

- Available and tested MELODIC platform
- Valid credentials for AWS and Openstack
- Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication)

CAMEL model attached.

Steps to Complete:

- Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials
- Upload xmi file with Model to GUI
- Select application id and cloud configuration
- Start application deployment choosing green button

Expected result:

- Application is deployed on the cheapest cloud and all components are available and functional

Regression (release 3.0) - UC-CET-15[T] Scale out 4-component dockerized application
Input Conditions:
• Available and tested MELODIC platform • Valid credentials for AWS • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials • Upload xmi file with Model to GUI • Select application id and cloud configuration • Start application deployment choosing green button
Expected result:
• Worker component scales out - new VMs are launched and take over the execution of part of the simulations

Regression (release 3.0) - UC-CET-14[T] Deploy 4-component dockerized application on AWS
Input Conditions:
• Available and tested MELODIC platform • Valid credentials for AWS and Openstack • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials • Upload xmi file with Model to GUI • Select application id and cloud configuration • Start application deployment choosing green button
Expected result:
• Application is deployed on the cheapest cloud and all components are available and functional

Regression (release 3.0) - UC-CET-13[T] Scale in 1 of 3 component
Input Conditions:
• Available and tested MELODIC platform

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

• Valid credentials for AWS and Openstack • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials • Upload xmi file with Model to GUI • Select application id and cloud configuration • Start application deployment choosing green button
Expected result:
• Worker component scales in - worker VMs are deleted when all calculations are finished

Regression (release 3.0) - UC-CET-12[T] Scale out 1 of 3 component
Input Conditions:
• Available and tested MELODIC platform • Valid credentials for AWS and Openstack • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:
• Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials • Upload xmi file with Model to GUI • Select application id and cloud configuration • Start application deployment choosing green button
Expected result:
Worker component scales in - worker VMs are deleted when all calculations are finished

Regression (release 3.0) - UC-CET-11[T] Deploy 3-component app on Openstack when one component starts from 0 instaces
Input Conditions:
• Available and tested MELODIC platform • Valid credentials for AWS and Openstack • Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication) CAMEL model attached.
Steps to Complete:

- Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials
- Upload xmi file with Model to GUI
- Select application id and cloud configuration
- Start application deployment choosing green button

Expected result:

- Application and all components are available and functional. Reconfiguration of application and scaling out of needed Workers is executed, while Manager sends relevant metrics because of need of simulations to be executed.

Regression (release 3.0) - UC-CET-10[T] Deploy 3-component app when one component starts from 0 instaces

Input Conditions:

- Available and tested MELODIC platform
- Valid credentials for AWS and Openstack
- Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication)

CAMEL model attached.

Steps to Complete:

- Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials
- Upload xmi file with Model to GUI
- Select application id and cloud configuration
- Start application deployment choosing green button

Expected result:

- Application and all components are available and functional. If there is a need of simulations execution, Manager sends relevant metrics to initialize a reconfiguration of application and deployment of needed Workers.

Regression (release 3.0) - UC-CET-9[T] Deploy 3-component app on AWS

Input Conditions:

- Available and tested MELODIC platform
- Valid credentials for AWS and Openstack
- Complete CAMEL model of the three-component application (which includes the definition of these three components, their installation/maintenance scripts and required communication)

CAMEL model attached.

Steps to Complete:

- Go to https://{PUBLIC_MELODIC_IP} and log in with your lpad credentials
- Upload xmi file with Model to GUI
- Select application id and cloud configuration
- Start application deployment choosing green button

Expected result:
• Application and all components are available and functional.

Regression (release 3.0) - UC-CET-3 [T] Deploy instance with GB RAM >= 8
Input Conditions:
1. Available and tested MELODIC platform 2. Valid credentials for AWS 3. CAMEL model where we request 8 or more GB RAM
Steps to Complete:
1. Upload CAMEL model 2. Trigger deployment
Expected result:
1. Instance is deployed with 8+ GB RAM.

Regression (release 3.0) - UC-CET-2 [T] Deploy app on AWS in United Kingdom
Input Conditions:
1. Available and tested MELODIC platform 2. Valid credentials for AWS 3. CAMEL model where we request deployment in UK
Steps to Complete:
1. Upload CAMEL model 2. Trigger deployment
Expected result:
1. Instance is deployed in UK on AWS.

Regression (release 3.0) - UC-CET-1 [F] Request VM with exactly 3 cores at AWS
Input Conditions:
1. Available and tested MELODIC platform 2. Valid credentials for AWS 3. CAMEL model with cores = 3
Steps to Complete:
1. Upload CAMEL model where we request 3 cores 2. Trigger deployment
Expected result:
Deployment fails as of today (2019-02-13) there are no EC2 VMs with 3 cores

Regression (release 3.0) - AC-13[F] At least one node of Deployment Instance Model (given to Adapter) has less than 16 GB RAM
Input Conditions:
1. Upperware installed and configured.

2. CAMEL model for a 2-component app, including a constraint that a VM has a maximum of 8 GB RAM. 3. Pre-authorization Policy that permits models where each node has >16 GB RAM.
Steps to Complete:
1. Login to the machine with installed MELODIC 2. Apply pre-authorization policy from link: https://bitbucket.7bulls.eu/projects/MEL/repos/security/browse/authorization-service/server/src/main/resources/config/policies/test-cases/tc-ac-12+13.xml?at=RC2.0 3. Upload model of 2-component app to CDO 4. Start deployment process
Expected result:
1. Deployment plan being Rejected.

Regression (release 3.0) - AC-12[T] The nodes of Deployment Instance Model (given to Adapter) have 16 GB RAM or more (each)
Input Conditions:
1. Upperware installed and configured 2. CAMEL model for a 2-component app, including a constraint that every VM has at least 16 GB RAM 3. Pre-authorization Policy that permits models where each node has >16 GB RAM
Steps to Complete:
1. Login to the machine with installed MELODIC 2. Apply pre-authorization policy from link: https://bitbucket.7bulls.eu/projects/MEL/repos/security/browse/authorization-service/server/src/main/resources/config/policies/test-cases/tc-ac-12+13.xml?at=RC2.0 3. Upload model of 2-component app to CDO 4. Start deployment process
Expected result:
1. Deployment plan being Accepted

Regression (release 3.0) - AC-11[F] Deployment Instance Model (given to Adapter) has a total number of GB RAM (across all nodes) higher than 64
Input Conditions:
1. Upperware installed and configured

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

2. CAMEL model for a 2-component app, including a constraint that there will be at least 2 VM instances per component and every VM instance has at least 32 GB RAM 3. Pre-authorization Policy that permits models where total RAM sums up to 64 GB RAM
Steps to Complete:
1. Login to the machine with installed MELODIC 2. Apply pre-authorization policy from link: https://bitbucket.7bulls.eu/projects/MEL/repos/security/browse/authorization-service/server/src/main/resources/config/policies/test-cases/tc-ac-10+11.xml?at=RC2.0 3. Upload model of 2-component app to CDO 4. Start deployment process
Expected result:
1. Deployment plan being Rejected.

Regression (release 3.0) - AC-10[T] Deployment Instance Model (given to Adapter) has a total number of GB RAM (across all nodes) lower or equal than 64
Input Conditions:
1. Upperware installed and configured. 2. CAMEL model for a 2-component app. 3. Pre-authorization Policy that permits models where total RAM sums up to 64 GB RAM.
Steps to Complete:
1. Login to the machine with installed MELODIC. 2. Apply pre-authorization policy from link: https://bitbucket.7bulls.eu/projects/MEL/repos/security/browse/authorization-service/server/src/main/resources/config/policies/test-cases/tc-ac-10+11.xml?at=RC2.0 3. Upload model of 2-component app to CDO. 4. Start deployment process.
Expected result:
1. Deployment plan being Accepted.

Regression (release 3.0) - AC-9[F] Deployment Instance Model (given to Adapter) contains at least one node with NO location information
Input Conditions:
• Upperware installed and configured • CAMEL model for a 2-component app (1) with no location constraints

(2) with such a constraint that will lead to selecting a *node candidate that (we know) has no location information (e.g. BYON candidate)* • Pre-authorization Policy that permits models where *all nodes are located in DE*
Steps to Complete:
1. Login to the machine with installed MELODIC 2. Apply pre-authorization policy from link: https://bitbucket.7bulls.eu/projects/MEL/repos/security/browse/authorization-service/server/src/main/resources/config/policies/test-cases/tc-ac-07+08+09.xml?at=RC2.0 3. Upload model of 2-component app to CDO 4. Start deployment process
Expected result:
Deployment plan being Rejected

Regression (release 3.0) - AC-8[F] Deployment Instance Model (given to Adapter) contains at least one node NOT located in DE
Input Conditions:
• Upperware installed and configured • CAMEL model for a 2-component app, including a constraint requiring that at least one VM is NOT located in DE • Pre-authorization Policy that permits models where all nodes are located in DE
Steps to Complete:
1. Login to the machine with installed MELODIC 2. Apply pre-authorization policy from link: https://bitbucket.7bulls.eu/projects/MEL/repos/security/browse/authorization-service/server/src/main/resources/config/policies/test-cases/tc-ac-07+08+09.xml?at=RC2.0 3. Upload model of 2-component app to CDO 4. Start deployment process
Expected result:
Deployment plan being Rejected

Regression (release 3.0) - AC-7[T] Deployment Instance Model (given to Adapter) contains node located in DE (all of them)
Input Conditions:
• Upperware installed and configured • CAMEL model for a 2-component app, including a constraint requiring that at least one VM is NOT located in DE

• Pre-authorization Policy that permits models where all nodes are located in DE
Steps to Complete:
1. Login to the machine with installed MELODIC 2. Apply pre-authorization policy from link: https://bitbucket.7bulls.eu/projects/MEL/repos/security/browse/authorization-service/server/src/main/resources/config/policies/test-cases/tc-ac-07+08+09.xml?at=RC2.0 3. Upload model of 2-component app to CDO 4. Start deployment process
Expected result:
Deployment plan being Accepted

Regression (release 3.0) - AC-6[F] At least one node of Deployment Instance Model (given to Adapter) has exactly 1 core
Input Conditions:
• Upperware installed and configured • CAMEL model for a 2-component app, including a constraint that every VM has 1 core • Pre-authorization Policy that permits models where all nodes have >1 cores each
Steps to Complete:
1. Login to the machine with installed MELODIC 2. Apply pre-authorization policy from link: https://bitbucket.7bulls.eu/projects/MEL/repos/security/browse/authorization-service/server/src/main/resources/config/policies/test-cases/tc-ac-07+08+09.xml?at=RC2.0 3. Upload model of 2-component app to CDO 4. Start deployment process
Expected result:
Deployment plan being Rejected

Regression (release 3.0) - AC-5[T] The nodes of Deployment Instance Model (given to Adapter) have more than 1 cores (each)
Input Conditions:
• Upperware installed and configured • CAMEL model for a 2-component app, including a constraint that every VM has at least 2 cores • Pre-authorization Policy that permits models where all nodes have >1 cores each
Steps to Complete:

1. Login to the machine with installed MELODIC 2. Apply pre-authorization policy from link: https://bitbucket.7bulls.eu/projects/MEL/repos/security/browse/authorization-service/server/src/main/resources/config/policies/test-cases/tc-ac-07+08+09.xml?at=RC2.0 3. Upload model of 2-component app to CDO 4. Start deployment process
Expected result:
Deployment plan being Accepted

Regression (release 3.0) - AC-4[F] Deployment Instance Model (given to Adapter) has a total number of cores (across all nodes) higher than 4.
Input Conditions:
• MELODIC installed and configured • CAMEL model for a 2-component app, including a constraint that every VM has at least 3 cores • Pre-authorization Policy that permits models where total number of cores is lower or equal to 4 cores
Steps to Complete:
1. Login to the machine with installed MELODIC 2. Apply pre-authorization policy from link: https://bitbucket.7bulls.eu/projects/MEL/repos/security/browse/authorization-service/server/src/main/resources/config/policies/test-cases/tc-ac-07+08+09.xml?at=RC2.0 3. Upload model of 2-component app to CDO 4. Start deployment process
Expected result:
Deployment plan being Rejected

Regression (release 3.0) - AC-3[T] Deployment Instance Model (given to Adapter) has a total number of cores (across all nodes) lower or equal than 4
Input Conditions:
• Upperware installed and configured • CAMEL model for a 2-component app • Pre-authorization Policy that permits models where total number of cores sums up to 4 cores
Steps to Complete:
1. Login to the machine with installed MELODIC 2. Apply pre-authorization policy from link: https://bitbucket.7bulls.eu/projects/MEL/repos/security/browse/authorization

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

-service/server/src/main/resources/config/policies/test-cases/tc-ac-07+08+09.xml?at=RC2.0 3. Upload model of 2-component app to CDO 4. Start deployment process
Expected result:
Deployment plan being Accepted

Regression (release 3.0) - AC-2[F] Successfully connect to Authorization Server and get a Negative response
Input Conditions:
• Upperware installed and configured • CAMEL model for a 2-component app • Pre-authorization Policy that permits all requests
Steps to Complete:
1. Login to the machine with installed MELODIC 2. Apply pre-authorization policy from link: https://bitbucket.7bulls.eu/projects/MEL/repos/security/browse/authorization-service/server/src/main/resources/config/policies/test-cases/tc-ac-07+08+09.xml?at=RC2.0 3. Upload model of 2-component app to CDO 4. Start deployment process
Expected result:
See log of authorization server that deployment plan being Rejected (i.e. get a negative response)

Regression (release 3.0) - AC-1[T] Successfully connect to Authorization Server and get a Positive response
Input Conditions:
• Upperware installed and configured • CAMEL model for a 2-component app • Pre-authorization Policy that permits all requests
Steps to Complete:
1. Login to the machine with installed MELODIC 2. Apply pre-authorization policy from link: https://bitbucket.7bulls.eu/projects/MEL/repos/security/browse/authorization-service/server/src/main/resources/config/policies/test-cases/tc-ac-07+08+09.xml?at=RC2.0 3. Upload model of 2-component app to CDO 4. Start deployment process
Expected result:

See log of authorization server that deployment plan being Pre-authorized (i.e. get a positive response)

Regression (release 3.0) - T7.4[F] Deployment preauthorisation for FCRnew application

Input Conditions:

1. Installed and configured MELODIC platform, without any application related artifacts.
2. At least one cloud provider has been integrated with the MELODIC platform; the user credentials for this provider should have also been supplied.
3. Cloudiator properly connected to the relevant Cloud Providers
4. Complete CAMEL model of FCR application.
 CAMEL model can be downloaded from: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases
5. Pre-authorization Policy that permits models where total cores sums <= 2
 This can be changed in: ~/conf/policies$ vi sample-PREAUTHORIZATION-policy.xml

```
<!-- ... ... total-number-of-cores <= 2-->              <xacml3:Apply
FunctionId="urn:oasis:names:tc:xacml:1.0:function:any-of">
<xacml3:Function
FunctionId="urn:oasis:names:tc:xacml:1.0:function:integer-greater-
than"/>                <xacml3:AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">2</xacml
3:AttributeValue>                <xacml3:AttributeDesignator
AttributeId="total-number-of-cores"
Category="urn:oasis:names:tc:xacml:3.0:attribute-
category:environment"
DataType="http://www.w3.org/2001/XMLSchema#integer"
MustBePresent="true"/>
```

Steps to Complete:

1. Login to the machine with installed MELODIC
2. Proceed to the ~/conf/policies and edit sample-PREAUTHORIZATION-policy.xml_OFF
 change name of sample-PREAUTHORIZATION-policy.xml_OFF to sample-PREAUTHORIZATION-policy.xml
3. Upload model of FCRnew.xmi app to CDO
4. Start deployment process
5. Check adapter.log
6. For each step, the status of the executed action should be positive except from the authorization step.

Expected result:

1. DENY decision is logged.
2. An authorization deny event is sent to ESB.
3. No virtual machine should be created.

Regression (release 3.0) - T7.3[T] Deployment preauthorisation for FCRnew application

Input Conditions:

1. Installed and configured MELODIC platform, without any application related artifacts.
2. At least one cloud provider has been integrated with the MELODIC platform; the user credentials for this provider should have also been supplied.
3. Cloudiator properly connected to the relevant Cloud Providers
4. Complete CAMEL model of FCR application.
 CAMEL model can be downloaded from:
 https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases
5. Pre-authorization Policy that permits models where total cores sums <= 2
 This can be changed in: ~/conf/policies$ vi sample-PREAUTHORIZATION-policy.xml

```
<!-- ... ... total-number-of-cores <= 3-->              <xacml3:Apply
FunctionId="urn:oasis:names:tc:xacml:1.0:function:any-of">
<xacml3:Function FunctionId="urn:oasis:names:tc:xacml:1.0:function:integer-
greater-than"/>                <xacml3:AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">3</xacml3:Attribute
Value>                <xacml3:AttributeDesignator AttributeId="total-number-of-
cores" Category="urn:oasis:names:tc:xacml:3.0:attribute-category:environment"
DataType="http://www.w3.org/2001/XMLSchema#integer"
MustBePresent="true"/>
```

Steps to Complete:

1. Login to the machine with installed MELODIC
2. Proceed to the ~/conf/policies and edit sample-PREAUTHORIZATION-policy.xml_OFF
 change name of sample-PREAUTHORIZATION-policy.xml_OFF to sample-PREAUTHORIZATION-policy.xml
3. Upload model of FCRnew.xmi app to CDO
4. Start deployment process
5. Check adapter.log

Expected result:

Positively passed

1. PERMIT authorization decision is logged.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

2. An authorization permit event is sent to ESB.
3. A certain virtual machine on the selected Cloud Provider should be created.
4. Two component application should be installed on that virtual machine.
5. The application should run properly (web page is properly displayed).

```
2020-01-08 10:19:57.738  INFO 242 --- [main-task-executor-14]
e.m.upperware.adapter.DeployCoordinator  : Authorizing deployment plan with
Authorization-Service...2020-01-08 10:19:57.739  INFO 242 --- [main-task-
executor-14] e.m.upperware.adapter.DeployCoordinator  : Deployment plan
authorized, executing...
```

Regression (release 3.0) - T7.2[F] Rejection ESB Authentication with invalid token
Input Conditions:
1. Installed and configured MELODIC platform, without any application related artifacts. • ESB configuration to require authentication (e.g. oAuth2.0) • ESB connected to a properly configured directory service (e.g. OpenLDAP) • all MELODIC components (connecting to ESB) should be configured to provide the appropriate credentials to ESB (the first time they connect) • provisioned credentials for Adapter should be invalid 2. At least one cloud provider integrated with the MELODIC platform; the user credentials for this provider should have also been supplied (included in CAMEL model- cloud credentials). 3. Meta solver configured to use CP solver for that case. 4. Cloudiator properly connected to the given Cloud Provider. 5. Complete CAMEL model of two component application (which includes the definition of the application components and their installation/maintenance scripts). CAMEL model of two component application can be downloaded from: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases?at=refs%2Fheads%2Frc3.0 6. CAMEL model of given Cloud Provider prepared and registered in the MELODIC platform with at least one virtual machine offer provided. There should be a proper configuration of the virtual machine both in CAMEL Provider model and on the Cloud Provider side. The configurations should be aligned.
Steps to Complete:
1. Go to https://{PUBLIC_MELODIC_IP} and log in with your ldap credentials 2. If you put incorrect value of ldap, uploading model and running application will not be possible
Expected result:
1. Not able to login to the MELODIC GUI

T1.5b[T] Installation and deployment of FCR application, where one component is installed in a Docker container and another on a normal VM on two different Cloud Providers
Input Conditions:
1. Installed and configured MELODIC platform, without any application related artifacts. 2. At least one cloud provider integrated with MELODIC platform, where the user has provided his/her own credentials for this provider (included in CAMEL model- cloud credentials). 3. Meta solver configured to use CP solver for that case. 4. Cloudiator properly connected to given Cloud Providers. 5. Complete CAMEL model of FCR application (which includes the definition of just one component along with its installation/management scripts). One application container will be installed as a Docker container, another as a unix process. Each component is to be deployed on different cloud providers. FCR CAMEL model can be downloaded from: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases?at=rc3.0 6. There should be a proper configuration of the virtual machine both in CAMEL Providers model and on the Cloud Providers sides.
Steps to Complete:
1. Go to http://{PUBLIC_MELODIC_IP} and log in with your ldap credentials 2. Select providers settings and add cloud definition 3. Upload xmi to GUI, file available under: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/TwoComponentApp/TwoComponentNativeAndDockerizedApp.xmi?at=refs%2Fheads%2Frc3.0 4. Add required information such as application id and select cloud configuration 5. Start application choosing green button
Expected result:
1. Two virtual machine instances should be created, one instance per each Cloud Provider. 2. The application should be installed on those VM instances (instance of first component should be installed on one VM instance and instance of second to the other VM instance). 3. The application should be run properly (The login page of DAM application should be displayed). 4. The specifc feature defined in input CAMEL model is properly applied.

T1.5a[T] Installation and deployment of application in Docker containers on two different Cloud Providers
Input Conditions:
1. Installed and configured MELODIC platform, without any application related artifacts. 2. At least one cloud provider integrated with MELODIC platform, where the user has provided his/her own credentials for this provider (included in CAMEL model- cloud credentials). 3. Meta solver configured to use CP solver for that case. 4. Cloudiator properly connected to given Cloud Providers. 5. Complete CAMEL model of FCR application (which includes the definition of just one component along with its installation/management scripts). One application container will be installed as a Docker container, another as a unix process. Each component is to be deployed on different cloud providers. FCR CAMEL model can be downloaded from: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases?at=rc3.0 6. There should be a proper configuration of the virtual machine both in CAMEL Providers model and on the Cloud Providers sides.
Steps to Complete:
1. Go to http://{PUBLIC_MELODIC_IP} and log in with your ldap credentials 2. Select providers settings and add cloud definition 3. Upload xmi to GUI, file available under: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/TwoComponentApp/TwoComponentNativeAndDockerizedApp.xmi?at=refs%2Fheads%2Frc3.0 4. Add required information such as application id and select cloud configuration 5. Start application choosing green button For each step, the status of the executed action should be positive.
Expected result:
Positively passed. Executed during Release 3.0 1. Two VM instances should be created+ using the selected Cloud Provider. 2. The application should be installed on those VM instances_ (where each application component is deployed and installed on a different VM instance). 3. The application should be run properly.

T1.4[T] Installation and deployment of application in Docker container on one Cloud Provider

Input Conditions:
1. Installed and configured MELODIC platform, without any application related artifacts. 2. At least one cloud provider integrated with MELODIC platform, where the user has provided his/her own credentials for this provider (included in CAMEL model- cloud credentials). 3. Meta solver configured to use CP solver for that case. 4. Cloudiator properly connected to given Cloud Providers. 5. Complete CAMEL model of FCR application (which includes the definition of just one component along with its installation/management scripts). One application container will be installed as a Docker container, another as a unix process. Each component is to be deployed on different cloud providers. FCR CAMEL model can be downloaded from: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases?at=rc3.0 6. There should be a proper configuration of the virtual machine both in CAMEL Providers model and on the Cloud Providers sides.
Steps to Complete:
1. Go to http://{PUBLIC_MELODIC_IP} and log in with your ldap credentials 2. Select providers settings and add cloud definition 3. Upload xmi to GUI, file available under: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/TwoComponentApp/TwoComponentNativeAndDockerizedApp.xmi?at=refs%2Fheads%2Frc3.0 4. Add required information such as application id and select cloud configuration 5. Start application choosing green button For each step, the status of the executed action should be positive.
Expected result:
Positively passed. 1. Virtual machine on the selected Cloud Provider should be created. 2. The application should be installed on that machine. 3. The application should be run properly (Website should be displayed properly).

Regression (release 3.0) - T4.6[T]Global reconfiguration - GenomWithSpark, time = 3600s
Input Conditions:
1. Installed and configured MELODIC platform, without any application related artifacts. 2. AWS Cloud Provider integrated with MELODIC platform for which the respective user credentials have been supplied.

3. Cloudiator properly connected to given Cloud Providers.
4. Complete CAMEL model of a GenomWithSpark application. The CAMEL model should include definition of events and metrics needed to execute the particular test case, parameter of time should be = 3600s.
5. Model .xmi can be downloaded from: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/ and it's attached as attachment.

Steps to Complete:

1. Go to http://{PUBLIC_MELODIC_IP} and log in with your ldap credentials
2. Select providers settinngs and add cloud definition
3. Upload xmi to GUI, file available under: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/TwoComponentApp/TwoComponentNativeAndDockerizedApp.xmi?at=refs%2Fheads%2Frc3.0
4. Add required information such as application id and select cloud configuration
5. Start application choosing green button

For each step, the status of the executed action should be positive.

Expected result:

1. Application should be reconfigured according to defined SLOs.
 For time 3600s should be more workers deployed.
2. EMS properly delivers events (produced according to camel model specifications) to metasolver.
3. Application should work properly; this means that Spark Master on {{MELODIC_IP:8181}} should display the workers and application status.

Regression (release 3.0) - T4.5[T]Global reconfiguration - GenomWithSpark, time = 7200s

Input Conditions:

1. Installed and configured MELODIC platform, without any application related artifacts.
2. AWS Cloud Provider integrated with MELODIC platform for which the respective user credentials have been supplied.
3. Cloudiator properly connected to given Cloud Providers.
4. Complete CAMEL model of a GenomWithSpark application. The CAMEL model should include definition of events and metrics needed to execute the particular test case, parameter of time should be = 3600s.
5. Model .xmi can be downloaded from: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/ and it's attached as attachment.

Steps to Complete:

1. Go to http://{PUBLIC_MELODIC_IP} and log in with your ldap credentials
2. Select providers settings and add cloud definition
3. Upload xmi to GUI, file available under: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/TwoComponentApp/TwoComponentNativeAndDockerizedApp.xmi?at=refs%2Fheads%2Frc3.0
4. Add required information such as application id and select cloud configuration
5. Start application choosing green button

For each step, the status of the executed action should be positive.

Expected result:

1. Application should be reconfigured according to defined SLOs.
   For time 3600s should be more workers deployed.
2. EMS properly delivers events (produced according to camel model specifications) to metasolver.
3. Application should work properly; this means that Spark Master on {{MELODIC_IP:8181}} should display the workers and application status.

Regression (release 3.0) - T4.4b[T]Global reconfiguration - GenomWithSpark deployed on OpenStack Cloud Provider

Input Conditions:

1. Installed and configured MELODIC platform, without any application related artifacts.
2. Openstack Cloud Provider integrated with MELODIC platform for which the respective user credentials have been supplied.
3. Cloudiator properly connected to given Cloud Providers.
4. Complete CAMEL model of a GenomWithSpark application. The CAMEL model should include definition of events and metrics needed to execute the particular test case, parameter of time should be = 3600s.
5. Model.xmi can be downloaded from: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/ and it's attached as attachment.

Steps to Complete:

1. Go to http://{PUBLIC_MELODIC_IP} and log in with your ldap credentials
2. Select providers settings and add cloud definition
3. Upload xmi to GUI, file available under: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/TwoComponentApp/TwoComponentNativeAndDockerizedApp.xmi?at=refs%2Fheads%2Frc3.0
4. Add required information such as application id and select cloud configuration

5. Start application choosing green button

For each step, the status of the executed action should be positive.

Expected result:

1. Application should be reconfigured according to defined SLOs.
   One worker created as first then second one should be added.

```
slo NotFinished constraint GenomConstraintModel.NotFinishedOnTime
constraint model GenomConstraintModel{
     variable constraint WorkerCoresGreaterThanMinimumCores :
GenomMetricModel.WorkerCoresGreaterThanMinimumCores > 0.0
     metric constraint NotFinishedOnTime :
[GenomMetricModel.NotFinishedOnTimeContext] >= 0.0
  }
```

2. Application should work properly; this means that its web page should be properly displayed (continuing the previous example with an Apache web server).

Regression (release 3.0) - T4.4a[T] Global reconfiguration - GenomWithSpark deployed on AWS Cloud Provider

Input Conditions:

1. Installed and configured MELODIC platform, without any application related artifacts.
2. AWS Cloud Provider integrated with MELODIC platform for which the respective user credentials have been supplied.
3. Cloudiator properly connected to given Cloud Providers.
4. Complete CAMEL model of a GenomWithSpark application. The CAMEL model should include definition of events and metrics needed to execute the particular test case, parameter of time should be = 3600s.
5. Model .xmi can be downloaded from: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/ and it's attached as attachment.

Steps to Complete:

1. Go to http://{PUBLIC_MELODIC_IP} and log in with your ldap credentials
2. Select providers settings and add cloud definition
3. Upload xmi to GUI, file available under: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/TwoComponentApp/TwoComponentNativeAndDockerizedApp.xmi?at=refs%2Fheads%2Frc3.0
4. Add required information such as application id and select cloud configuration
5. Start application choosing green button

For each step, the status of the executed action should be positive.

Expected result:

1. Application should be reconfigured according to defined SLOs.
   One worker created as first then second one should be added.

slo NotFinished constraint GenomConstraintModel.NotFinishedOnTime
constraint model GenomConstraintModel{
    variable constraint WorkerCoresGreaterThanMinimumCores :
GenomMetricModel.WorkerCoresGreaterThanMinimumCores > 0.0
    metric constraint NotFinishedOnTime :
[GenomMetricModel.NotFinishedOnTimeContext] >= 0.0
  }

2. Application should work properly; this means that its web page should be properly displayed (continuing the previous example with an Apache web server).

Regression (release 3.0) - DEP4 [T] Deploy dockerized application from private registry

Input Conditions:

- image in private registry present
- Use CAS use case application

Steps to Complete:

- create a Job with the DockerInterface and specify the private registry with valid credentials to retrieve image

Expected result:

- Application successfully deployed from private registry, running and usable

Regression (release 3.0) - DEP3 [T] Deploy application with native and dockerized component

Input Conditions:

1. Installed and configured MELODIC platform, without any application related artifacts.
2. At least one cloud provider has been integrated with the MELODIC platform; the user credentials for this provider should have also been supplied.
3. Cloudiator properly connected to the relevant Cloud Providers
4. Complete CAMEL model of TwoComponentApp application.
   CAMEL model can be downloaded from:
   https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases

Steps to Complete:

1. Go to http://{PUBLIC_MELODIC_IP} and log in with your ldap credentials
2. Select providers settinngs and add cloud definition

3. Upload xmi to GUI, file available under:
   https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/TwoComponentApp/TwoComponentNativeAndDockerizedApp.xmi?at=refs%2Fheads%2Frc3.0
4. Add required information such as application id and select cloud configuration
5. Start application choosing green button

For each step, the status of the executed action should be positive.

Expected result:

- Application and both components successfully deployed, running and usable

Regression (release 3.0) - DEP2 [T] Deploy dockerized application
Input Conditions:
1. Installed and configured MELODIC platform, without any application related artifacts. 2. At least one cloud provider has been integrated with the MELODIC platform; the user credentials for this provider should have also been supplied. 3. Cloudiator properly connected to the relevant Cloud Providers 4. Complete CAMEL model of TwoComponentApp application. CAMEL model can be downloaded from: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases
Steps to Complete:
1. Go to http://{PUBLIC_MELODIC_IP} and log in with your ldap credentials 2. Select providers settings and add cloud definition 3. Upload xmi to GUI, file available under: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/TwoComponentApp/TwoComponentNativeAndDockerizedApp.xmi?at=refs%2Fheads%2Frc3.0 4. Add required information such as application id and select cloud configuration 5. Start application choosing green button For each step, the status of the executed action should be positive.
Expected result:
Application and both components successfully deployed, running and usable

Regression (release 3.0) - DEP1 [T] Deploy native application
Input Conditions:
1. Installed and configured MELODIC platform, without any application related artifacts. 2. At least one cloud provider has been integrated with the MELODIC platform; the user credentials for this provider should have also been supplied.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664

3. Cloudiator properly connected to the relevant Cloud Providers 4. Complete CAMEL model of TwoComponentApp application. CAMEL model can be downloaded from: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases
Steps to Complete:
1. Go to http://{PUBLIC_MELODIC_IP} and log in with your ldap credentials 2. Select providers settings and add cloud definition 3. Upload xmi to GUI, file available under: https://bitbucket.7bulls.eu/projects/TST/repos/MELODIC/browse/TestCases/TwoComponentApp/TwoComponentNativeAndDockerizedApp.xmi?at=refs%2Fheads%2Frc3.0 4. Add required information such as application id and select cloud configuration 5. Start application choosing green button For each step, the status of the executed action should be positive.
Expected result:
Positively passed Regression Testing according to Test Case: MT-142. Executed during Release 3.0 • application is running and can be accessed via web dashboard or API (depending on the application)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731664