D3.4 DEMETER Technology Integration Tools – Release 2

Dissemination level: Public
Submission date: 30 June 2021

Contents

1 Executive Summary 7
2 Acronyms 9
3 List of Authors 11
4 Document History 11
5 Introduction 12
6 Architecture of the Reference Implementation 14
6.1 Architecture (Physical view, Process view) 14
6.2 Requirements Mapping 19
7 Brokerage Service Environment 23
7.1 Description 23
7.2 Development View 23
7.2.1 Component diagram 23
7.2.2 Building blocks 24
7.3 Process View 25
7.4 Interfaces 25
7.4.1 Data models used in interfaces 25
7.4.2 Description of API's 27
7.5 Technologies and implementation details 32
8 Access Control Server 34
8.1 Description 34
8.2 Development View 35
8.2.1 Component diagram 35
8.2.2 Building blocks (components) 35
8.2.3 Process View 40
8.3 Identity Manager 41
8.3.1 Interfaces 41

8.3.2 Technologies and implementation details ... 45
8.4 XACML PDP ... 45
8.4.1 Interfaces ... 45
8.4.2 Technologies and implementation details ... 49
8.5 Capability Manager ... 49
8.5.1 Interfaces ... 50
8.5.2 Technologies and implementation details ... 53
8.6 PEP Proxy ... 53
8.6.1 Interfaces ... 53
8.6.2 Data models used in interfaces ... 53
8.6.3 Technologies and implementation details ... 53
8.7 Traceability Agent ... 53
8.7.1 Interfaces ... 54
9 DEMETER Enabler Hub ... 60
9.1 Description ... 60
9.2 Development view ... 61
9.2.1 Component diagram ... 61
9.2.2 Building blocks (components) ... 62
9.3 Process View ... 82
9.3.1 DEH Authentication & Authorisation ... 82
9.3.2 DEH Resource Registry Management ... 83
9.4 Interfaces ... 87
9.4.1 DEH Resource Data Model ... 88
9.4.2 Description of API's ... 89
9.5 Technologies and implementation details ... 113
10 Core Enablers for Integration ... 114
10.1 Functional Interoperability Enabler (FIE) ... 114
10.1.1 Functionality description ... 114
10.1.2 Interaction with other Enablers ... 114
10.1.3 Dependencies on other Core/Advanced Enablers ... 114
10.1.4 Deployment considerations ... 114
10.1.5 Technical description ... 114
10.2 Access Control Enabler ... 121
10.2.1 Authentication Security Enabler ... 121
10.2.2 Authorisation Security Enabler ... 125

10.2.3 Communications and Networking Enabler: TLS/DTLS 128
10.2.4 Communications and Networking Enabler: JSON/XML Encryption 131
10.3 DEH Client Enabler 136
11 Infrastructure and Tools 157
11.1 Tools in DEMETER 157
11.1.1 Version control 157
11.1.2 CI/CD 157
11.1.3 Issue Trackers 159
11.1.4 Requirements management and traceability 161
11.2 Infrastructure 163
11.3 Internal Documentation 164
12 Verification and Validation Report 168
12.1 Verification Plan 168
12.2 Validation Plan 169
12.3 Report collections status 170
13 Conclusions 171

List of Figures

Figure 1: DEMETER main elements 12
Figure 2: Reference Implementation Deployment diagram 14
Figure 3: Pilots Deployment diagram 15
Figure 4: Deployment of applications in relation to DEMETER 15
Figure 5: Sequence diagram - Provider 16
Figure 6: Sequence diagram - Consumer 17
Figure 7: Activity diagram - Provider 18
Figure 8: Activity diagram - Consumer 18
Figure 9: BSE component diagram 24
Figure 10: BSE sequence diagram 25
Figure 11: BSE online Swagger documentation 33
Figure 12: Security component diagram 35
Figure 13: XACML PDP authorisation sequence diagram 37
Figure 14: Capability Manager with capability token sequence diagram 38
Figure 15: PEP Proxy flow diagram 39
Figure 16: Traceability sequence diagrams 40
Figure 17: Authentication and Authorisation sequence diagram to access DEMETER resources 40
Figure 18: DEH Functional Architecture 60
Figure 19: DEMETER Enabler HUB component diagram 61
Figure 20: DEMETER Enabler Hub Dashboard building block components 63
Figure 21: DEH Dashboard user login 64
Figure 22: DEH Dashboard – Resource catalogue 65
Figure 23: DEH Dashboard - DEH Resource Details 65
Figure 24: DEH Dashboard search filter on resource catalogue 66
Figure 25: DEH Dashboard – create new DEH resource form first part 67
Figure 26: DEH Dashboard – create new DEH resource form second part 68
Figure 27: DEH Dashboard edit and delete functions 69
Figure 28: DEH Dashboard – edit existing DEH resource form first part 70
Figure 29: DEH Dashboard – edit existing DEH resource form second part 71
Figure 30: DEH Metrics Dashboard 72
Figure 31: DYMER administration dashboard 72
Figure 32: DYMER administration – template management 73
Figure 33: DYMER administration– models/forms management 74
Figure 34: DYMER administration– models/template manage css/javascript 74
Figure 35: DYMER administration– models/template crud css/javascript 75
Figure 36: DYMER administration– models/template manage edit css/javascript 75
Figure 37: DYMER administration - Bridge entities configuration list 76
Figure 38: DYMER administration - Bridge entities configuration first part 77
Figure 39: DYMER administration - Bridge entities configuration second part 78
Figure 40: Resource Registry Management building block diagram 79
Figure 41: DEMETER Enabler Hub Authentication and Authorisation sequence diagram 83
Figure 42: DEH Resource Registry activity diagram 84
Figure 43: DEH Resource Discovery activity diagram 85
Figure 44: DEH Resource Registry sequence diagram 86
Figure 45: DEH Resource Discovery sequence diagram 87
Figure 46: Functionality Enabler (FI) Sequence diagram 117

Figure 47: FI Activity diagram - Provider........118
Figure 48: FI Activity diagram - Consumer........118
Figure 49: Authentication function sequence diagrams........124
Figure 50: Authorisation DCapBAC access control sequence diagram........127
Figure 51: OpenSSL TLS/DTLS activity diagram........129
Figure 52: OpenSSL TLS/DTLS sequence diagram........130
Figure 53: XML encryption and decryption activity diagram........135
Figure 54: DEH Client Docker Engine API........137
Figure 55: DEH Client Docker Engine Remote API........137
Figure 56: DEH Client and Docker Host Secured Communication........138
Figure 57: DEH Client - Configure Docker Host........139
Figure 58: DEH Client CLI Interaction with Remote Docker Daemon Docker Host........139
Figure 59: DEH Client Configurable Parameters........141
Figure 60: DEH Client Monitored Docker Events........142
Figure 61: DEH Client Architecture Diagram........143
Figure 62: DEH Client UML Sequence Diagram........145
Figure 63: DEH Client Module Resource Consumption Metrics Data Model........150
Figure 64: DEH Client Module Generated Real Time Resource Consumption Metrics Data........155
Figure 65: CI/CD operations at work........157
Figure 66: Example job file........159
Figure 67: Issue Tracker - Board view........160
Figure 68: Issue Tracker - List view........160
Figure 69: Issue Tracker - Board view........161
Figure 70: Issue Tracker - List view........161
Figure 71: Requirements management in GitLab........162
Figure 72: Monitoring Production environment........163
Figure 73: DEMETER CI/CD infrastructure........164
Figure 74: Wiki page in DEMETER GitLab........164
Figure 75: README file in GitLab........165
Figure 76: Integration Flow document table of contents........166
Figure 77: Integration Flow document contents........167
Figure 78: DEMETER's Component Test Levels........169
Figure 79: Verification and Validation process........169
Figure 80: Example job file........272

List of Tables

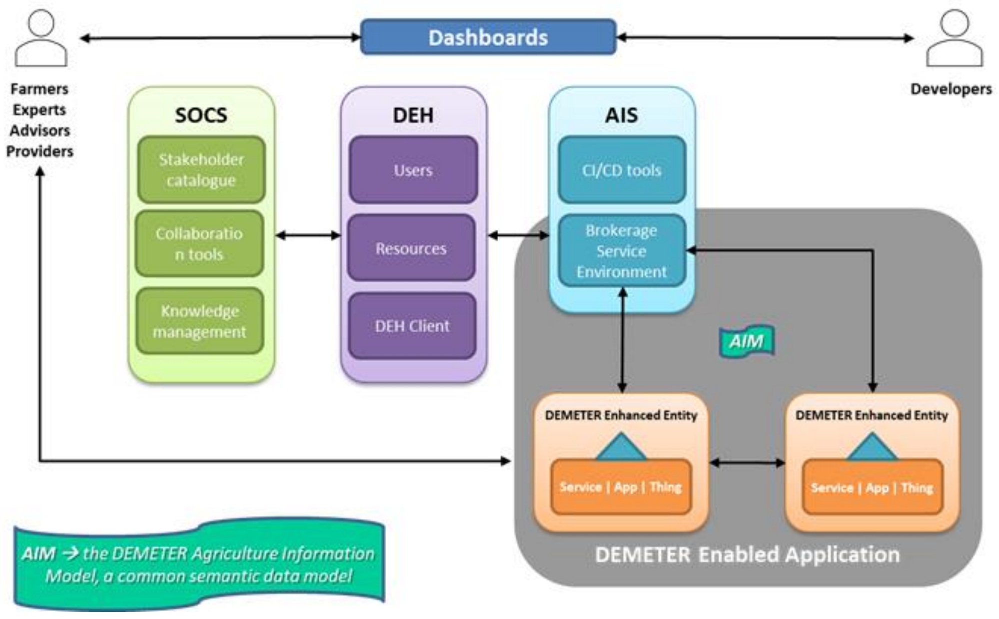
Table 1: Summary of Functional and Non-functional requirements for Reference Implementation .. 19
Table 2: BSE Service data model 25
Table 3: BSE meta data model 26
Table 4: BSE apiModel data model 26
Table 5: BSE - Register service endpoint 27
Table 6: BSE - Remove service endpoint 28
Table 7: BSE - Discover services endpoint 29
Table 8: BSE - Discover service by deh_id endpoint 30
Table 9: BSE - Discover service by service_name endpoint 31
Table 10: BSE - Discover service by tag endpoint 32
Table 11: User Data Model 41
Table 12: Application Data Model 41
Table 13: Organisation Data Model 42
Table 14: Role Data Model 42
Table 15: Authentication Token Data Model 42
Table 16: DEH Resource Data Model 88
Table 17: DEH Attachment and image Data model 88
Table 18: DEH Metrics Data Model 88
Table 19: DEH Metrics Data Data Model 89
Table 20: DEH Dashboard linked resources 113
Table 21: DEH Resource Registry Management linked resources 113
Table 22: Functional Interoperability enabler CompatibilityChecker data model 115
Table 23: Functional Interoperability compatibility endpoint 115
Table 24: Authorisation Enabler Data Model Information 126
Table 25: Encryption enabler, json data model 131
Table 26: Encryption enabler, XML data model 132
Table 27: DEH Client Module Configurable Parameters 147
Table 28: DEH Client Module Resource Consumption Metrics Data Model 150
Table 29: Component's general description 267

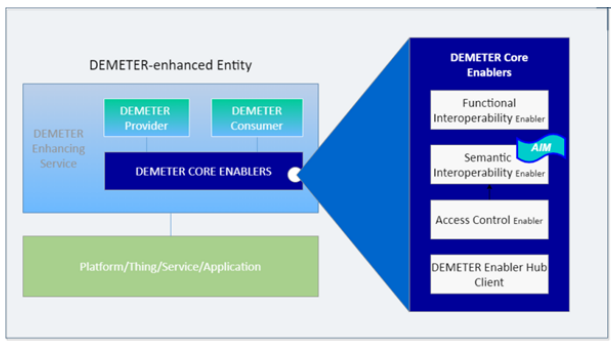
1 Executive Summary

D3.4 is a demonstrator deliverable of the DEMETER project. This document is the accompanying report of this deliverable. D3.4 provides the second release of components and tools that enable solution integration, interoperability with external platforms and deployment support for pilot cases (D3.2 provided the first release).

The diagram below illustrates the main DEMETER elements to be deployed:

- Stakeholders Open Collaboration Space (SOCS): a knowledge base and co-creation space where farmers, advisors and providers connect.
- DEMETER Enabler HUB (DEH): collects all the resources that are available to be used by a solution and enables access to them.
- Agricultural Interoperability Space (AIS): provides interoperability mechanisms to develop and deploy a solution.
- Dashboards: sole entry points to the DEMETER ecosystem.
- DEMETER-enhanced Entity (DEE): A Service, Application, Platform, or Thing wrapped with DEMETER enabler functionalities to act as a DEMETER consumer and/or producer. Many of these DEE's interoperate with each other to form an application solution.
- Agriculture Information Model (AIM): a common semantic data model to be used for the information exchange.




DEMETER-enhanced Entity
DEMETER Enhancing Service
DEMETER Provider
DEMETER Consumer
DEMETER CORE ENABLERS
Platform/Thing/Service/Application
DEMETER Core Enablers
Functional Interoperability Enabler
Semantic Interoperability Enabler
AIM
Access Control Enabler
DEMETER Enabler Hub Client

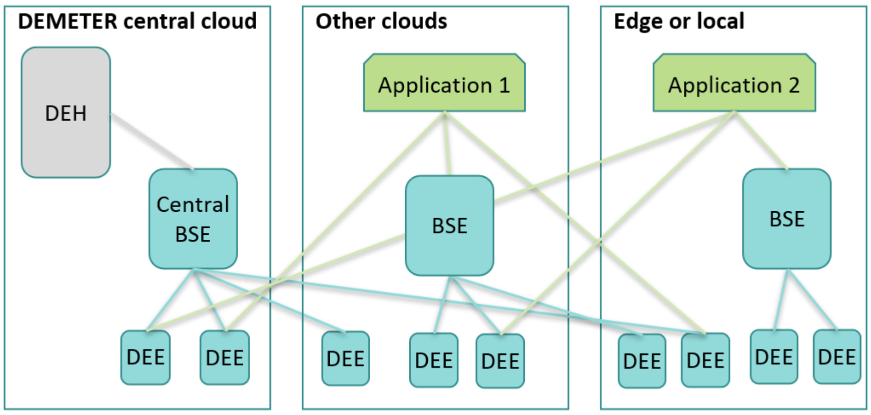
For each of the below components this document provides description, multiple architectural views, interface definition and notable details about implementation and the technologies used:

- Brokerage Service Environment (BSE): a microservices-based environment used to facilitate the registration, discovery, and communication of the DEE's.
- Access Control Server (ACS): offers authentication, authorisation, and traceability functionalities to the brokerage environment.
- DEMETER Enabler HUB (DEH): collects all the resources that are available to be used by a solution and enables access to them.
- Core Enablers for integration: specifications for core enablers that need to be implemented by DEE's in order to interoperate in a DEMETER application.

Furthermore, the document describes the deployed DEMETER infrastructure and the tools offered to integrate the platform and deploy the solution. Validation reports of the used components have been collected (as part of the Verification & Validation plan) and summarised.

The **key achievements of the D3.4** are the updated implementations and deployments of the BSE, ACS, DEH, Core Enablers for integration, along with the Verification and Validation activities over the available DEMETER enablers and modules.

DEMETER allows application to be deployed in many ways (from cloud to edge to local infrastructure), according to the business needs, as depicted in the diagram below:



2 Acronyms

ACE	Access Control Enabler
ACS	Access Control Server
AIS	Agricultural Interoperability Space
API	Application Programming Interface
BID	Business Intelligence Dashboard tool
BS	Brokerage Server
BSE	Brokerage Service Environment
CI/CD	Continuous Integration / Continuous Deployment
CoAP	Constraint Application Protocol
CRUD	Create Read Update Delete
DAE	DEMETER Advanced Enabler
DAO	Data Access Object
DEE	DEMETER-enhanced Entity
DEH	DEMETER Enabler HUB
DSS	Decision Support System
DTLS	Datagram Transport Layer Security
ETSI	European Telecommunications Standards Institute
FMIS	Farm Management Information System
GA	Grant Agreement
GDPR	General Data Policy Regulations
GE	Generic Enablers
GUI	Graphical User Interface
HTML	Hyper Text Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
IdM	Identity Management
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IOT	Internet of Things
IP	Internet Protocol
ISO	International Organization for Standardisation
IT	Information Technology
JSON	Java Script Object Notation
KPI	key Performance Indicator
LAN	Local Area Network
MQTT	Message Queuing Telemetry Transport
NGSI	Next Generation Sensors Initiative
NGSI-LD	Next Generation Sensors Initiative - Linked Data
OneM2M	One Machine to Machine
PAN	Personal Area Network
PDP	Policy Decision Point
PEP	Policy Enforcement Point
RDF	Resource Description Framework
REST	Representational State Transfer
RFID	Radio Frequency Identification Device
RPC	Remote Procedure Calls
RTPS	Real Time Publish Subscribe
SaaS	Software as a Service

SDK	Software Development Kit
SOCS	Stakeholders Open Collaboration Space
SQL	Structured Query Language
SSL	Secure Sockets Layer
SR	Service Registry
TBD	To Be Determined
TDD	Test Driven Development
TLS	Transport Layer Security
UAV	Unmanned Aerial Vehicle
UGV	Unmanned Ground Vehicle
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UUID	Universally Unique Identifier
WAN	Wide Area Network
WSN	Wireless Sensor Network
XACML	Extensible Access Control Markup Language
XML	Extensible Markup Language

3 List of Authors

Organization	Author
INTRA	Ioannis Oikonomidis, Athanasios Poulakidas, Dimitrios Skias
ENG	Antonio Caruso, Maria Francesca Cantore
VICOM	Álvaro Fernández Carrasco, Raul Orduna
ODINS	J. Andres, J. A. Martinez
TECNALIA	Belén Martínez, Sonia Bilbao
ICCS	Ioannis A. Vetsikas, Georgios Routis
WIT	Sundaresan Venkatesan
UMU	Antonio Skármeta, Manuel Mora
ATOS	Sergio Salmerón, Tomás Lobo

Organization	Reviewer
ATOS	Jesus Benedicto, Jesus Martinez
ICE	John Beattie, Oscar Garcia Perales

4 Document History

Version	Date	Change editors	Changes
0.1	19/05/2021	I. Oikonomidis, D. Skias, A. Poulakidas (INTRA)	First draft with TOC
0.2	21/05/2021	I. Oikonomidis, D. Skias, A. Poulakidas (INTRA)	Comments to first draft + final TOC
0.3	26/05/2021	I. Oikonomidis, D. Skias, A. Poulakidas (INTRA)	First input
0.4	02/06/2021	I. Oikonomidis, D. Skias, A. Poulakidas (INTRA)	Input for Appendices
0.5	07/06/2021	I. Oikonomidis, D. Skias, A. Poulakidas (INTRA), M. Belen (TECNALIA)	Requirements update
0.9	09/06/2021	I. Oikonomidis, D. Skias, A. Poulakidas (INTRA), R. Orduna, A. Carrasco (VICOM), A. Caruso, M. Cantore (ENG)	Merged nput for several sections
0.10	10/06/2021	I. Oikonomidis, D. Skias, A. Poulakidas (INTRA), J. Andres, A. Martinez (ODINS), S. Venkatesan (WIT)	Input for several sections
0.11	11/06/2021	I. Oikonomidis, D. Skias, A. Poulakidas (INTRA)	Doc ready for Internal review
0.12	23/06/2021	I. Oikonomidis, D. Skias, A. Poulakidas (INTRA), R. Urutia (VICOM), S. Venkatesan (WIT)	Addressed internal review comments
0.13	25/06/2021	I. Oikonomidis, D. Skias, A. Poulakidas (INTRA)	Prepared for final quality check

5 Introduction

D3.4 is a demonstrator deliverable of the DEMETER project and updates D3.2 which was submitted a year ago (M10). This document is part of that deliverable. D3.4 provides the second release of modules and tools that enable solution integration, interoperability with external platforms and deployment support for pilot cases. The following Tasks contributed to D3.4: T3.2, T3.3, T3.4, T3.5, T3.6.

Figure 1 illustrates the main DEMETER elements to be deployed:

- Stakeholders Open Collaboration Space (SOCS): a knowledge base and co-creation space where farmers, advisors and providers connect.
- DEMETER Enabler HUB (DEH): collects all the resources that are available to be used by a solution and enables access to them.
- Agricultural Interoperability Space (AIS): provides interoperability mechanisms to develop and deploy a solution.
- Dashboards: sole entry points to the DEMETER ecosystem.
- DEMETER-enhanced Entity (DEE): A service, application, platform, or thing wrapped with DEMETER enabler functionalities to act as a DEMETER consumer and/or producer. Many of these DEEs interoperate with each other to form an application solution.
- Agriculture Information Model (AIM): a common semantic data model to be used for the information exchange.

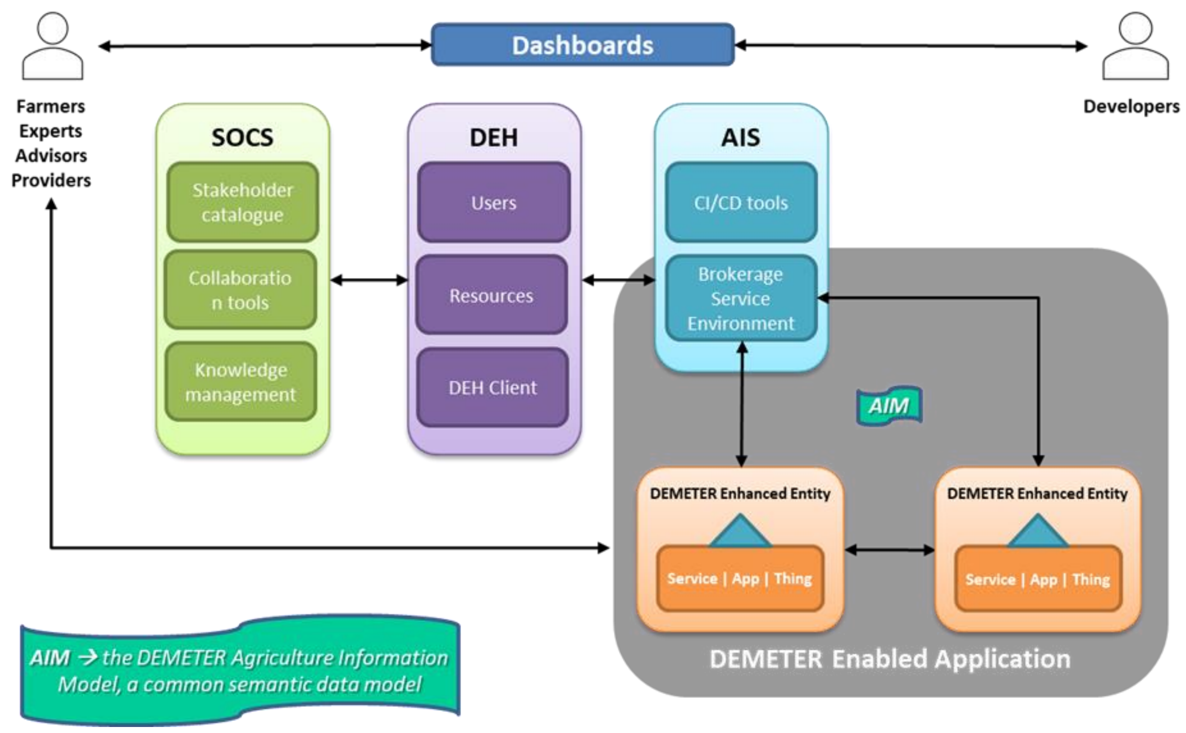


Figure 1: DEMETER main elements

The remainder of this document is comprised of the following sections:

- Section 6 provides the overall architecture of the DEMETER reference implementation. It also provides an overview of its updated requirements along with their current status and their mapping to the implementation components.
- Section 7 presents the Brokerage Service Environment, a microservices-based environment used to facilitate the registration, discovery, and communication of the DEE's.
- Section 8 presents the Access Control Server, which offers authentication, authorisation, traceability functionalities to the brokerage environment.
- Section 9 presents the DEMETER Enabler HUB (DEH) which offers all available Enablers in a catalogue for users.
- Section 10 specifies the core Enablers for integration. These enablers need to be implemented by DEE's in order to interoperate in a DEMETER application. This section reflects the changes presented in D3.3 (Reference Architecture Release 2) regarding the core enablers. More specifically, the "Security" and "Communication and Networking" core enablers functionalities were placed under one single core enabler, namely "Access Control Enabler for better management; no functionality was removed.
- Section 11 describes the infrastructure and tools that have been setup, deployed, configured, and how they are leveraged to assist in integrating and deploying a DEMETER module and enabler.
- Section 12 includes the verification and validation plan and summarizes the Verification and Validation reports collected so far for the various DEMETER modules and enablers.
- Section 13 provides the conclusions and next steps.
- Annex A lists the full details of the updated requirements whose overview was provided in Chapter 6.
- Annex B provides the template used to specify each DEMETER Enabler.
- Annex C provides the updated template used to represent the general information of the validation of any DEMETER module and enabler.

6 Architecture of the Reference Implementation

Following DEMETER's revised reference architecture presented in deliverable D3.3, this deliverable illustrates the Reference Implementation.

6.1 *Architecture (Physical view, Process view)*

This section depicts the Physical and Process View of the DEMETER Reference Implementation. More specifically, Figure 2 describes the 3 major modules of DEMETER platform, namely Stakeholder Open Collaboration Space, DEMETER Enabler HUB and Brokerage Service environment. Included in these three major modules lies, the Security module. In addition, it illustrates the interoperation activities between DEMETER Enhanced Entities (DEE) and DEMETER's Reference Implementation. DEE consists of a set of either an app, a service, or a device along with a set of core Enablers and Advanced Enablers.

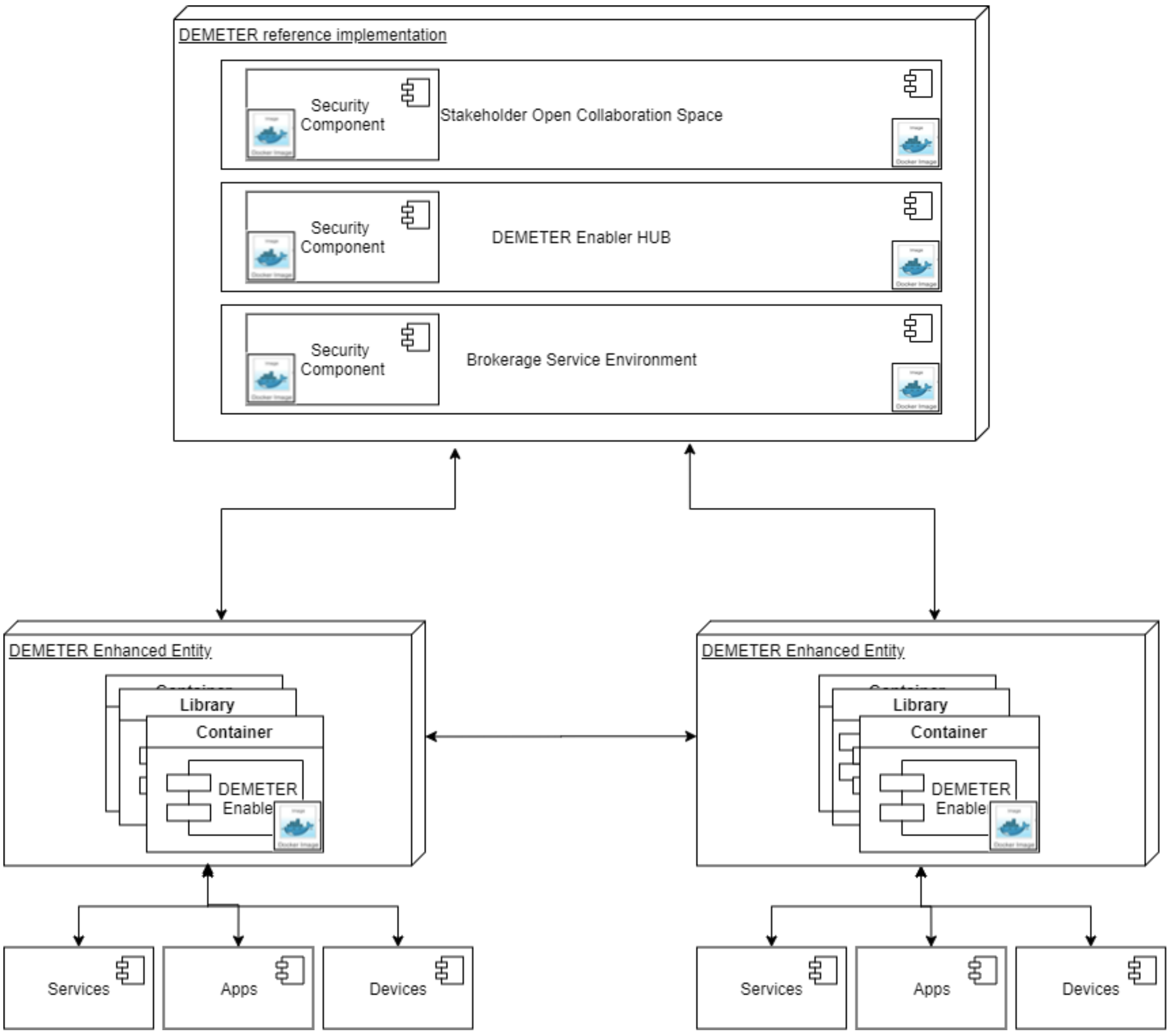


Figure 2: Reference Implementation Deployment diagram

Figure 3 depicts the Pilots' deployment schema. DEMETER's Pilots can either use their own infrastructure and deploy the DEMETER Brokerage Service Environment and the DEMETER Enhanced Entities (ellipse on the right) in their premises or they can rely on the BSE that is deployed in the DEMETER's Central Cloud and use this infrastructure in order to enable the communication of their

DEE (ellipse in the centre). In the same figure, the box on the left depicts the DEMETER Central Cloud where the BSE, the SOCS, the DEH and DEE resides.

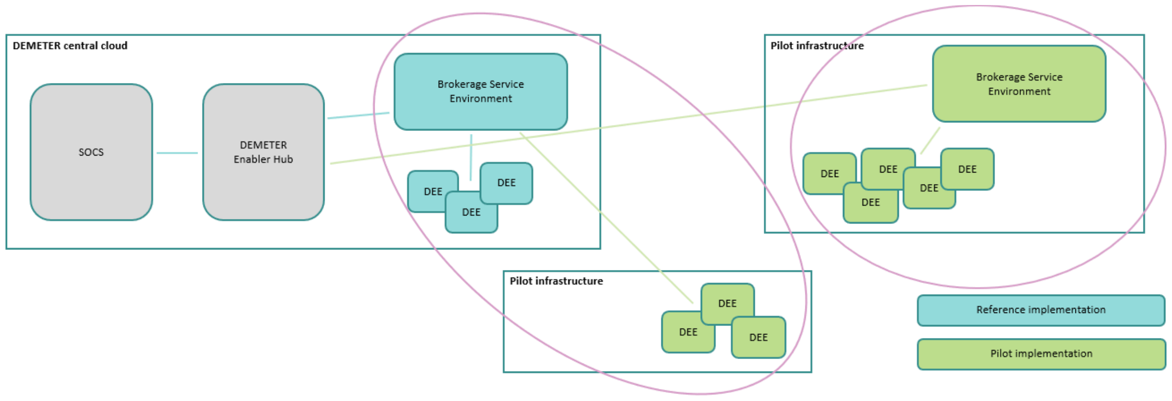


Figure 3: Pilots Deployment diagram

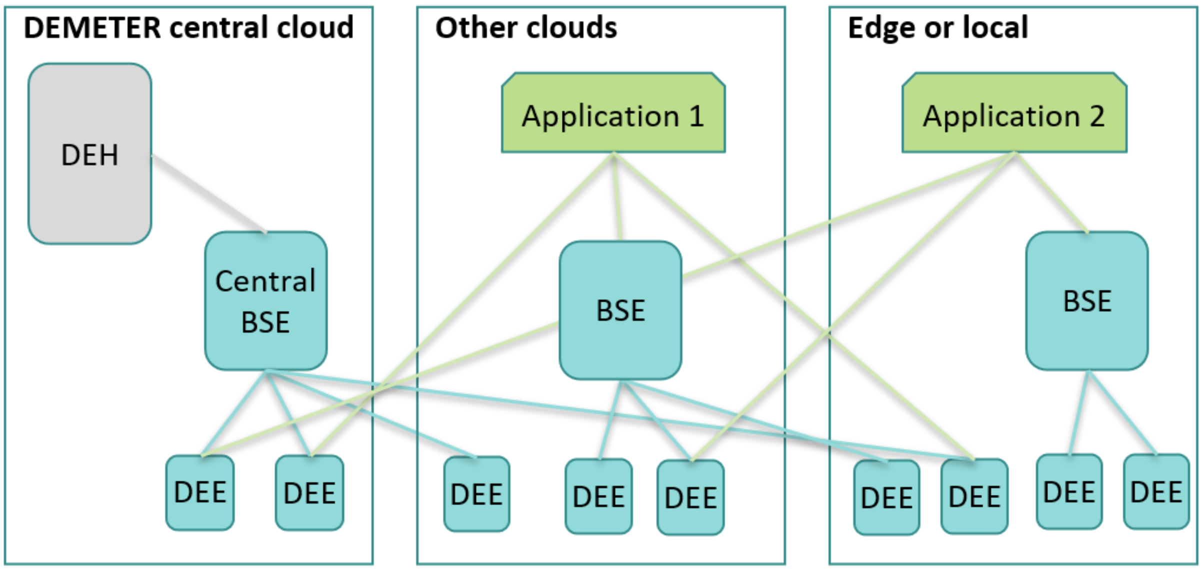


Figure 4: Deployment of applications in relation to DEMETER

Figure 4 illustrates how BSE's and DEE's can be deployed from cloud to edge or to local infrastructure. It is envisioned that BSE instances will coordinate the discovery and execution of running DEE's. Therefore, DEE's will be able to offer services whose access is public, or local to the BSE, or customisable (i.e., hybrid access policy), while applications will be able to use services from other BSE instances since running services can be discovered from remote BSE instances (with the assistance of the DEH and the central BSE, i.e., the one deployed on DEMETER cloud).

Figure 5 and Figure 6 present sequence diagrams that illustrates a set of processes that are being offered by the DEMETER reference implementation platform. The main actors of the diagrams are the DEMETER Provider who "provides" some resource to the DEMETER platform, the DEMETER reference implementation platform which comprises of the ACS component, the BSE, the DEH and the SOCS, and finally the DEMETER Consumer who "consumes" some resource that is being offered by the DEMETER platform.

Figure 5 illustrates the resource registration process that a provider initiates to register the resource in DEMETER's catalogue. DEMETER's security component (ACS) facilitates the Authentication and Authorisation process, subsequently the provider registers the resource to the DEH. Once the registration is confirmed and a unique identified is generated, the provider can register the running service with the BSE.

Figure 6 illustrates the process of resource discovery from the side of the consumer. Firstly, DEMETER consumer logs in to the DEMETER platform through the ACS component. Then, through the SOCS environment, the consumer can search a solution that matches their needs. Subsequently, the consumer browses on the DEH to find the right Enablers that would facilitate the access towards the resource that needs to consume. Once those Enablers are deployed, he discovers the relevant running service via the BSE. The BSE returns to the consumer the access information for that specific resource.

The final part of the process described depicts the consumer, possessing the access information that was given to him by the BSE, finding the requested resource and proceed in making requests and receiving the subsequent responses.

Figure 7 and Figure 8 complement the sequence diagrams described above and present the activity diagrams of the process described above.

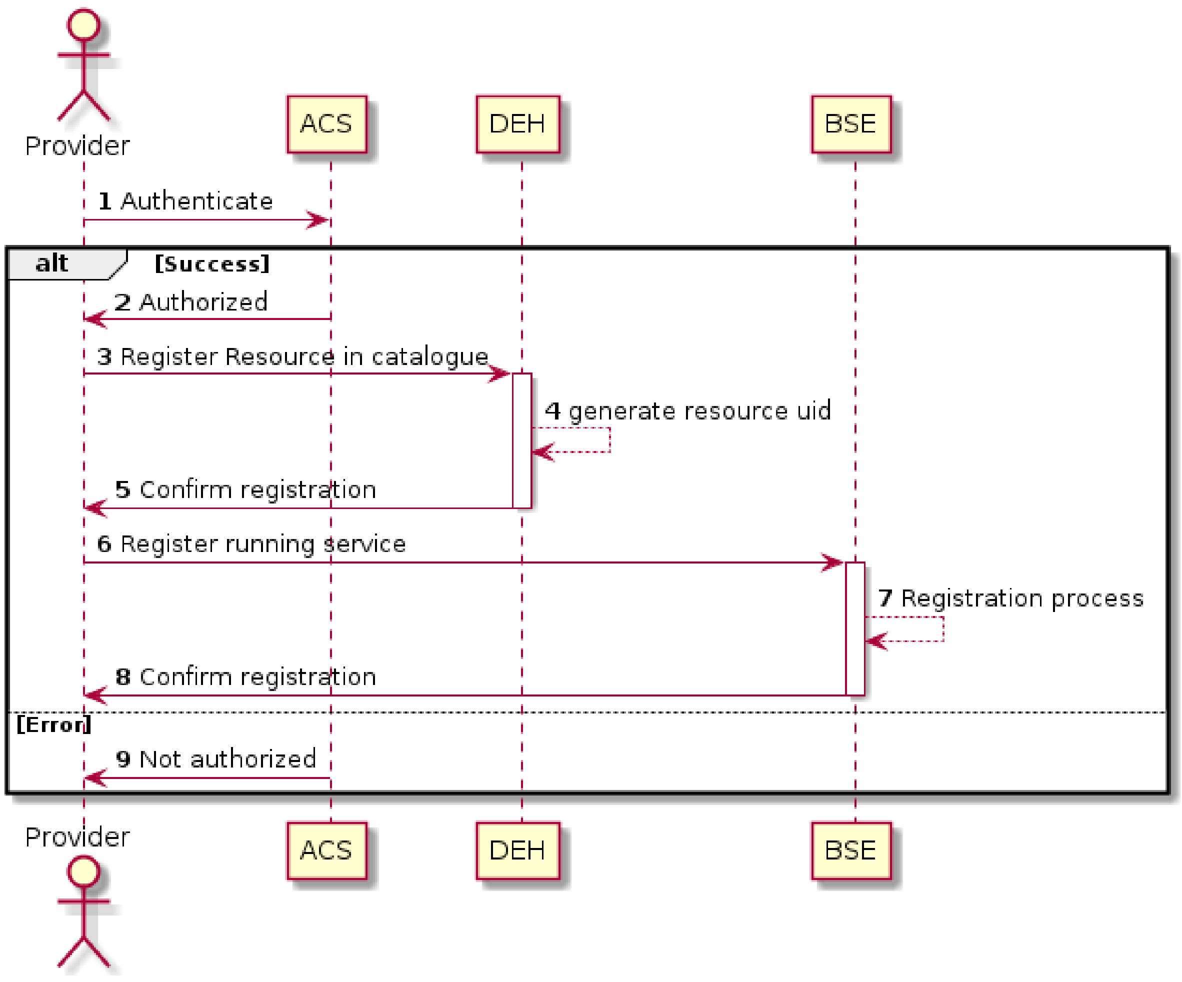


Figure 5: Sequence diagram - Provider

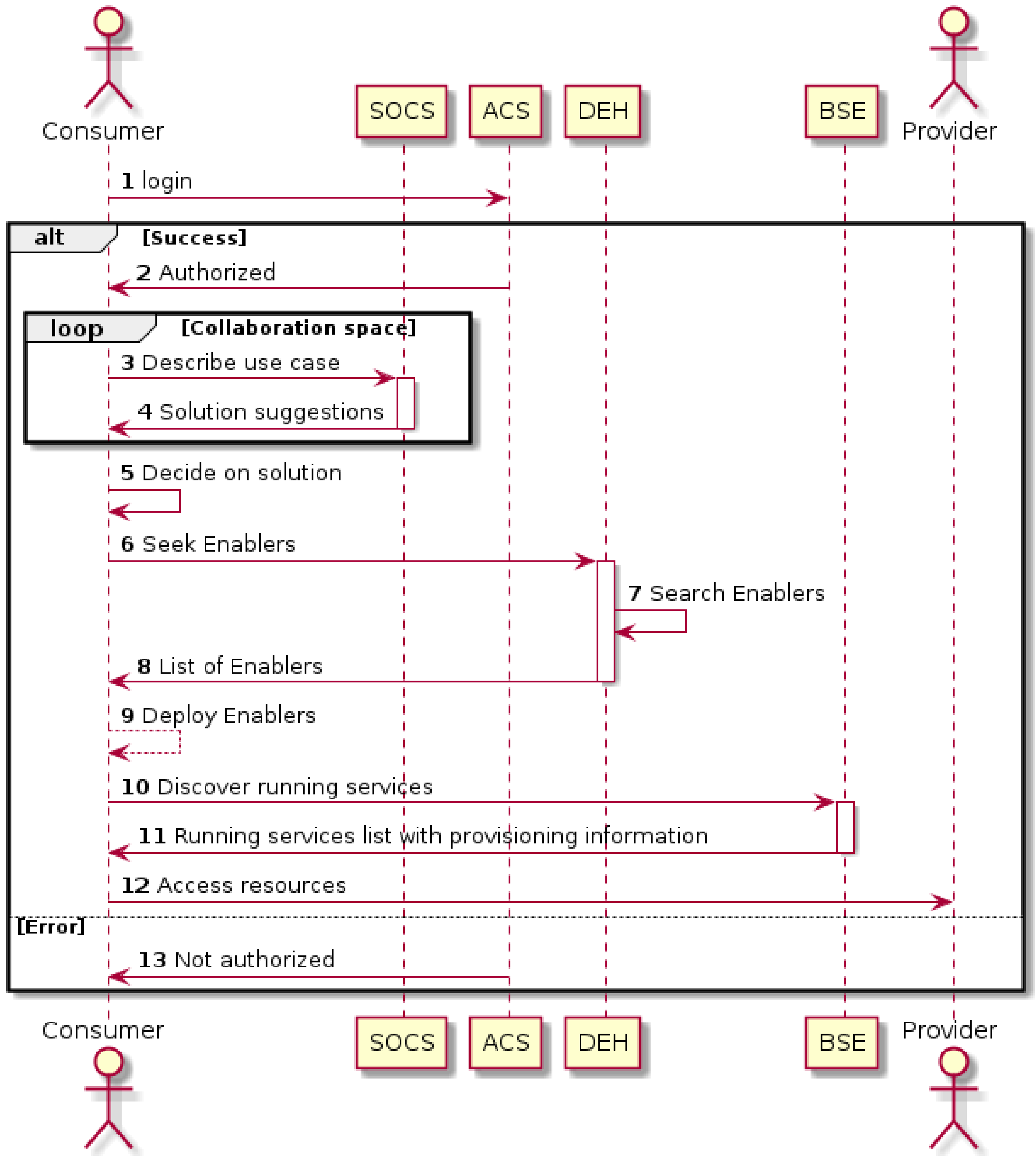


Figure 6: Sequence diagram - Consumer

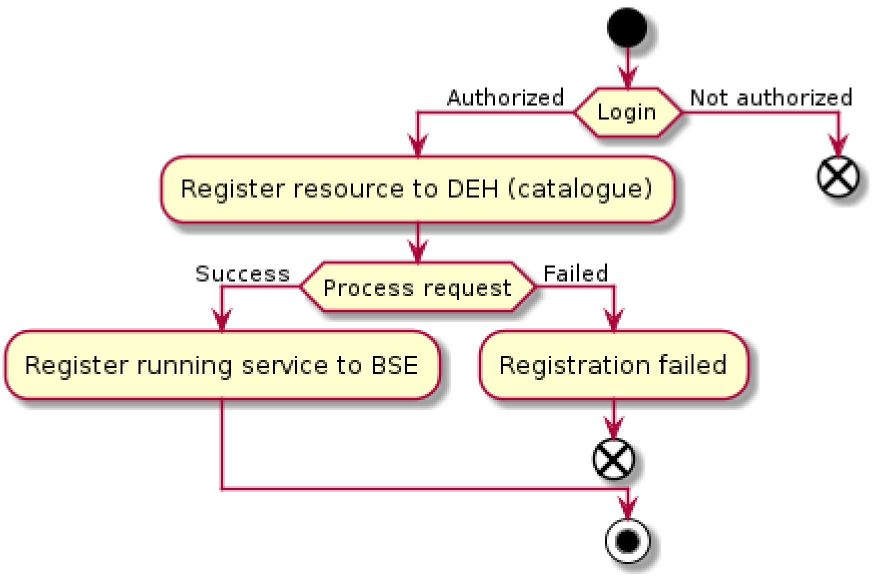


Figure 7: Activity diagram - Provider

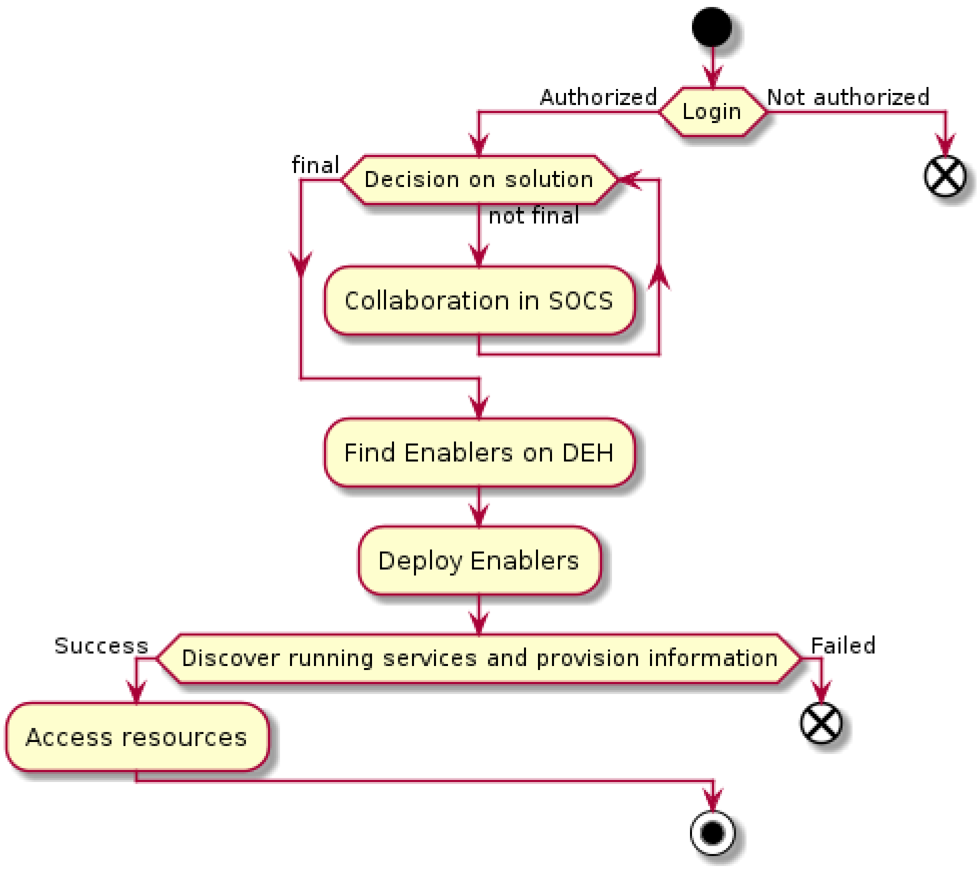


Figure 8: Activity diagram - Consumer

6.2 Requirements Mapping

Table 1 below summarizes the functional and non-functional requirements that refer to the Reference Implementation.

As already mentioned in D3.1, the requirement classes of WP3 are:

- TI1. Technical and Syntactic Interoperability of pilot technologies/platform,
- TI2. Environment for service discovery and provisioning,
- TI3. Networking and Communication,
- TI4. Security,
- TI5. Device/resource Management (including databases),
- TI6. Runtime Environment, Deployment Management & Orchestration,
- TI7. Service / application life-cycle management,
- TI8. APIs and Application development support,
- TI9. Enabler registration, discovery, provision, management, composition, accounting, billing,
- TI10. Stakeholder account management,
- TI11. Monitoring, Awareness, Feedback,
- GNFR. General (Non) Functional Requirements.

Table 1: Summary of Functional and Non-functional requirements for Reference Implementation

ID	Name	Related Module	Status
TI1.1	Utilization of existing standards	Enablers	Proposed More related to WP2/4
TI1.2	Support of Communication Protocol Standards	Enablers	Rejected (DEMETER targets higher layers, e.g., at IoT platform level)
TI1.3	Support of Geospatial Interoperability Standards	Enablers	Proposed More related to WP2/4
TI1.4	Provide interoperability with existing cloud platforms	Enablers	Fulfilled
TI1.5	HTTP REST API(s)	Enablers, BSE, ACS	Fulfilled
TI1.6	Pub/sub and messaging queue mechanisms	Enablers, StreamHandler	Fulfilled
TI1.7	Compliance with system domain standards	Enablers, BSE, DEH, ACS	Proposed
TI1.8	Data formats	Enablers, BSE, FIE, DEH, DEH Client, ACS	Fulfilled
TI2.1	Service description definition	Enablers, BSE, FIE, DEH	Fulfilled
TI2.2	Services provisioning maintaining data security and privacy	Enablers, ACS	Fulfilled
TI2.3	Services registration to DEMETER Enabler Hub	Enablers, BSE, FIE DEH	Fulfilled

ID	Name	Related Module	Status
TI2.4	Services' categorization	DEH	Fulfilled
TI3.1	Secure transport layer (TLS, SSH, etc.)	Enablers, BSE, DEH, ACS	Fulfilled
TI3.2	GDPR technical requirements	Enablers, BSE, DEH, ACS, ACE	Fulfilled
TI3.3	Combination of physical/wireless communications and Internet backbone networks	Enablers	Proposed
TI3.4	Control devices sharing information	Enablers	Proposed
TI4.1	Attribute Based Access Control or Distributed Capabilities Access Control component	Enablers, ACS	Fulfilled
TI4.2	Authentication and authorization mechanisms for services, accessing resources and information audit tools	Enablers, ACS	Fulfilled
TI4.3	Data protection and privacy on software execution, network communications and integrated solution security	Enablers, ACS, ACE	Fulfilled
TI4.4	Identity management, access control and audit log	Enablers, ACS	Fulfilled
TI4.5	Encrypted communications, integrity controls and electronic signature functionalities	ACE	Fulfilled
TI5.1	Data storage systems access management	Enablers	Proposed
TI5.2	Registration the capabilities of a resource	Enablers, DEH, BSE, FIE	Fulfilled
TI5.3	Multiple devices bulk operations	Enablers	Proposed
TI5.4	Resource/device sharing rules	Enablers, DEH, BSE	Fulfilled
TI6.1	DEMETER Enablers deployment	Enablers, BSE, FIE, DEH, DEH Client, ACS	Fulfilled
TI6.2	DEMETER Enablers compliance	Enablers, BSE, FIE	Fulfilled
TI6.3	DEMETER deployment tests	Enablers, BSE, FIE	Fulfilled
TI6.4	DEMETER runtime environment agnostic	Enablers, BSE, FIE	Fulfilled
TI6.5	Deployment process documentation	ALL WP3 modules/enablers	Fulfilled
TI6.6	Deployment software life-cycle management	Enablers, DEH	Proposed
TI6.7	Deployment process security	Enablers, BSE, FIE, DEH, DEH Client ACS, ACE	Fulfilled
TI7.1	Service/application life-cycle management methodology	ALL WP3 modules/enablers	Fulfilled
TI7.2	Technical requirements review	ALL WP3 modules/enablers	Fulfilled
TI7.3	Components' testing	ALL WP3 modules/enablers	Fulfilled
TI7.4	Development teams' communication	ALL WP3 modules/enablers	Fulfilled

ID	Name	Related Module	Status
TI7.5	Component maintenance	ALL WP3 modules/enablers	Fulfilled
TI7.6	Service/application life-cycle management software suites	Enablers, DEH, BSE, ACS	Fulfilled
TI8.1	CRUD to HTTP methods mapping	Enablers, DEH, BSE, FIE	Fulfilled
TI8.2	Proper HTTP response codes	ALL WP3 modules/enablers	Fulfilled
TI8.3	Searching, sorting, filtering, and pagination	Enablers, DEH, BSE, FIE	Fulfilled
TI8.4	Stateless Authentication & Authorization	Enablers, ACS	Fulfilled
TI8.5	Usage of Swagger for Documentation	Enablers, BSE, FIE, DEH	Fulfilled
TI8.6	REST-based services	ALL WP3 modules/enablers	Fulfilled
TI8.7	Access control mechanisms in API(s)	ACS	Fulfilled
TI8.8	API and application documentation	ALL WP3 modules/enablers	Fulfilled
TI9.1	DEH resource registry	DEH	Fulfilled
TI9.2	Discovery Management	DEH	Fulfilled
TI9.3	Query Management	DEH	Fulfilled
TI9.4	Rate services	DEH	Fulfilled
TI9.5	Resource Access Control	DEH	Fulfilled
TI9.6	Query Management	DEH	Fulfilled
TI9.7	Publish & Subscribe Notification	DEH	Not implemented Optional
TI9.8	Enablers Information Management	DEH, DEH Client	Fulfilled
TI9.9	DEH Scalability & Availability	DEH	Fulfilled
TI9.10	Licensing	DEH	Fulfilled
TI9.11	Data encryption in communications	DEH, DEH Client, ACE	Fulfilled
TI9.12	Service User Advisory	DEH	Fulfilled
TI9.13	Accounting Management	DEH	Not to be implemented Out of scope
TI9.14	Semantic Interoperability Framework	DEH	Fulfilled
TI9.15	Application portability	DEH	Fulfilled
TI9.16	System security services	DEH	Not implemented Not applicable
TI9.17	System availability	DEH	Fulfilled
TI9.18	External registration and provisioning	DEH	Fulfilled
TI9.19	Data synchronization	DEH	Fulfilled
TI9.20	Data federation	DEH	Fulfilled
TI9.21	Technology specification	DEH	Fulfilled
TI9.22	DEH modules characteristic definition	DEH	Fulfilled
TI9.23	Data management	DEH	Fulfilled
TI9.24	Data fusion	DEH	Fulfilled
TI9.25	Monitoring & Audit	DEH, ACS	Fulfilled
TI9.26	Information Management	DEH	Fulfilled

ID	Name	Related Module	Status
TI9.27	Metadata collection	DEH	Fulfilled
TI9.28	Data Resource Definition	DEH	Fulfilled
TI9.29	Resource Management (CRUD operations)	DEH	Fulfilled
TI9.30	Web service interoperability	DEH	Fulfilled
TI9.31	Resource compatibility checker	DEH	Fulfilled
TI9.32	Agriculture interoperability space resources	DEH	Fulfilled
TI9.33	Data Discovery Management	DEH	Fulfilled
TI9.34	Rating service	DEH	Fulfilled
TI9.35	Resource download report	DEH	Fulfilled
TI9.36	Collection of enablers system	DEH	Fulfilled
TI9.37	User profile management	ACS	Fulfilled
TI9.38	Responsive web GUI	DEH	Fulfilled
TI9.39	User account management	DEH, ACS	Fulfilled
TI9.40	User private home page	DEH	Fulfilled
TI9.41	User registration web page	DEH	Fulfilled
TI9.42	Resources Management web page	DEH	Fulfilled
TI9.43	Interoperability marketplace and catalogues solution	DEH	Fulfilled
TI9.44	DEH solutions web page	DEH	Fulfilled
TI9.45	Team services	DEH	Fulfilled
TI10.1	Stakeholder access	DEH, ACS	Fulfilled
TI10.2	Account management roles functionality	Enablers, DEH, BSE, FIE, ACS	Fulfilled
TI10.3	Distinguishing a) internal and external stakeholders and b) primary and secondary stakeholders	DEH, ACS	Not implemented Not applicable
TI10.4	Stakeholders' categorization	DEH, ACS	Fulfilled
TI11.1	Feedback from end-users	DEH	Fulfilled
TI11.2	Upvoting mechanism	DEH	Fulfilled
GNFR.1	Business analytic data visualization suite	Enablers	Proposed More related to WP4
GNFR.2	Decision Support System Dashboards	DEH	Proposed More related to WP4
GNFR.3	Web applications usability	DEH	Fulfilled
GNFR.4	Web application stylesheet	DEH	Fulfilled
GNFR.5	Web application friendliness	DEH	Fulfilled
GNFR.6	Business analytic data visualization suite	Enablers, DEH	Fulfilled
GNFR.7	DSS dashboard outcomes data visualization	Enablers	Proposed More related to WP4
GNFR.8	DSS dashboard notification	Enablers	Proposed More related to WP4
GNFR.9	DSS Dashboard widget	Enablers	Proposed More related to WP4

7 Brokerage Service Environment

7.1 Description

The Brokerage Service Environment (BSE) is a core module of DEMETER architecture, which facilitates the registration, discovery and ultimately communication process for the DEMETER-enabled resources in a secure and privacy preserving manner. In the framework of DEMETER, a resource coupled with the necessary enablers (core and advanced) is named a DEMETER enhanced entity (DEE). A DEE, once authenticated and authorised by the BSE can register as a service with the BSE specific registry. Subsequently, it becomes discoverable by all the other registered DEE’s. Finally, based on the suitable core and advanced enablers that each DEE implement and after resource provisioning information from the BSE, DEE’s can communicate directly with each other. In addition to the functionalities, BSE can interconnect (interface) with DEMETER HUB in case a useful flow is identified or required.

The BSE is implemented as a self-contained application that enables an external party to deploy it as a complete brokerage service solution. Each DEMETER-enabled application should utilise at least one BSE. The BSE accompanied by a publish-subscribe communication mechanism that addresses the required communication data throughput realises the backbone of the DEMETER reference architecture.

The following sections describe BSE's core subcomponents and their interactions, along with the sequence diagrams that illustrate the data flow between them.

7.2 Development View

The development view illustrates a system from a programmer's perspective and is also known as the implementation view. It uses the UML Component diagram to describe BSE components.

7.2.1 Component diagram

Figure 9 below illustrates the major components of the BSE. Its core components are the Access Control Server (ACS), the Brokerage Server (BS) and the Service Registry (SR). While the ACS provides for the authentication and the authorisation of the DEE’s that request to be included in the BSE, the BS realises the DEE registration, discovery, and the provisioning functionality.

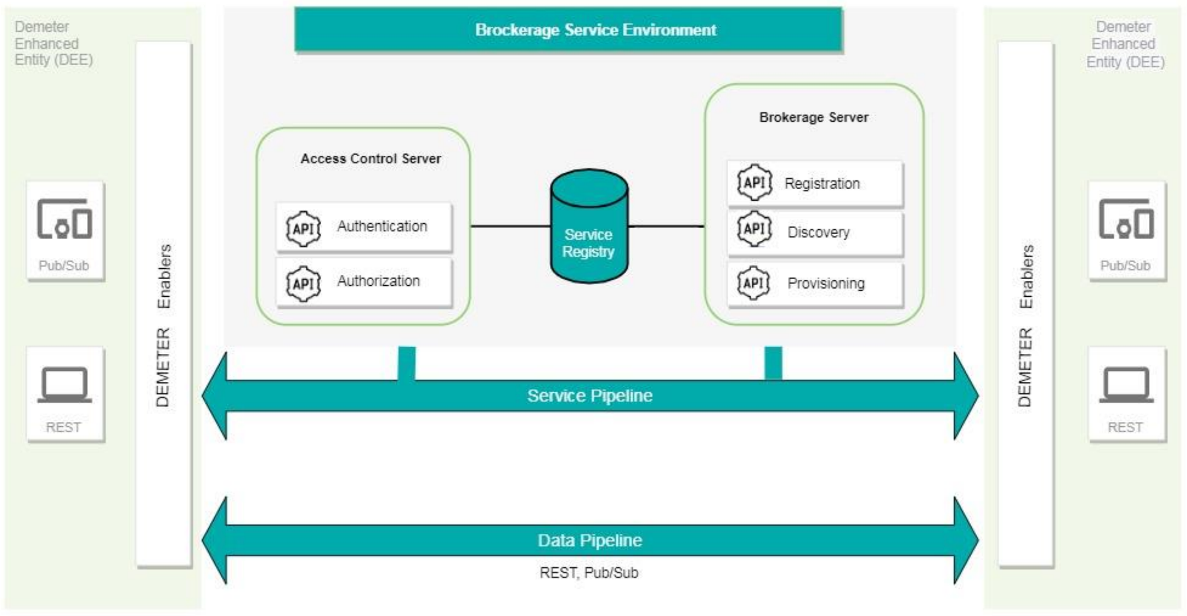


Figure 9: BSE component diagram

7.2.2 Building blocks

7.2.2.1 Access Control Server

Access Control Server (ACS) and its sub-components are described in detail in section 8. BSE is utilizing the functionality provided by this component. Each BSE instance, independently of where it is deployed, it should be paired/registered to an ACS instance.

7.2.2.2 Service Registry

In the context of Brokerage Service Environment (BSE), the Service Registry implements a RESTful interface through which it communicates on one hand with Access Control Server (ACS) and on the other with Brokerage Server (BS). Service Registry is used to store user and service-related metadata in a persistent manner. More specifically, it holds, user authentication credential information (where necessary) and access tokens that are generated by the ACS and are used from third party services to access and interoperate with BSE endpoints. Furthermore, it stores service-related meta-data that is required or generated by the Brokerage Server (BS).

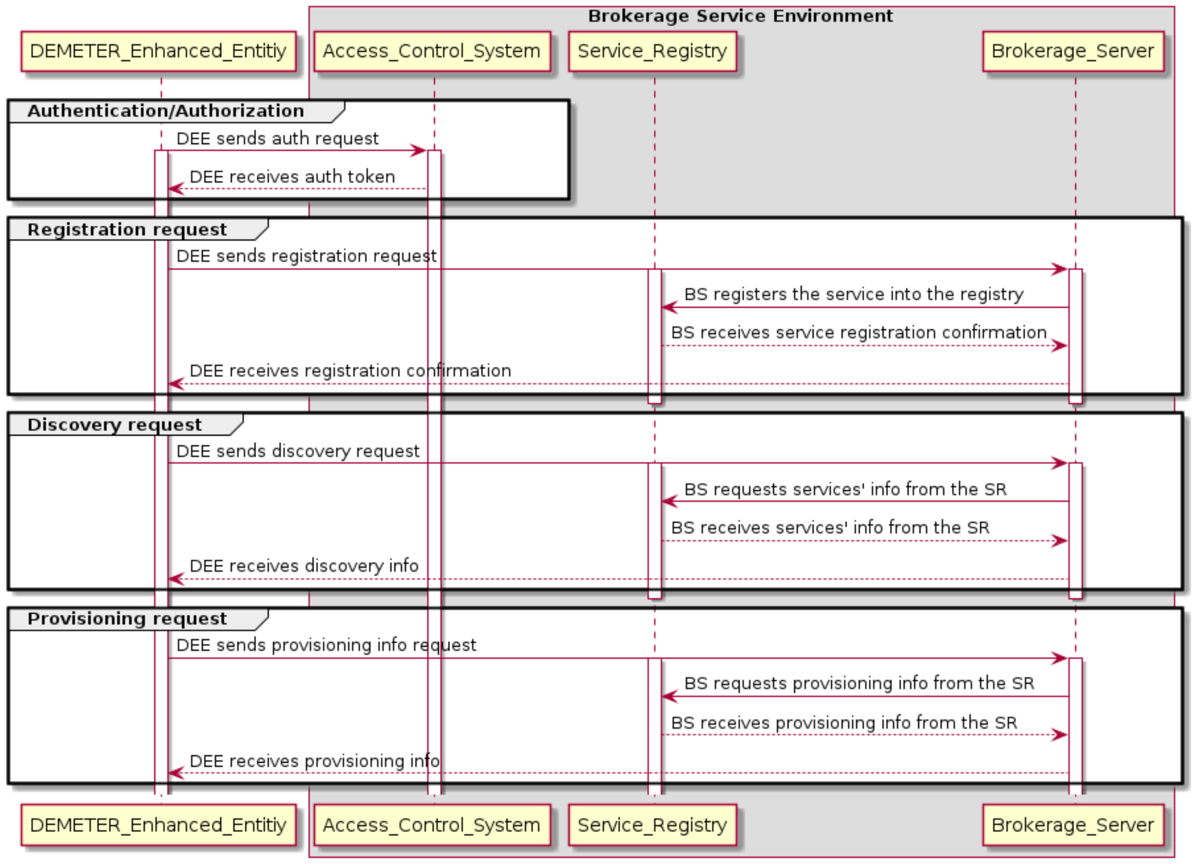
7.2.2.3 Brokerage Server

In the context of Brokerage Service Environment (BSE), the Brokerage Server's (BS) purpose is to facilitate the registration, discovery, and provisioning service. It is built on top of Consul[1] which is a service mesh solution providing a full featured control plane with service discovery, configuration, and segmentation functionality. Through Brokerage Server (BS) and the RESTful interface that it implements, a third-party service can get registered, discovered, and queried through the BSE. The BS, interfaces with Service Registry where it stores services' meta-data in a persistent manner. In addition, where it is necessary from a security or administrative point of view, the BS incorporates Access Control Lists through tokens that can be used to confine each service discovery environment.

[1] https://www.consul.io/

7.3 Process View

Figure 10 illustrates a Brokerage Service Environment (BSE) sequence diagram that depicts an overview of the core functionalities provided by the BSE. Each functionality is presented in its own frame where the data flow is described.


Brokerage Service Environment
DEMETER_Enhanced_Entitiy
Access_Control_System
Service_Registry
Brokerage_Server
Authentication/Authorization
DEE sends auth request
DEE receives auth token
Registration request
DEE sends registration request
BS registers the service into the registry
BS receives service registration confirmation
DEE receives registration confirmation
Discovery request
DEE sends discovery request
BS requests services' info from the SR
BS receives services' info from the SR
DEE receives discovery info
Provisioning request
DEE sends provisioning info request
BS requests provisioning info from the SR
BS receives provisioning info from the SR
DEE receives provisioning info
DEMETER_Enhanced_Entitiy
Access_Control_System
Service_Registry
Brokerage_Server

BSE: Brokerage Service Environment
DEE: DEMETER Enhanced Entity
BS: Brokerage Server
ACS: Access Control Server
SR: Service Registry

Figure 10: BSE sequence diagram

7.4 Interfaces

7.4.1 Data models used in interfaces

Table 2: BSE Service data model

Name	Service Data Model	
Property	Type	Description
service_name	string	*service name*
tags	String array	*Tag(s) that describe the service*
meta	Object	*meta-data that describe the service*
port	integer	*port of the service*
address	string	*IP address of the service*

Table 3: BSE meta data model

Name	meta Data Model	
Property	Type	Description
applicationCategory	string	*Type of service*
description	string	*Description of the service*
version	number	*service's version*
deh_id	string	*Service DEMETER HUB unique identifier*
featureList	String array	*service's feature list*
dataEncryption	boolean	*Whether data are encrypted or not*
authentication	boolean	*Whether the user needs to authenticate*
conditionsOfAccess	string	*Process to authenticate user*
timeRequired	integer	*max seconds between when a user makes a request and system response*
quota	string	*the maximum number of requests that can be done to the service*
offers	number	*cost of the service*
TermsOfService	string	*License model of the service*
usageInfo	string	*Link to extra information about the service*
provider	string	*The name of the service provider*
spatial	string	*country where the service is hosted*
aggregateRating	integer	*The score representing the service 1-5*
apiModel	Object	*Service's API description data model*

Table 4: BSE apiModel data model

Name	apiModel Data Model	
Property	Type	Description
dataProtocol	string	*endpoint's data protocol (REST or PUB/SUB)*
baseUrl	string	*endpoint's base URL*
relativePath	string	*endpoint's relative path*
method	string	*endpoint's method described (GET/POST/PUT/DELETE)*
URLRequiredParams	JSON object	*Required parameters – JSON*
URLOptionalParams	JSON object	*Optional parameters – JSON*
dataParams	JSON object	*Endpoint’s body payload – JSON*
successResponse	Integer array	*endpoint's success response list*
errorResponse	Integer array	*endpoint's error response list*
sampleCall	string	*endpoint's CURL sample call*
topic	string	*label text about topic*
payloadFormat	string	*endpoint's payload format JSON, JSON-LD, NGSI-LD*
payloadRepresentation	JSON object	*JSON formatted payload representation*

7.4.2 Description of API's

Table 5: BSE - Register service endpoint

Title	Register service to BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
https://bse.h2020-demeter-cloud.eu/api/BSE/register	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
service_name	*service name*
tags	*Tag(s) that describe the service*
meta	*meta-data that describe the service*
port	*port of the service*
address	*IP address of the service*
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
201 Content: {service}	Request was successful
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorised
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

Table 6: BSE - Remove service endpoint

Title	Remove registered service from BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
https://bse.h2020-demeter-cloud.eu/api/BSE/deregister/{service_id}	
Method This field holds the type of the Method used	
DELETE	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
service_id	The unique identifier of the service
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
204 Content: { }	Resource was successfully deleted
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorised
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

Table 7: BSE - Discover services endpoint

Title	Discover registered services in BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
https://bse.h2020-demeter-cloud.eu/api/BSE/services	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: [{service}]	An array of service objects discovered
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorised
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

Table 8: BSE - Discover service by deh_id endpoint

Title	Discover registered service by deh_id in BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
https://bse.h2020-demeter-cloud.eu/api/BSE/service/deh/{deh_id}	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
deh_id	DEMETER Enabler HUB ID
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: {service}	An array of service objects discovered
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorised
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	
Deh_id is unique	

Table 9: BSE - Discover service by service_name endpoint

Title	Discover registered service by service name in BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
https://bse.h2020-demeter-cloud.eu/api/BSE/service/{service_name}	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
service_name	Service name
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: [{service}]	An array of service objects discovered
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorised
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	
Service name might not be unique	

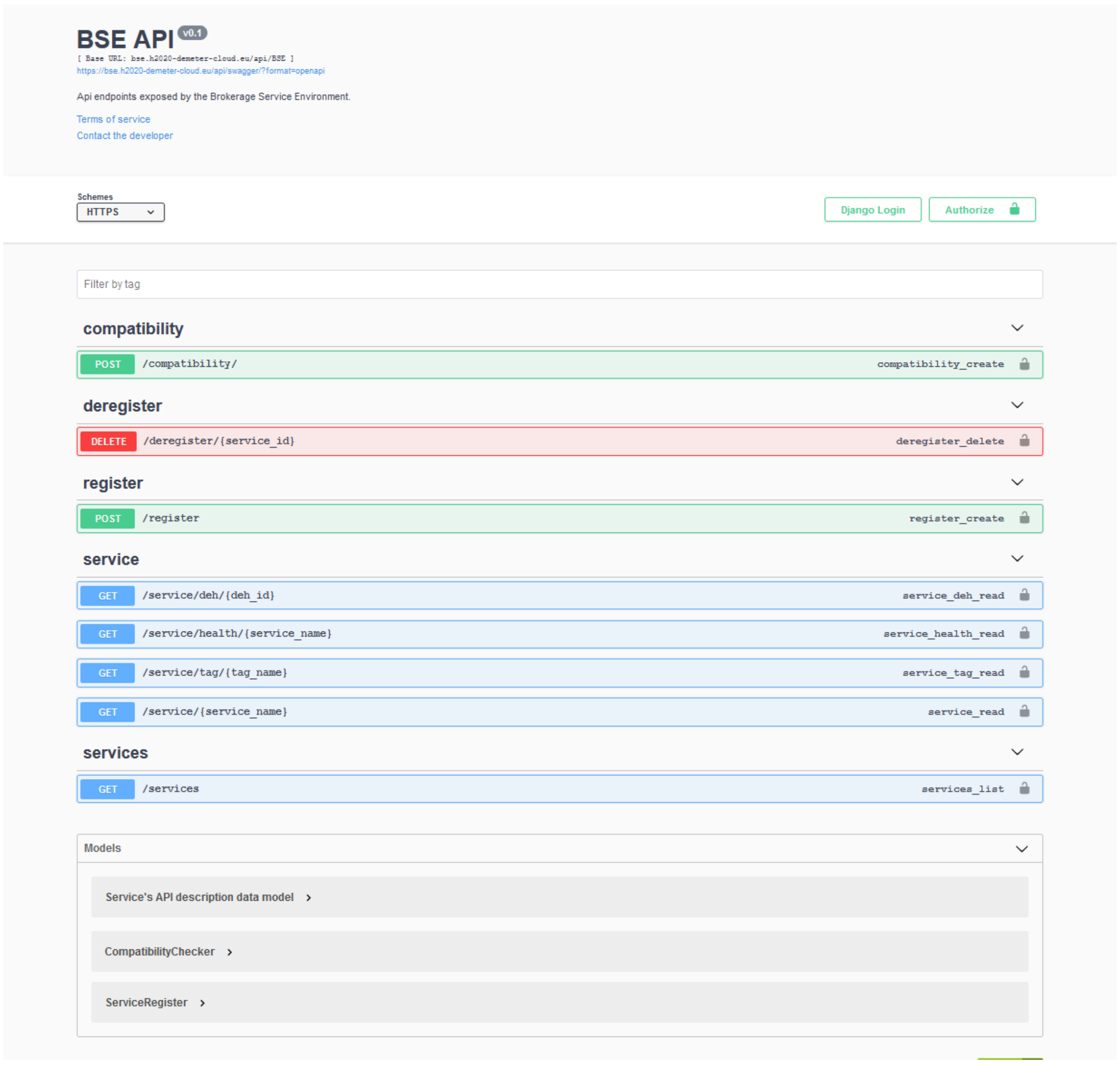
Table 10: BSE - Discover service by tag endpoint

Title	Discover registered service by tag in BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
https://bse.h2020-demeter-cloud.eu/api/BSE/service/tag/{tag_name}	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Tag_name	Tag of the service
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: [{service}]	An array of service objects discovered
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorised
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	
Same tag might be included in several services	

7.5 Technologies and implementation details

The Brokerage Service Environment (BSE) is implemented using the Django Framework[2] (Python-based framework) and is realised as a containerised application with a self-contained execution environment. It consists of a set of Docker containers that hold the Brokerage Server (BS), the Service Registry, and the Access Control Server (ACS). As described in section 7, BSE is bundled with the Functional Interoperability core enabler; hence, their deployment is described in that section. In addition, BSE also implements a REST API which is based on the Django Rest Framework. The Brokerage server is developed based on and integrated with the Consul service discovery and configuration system. Swagger online documentation is provided for the REST API of the BSE. For DEMETER cloud BSE instance, it can be found on DEMETER cloud[3].

[2] https://www.django-rest-framework.org/
[3] https://bse.h2020-demeter-cloud.eu/api/swagger/


BSE API v0.1

[Base URL: bse.h2020-demeter-cloud.eu/api/BSE]

https://bse.h2020-demeter-cloud.eu/api/swagger/?format=openapi

Api endpoints exposed by the Brokerage Service Environment.

Terms of service

Contact the developer

Schemes: HTTPS

Django Login | Authorize

Filter by tag

compatibility

```
POST    /compatibility/                     compatibility_create
```

deregister

```
DELETE  /deregister/{service_id}            deregister_delete
```

register

```
POST    /register                           register_create
```

service

```
GET     /service/deh/{deh_id}               service_deh_read
GET     /service/health/{service_name}      service_health_read
GET     /service/tag/{tag_name}             service_tag_read
GET     /service/{service_name}             service_read
```

services

```
GET     /services                           services_list
```

Models

- Service's API description data model
- CompatibilityChecker
- ServiceRegister

Figure 11: BSE online Swagger documentation

Future work for BSE includes finalising the functionality that will allow inter-BSE communication, which will allow for a federation of BSE instances, thus, extending DEE discovery capabilities to more domains.

In addition to the REST API, a solution, namely StreamHandler, for data streams based on the pub/sub paradigm has been adjusted, configured, and deployed in the context of WP2 and is offered (to pilots and other external parties) in DEMETER as part of the Data Management System. StreamHandler is a high-performance (low latency and high throughput) distributed streaming platform for handling real-time data based on Apache Kafka. It can efficiently ingest and handle massive amounts of data into processing pipelines, for both real-time and batch processing. The platform and its underlying technologies can support any type of data-intensive ICT services (Artificial Intelligence, Business Intelligence, etc.) from cloud to edge.

The key capabilities and features offered by the platform are:

- Real-time monitoring and event-processing
- Interoperability with all modern data storage technologies and popular data sources

- Distributed messaging system
- High fault-tolerance - Resiliency to node failures and support of automatic recovery
- Elasticity - High scalability
- Security (encryption, authentication, authorisation)

In particular, the platform is a fully featured industrial grade solution which i) is able to scale and accommodate Big Data sources from different domains, interoperating with all modern data storage technologies as well as other persistence approaches and ii) can support all important Big Data languages including Python, Java, R and Scala as well as other traditional programming approaches. The StreamHandler platform consists of i) Connectors, ii) Streaming Core component, iii) Schema Registry, iv) Security Management and v) Platform Admin and Monitoring Dashboard. Data sources, Data stores, Data Analytics and Visualisation applications and well as the supporting Processing Infrastructure are components that complement the offered Big Data solution and their choice and implementation are dependent on the targeted use cases and scenarios.

8 Access Control Server

8.1 Description

The security components provide the following three functionalities to other DEMETER components and pilots implementations:

- Authentication
- Authorisation
- Traceability

These functionalities have been implemented in six main security components: Identity Manager, XACML PDP, Capability Manager, PEP Proxy, Traceability Agent and Traceability blockchain repository. These components expose methods using a REST API as described in the following sub sections.

8.2 *Development View*

8.2.1 Component diagram

The following diagram depicts the security components and their relationships in order to provide the authentication, authorisation and traceability functionalities:

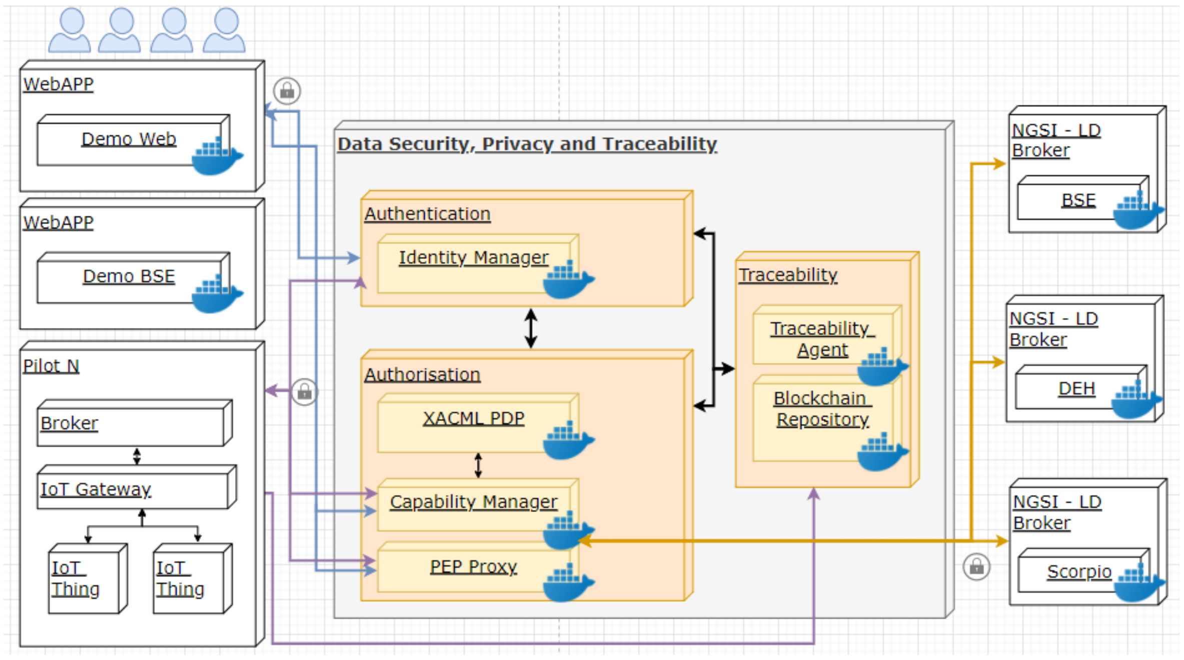


Figure 12: Security component diagram

The authentication functionalities will be provided by the FIWARE Identity Manager component. The authorisation functionalities will be provided by the DCapBAC module, which contains three subcomponents: XACML PDP, Capability Manager and PEP Proxy. The traceability functionalities will be provided by the Traceability Agent, which will log the use of the authentication/authorisation token within the traceability blockchain repository.

These building blocks are described in the following sub section.

8.2.2 Building blocks (components)

Description of the Data Security Components:

- Identity Manager
- XACML PDP
- Capability Manager
- PEP Proxy
- Traceability Agent
- Traceability Blockchain Repository

8.2.2.1 Identity Manager

The Demeter Identity Manager (IdM) component is based on the FIWARE Keyrock GE[4] and will provide the Keyrock's REST API for authentication based on the OAuth 2.0 protocol. The OAuth 2.0

[4] https://fiware-idm.readthedocs.io/en/7.4.0/

protocol supports several grants ("methods") types for a client application to acquire an access token (which represents a user´s permission for the client to access their data) which can be used to authenticate a request to the Keyrock API endpoint. The following methods for authentication are provided:

- **Authorisation Code**: defined for apps running on a web server, where the user will be redirected to the Keyrock server.
- **Username and Password**: for logging in with a username and password directly in the web server.
- **Client credential**: suitable for machine-to-machine authentication where specific user´s permission to access data is not required

Refresh token: to refresh the authentication token before its expiration time.

8.2.2.2 XACML PDP

The XACML PDP manages the access control policies and decides who can access a resource and what actions they can perform with that resource.

The PDP (*Policy Decision Point*) evaluates XACML (*eXtensible Access Control Markup Language*) policies in XML representation. With the specified policies, a request from the Capability Manager made to the PDP, that has the location of the policies, is evaluated to decide whether the access or action in the request can be performed or not, sending back a response. This communication is sent encoded in JSON, which provides a less verbose representation of the information and improves the request processing as well. The next code listing shows an example of an XACML policy in XML format:

```
<Policy PolicyId="example">
    <Rule Effect="Permit" RuleId="001">
            <Target>
          <Subject>
              <SubjectMatch 
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                  <AttributeValue 
DataType="http://www.w3.org/2001/XMLSchema#string">Peter</AttributeValue>
              </SubjectMatch>
          </Subject>

          <Resource>
              <ResourceMatch 
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                  <AttributeValue 
DataType="http://www.w3.org/2001/XMLSchema#string">https://215.64.19.203:1020/ngsi
-ld/v1/entities?type=http://www.w3.org/ns/sosa/Sensor;idPattern=urn:ngsi-
ld:Sensor:temperature.*
            </AttributeValue>
              </ResourceMatch>
          </Resource>

          <Action>
              <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-
equal">
                  <AttributeValue 
DataType="http://www.w3.org/2001/XMLSchema#string">GET</AttributeValue>
              </ActionMatch>
          </Action>
```

```
            </Target>
    </Rule>
</Policy>
```

The PDP is deployed as a Web Service to be accessed by the authorisation entity acting as Policy Enforcement Point (PEP) through the exchange of HTTP messages with JSON payloads containing the XACML requests or responses. The PDP is based on Web technologies to be a scalable and lightweight solution so it can be applied to any large-scale deployment that requires XACML as policy language. XACML PDP achieves clear performance improvements over other existing solutions in terms of scalability and efficiency.

The next figure shows a flow chart with an XACML PDP in an authorisation process example:

1. The Capability Manager asks the XACML PDP sending an authorisation (AuthZ) request to determine whether the requested credential must be generated or not (fig. step 1).
2. The XACML PDP evaluates the AuthZ request using the defined XACML policies and sends back its verdict to the Capability Manager (fig. steps 2, 3).

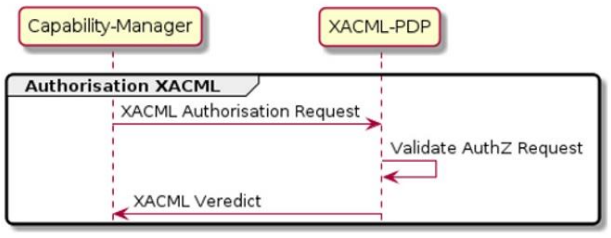


Figure 13: XACML PDP authorisation sequence diagram

8.2.2.3 Capability Manager

The Capability Manager is the component for generating capability tokens for the user in the event of an affirmative authorisation decision from the XACML PDP following a request about an action or access to a resource. The Capability Manager signs the generated capability token that includes the client's public key and time restrictions associated with the specific policy delimiting the validity period for this credential.

The figure below shows a sequence chart with a Capability Manager in an authorization process example:

1. When an authenticated user wants to get access to a resource or to perform an action an authorisation request is sent to the Capability Manager (fig. step 1).
2. When this request, that includes the user’s authentication (AuthN) token, is received by the Capability Manager it validates the token on the IdM getting back the user’s identity attributes (fig. steps 2, 3) and then validates them (fig. step 4).
3. Once validated and with these attributes, the Capability Manager asks the XACML PDP sending an authorisation (AuthZ) request to determine whether the requested credential must be generated or not (fig. step 5).
4. The XACML PDP evaluates the AuthZ request using the defined policies and sends back its verdict to the Capability Manager (fig. steps 6, 7).
5. The Capability Manager generates then the Cap.Token and send it back to the user in a response (fig. steps 8, 9).

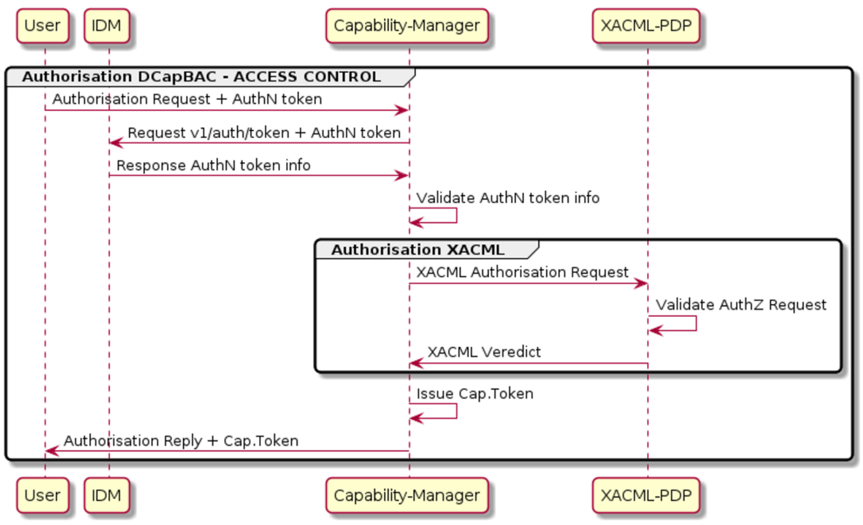

User
IDM
Capability-Manager
XACML-PDP
Authorisation DCapBAC - ACCESS CONTROL
Authorisation Request + AuthN token
Request v1/auth/token + AuthN token
Response AuthN token info
Validate AuthN token info
Authorisation XACML
XACML Authorisation Request
Validate AuthZ Request
XACML Veredict
Issue Cap.Token
Authorisation Reply + Cap.Token
User
IDM
Capability-Manager
XACML-PDP

Figure 14: Capability Manager with capability token sequence diagram

The format of the capability token is based on JSON as it can provide a simple, lightweight, efficient, and expressive data representation, which is suitable to be used on constrained networks and devices in IoT scenarios. The next text shows an example of capability token in JSON format:

```
{
  "id": "nlqfnfa6nqrlbh9h7tigg28ga1",
  "ii": 1586166961,
  "is": "capabilitymanager@odins.es",
  "su": "Lucas",
  "de": "https://153.55.55.120:2354",

"si":"MEUCIEEGwTGdlEeUxZv7jsh0UdWoLud3uqpMDvlT+GD7AiEAmwEuFHuG+XyRi9BEAMVPBIqRvOJl
SIBkBT3K7LHCw=",
   "ar": [
    {
      "ac": "GET",
      "re":"/ngsi-ld/v1/entities/urn:ngsi-ld:Sensor:temperature.21"
    }
],
  "nb": 1586167961,
  "na": 1586178916
}
```

- The identifier (ID): It is used to un-equivocally identify a capability token.
- The issuer (IS): Entity issuing and signing the capability token.
- The signature (SI): It carries the digital signature of the token.
- Access Rights (AR): The set of rights granted to the subject.
    - Action (AC): Its purpose is to identify a specific granted action (“get”).
    - Resource (RE): The resource (“temperature”) for which the action is granted.

8.2.2.4 PEP Proxy

The PEP (*Policy Enforcement Point*) is responsible for validating a generated assertion in an authentication token (X-AUTH-TOKEN) with the capability token that was already generated in a response by the Capability Manager to a user's authorization request. The PEP Proxy verifies that the public key contained in the received capability token is the same key that was used in the authentication process and verifies the token's signature by making use of the Capability Manager's public key. This component simplifies the access control mechanism to the resources, and it is a relevant feature on IoT scenarios since complex access control policies are not required to be deployed on end devices.

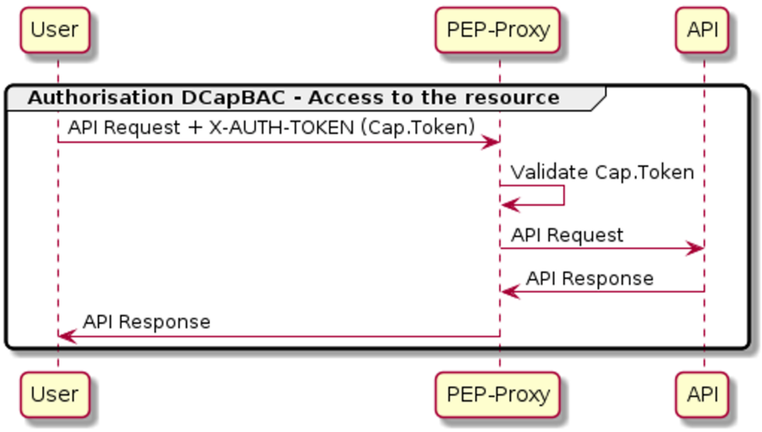


Figure 15: PEP Proxy flow diagram

In the figure above, a user that has already received the capability token from the Capability Manager attempts to access a resource. For this purpose, the user generates a request in which the Cap. Token is attached and that is handled by the PEP Proxy (fig. step 1) which validates the token (fig. step 2). After a positive validation, the PEP Proxy forwards the query to the system (Context Broker) (fig. step 3) as well as forwarding the response (fig. step 4) to the user (fig. step 5).

8.2.2.5 Traceability Agent

The authentication and authorisation traceability component will log the access to DEMETER resources by logging the issue and use of authentication and authorisation tokens. These tokens contain the information about the user who is logged on to the system and the resources the user is intending to access.

The traceability agent will expose a REST API to register authentication and authorisation events (POST) and retrieve their details (GET). The REST API has been designed to be flexible enough to be able to use different traceability blockchain repositories (e.g., Quorum, HyperLedger Fabric, etc.)

The events logged will contain information about the receiver of the token, the sender of the token, the timestamp, the token details, and an optional data field to extend the information of the event.

The UML sequence diagrams are as follows:

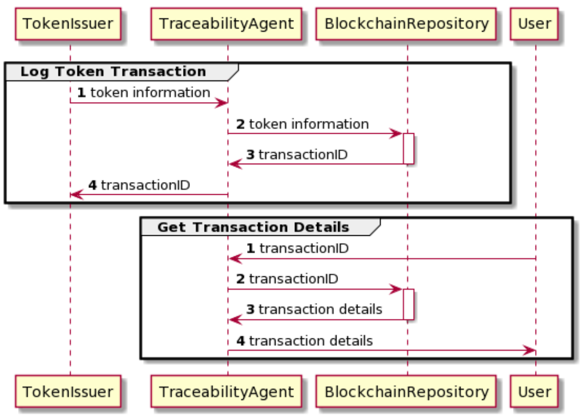


Figure 16: Traceability sequence diagrams

8.2.2.6 *Traceability Blockchain Repository*

A permissioned blockchain repository has been chosen to provide the characteristics of immutability, privacy and compatibility required by the DEMETER Traceability Component. It supports both public and private transactions and smart contracts, and their states derived from a single, common, complete blockchain for transactions validated by every node in the network.

8.2.3 Process View

A user trying to access a DEMETER resource should first get authenticated at the Identity Manager to obtain an authentication token. Once the user is authenticated, the authentication token will be used to request access to DEMETER resources through the authorisation component, as described in the following sequence diagram:

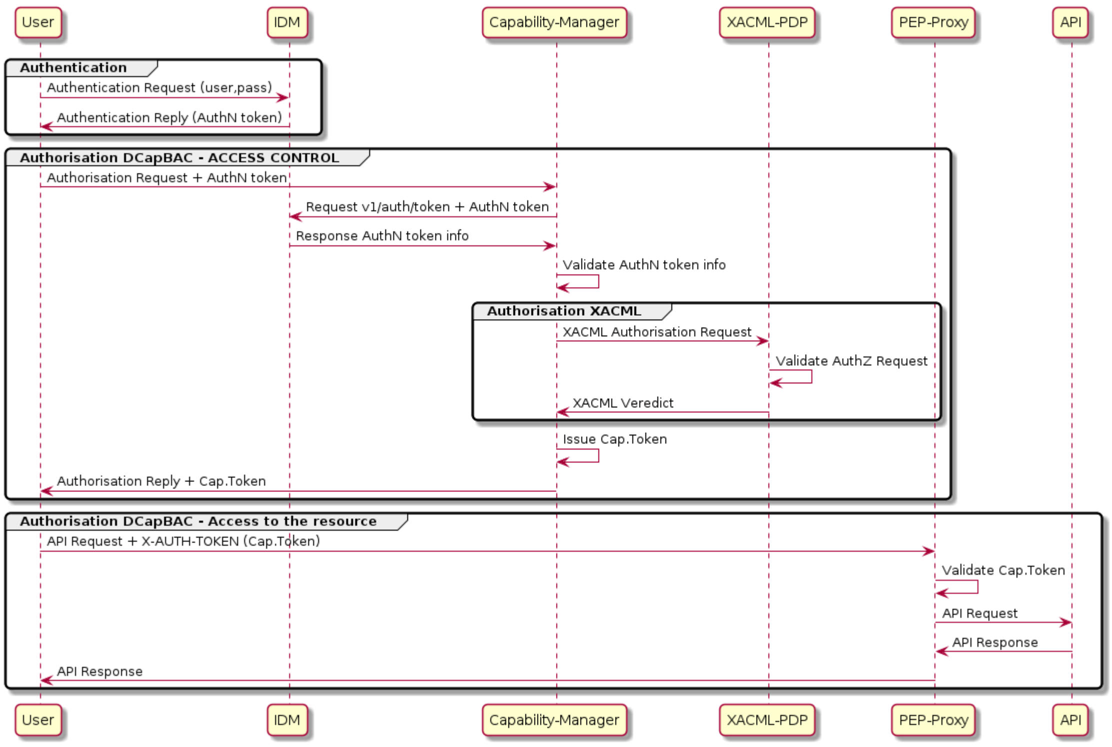

User
IDM
Capability-Manager
XACML-PDP
PEP-Proxy
API
Authentication
Authentication Request (user,pass)
Authentication Reply (AuthN token)
Authorisation DCapBAC - ACCESS CONTROL
Authorisation Request + AuthN token
Request v1/auth/token + AuthN token
Response AuthN token info
Validate AuthN token info
Authorisation XACML
XACML Authorisation Request
Validate AuthZ Request
XACML Veredict
Issue Cap.Token
Authorisation Reply + Cap.Token
Authorisation DCapBAC - Access to the resource
API Request + X-AUTH-TOKEN (Cap.Token)
Validate Cap.Token
API Request
API Response
API Response

Figure 17: Authentication and Authorisation sequence diagram to access DEMETER resources

8.3 *Identity Manager*

8.3.1 Interfaces

8.3.1.1 Data models used in interfaces

The following data models are used by Keyrock to store the information for the application, user, organisation, roles, and authentication tokens:

Table 11: User Data Model

Name	Keyrock User Data Model	
Property	Type	Description
Id	UUID	universally unique identifier
Username	String	sequence of characters that identifies a user
Description	String	text that provides further details about the user
Website	String	URL
Image	String	image to be used by an application representing the user
Gravatar	Integer	Gravatar image
Email	String (unique)	email provided by the user at registration
Password	String	string of characters, used to confirm the identity of the user
Date_Password	DateTime	date when the password was set
Admin	Integer	Boolean value indicating whether the user has administration rights
Extra	JSON	field where a JSON object can be stored to provided extra information

Table 12: Application Data Model

Name	Keyrock Application Data Model	
Property	Type	Description
Id	UUID	universally unique identifier
Name	String	string of characters that identifies the application
Description	String	text that provides further details about the application
URL	String	application's URL
Redirect_URL	String	URL required by the OAuth protocol
Redirec_sign_out_URL	String	the URL to which Keyrock will redirect a user if a sign out is performed from a service
Grant_Type	String	list of grant type authentication allowed for the application
Provider	String	Specify the provider of the application
Extra	JSON	field where a JSON object can be stored to provided extra information

Table 13: Organisation Data Model

Name	Keyrock Organisation Data Model	
Property	Type	Description
Id	UUID	universally unique identifier
Name	String	sequence of characters that identifies the organisation
Description	String	text that provides further details about the organization
Website	String	URL provided

Table 14: Role Data Model

Name	Keyrock Role Data Model	
Property	Type	Description
Id	UUID	universally unique identifier
Name	String	sequence of characters that identifies the role

Table 15: Authentication Token Data Model

Name	*Keyrock* Authentication Token *Data Model*	
Property	Type	Description
Access_Token	String (unique)	string issued by Keyrock as a token identifier
Method	String	specifies the grant type method used for the authentication
Expire_at	DateTime	Date and Time for the expiration of the authentication token

8.3.1.2 Description of API's

The following tables provide the REST API details for the user to obtain a token from Keyrock using username and password, how to refresh that token and how to delete.

More information about Keyrock API can be found at:

- https://keyrock.docs.apiary.io/#introduction/preface/status
- https://swagger.lab.fiware.org/?url=https://raw.githubusercontent.com/FIWARE/specifications/master/OpenAPI/security.Idm/Idm-openapi.json

(REST API)

Title	Create token with Password
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://keyrock/v1/auth/tokens	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Data Params This field holds the body payload of a request.	
Required:	
"name"=[string]	Username set by the user (email)

Required:	
“password”=[string]	Password set by the user
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	

200
 	
Content:
 	
Header:
Content-Type:application/json,application/json; charset=utf-8
X-Subject-Token: 04c5b070-4292-4b3f-911b-36a103f3ac3f
Content-Length:74
ETag:W/"4a-jYFzvNRMQcIZ2P+p5EfmbN+VHcw"
Date:Mon, 19 Mar 2018 15:05:35 GMT
Connection:keep-alive
 	
Body:
{
 "token": {
  "methods": ["password"],
  "expires_at": "2018-03-20T15:05:35.697Z"
 }
}
	Authentication token provided in the header (X-Subject-Token) and token details provided in the Body (method used to obtain the token and token expiration time)

Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
400 Bad Request "Invalid grant: user credentials are invalid" "Invalid client: client is invalid"	 Error response for wrong username and/or wrong password. Error response for wrong client
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
curl --include \\ --request POST \\ --header "Content-Type: application/json" \\ --data-binary "{ \\"name\\": \\"alice@test.com\\", \\"password\\": \\"passw0rd\\" }" \\	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Refresh token
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://keyrock/v1/auth/tokens	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Data Params This field holds the body payload of a request.	
Required:	
“token”=[string]	Token previously obtained
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	

200
 	
Content:
 	
Header:
Content-Type:application/json,application/json; charset=utf-8
X-Subject-Token: 65c6b870-3535-6b4f-345b-34a345f3ac7f
Content-Length:74
ETag:W/"4a-jYFzvNRMQcIZ2P+p5EfmbN+VHcw"
Date:Mon, 19 Mar 2018 16:05:35 GMT
Connection:keep-alive
 	
Body:
{
 "token": {
  "methods": ["password"],
  "expires_at": "2018-03-20T16:05:35.697Z"
 }
}
	Authentication token provided in the header (X-Subject-Token) and token details provided in the Body (method used to obtain the token and token expiration time)

Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
400 Bad Request “Invalid grant: refresh token is no longer valid”	The token provided is no longer valid, therefore, a new authentication token is not provided.
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	

```
curl --include \
     --request POST \
     --header "Content-Type: application/json" \
     --data-binary "{
  \"token\": \"token_id\"
}" \
```

Notes This field holds any additional helpful info related to this endpoint.

Title	Revoke token
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://keyrock/v1/auth/tokens	
Method This field holds the type of the Method used	
DELETE	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
"X-Auth-token: auth_token"	Authentication token previously obtained for the user
"X-Subject-token: subj_token"	Authentication token previously obtained for the user
Data Params This field holds the body payload of a request.	
Required:	
“token”=[string]	Token previously obtained
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
204	Success response for token deletion
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
400 Bad Request “Invalid grant: refresh token is no longer valid”	The token provided is no longer valid.
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	

`curl --include \`
`     --request DELETE \`
`     --header "X-Auth-token: 65c6b870-3535-6b4f-345b-34a345f3ac7f" \`
`     --header "X-Subject-token: 65c6b870-3535-6b4f-345b-34a345f3ac7f" \`	

Notes This field holds any additional helpful info related to this endpoint.	

8.3.2 Technologies and implementation details

The DEMETER Identity Manager has been implemented using the FIWARE Keyrock GE and it will be deployed (along with its database) using Docker containers.

8.4 XACML PDP

The XACML PDP manages the access control policies.

8.4.1 Interfaces

The PDP offers a RES interface to offer to the Capability Manager the verification of an authorisation policy returning a verdict.

8.4.1.1 Data models and interfaces

In an XML format there is a PolicySet with a set of policies, each one with an ID and a set of Rules, Subjects and Actions. The next table of properties shows the most relevant elements that are used by the PDP:

Name	XACML_Policy_Set	
Property	Type	Description
PolicySet.PolicySetId	String	PolicySet ID
Policy.PolicyId	String	Policy ID
Rule.RuleId	String	Rule ID
Rule.Effect	String	Rule effect required (permit/deny/…)
Subject.SubjectMatch.MatchId	String	Subject match XACML function. Examples: string-equal, etc.
Subject.SubjectMatch.AttributeValue.DataType	<any>	Subject value type as XMLSchema type. Example: string, number, etc.
Subject.SubjectMatch.AttributeValue.value	<any>	Subject value
Resource.ResourceMatch.MatchId	String	Resource match XACML function. Examples: string-equal, etc.
Resource.ResourceMatch.AttributeValue.DataType	String	Resource value type as XMLSchema type. Example: string, number, etc.
Resource.ResourceMatch.AttributeValue.value	String	Resource value as an entry point
Action.ActionMatch.MatchId	String	Action match XACML function. Examples: string-equal, etc.
Action.ActionMatch.AttributeValue.DataType	String	Action value type as XMLSchema type. Example: string, number, etc.
Action.ActionMatch.AttributeValue.value	<any>	Action value. Examples: “GET”, “PUT”, etc.

Next there is an example of an XACML policySet in XML format:

```
<PolicySet xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
PolicyCombiningAlgId="urn:oasis:names:tc:xacml:1.0:policy-combining-
algorithm:first-applicable" PolicySetId="POLICY_SET">
    <Policy PolicyId="test1"
RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:first-
applicable">
        <Rule Effect="Permit" RuleId="001">
            <Target>
       <Subjects>
           <Subject>
               <SubjectMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                   <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Peter</AttributeValue>
<SubjectAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:2.0:subject:role"
  DataType="http://www.w3.org/2001/XMLSchema#string" />
               </SubjectMatch>
           </Subject>
       </Subjects>
       <Resources>
           <Resource>
               <ResourceMatch
```

```
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                  <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">https://215.64.19.203:1020/ngsi
-ld/v1/entities?type=http://www.w3.org/ns/sosa/Sensor;idPattern=urn:ngsi-
ld:Sensor:temperature.*
            </AttributeValue>
                  <ResourceAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
              DataType="http://www.w3.org/2001/XMLSchema#string" />
              </ResourceMatch>
          </Resource>
      </Resources>
      <Actions>
          <Action>
              <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-
equal">
                  <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">GET</AttributeValue>
                  <ActionAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
              DataType="http://www.w3.org/2001/XMLSchema#string" />
              </ActionMatch>
          </Action>
      </Actions>
            </Target>
        </Rule>
    </Policy>
</PolicySet>
```

8.4.1.2 Description of API's

Title	Obtain XACML PDP decision
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
/XACMLServletPDP/	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Data Params This field holds the body payload of a post request.	
Required:	
subject= [alphanumeric] <Subject SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"> <Attribute AttributeId="urn:oasis:names:tc:xacml:2.0:subject:role" DataType="http://www.w3.org/2001/XMLSchem	Subject of the resource's request. In DCapBAC scenario, it could correspond with a username (IDM). For example: "Peter"

a#string">
        <AttributeValue>**subject**</AttributeValue>
      </Attribute>
</Subject>	

resource= [alphanumeric]
 	
<Resource>
      <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id" DataType="http://www.w3.org/2001/XMLSchema#string">
 	
<AttributeValue>resource</AttributeValue>
      </Attribute>
   </Resource>
	Resource: endpoint+path of the resource's request (protocol+IP+PORT+path).
	For example: "https://153.55.55.120:2354/ngsi-ld/v1/entities/urn:ngsi-ld:Sensor:humidity.201".
	
	In DCapBAC scenario, endpoint corresponds with the PEP-Proxy one.

action=[alphanumeric]
 	
<Action>
      <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id" DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue>action</AttributeValue>
      </Attribute>
   </Action>
	Action: method of the resource's request
	For example: "POST", "GET", "PATCH", etc.

Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	

200
XACML – Permit
	`<Response>`
	`  <Result ResourceID="resource">`
	`    <Decision>`**`Permit`**`</Decision>`
	`    <Status>`
	`      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>`
	`    </Status>`
	`    <Obligations>`
	`      <Obligation ObligationId="liveTime" FulfillOn="Permit">`
	`      </Obligation>`
	`    </Obligations>`
	`  </Result>`
	`</Response>`

Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	

200
XACML – NotApplicable
	`<Response>`
	`  <Result ResourceID="resource">`

	<Decision>**NotApplicable**</Decision>
	    <Status>
	      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
	    </Status>
	  </Result>
	</Response>

200
XACML – Deny
	<Response>
	  <Result ResourceID="resource">
	    <Decision>**Deny**</Decision>
	    <Status>
	      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
	    </Status>
	  </Result>
	</Response>

Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	

```
curl --location --request POST 'http://<PDP-IP>:8080/XACMLServletPDP/' \
--header 'Content-Type: text/plain' \
--data-raw '<Request xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os">
   <Subject SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject">
       <Attribute AttributeId="urn:oasis:names:tc:xacml:2.0:subject:role"
         DataType="http://www.w3.org/2001/XMLSchema#string">
         <AttributeValue>Peter</AttributeValue>
       </Attribute>
   </Subject>

   <Resource>
       <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
         DataType="http://www.w3.org/2001/XMLSchema#string">
         <AttributeValue>https://153.55.55.120:2354/ngsi-ld/v1/entities/urn:ngsi-ld:Sensor:humidity.201</AttributeValue>
       </Attribute>
   </Resource>

   <Action>
       <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
         DataType="http://www.w3.org/2001/XMLSchema#string">
         <AttributeValue>GET</AttributeValue>
       </Attribute>
   </Action>

   <Environment/>
</Request>'
```

Notes This field holds any additional helpful info related to this endpoint.	

8.4.2 Technologies and implementation details

It is developed using XML and Java Servlets deployed on a Tomcat server.

8.5 Capability Manager

The Capability Manager is the endpoint for authorisation requests.

8.5.1 Interfaces

The Capability Manager offers an interface to respond to an authorisation request to access a resource or to perform an action. If the access is granted the capability token is sent back in the response.

8.5.1.1 Data models used in interfaces

As the Capability Manager acts as an intermediate making a translation of an authorisation request to an XACML authorisation passing it to the XACML PDP, it does not have a specific data model although, in a later stage after the PDP verdict, the Capability Manager creates a capability token which is a JSON signed document with the fields shown in the next table:

Name	Capability Manager Data Models	
Property	Type	Description
"id"	String	Identifier (ID). This field is used to un-equivocally identify a capability token.
"ii"	Numeric	Issued-time (II). It identifies the time at which the token was issued as the number of seconds from 1970-01-01T0:0:0Z.
"is"	String	Issuer (IS). Entity issuing and signing the capability token.
"su"	String	Subject (SU). The subject to which the rights from the token are granted. A public key has been used to validate the legitimacy of the subject.
"de"	String	Device (DE). It is a URI used to unequivocally identify the device to which the token applies.
“si”	String	Signature (SI). It carries the digital signature of the token.
"ar"	String	Access Rights (AR). This field represents the set of rights that the issuer has granted to the subject.
"ac"	String	Action (AC). Its purpose is to identify a specific granted action. Its value could be any CoAP method (GET, POST, PUT and DELETE).
"re"	String	Resource (RE). It represents the resource in the device for which the action is granted.
"nb"	Number	Not Before (NB). It expresses a time value. Before NB the token must not be accepted. Its value cannot be earlier than the II field and it implies the current time must be after or equal than NB.
"na"	Number	Not After (NA). It represents the time after which the token must not be accepted.

An example in JSON format is shown next:

```
{
  "id": "nlqfnfa6nqrlbh9h7tigg28ga1",
  "ii": 1586166961,
  "is": "capabilitymanager@odins.es",
  "su": "Peter",
  "de": "https://153.55.55.120:2354",
"si":"MEUCIEEGwsTKGdlEeUxZv7jsh0UdWoFLud3uqpMDvnlT+GD7AiEAmwEu0FHuG+XyRi9BEAMaVPBI
qRvOJlSIBkBT3K7LHCw=",
  "ar": [
    {
      "ac": "GET",
      "re":"/ngsi-ld/v1/entities/urn:ngsi-ld:Sensor:humidity.201"
    }
  ],
  "nb": 1586167961,
  "na": 1586177961
}
```

8.5.1.2 *Description of API's*

Title	Obtain Capability Token.
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
/	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Data Params This field holds the body payload of a post request.	
Required:	
token=[alphanumeric]	Subject of the resource's request. In DCapBAC scenario, it could correspond with an authentication token (IDM-KeyRock). For example: "753f103c-d8e5-4f4e-8720-13d5e2f55043"
de=[alphanumeric]	Endpoint: resource's request endpoint. Example, for a device (protocol+IP+PORT): "https://153.55.55.120:2354".
ac=[alphanumeric]	Action: method of the resource's request. Example: "POST", "GET", etc.
re=[alphanumeric]	Resource: path of the resource request. Example: "/ngsi-ld/v1/entities/urn:ngsi-ld:Sensor:humidity.201"
Optional:	

Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	

200
Body:
`{`
`"id": "nlqfnfa6nqrlbh9h7tigg28ga1",`
`"ii": 1586166961,`
`"is": "capabilitymanager@odins.es",`
`"su": "Peter",`
`"de": "https://153.55.55.120:2354",`
`"si":"MEUCIEEGwsTKGdlEeUxZv7jsh0UdWoFLud3uqpMDvnlT+GD7AiEAmwEu0FHuG+XyRi9BEAMaVPBIqRvOJlSIBkBT3K7LHCw=",`
`"ar": [`
`{`
`"ac": "GET",`
`"re":"/ngsi-ld/v1/entities/urn:ngsi-ld:Sensor:humidity.201"`
`}`
`],`
`"nb": 1586167961,`
`"na": 1586177961`
`}`
	Capability Token
	"id": Identifier (ID). This field is used to un-equivocally identify a capability token.
	"ii": Issued-time (II). It identifies the time at which the token was issued as the number of seconds from 1970-01-01T0:0:0Z.
	"is": Issuer (IS). Entity issuing and signing the capability token.
	"su": Subject (SU). The subject to which the rights from the token are granted. A public key has been used to validate the legitimacy of the subject.
	"de": Device (DE). It is a URI used to unequivocally identify the device to which the token applies.
	Signature (SI). It carries the digital signature of the token.
	"ar": Access Rights (AR). This field represents the set of rights that the issuer has granted to the subject.
	"ac": Action (AC). Its purpose is to identify a specific granted action. Its value could be any CoAP method (GET, POST, PUT and DELETE).
	"re":" Resource (RE). It represents the resource in the device for which the action is granted.
	"nb": Not Before (NB). It expresses a time value. Before NB the token must not be accepted. Its value cannot be earlier than the II field and it implies the current time must be after or equal than NB.
	"na": Not After (NA). It represents the time after which the token must not be accepted.

Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
500	"Can't generate capability token"
An IdM error code in case of error validating the authentication token (AuthN) on the IdM.	Text error associated to the IdM error code.
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	

`curl --location --request POST 'https://<CAPMAN-IP>:<CAPMAN-PORT>' \`
`--header 'Content-Type: application/json' \`
`--data-raw '{`
`"token": "753f103c-d8e5-4f4e-8720-13d5e2f55043",`
`"de": "https://153.55.55.120:2354",`
`"ac": "GET",`
`"re": "/ngsi-ld/v1/entities/urn:ngsi-ld:Sensor:humidity.201"`
`}'`	

Notes This field holds any additional helpful info related to this endpoint.	

8.5.2 Technologies and implementation details

There is a Java implementation using Eclipse that can be used as an imported JAR file or via an API implemented in Python using Visual Studio Code that uses this same JAR file.

8.6 PEP Proxy

The PEP acts as an intermediate between a user, service, device, etc. and the Information Repository (Broker) and also as a component of validation of the capability token.

8.6.1 Interfaces

The PEP acts as an intermediate and, nowadays, its implementation is integrated with NGSI or NGSI-LD Brokers (i.e. Orion, Scorpio) and, as a component of validation of the capability token, it is mandatory in both cases to include in the headers the X-AUTH-TOKEN that will include the capability token.

8.6.2 Data models used in interfaces

The data model will be the same as the one in the implementation of NGSI or NGSI-LD used in the integrated Broker[5]. Also, as a component of validation of the capability token we need to include the structure of this token that was defined in section 8.5.

8.6.2.1 Description of API's

As mentioned before, this component acts as an intermediate and will use NGSI or NGSI-LD API's depending on the implementation used in the integrated Broker. These API's are offered by Brokers as Orion (NGSI[6]) and Scorpio (NGSI-LD[7]).

8.6.3 Technologies and implementation details

The implementation is made in Python.

8.7 Traceability Agent

The authentication and authorisation traceability component will log the access to DEMETER resources by logging the issue and use of authentication and authorisation tokens. These tokens contain the information about the user who is logged to the system and the resources the user is intended to access.

A permissioned version of a blockchain repository has been chosen to provide the characteristics of immutability, privacy and compatibility required by the DEMETER Traceability Component. It supports both public and private transactions and smart contracts, and their states derived from a single, common, complete blockchain for transactions validated by every node in the network.

[5] NGSI, NGSI-LD Data Models: https://www.fiware.org/developers/data-models/
[6] Orion, NGSI API: https://fiware-orion.readthedocs.io/en/master/user/walkthrough_apiv2/index.html
[7] Scorpio, NGSI-LD API: https://docbox.etsi.org/ISG/CIM/Open/PDF-Copy-of-GS-CIM-009-NGSI-LD-API-V1.2.1-with-line-numbers.pdf

8.7.1 Interfaces

8.7.1.1 Data models used in interfaces

Name	Traceability Agent Data Model	
Property	Type	Description
Receiver	String	user that obtain the right to access a Demeter resource
Sender	String	identification of who is issuing the right
Timestamp	DateTime	time of occurrence of the event
OptionalData	JSON	optional data to extend the information of the registered event

8.7.1.2 Description of API's

Title	Register General Event
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://audit_tool /registerEvent	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
N/A	parameter description
Optional:	
N/A	parameter description
Data Params This field holds the body payload of a request.	
Required:	
Sender=[string]	identification of who is issuing the right
Optional:	
Recipient=[string]	the beneficiary of the right
Payload=[JSON]	Token information payload with the details of the transaction
OptionalData=[JSON]	Optional data
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: {"transactionId"="123e4567-e89b-12d3-a456-426614174000"}	response description value of a key as a transaction ID
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
400	Error response
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	

```
curl --location --request POST ' http://audit_tool /registerEvent ' \
--header 'Content-Type: application/json' \
--data-raw '{
"sender": "753f103c-d8e5-4f4e-8720-13d5e2f55043",
"recipient": "534f503d-f8e6-5g7e-1234-53d8g4d55043",
"payload": {"authentication_token"="753f103c-d8e5-4f4e-8720-13d5e2f55043",
            "timestamp"="2018-03-20T15:05:35.697Z"}
"optionalData": {}
```

}'
Notes This field holds any additional helpful info related to this endpoint.

Title	General Event Details
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://audit_tool/getEvent	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
N/A	parameter description
Optional:	
N/A	parameter description
Data Params This field holds the body payload of a request.	
Optional:	
transaction=[string]	Key value as a transaction ID
t1=YYYY-MM-DD hh:mm:ss[string]	Start of requested time period
t2=YYYY-MM-DD hh:mm:ss[string]	End of requested time period
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	

200
Content:
 	
`['{`
`"sender": “753f103c-d8e5-4f4e-8720-13d5e2f55043”,`
`"tecipient": “https://153.55.55.120:2354”,`
`"payload": {“authentication_token”=“753f103c-d8e5-4f4e-8720-13d5e2f55043”,`
`“timestamp”="2018-03-20T15:05:35.697Z"}`
`"optionalData": {}`
`}’]`
	Event or Events details

Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
400	Error response, transaction ID does not exist
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
`curl -location -request GET "http://audit_tool /getEvent? t1=2021-01-27%2010%3A10%3A10&t2=2021-04-27%2010%3A10%3A10" -H "accept: text/plain"`	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Register Policy
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://audit_tool/policy/register	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
N/A	parameter description
Optional:	
N/A	parameter description
Data Params This field holds the body payload of a request.	
Required:	
domain=[str]	Identification of the policy
digest=[str]	The rules, subjects and actions of the policy
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: "{'version': 1}"	Version of the policy
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
500	Error response, policy already registered
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
`curl -location -request POST "http:// audit_tool /policy/register" -H  "accept: text/plain" -H  "Content-Type: text/plain" -d "{'domain':demeter, 'digest':'ad214c6e2996b6801896d07d90b8f793b1d2c32c1ac56b3724ce0c68f7bb82cg'}"`	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Update Policy
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://audit_tool/policy/update	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
N/A	parameter description
Optional:	
N/A	parameter description
Data Params This field holds the body payload of a request.	
Required:	
domain=[str]	Identification of the policy
digest=[str]	The rules, subjects and actions of the policy
Success response <What should the status code be on success and is there any returned data? This	

is useful when people need to know what their call-backs should expect>	
200 Content: "{'version': 2}"	Version of the policy
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
500	Error response, policy not registered
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
curl -location -request POST "http:// audit_tool /policy/update" -H "accept: text/plain" -H "Content-Type: text/plain" -d "{'domain':demeter, 'digest':'ad214c6e2996b6801896d07d90b8f793b1d2c32c1ac56b3724ce0c68f7bb82cd'}"	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Get Policy
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://audit_tool/policy/{domain}	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
domain	Identification of the policy
Optional:	
N/A	parameter description
Data Params This field holds the body payload of a request.	
Optional:	
N/A	parameter description
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: "{'digest': 'ad214c6e2996b6801896d07d90b8f793b1d2c32c1ac56b3724ce0c68f7bb82cg', 'version': 4}"	Policy details
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
500	Error response, policy not registered
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
curl -location -request GET "http://10.200.10.46:8080/policy/demeter" -H "accept: text/plain"	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Register Token
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://audit_tool/token/register	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
N/A	parameter description
Optional:	
N/A	parameter description
Data Params This field holds the body payload of a request.	
Optional:	
id=[str]	Identification of the policy
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: "{'state': 1}"	Token state
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
500	Error response, policy is already registered
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
`curl -location -request POST "http:// audit_tool /token/register" -H  "accept: text/plain" -H  "Content-Type: text/plain" -d "{'id':'RegisterId2'}"`	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Revoke Token
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://audit_tool/token/revoke	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
N/A	parameter description
Optional:	
N/A	parameter description
Data Params This field holds the body payload of a request.	
Required:	
id=[str]	Identification of the token
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	

200 Content: `"{'state': 0}"`	Revoked Token's state
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
500	Error response, token not registered
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
`curl -location -request POST "http:// audit_tool /token/revoke" -H  "accept: text/plain" -H  "Content-Type: text/plain" -d "{'id':'RegisterId2'}"`	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Verify Token
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://audit_tool/token/{id}	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
id	Identification of the token
Optional:	
N/A	parameter description
Data Params This field holds the body payload of a request.	
Optional:	
N/A	parameter description
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: "{'state': 0}"	Token state
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
500	Error response, token not registered
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
curl -location -request GET "http:// audit_tool /token/RegisterId2" -H "accept: text/plain"	
Notes This field holds any additional helpful info related to this endpoint.	

8.7.1.3 Technologies and implementation details

The traceability agent has been implemented using Django REST Framework and it is deployed using Docker containers.

9 DEMETER Enabler Hub

9.1 Description

The DEMETER Enabler HUB (DEH) is one of the most crucial components of the DEMETER project; it represents the digital space dedicated to (technically capable) end users of DEMETER where they are able to create and register their own resources. Users have two roles; they act as DEMETER Provider and DEMETER Consumer. A DEMETER Provider is able to offer and manage his resources (components, services, data sources, devices, platforms, etc), while DEMETER Consumers will be able to browse it and find suitable resources matching their requirements. In order to support this, the DEH involves communication between various DEMETER components. Taking this into account, each component inside DEH exposes endpoints through their internal API's. Data entities from any Platform, Thing, Application, Service will be managed through these API's, but for the sole purpose of discovery and management of the resource registry maintained by the DEH. To make the solution more flexible and easier to maintain, all components inside the DEH will be developed as separate services and deployed as standalone Docker containers. The DEH Dashboard (DEH Dymer sub-component described below), provided as an external component communicates with Resource Registry Management (RRM), which is composed out of 3 logical modules: Compatibility Checker, Resource Management, and Discovery Management as shown in the figure below:

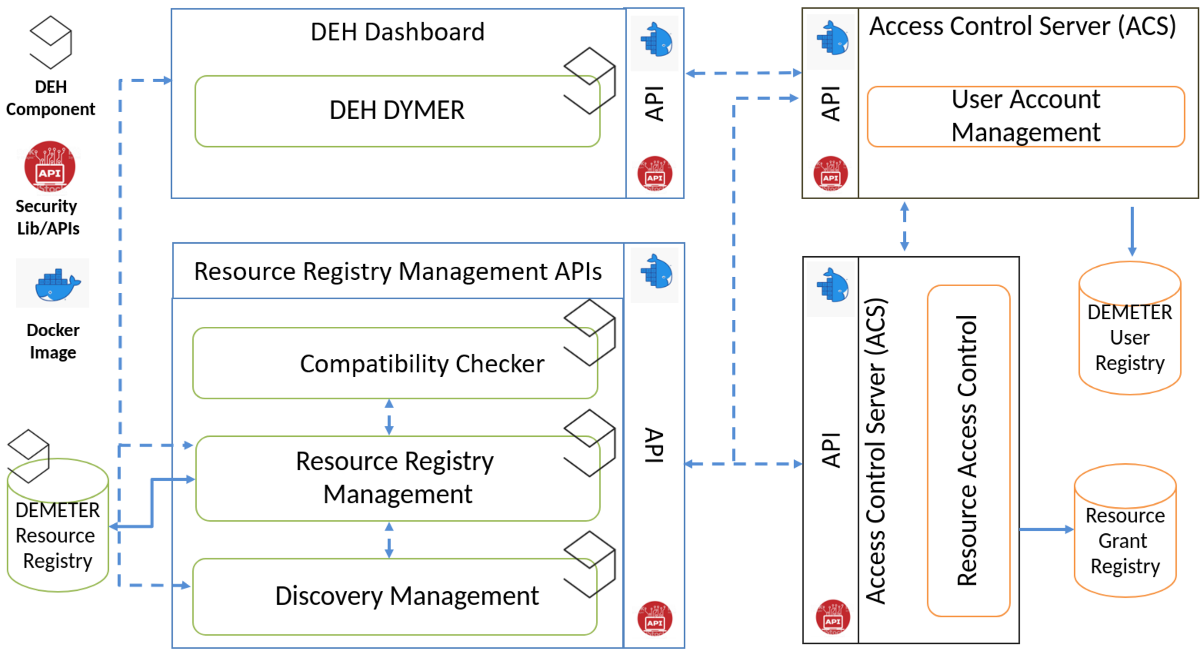


Figure 18: DEH Functional Architecture

Secured communication among all components will be provided by a Security Enabler, more specifically by the Identity Manager, and Access Control Server (Capability Manager) components. The DEH Dashboard communicates with Resource Registry Management (RRM), where all data related to future DEH Resources is inserted by the user, and passed to be verified by Compatibility Checker, and if the data satisfies all necessary requirements, it will be stored inside the DEMETER Resource Registry. In the end, the DEH Dashboard is able to show these resources to the end-users of DEMETER, who intend to view them.

All Hub components are made available for deployment via Docker containerisation. This increases the configurability of the API's and the flexibility of the DEH components by allowing different deployment modes such as cloud-centric or Pilot environments.

9.2 Development view

The development view, also known as an implementation view, depicts the system from an engineer's point of view with a focus on the software module organisation. For the purpose of representing the Development view, UML Component Diagram will be used to depict building blocks, components and their internal modules.

9.2.1 Component diagram

The component diagram, which is also known as a UML component diagram, is used to represent the physical aspect of a system and depicts the organization and the connection between internal components.

Figure 19 shows the component diagram that depicts the decomposition of the DEH into Building Blocks and the relations between them.

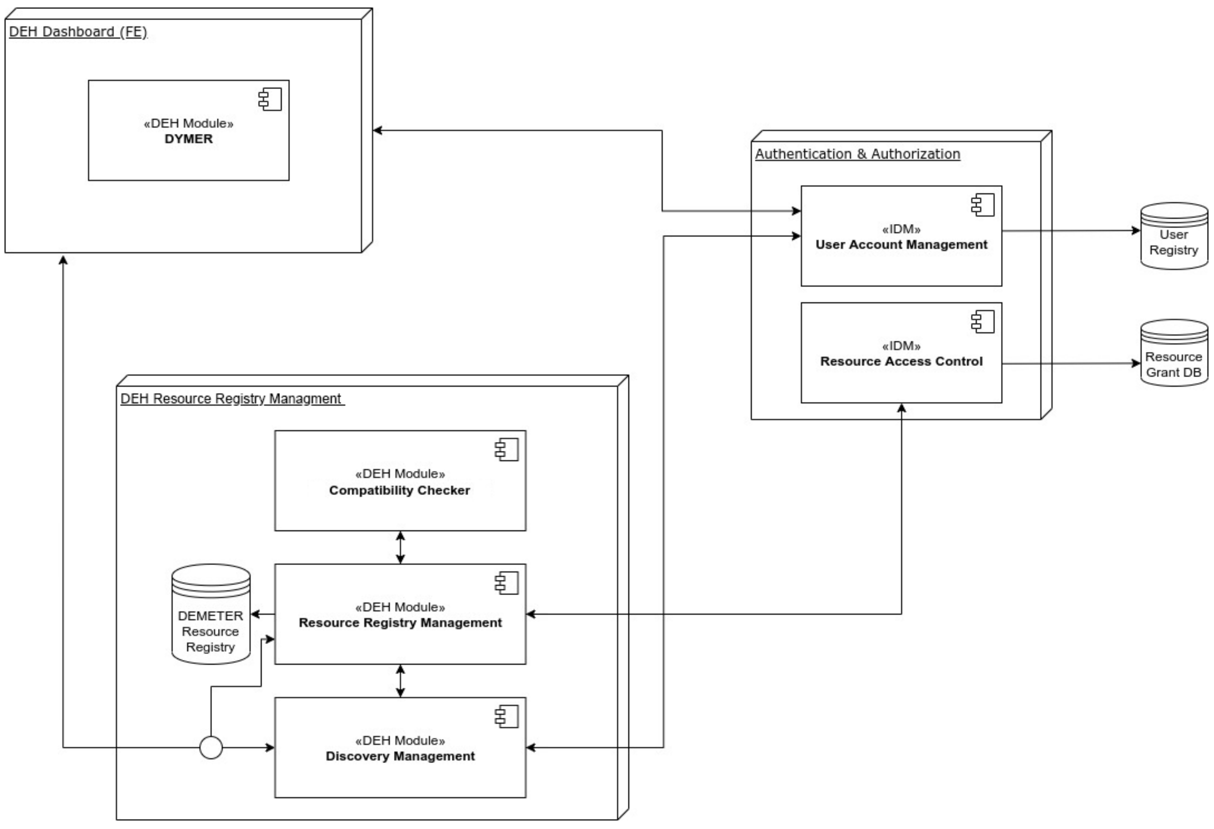


Figure 19: DEMETER Enabler HUB component diagram

DEH Components is composed of three main modules:

- **DEH Dashboard** functional module is in charge of User Interaction & Data Visualisation. It allows users to login to DEH, discover, register and manage DEH Resources.
- **DEH Authentication & Authorization** functional module is in charge of User Authentication and Authorization. It contains information related to users.

DEH Resource Registry Management functional module is in charge for managing DEH Resources. It manages creating, validating, editing, deleting, discovery and consumption of DEH Resources.

9.2.2 Building blocks (components)

This section contains detailed descriptions of components inside the DEH. Each component will be presented in subsections, which contain information about it, its purpose inside the DEH and component diagrams which depict their internal functional modules.

9.2.2.1 DEH Dashboard

DEH Dashboard represents the DEH front-end application, which is used by end-users or DEMETER Stakeholders for resource creation or discovery. The DEH resources are represented by a set of entities such as Component, Device, Service, Dataset, Platform which can be added via DEH Dashboard (User Interface) or Resource Registry Management API's.

This fronted application for Stakeholders is provided by ENG, which in the context of the DEMETER project for this specific component is using technology developed within its research laboratories, namely the DYMER. The DYMER is an open-source suite for resource catalogue visualisation. DYMER provides, on one hand, advanced mapping capabilities between a data model in JSON format and its graphic template, and, on the other hand, a JavaScript framework for integrating the DYMER template into a web-based application. The software is flexible because it adopts open technologies and can be used in various environments without considerable requirements.

The DYMER consists of two main components:

- DYMER-Core (or DYMER business service module)
- DYMER-Viewer

DYMER Core is based on a microservice architectural style with an approach to develop a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms using HTTP/REST protocols alongside JSON.

The diagram in Figure 20 depicts the DEMETER Enabler HUB Dashboard building block components:

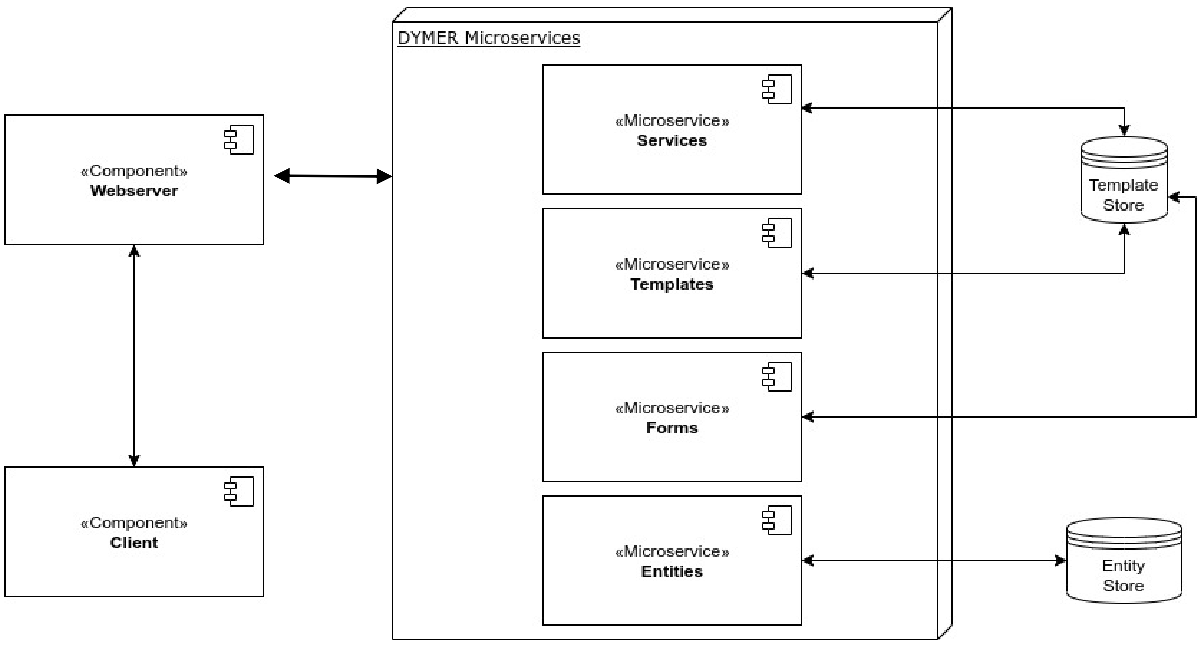


Figure 20: DEMETER Enabler Hub Dashboard building block components

Each microservice is developed with a specific role, however among the main ones we can identify three that have most impact on DEH:

- **Templates microservice** is responsible for generating graphic templates that can be used in order to display products and services using logic-less templates.
- **Forms microservice** is responsible for modelling data and metadata inherent to the products and services offered in DEH.
- **Entities microservice** is responsible for managing the storage and usage of the product and its services.

These microservice are developed with Express.js[8] framework for Node.js[9], designed for building web applications and API's, released as free and open-source software under the MIT License[10].

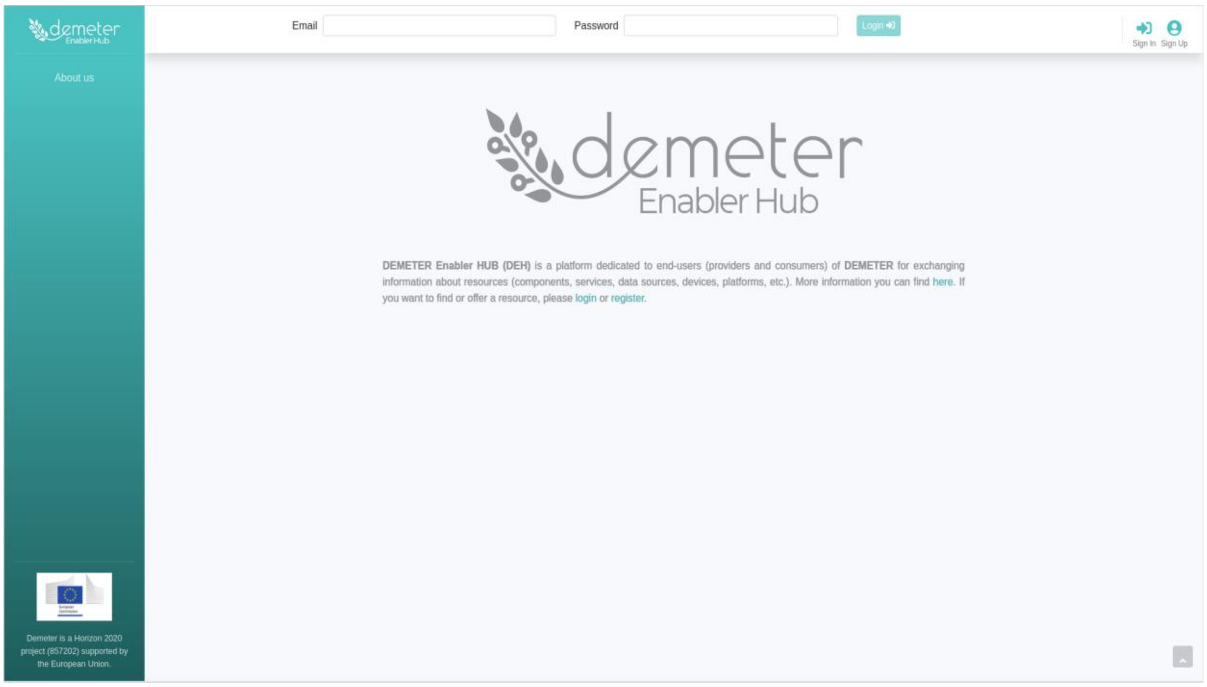
The information is stored in NoSQL[11] Database that provides high performance, high availability, and automatic scaling. Service-Entities use Elasticsearch[12] that is a distributed, open-source search and analytics engine for all types of data, including textual, numerical, geospatial, structured, and unstructured that stores data in JSON format.

Interaction with the **DYMER-Core** takes place through the **DYMER-Viewer** that is a fast, small, and feature-rich JavaScript library. Thanks to it, it is possible to interact with the platform facilitating the user in the use of the data by offering a single search point and displaying the results in special graphic templates.

[8] https://expressjs.com/
[9] https://nodejs.org/
[10] https://opensource.org/licenses/MIT
[11] https://en.wikipedia.org/wiki/NoSQL
[12] https://www.elastic.co/

When it comes to the security part, DYMER will communicate with User Account management and Resource Access Control components.

DYMER created the frontend technology that allows the creation of the Dashboard user for the DEMETER resource catalogue. The DEH end-user can register themselves to the DEH catalogue through a specific registration form made available by the Fiware-IdM component which is integrated with the catalogue through its API. At the end of the registration the user will be able to access to the DEH resource catalogue service as shown in the figure below:



DEMETER Enabler HUB (DEH) is a platform dedicated to end-users (providers and consumers) of **DEMETER** for exchanging information about resources (components, services, data sources, devices, platforms, etc.). More information you can find here. If you want to find or offer a resource, please login or register.

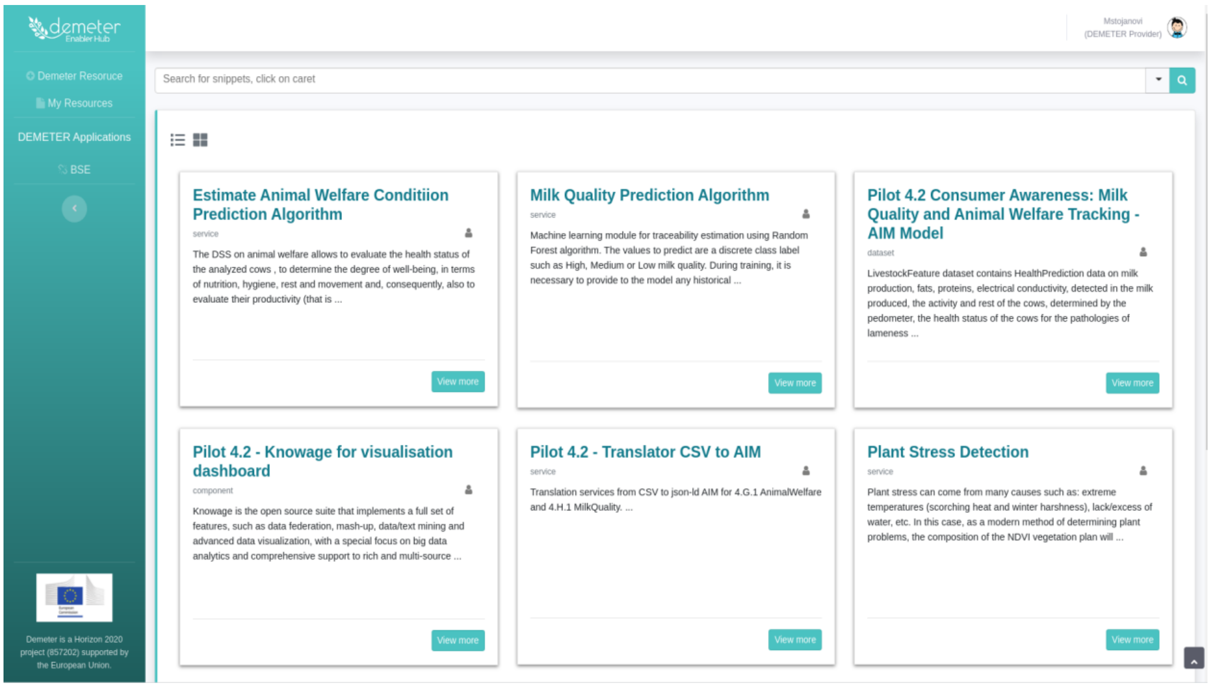
Demeter is a Horizon 2020 project (857202) supported by the European Union.

Figure 21: DEH Dashboard user login

By entering the credentials (Username and Password) the user can access the DEH catalogue of resources and to a whole series of features that allows the logged in user to discover the existing resources in the catalogue, or to create new ones if he has a PROVIDER access role. Most of the functionality that the catalogue was to offer were already developed during the first iteration of the DEH development process. A first software release has already been released after careful functionality and integration tests with the other DEMETER modules. However, refinements and additions to existing functionalities are not excluded to meet the needs of DEMETER Pilots who will make requests in this sense from time to time.

The DEH Dashboard is deployed and available on DEMETER cloud[13]. The following screenshots describe the features available in DEMETER Enabler Hub (DEH). Figure 22 shows the dashboard with the resource catalogue that appears to the DEH user after successful login. The DEH user can navigate through the list of resources, which will be paged if the number of resources exceeds a specific threshold. In the list, the DEH user can see the name, type (of Component, Device, Service, Dataset, Platform), and a short description of DEH Resources, but it also has a button for displaying more information about specific DEH Resource. Figure 23 shows DEH Resource Details page. DEH Users can also access a list of their own resources, this is possible by clicking on button the "My Resources" on the left side menu.

[13] https://deh.h2020-demeter-cloud.eu


demeter Enabler Hub
Demeter Resource
My Resources
DEMETER Applications
BSE
Mstojanovi (DEMETER Provider)
Search for snippets, click on caret
View more
Demeter is a Horizon 2020 project (857202) supported by the European Union.

Estimate Animal Welfare Conditiion Prediction Algorithm
service
The DSS on animal welfare allows to evaluate the health status of the analyzed cows , to determine the degree of well-being, in terms of nutrition, hygiene, rest and movement and, consequently, also to evaluate their productivity (that is ...

Milk Quality Prediction Algorithm
service
Machine learning module for traceability estimation using Random Forest algorithm. The values to predict are a discrete class label such as High, Medium or Low milk quality. During training, it is necessary to provide to the model any historical ...

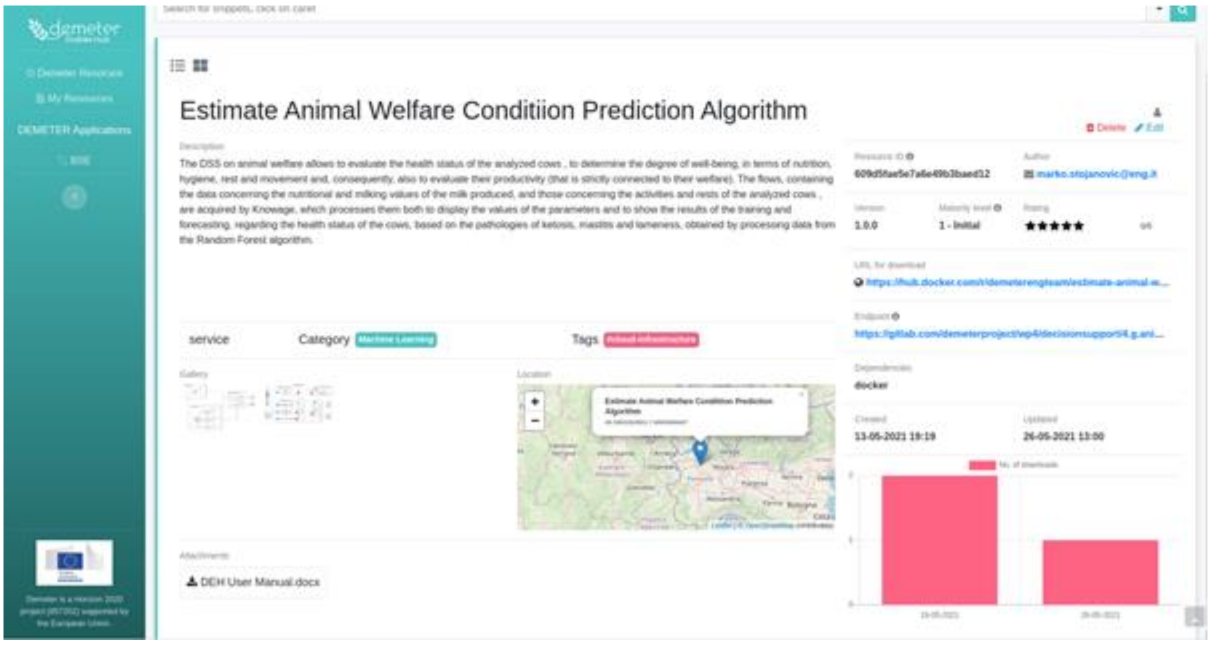
Pilot 4.2 Consumer Awareness: Milk Quality and Animal Welfare Tracking - AIM Model
dataset
LivestockFeature dataset contains HealthPrediction data on milk production, fats, proteins, electrical conductivity, detected in the milk produced, the activity and rest of the cows, determined by the pedometer, the health status of the cows for the pathologies of lameness ...

Pilot 4.2 - Knowage for visualisation dashboard
component
Knowage is the open source suite that implements a full set of features, such as data federation, mash-up, data/text mining and advanced data visualization, with a special focus on big data analytics and comprehensive support to rich and multi-source ...

Pilot 4.2 - Translator CSV to AIM
service
Translation services from CSV to json-ld AIM for 4.G.1 AnimalWelfare and 4.H.1 MilkQuality. ...

Plant Stress Detection
service
Plant stress can come from many causes such as: extreme temperatures (scorching heat and winter harshness), lack/excess of water, etc. In this case, as a modern method of determining plant problems, the composition of the NDVI vegetation plan will ...

Figure 22: DEH Dashboard – Resource catalogue



Estimate Animal Welfare Conditiion Prediction Algorithm

Description
The DSS on animal welfare allows to evaluate the health status of the analyzed cows , to determine the degree of well-being, in terms of nutrition, hygiene, rest and movement and, consequently, also to evaluate their productivity (that is strictly connected to their welfare). The flows, containing the data concerning the nutritional and milking values of the milk produced, and those concerning the activities and rests of the analyzed cows , are acquired by Knowage, which processes them both to display the values of the parameters and to show the results of the training and forecasting, regarding the health status of the cows, based on the pathologies of ketosis, mastitis and lameness, obtained by processing data from the Random Forest algorithm.

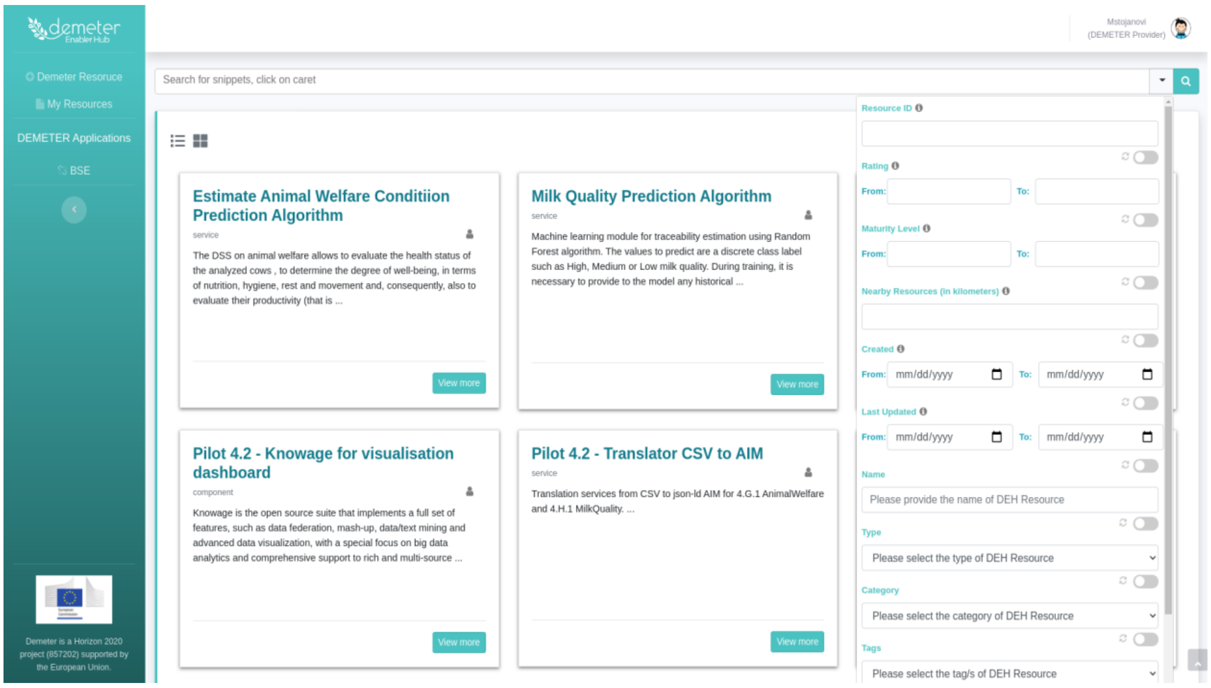
service Category Machine Learning Tags [illegible]

Attachments
DEH User Manual.docx

Resource ID: 609d5fae5e7a6e49b3baed12
Author: marko.stojanovic@eng.it
Version: 1.0.0
Maturity level: 1 - Initial
Rating: ★★★★★
URL for download: https://hub.docker.com/r/demeterengteam/estimate-animal-w...
Endpoint: https://gitlab.com/demeterproject/wp4/decisionsupport4.g.ani...
Dependencies: docker
Created: 13-05-2021 18:19
Updated: 26-05-2021 13:00

Figure 23: DEH Dashboard - DEH Resource Details

The user is also able to perform a search among the resources in the list, using a special search function. The search can be done by the next DEH Resource attributes: Resource ID, Rating (range), Maturity Level (range), Nearby Resources (in kilometres), Created (range), Last updated (range), Name, Type, Category, Tags and status. Figure 24 shows how a user can access the search filters made available by the DEH dashboard to perform targeted searches on the resources contained in the catalogue:



demeter Enabler Hub

Demeter Resoruce
My Resources
DEMETER Applications
BSE

Demeter is a Horizon 2020 project (857202) supported by the European Union.

Mstojanovi (DEMETER Provider)

Search for snippets, click on caret

Estimate Animal Welfare Conditiion Prediction Algorithm
service
The DSS on animal welfare allows to evaluate the health status of the analyzed cows , to determine the degree of well-being, in terms of nutrition, hygiene, rest and movement and, consequently, also to evaluate their productivity (that is ...
View more

Milk Quality Prediction Algorithm
service
Machine learning module for traceability estimation using Random Forest algorithm. The values to predict are a discrete class label such as High, Medium or Low milk quality. During training, it is necessary to provide to the model any historical ...
View more

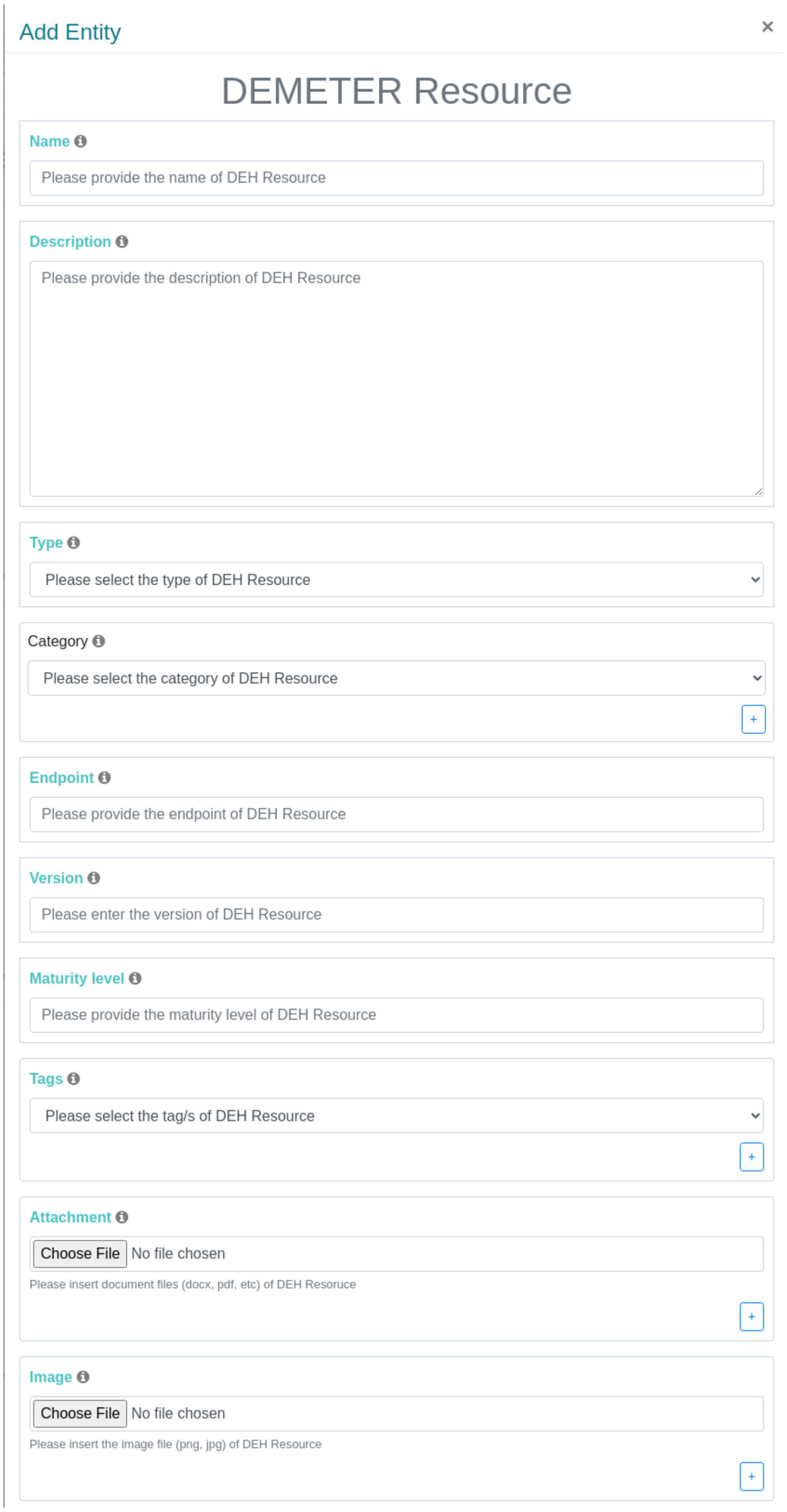
Pilot 4.2 - Knowage for visualisation dashboard
component
Knowage is the open source suite that implements a full set of features, such as data federation, mash-up, data/text mining and advanced data visualization, with a special focus on big data analytics and comprehensive support to rich and multi-source ...
View more

Pilot 4.2 - Translator CSV to AIM
service
Translation services from CSV to json-ld AIM for 4.G.1 AnimalWelfare and 4.H.1 MilkQuality. ...
View more

Resource ID
Rating From: To:
Maturity Level From: To:
Nearby Resources (in kilometers)
Created From: mm/dd/yyyy To: mm/dd/yyyy
Last Updated From: mm/dd/yyyy To: mm/dd/yyyy
Name: Please provide the name of DEH Resource
Type: Please select the type of DEH Resource
Category: Please select the category of DEH Resource
Tags: Please select the tag/s of DEH Resource

Figure 24: DEH Dashboard search filter on resource catalogue

If the DEH user is DEMETER Provider, it can insert a new DEH resource, edit or delete an existing one. By clicking on the "+ DEMETER Resource " button on the left side menu, the DEH user can access a mode that shows the data entry form for the DEH resource as shown on Figure 26 and Figure 27.



Add Entity

×

DEMETER Resource

Name

Please provide the name of DEH Resource

Description

Please provide the description of DEH Resource

Type

Please select the type of DEH Resource

Category

Please select the category of DEH Resource

Endpoint

Please provide the endpoint of DEH Resource

Version

Please enter the version of DEH Resource

Maturity level

Please provide the maturity level of DEH Resource

Tags

Please select the tag/s of DEH Resource

Attachment

Choose File No file chosen

Please insert document files (docx, pdf, etc) of DEH Resoruce

Image

Choose File No file chosen

Please insert the image file (png, jpg) of DEH Resource

Figure 25: DEH Dashboard – create new DEH resource form first part

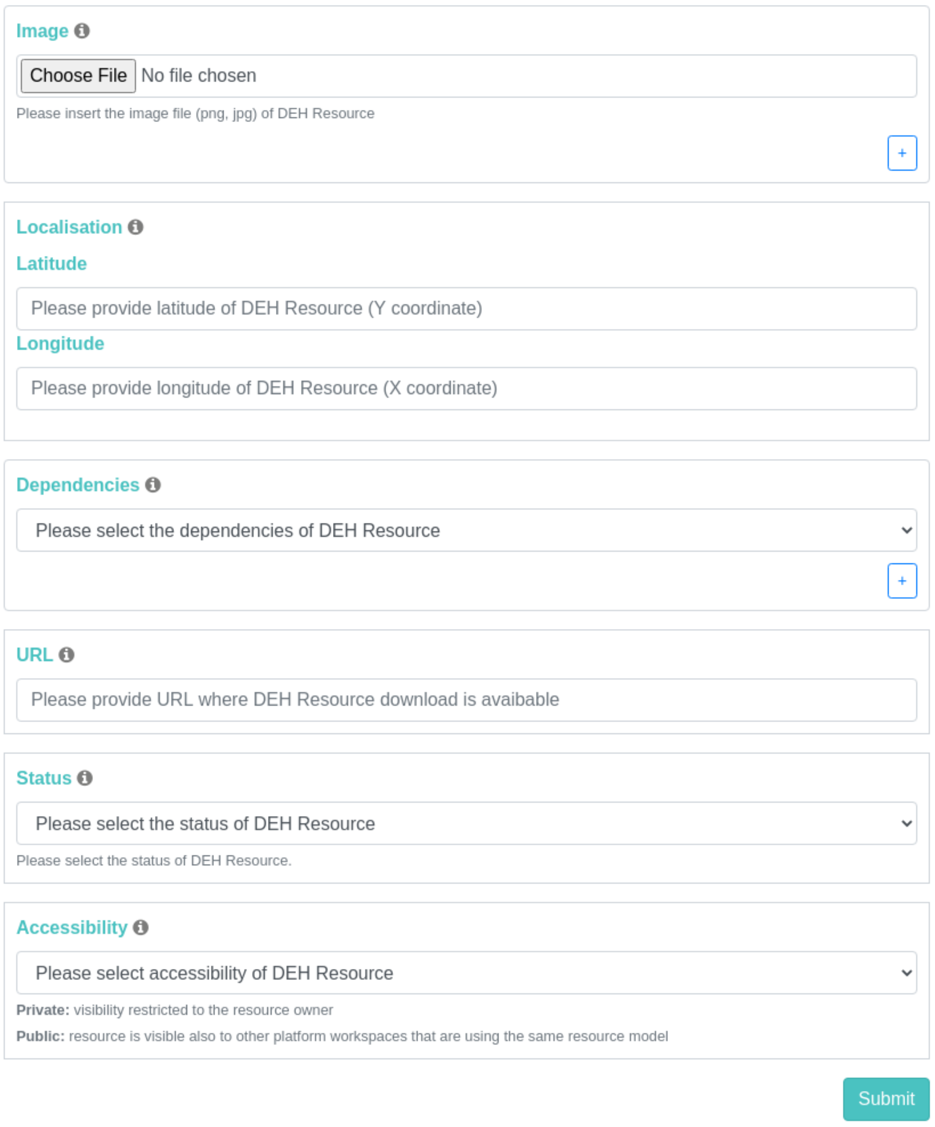


Image
Choose File No file chosen
Please insert the image file (png, jpg) of DEH Resource

Localisation
Latitude
Please provide latitude of DEH Resource (Y coordinate)
Longitude
Please provide longitude of DEH Resource (X coordinate)

Dependencies
Please select the dependencies of DEH Resource

URL
Please provide URL where DEH Resource download is avaibable

Status
Please select the status of DEH Resource
Please select the status of DEH Resource.

Accessibility
Please select accessibility of DEH Resource
Private: visibility restricted to the resource owner
Public: resource is visible also to other platform workspaces that are using the same resource model

Figure 26: DEH Dashboard – create new DEH resource form second part

The user can define a whole series of details relating to the resource he intends to create, including the name, description, type, category, endpoint (URL where information about DEH Resource is available. Also, in the case of REST APIs, the endpoint where DEH Resource is exposed), version, maturity level, tags, location of a resource, dependencies, URL where DEH Resource can be downloaded/used, the status of a resource, and finally the accessibility (this property is related to the user's need to make the resource public or discoverable by the other users of the DEH or private).

Finally, the DEH user can decide to delete the resource from the register or simply edit it (only if the user is the owner of the selected resource). Those functions will appear on the DEH Resource details page, in the upper right corner, as shown in Figure 23 and Figure 27.

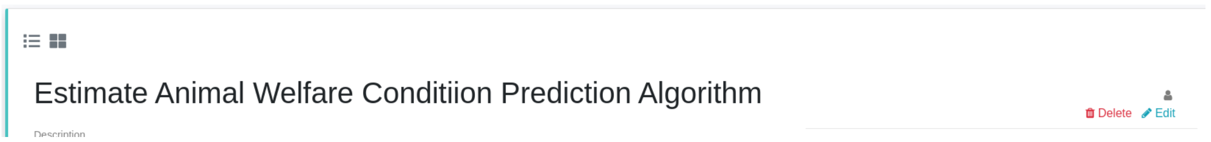
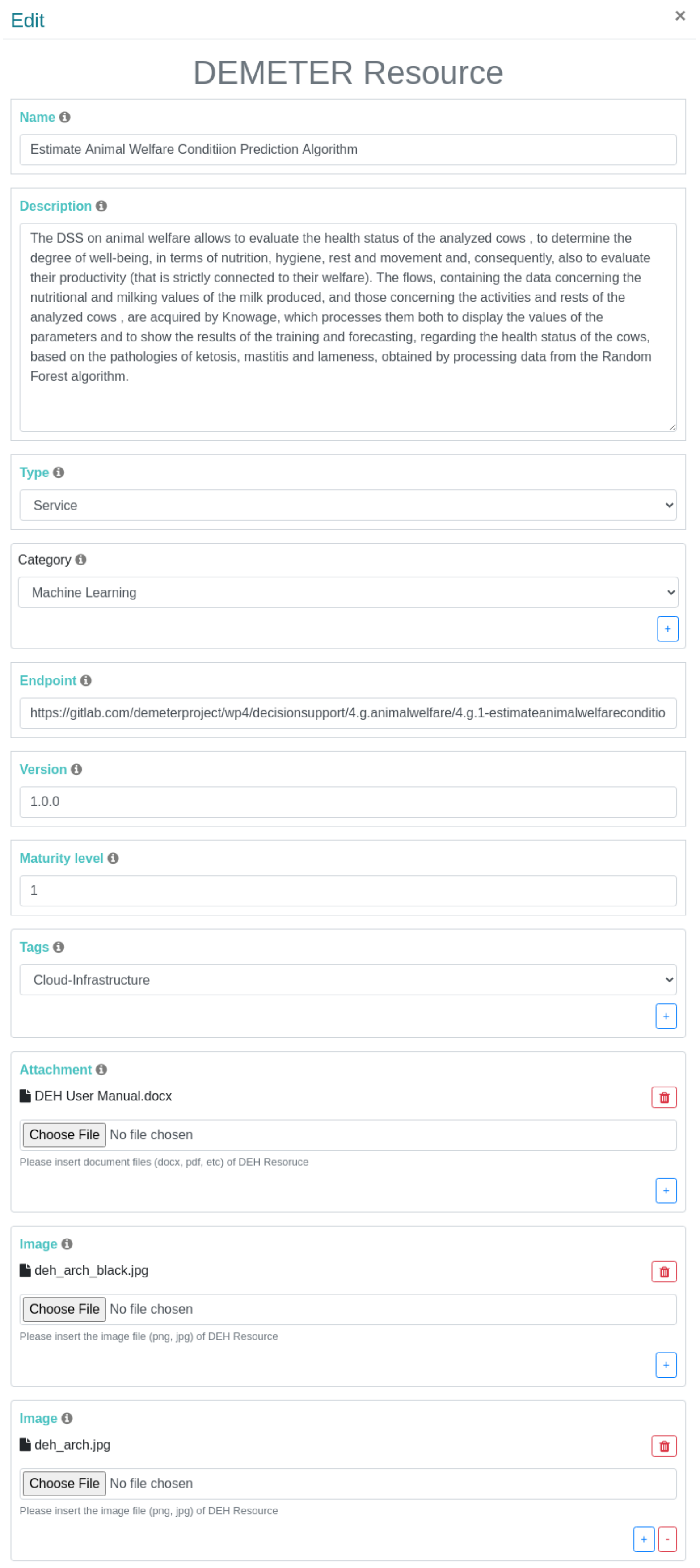

Estimate Animal Welfare Conditiion Prediction Algorithm

Figure 27: DEH Dashboard edit and delete functions

In the case of deleting DEH Resource, the user must click on the "Delete" button, confirm it and wait for the system to send a visual message with the result of deleting the selected resource from the resource registry. In the case of editing the resource, Figure 28 and Figure 29 show the forms that the user accesses by simply clicking on the “Edit” button:


Edit

DEMETER Resource

Name
Estimate Animal Welfare Conditiion Prediction Algorithm

Description
The DSS on animal welfare allows to evaluate the health status of the analyzed cows , to determine the degree of well-being, in terms of nutrition, hygiene, rest and movement and, consequently, also to evaluate their productivity (that is strictly connected to their welfare). The flows, containing the data concerning the nutritional and milking values of the milk produced, and those concerning the activities and rests of the analyzed cows , are acquired by Knowage, which processes them both to display the values of the parameters and to show the results of the training and forecasting, regarding the health status of the cows, based on the pathologies of ketosis, mastitis and lameness, obtained by processing data from the Random Forest algorithm.

Type
Service

Category
Machine Learning

Endpoint
https://gitlab.com/demeterproject/wp4/decisionsupport/4.g.animalwelfare/4.g.1-estimateanimalwelfareconditio

Version
1.0.0

Maturity level
1

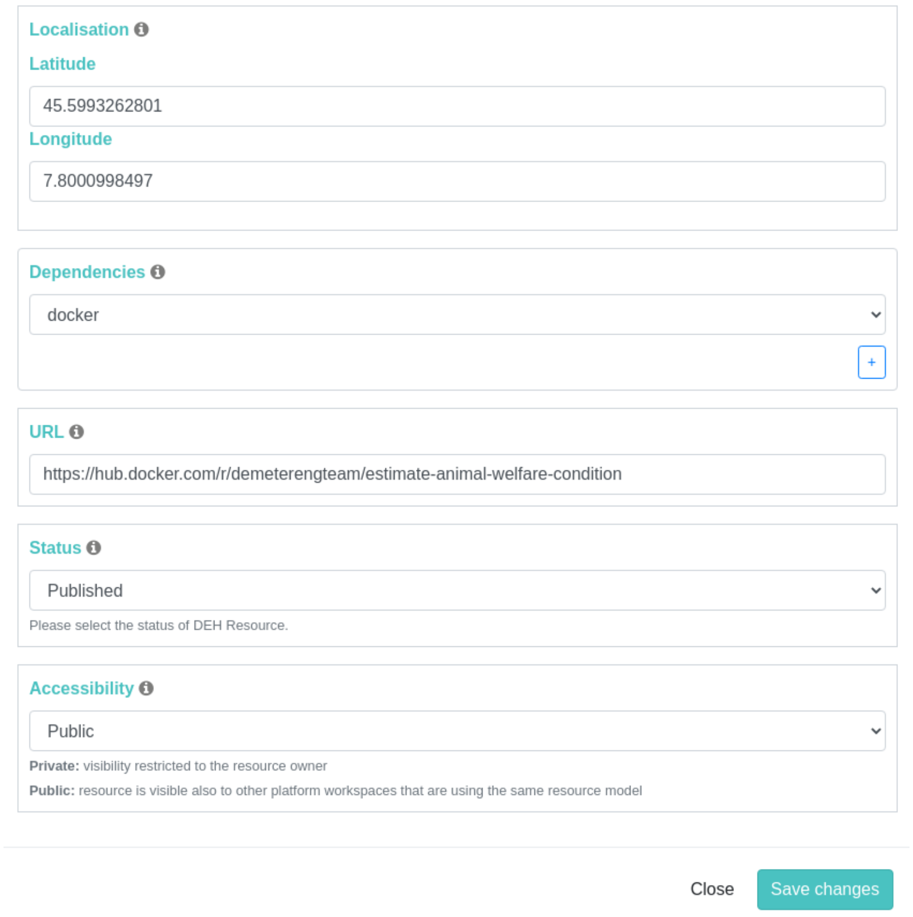
Tags
Cloud-Infrastructure

Attachment
DEH User Manual.docx
Choose File No file chosen
Please insert document files (docx, pdf, etc) of DEH Resoruce

Image
deh_arch_black.jpg
Choose File No file chosen
Please insert the image file (png, jpg) of DEH Resource

Image
deh_arch.jpg
Choose File No file chosen
Please insert the image file (png, jpg) of DEH Resource

Figure 28: DEH Dashboard – edit existing DEH resource form first part


Localisation
Latitude
45.5993262801
Longitude
7.8000998497

Dependencies
docker

URL
https://hub.docker.com/r/demeterengteam/estimate-animal-welfare-condition

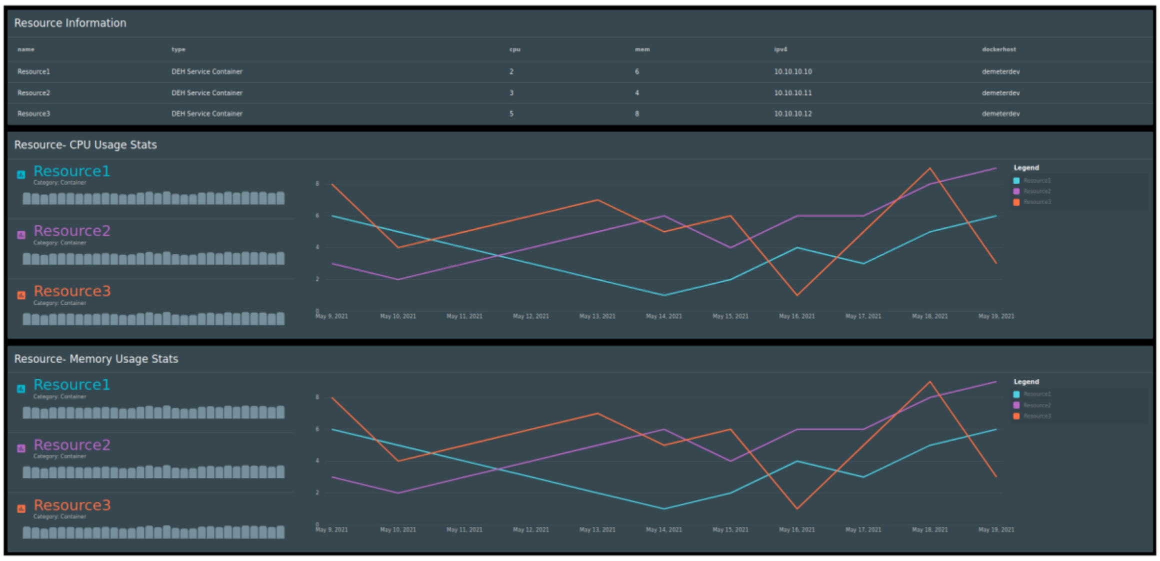
Status
Published
Please select the status of DEH Resource.

Accessibility
Public
Private: visibility restricted to the resource owner
Public: resource is visible also to other platform workspaces that are using the same resource model

Figure 29: DEH Dashboard – edit existing DEH resource form second part

Changing a resource means changing all the data previously defined in the creation phase, allowing the user to correct any errors such as bad typing of information.

Besides managing DEH resources, the user is able to track their consumption. This feature is part of the DEH Metrics Dashboard. On this dashboard, the user is able to see information about running container instances of his DEH Resources. He is able to see information such as a number of deployed instances of specific DEH Resource and information related to each running container such as container id, name, type, IP address, hostname, last update time, and CPU and memory history consumption on daily basis. In the first diagram, user is able to see the diagram with the highest peak of CPU and Memory usage per day, but he is also able to get a more detailed view if he clicks on a specific container. There, he will be able to see all captured peaks from DEH Client which are stored inside DEH Resource Registry DB. DEH Metrics dashboard will be updated based on feedback gathered from pilots, and participants of future workshops. Figure 30 shows current implementation of DEH Metrics Dashboard.

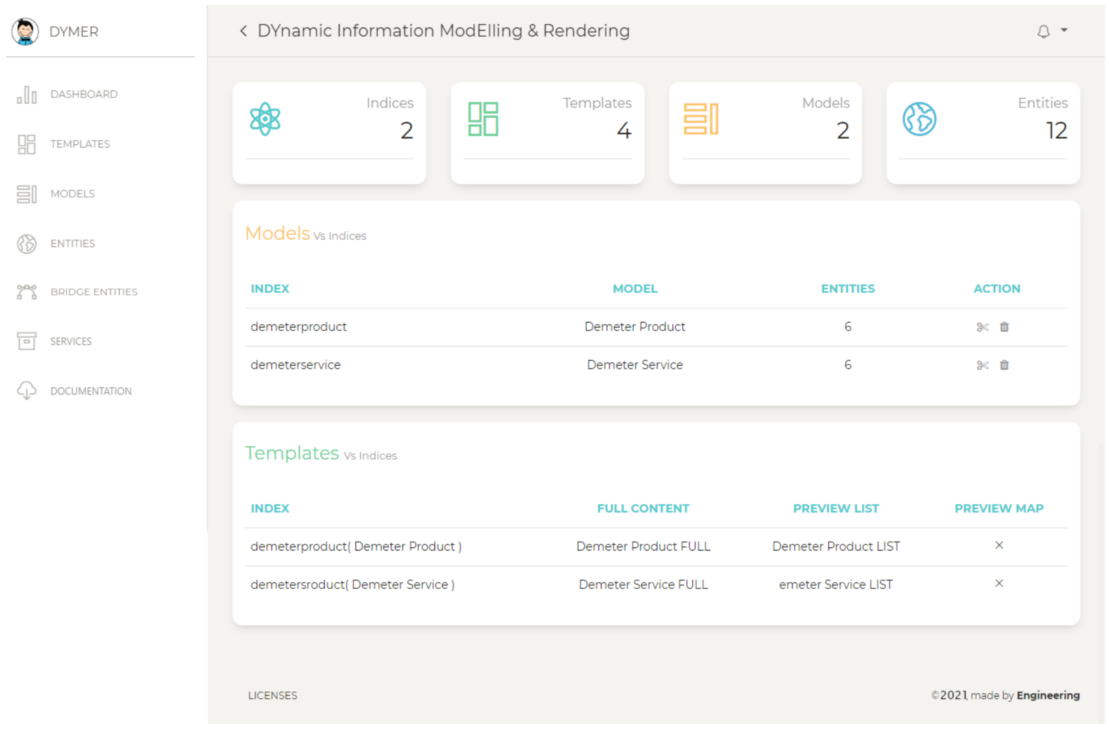


Resource Information

name	type	cpu	mem	ipv4	dockerhost
Resource1	DEH Service Container	2	6	10.10.10.10	demeterdev
Resource2	DEH Service Container	3	4	10.10.10.11	demeterdev
Resource3	DEH Service Container	5	8	10.10.10.12	demeterdev

Figure 30: DEH Metrics Dashboard

The DYMER also implements administration functionality represented by a web-based application, external to the DEH catalogue, to allow a user with Admin role to have complete management of Templates, Models, Forms and Entities. Figure 31 shows administration dashboard of DYMER component:



Models Vs Indices

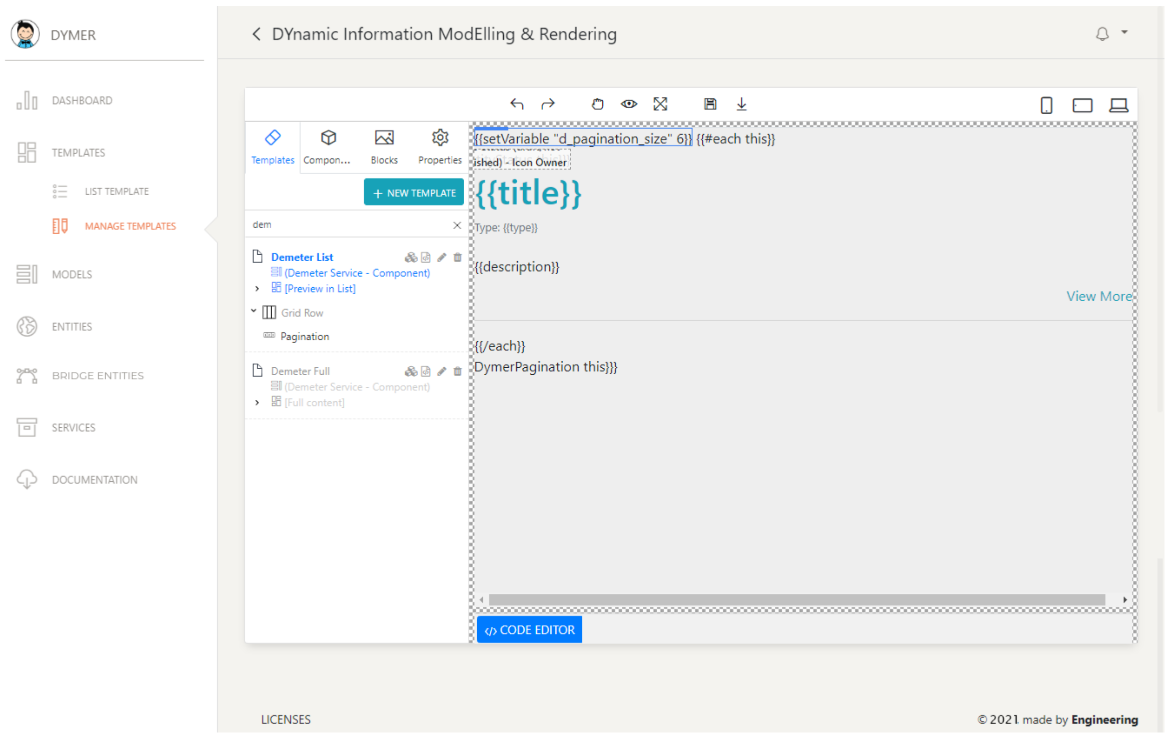
INDEX	MODEL	ENTITIES	ACTION
demeterproduct	Demeter Product	6	
demeterservice	Demeter Service	6	

Templates Vs Indices

INDEX	FULL CONTENT	PREVIEW LIST	PREVIEW MAP
demeterproduct(Demeter Product)	Demeter Product FULL	Demeter Product LIST	×
demetersroduct(Demeter Service)	Demeter Service FULL	emeter Service LIST	×

Figure 31: DYMER administration dashboard

By clicking on the Templates link menu, on the left in the drop-down list, the user can access to the list of the currently registered Templates, in order to view them or create new templates through "Manage Template" functionality or modify the existing ones. Figure 32 shows the Template list:



```
{{setVariable "d_pagination_size" 6}} {{#each this}}
ished) - Icon Owner
{{title}}
Type: {{type}}
{{description}}
{{/each}}
DymerPagination this}}}
```

Figure 32: DYMER administration – template management

The same management features are available respectively for the models/forms and for the entities as the figures below shown:

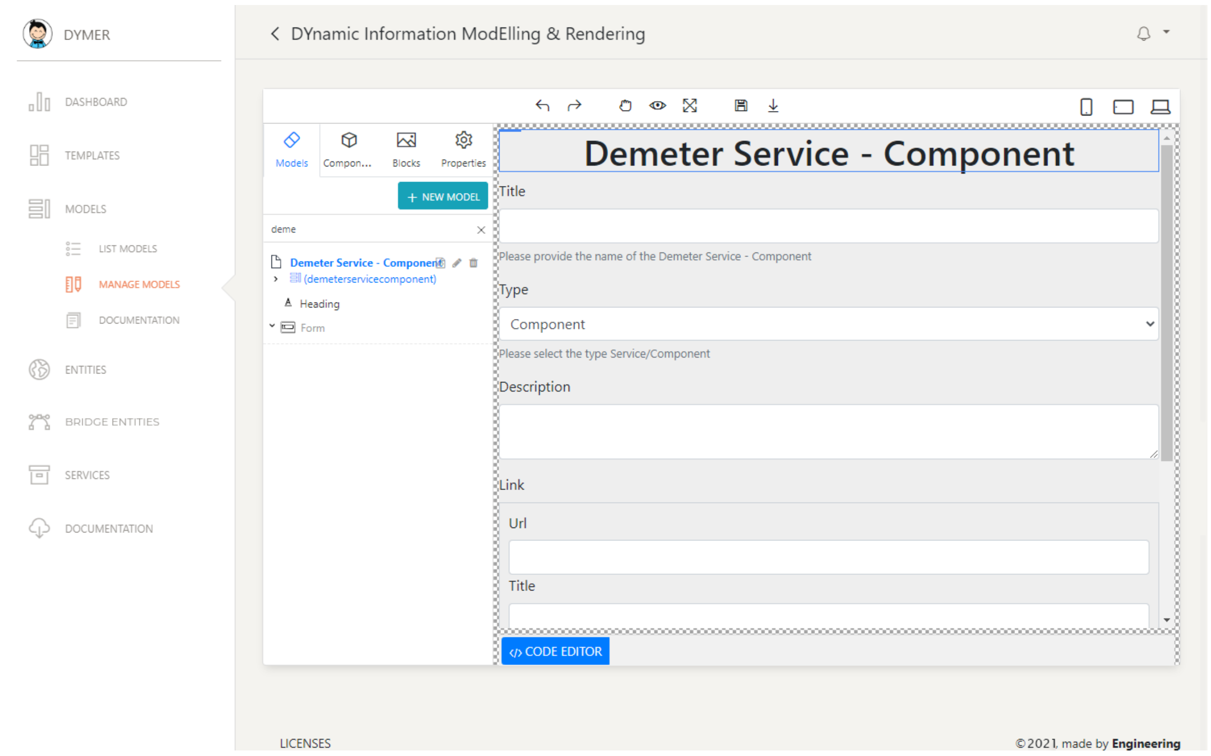

DYMER
DASHBOARD
TEMPLATES
MODELS
LIST MODELS
MANAGE MODELS
DOCUMENTATION
ENTITIES
BRIDGE ENTITIES
SERVICES
DOCUMENTATION
DYnamic Information ModElling & Rendering
Models
Compon...
Blocks
Properties
NEW MODEL
deme
Demeter Service - Component
(demeterservicecomponent)
Heading
Form
Demeter Service - Component
Title
Please provide the name of the Demeter Service - Component
Type
Component
Please select the type Service/Component
Description
Link
Url
Title
CODE EDITOR
LICENSES
©2021, made by Engineering

Figure 33: DYMER administration– models/forms management

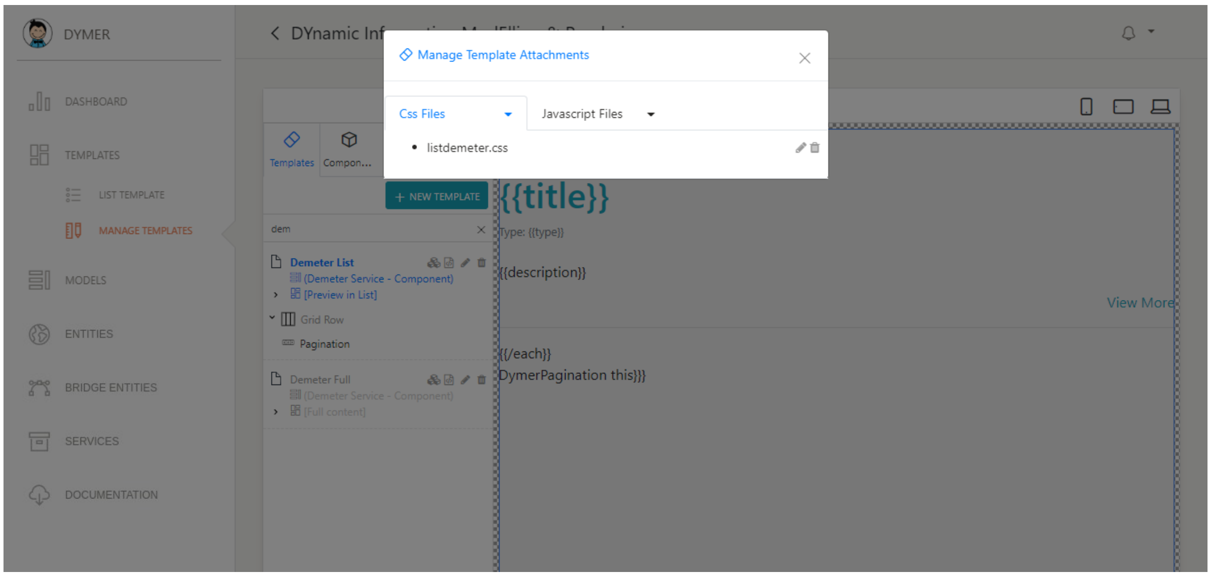


Figure 34: DYMER administration– models/template manage css/javascript

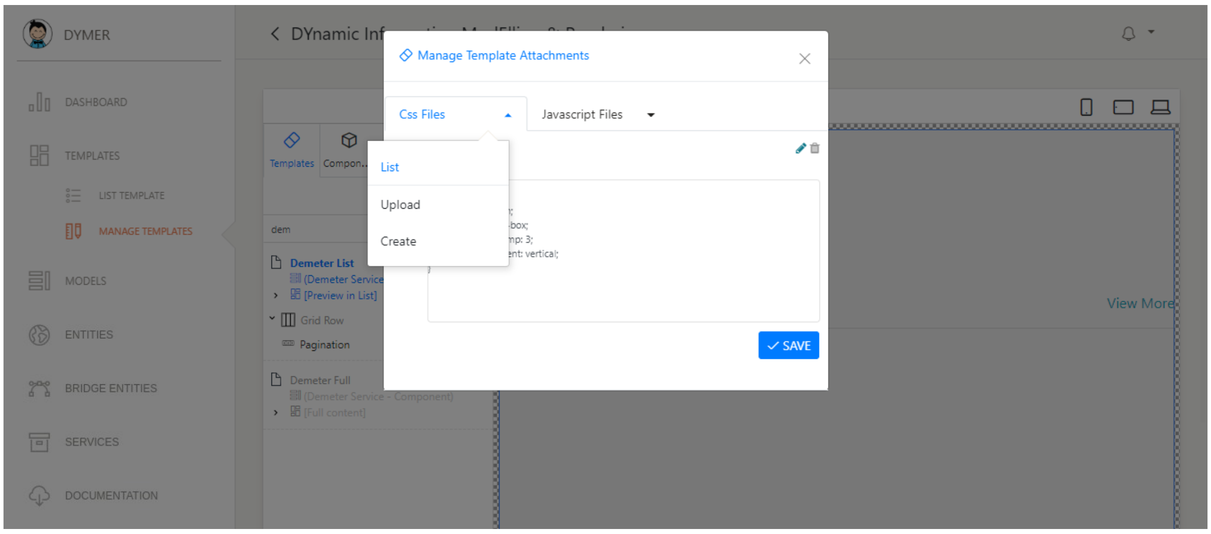
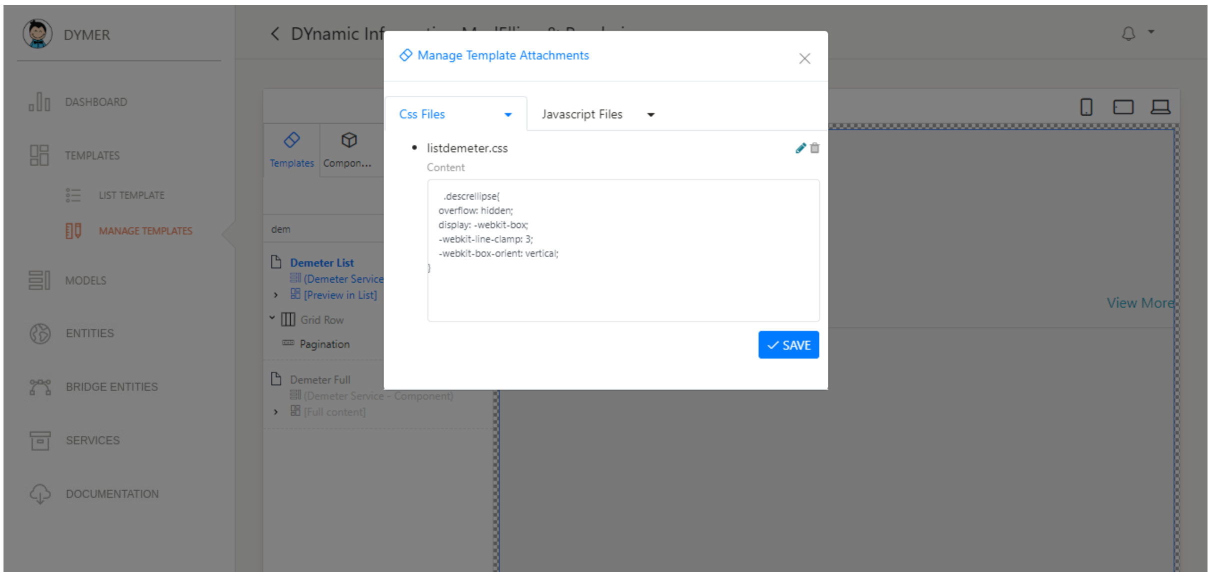


Figure 35: DYMER administration– models/template crud css/javascript



```
  .descrellipse{
 overflow: hidden;
 display: -webkit-box;
 -webkit-line-clamp: 3;
 -webkit-box-orient: vertical;
}
```

Figure 36: DYMER administration– models/template manage edit css/javascript

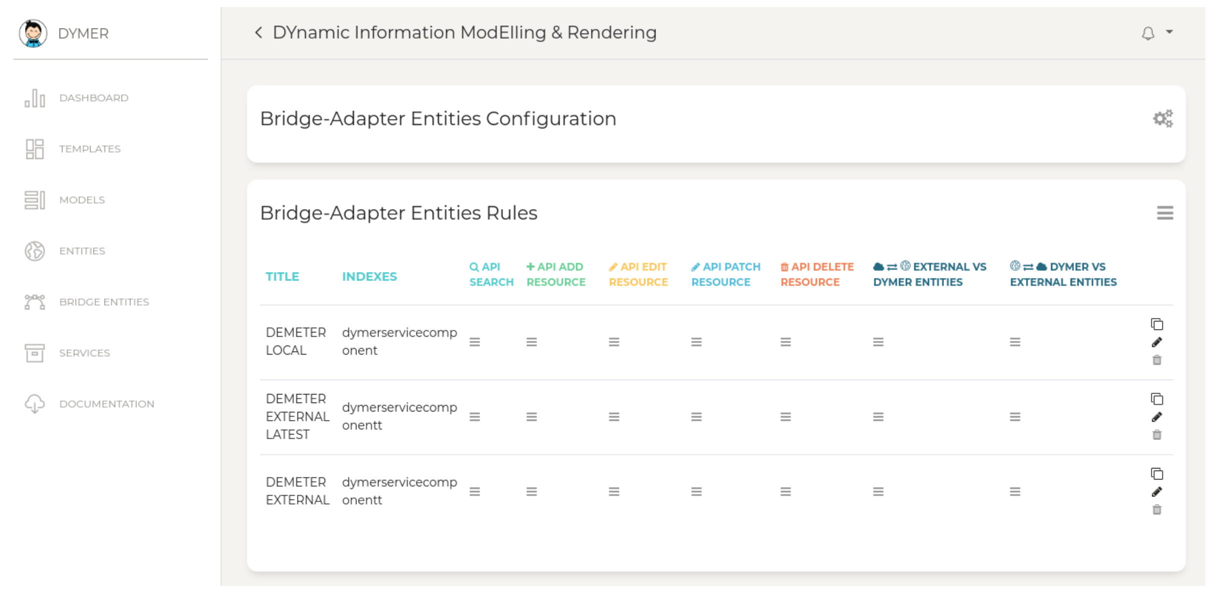
For the purpose of the DEMETER project, a new submodule Bridge Entities is developed inside the Entities microservice. The Bridge Entities allows entity mapping between existing DYMER templates and models with any external service. The essential part of Bridge Entities is Bridge Entities configuration, which is a JSON configuration file. Bridge Entities configuration is made of different parts that are related to external service API. In the beginning, the first thing that has to be defined is a name of Bridge configuration, and indexes that correspond to the existing DYMER model. After that, there is an option to choose whether an external service has token providers. If that is the case information related to it should be inserted. In the case of DEMETER, ACS is used. The next step is to configure things related to search capabilities defined inside of DYMER such as the method, host, port and path, and headers of the external service endpoint. After that, Mapping DYMER Query vs External Query Params should be configured. For this, purpose the JsonMapper [14]library is used. In

[14] https://github.com/marchah/json-mapper-json

this part, existing search queries inside DYMER should be mapped with external ones. The next configurations are related to CRUD operations. Like in the case of Search capabilities, basic information about endpoints like method, host, port, path, and headers should be defined, alongside with Mapping DYMER Entity vs External Entity configuration, where existing attributes inside the DYMER Entity model should be mapped with external ones used in endpoints.

After performing the configuration related to CRUD operations, Mapping External Entities vs DYMER Entities should also be mapped. In this case, attributes from the existing DYMER model should be mapped with the external model.

The last step is in-depth configuration of Mapping DYMER Entities vs External Entities. In this case, the index, type and id of the DYMER entity models must be mapped with the external entity model, along with the _source field, which corresponds to the DYMER model's attribute mapping, and finally, the _properties value. These five are the mandatory values that must be mapped to the external entity model. Figure 37 shows the Bridge configuration and Figure 38 and Figure 39 shows configuration details.

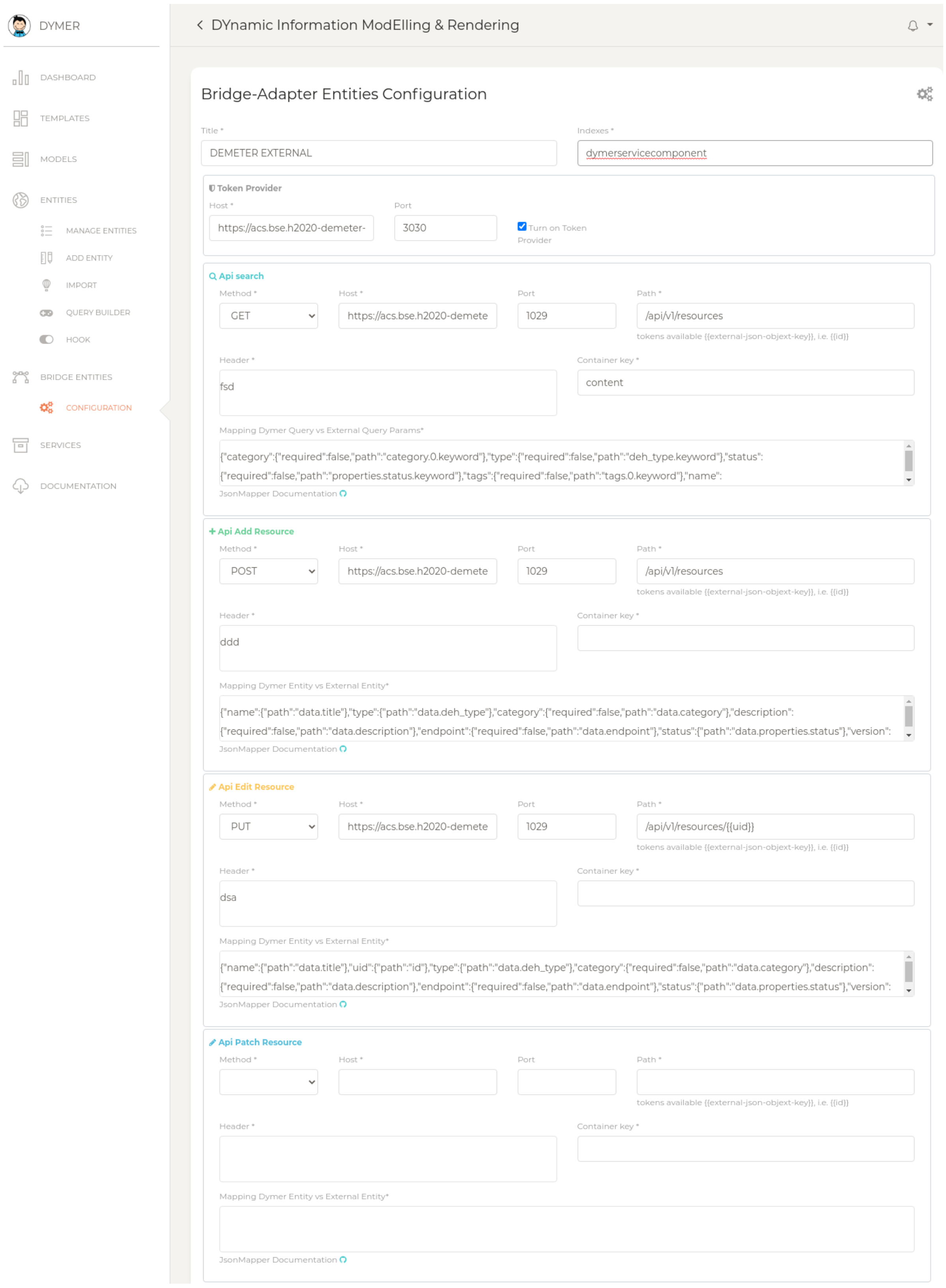


Bridge-Adapter Entities Configuration

Bridge-Adapter Entities Rules

TITLE	INDEXES	API SEARCH	API ADD RESOURCE	API EDIT RESOURCE	API PATCH RESOURCE	API DELETE RESOURCE	EXTERNAL VS DYMER ENTITIES	DYMER VS EXTERNAL ENTITIES
DEMETER LOCAL	dymerservicecomponent	≡	≡	≡	≡	≡	≡	≡
DEMETER EXTERNAL LATEST	dymerservicecomponentt	≡	≡	≡	≡	≡	≡	≡
DEMETER EXTERNAL	dymerservicecomponentt	≡	≡	≡	≡	≡	≡	≡

Figure 37: DYMER administration - Bridge entities configuration list



Bridge-Adapter Entities Configuration

Title *: DEMETER EXTERNAL
Indexes *: dymerservicecomponent

Token Provider
Host *: https://acs.bse.h2020-demeter-
Port: 3030
[✓] Turn on Token Provider

Api search
Method *: GET
Host *: https://acs.bse.h2020-demete
Port: 1029
Path *: /api/v1/resources
tokens available {{external-json-objext-key}}, i.e. {{id}}
Header *: fsd
Container key *: content
Mapping Dymer Query vs External Query Params*:

```
{"category":{"required":false,"path":"category.0.keyword"},"type":{"required":false,"path":"deh_type.keyword"},"status":
{"required":false,"path":"properties.status.keyword"},"tags":{"required":false,"path":"tags.0.keyword"},"name":
```

JsonMapper Documentation

Api Add Resource
Method *: POST
Host *: https://acs.bse.h2020-demete
Port: 1029
Path *: /api/v1/resources
tokens available {{external-json-objext-key}}, i.e. {{id}}
Header *: ddd
Container key *:
Mapping Dymer Entity vs External Entity*:

```
{"name":{"path":"data.title"},"type":{"path":"data.deh_type"},"category":{"required":false,"path":"data.category"},"description":
{"required":false,"path":"data.description"},"endpoint":{"required":false,"path":"data.endpoint"},"status":{"path":"data.properties.status"},"version":
```

JsonMapper Documentation

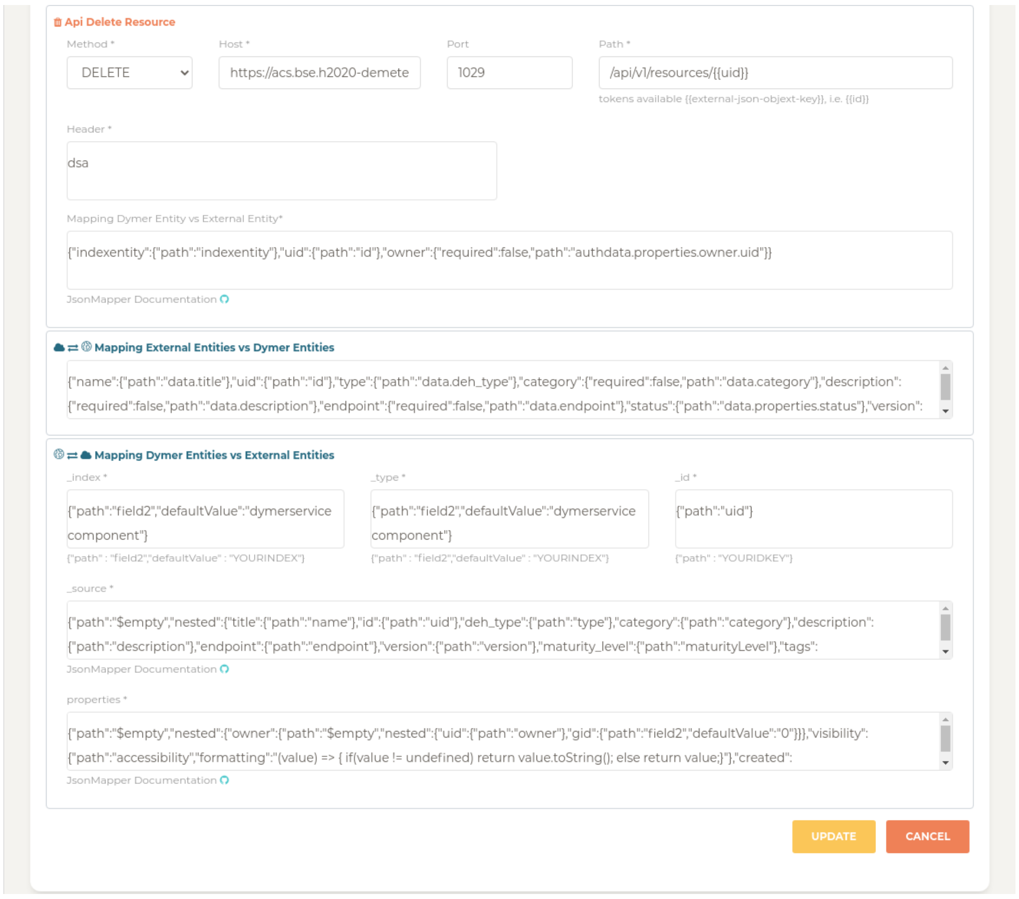
Api Edit Resource
Method *: PUT
Host *: https://acs.bse.h2020-demete
Port: 1029
Path *: /api/v1/resources/{{uid}}
tokens available {{external-json-objext-key}}, i.e. {{id}}
Header *: dsa
Container key *:
Mapping Dymer Entity vs External Entity*:

```
{"name":{"path":"data.title"},"uid":{"path":"id"},"type":{"path":"data.deh_type"},"category":{"required":false,"path":"data.category"},"description":
{"required":false,"path":"data.description"},"endpoint":{"required":false,"path":"data.endpoint"},"status":{"path":"data.properties.status"},"version":
```

JsonMapper Documentation

Api Patch Resource
Method *:
Host *:
Port:
Path *:
tokens available {{external-json-objext-key}}, i.e. {{id}}
Header *:
Container key *:
Mapping Dymer Entity vs External Entity*:
JsonMapper Documentation

Figure 38: DYMER administration - Bridge entities configuration first part


Api Delete Resource

Method *: DELETE

Host *: https://acs.bse.h2020-demete

Port: 1029

Path *: /api/v1/resources/{{uid}}

tokens available {{external-json-objext-key}}, i.e. {{id}}

Header *: dsa

Mapping Dymer Entity vs External Entity*

```
{"indexentity":{"path":"indexentity"},"uid":{"path":"id"},"owner":{"required":false,"path":"authdata.properties.owner.uid"}}
```

JsonMapper Documentation

Mapping External Entities vs Dymer Entities

```
{"name":{"path":"data.title"},"uid":{"path":"id"},"type":{"path":"data.deh_type"},"category":{"required":false,"path":"data.category"},"description":
{"required":false,"path":"data.description"},"endpoint":{"required":false,"path":"data.endpoint"},"status":{"path":"data.properties.status"},"version":
```

Mapping Dymer Entities vs External Entities

_index *: {"path":"field2","defaultValue":"dymerservice component"}

{"path" : "field2","defaultValue" : "YOURINDEX"}

_type *: {"path":"field2","defaultValue":"dymerservice component"}

{"path" : "field2","defaultValue" : "YOURINDEX"}

_id *: {"path":"uid"}

{"path" : "YOURIDKEY"}

_source *

```
{"path":"$empty","nested":{"title":{"path":"name"},"id":{"path":"uid"},"deh_type":{"path":"type"},"category":{"path":"category"},"description":
{"path":"description"},"endpoint":{"path":"endpoint"},"version":{"path":"version"},"maturity_level":{"path":"maturityLevel"},"tags":
```

JsonMapper Documentation

properties *

```
{"path":"$empty","nested":{"owner":{"path":"$empty","nested":{"uid":{"path":"owner"},"gid":{"path":"field2","defaultValue":"0"}}},"visibility":
{"path":"accessibility","formatting":"(value) => { if(value != undefined) return value.toString(); else return value;}"},"created":
```

JsonMapper Documentation

Figure 39: DYMER administration - Bridge entities configuration second part

9.2.2.2 *DEH Resource Registry Management*

DEH Resource Registry management component represents essentially a master service included in the DEH. It is the only component inside the DEH that has direct access to the DEH Resource Registry and thus the component will act like a Data Access Object (DAO) Layer. The functionality that the DEH Resource Registry Management module is providing is being able to manage and monitor resources.

Resource management includes features that provide storage of new resources, as well as updating the properties of existing ones and deleting them when they are no longer available. Besides the mandatory information stored about any resource offered by the DEH, the DEH Audit component records any changes related to the registry. It stores the download history of DEH resources, but it also tracks version changes and their time. Part of the process for registering (or updating) a resource also includes passing the relevant information to the DEH Compatibility Checker module to verify compatibility with the DEH data model, only after this check has been completed successfully can any resource be registered.

In addition, DEH contains a Resource Discovery module, which searches for specific resources registered in the DEH Resource Registry DB.

Finally, the DEH Resource Registry Management collects data from DEH Client Enabler, and through DEH Metric module it also provides the information and enables the usage of the stored resources to the end users.

As part of the previous feature, resource usage by end users, it offers tools for monitoring these resources and this includes features that will record the history of specific resource consumption and their workload.

Figure 40 shows a diagram that depicts the internal functional modules of the DEH Resource Registry Management and communications with other DEH modules.

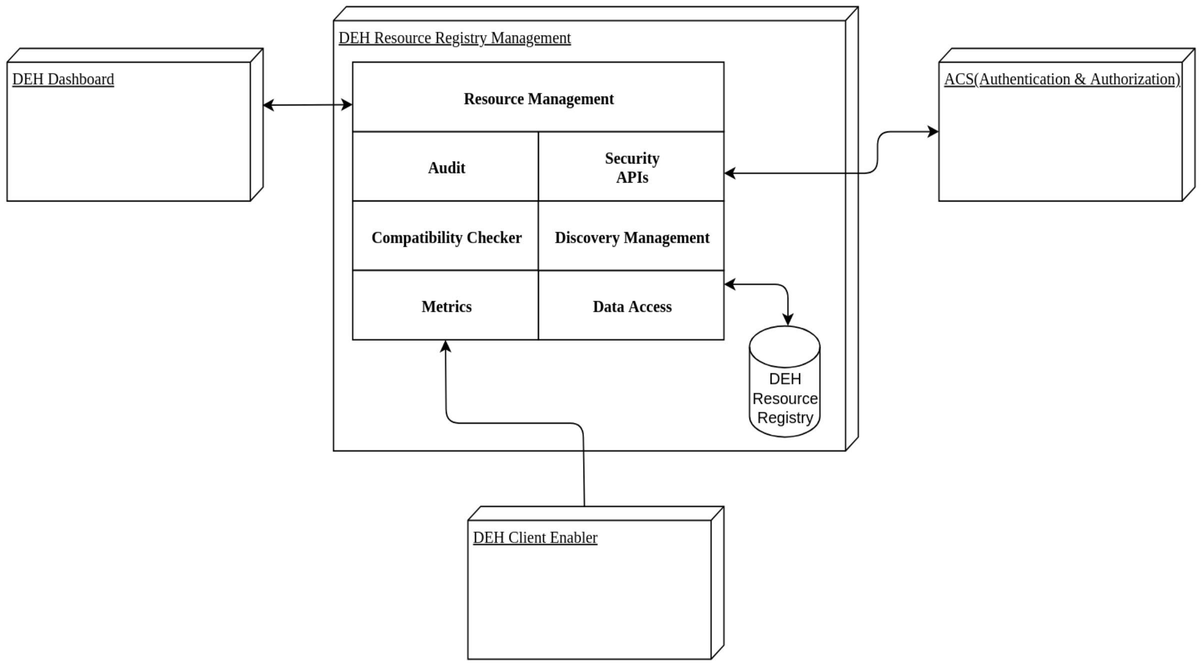


Figure 40: Resource Registry Management building block diagram

The Resource Registry management is composed of seven internal modules:

- Audit - responsible for collecting data related to resource changing, including their download history, version changing history and rating.
- Data Access - responsible for accessing the actual data for a resource as stored in the Resource Registry DB.
- Resource Management - responsible in general for the resource management process and for interfacing with other modules such as the compatibility checker and the discovery management.
- Security API's – responsible for providing authentication and authorisation.
- Metrics – responsible for storing consumption data gathered from DEH Clients.
- DEH Discovery Management - responsible for advanced querying DEH Resources.
- DEH Compatibility Checker - responsible for validating input data in DEH Resource registration process.

The **Audit module** provides support for collecting data about resources such as the date and time of their creation or any edit or update to the resource, rating, version, and download history The Audit module communicates all this data to the Resource Management, in order to pass all of that information along to other components.

The Data Access module provides support for direct access to the Resource Repository DB in order to manipulate the data stored about a registered resource allowing to store, read, edit and delete its relevant data.

The Resource Management module provides support related to preparing a resource for storing. After the resource is verified by DEH Compatibility Checker, and prepared, data is passed to the Data Access modules in order to store it inside the Resource Registry DB. The research capabilities of this module may be integrated and improved from time to time during the development phase of the DEH, in order to support new application needs as much as possible or to cover new integration requirements with the other DEMETER software components.

For purposes of the **DEH Resource Registry**, a NoSQL document-oriented database is used. NoSQL systems are generally more efficient than relational databases because of their ease of management. Since the information to be collected, stored, consulted essentially refer to DEH resources metadata, the challenge will be to use a NoSQL technology for this purpose.

The DEH Resource Registry represents the solution to the metadata store based on NoSQL data archives. DEH Resource Registry works as part of store data and take advantage of the underlying NoSQL functionality to provide its services. This database will face a number of important challenges:

- First, gathering metadata without imposing too much overhead for CRUD operations arriving from client application,
- Second, it must allow a user to specify (via API) what metadata are to be collected and how to use them.
- Finally, it must manage the size of the stored data, the number of incoming CRUD operations and obviously the size of the metadata itself.

Probably if the size of the metadata and the operations managed by the NoSQL database (in the continuous collection of metadata), should cause a great overload of the system, configuration-level facilities could be introduced which would allow a more moderate acquisition of metadata by sizing the read and write operations performed by client applications on the NoSQL store.

The **DEH Resource Registry** is a database where information about DEH Resources is stored. Each DEH Resource is identified by *uid (unique identifier).* Alongside with collection that stores information about DEH Resources (deh_resource), we have two more collections: a collection for storing DEH Resources audit data (resources_audit) and a collection for storing DEH Resources consumption gathered from DEH Client Enabler (resources_metrics).

The Access Control Server component provides the solution for controlling the access to the stored resources. **The Security API's module** provides support for connection to the component in order to get respective access policies. These API's implement OAuth2 for the implementation of the Identity Management Keyrock (authentication) and a REST API for the implementation of DCapBAC (authorisation).

The DCapBAC REST API end point function returns the authorization or Capability Token. A definition of this is shown next, a definition that can be found in more detail in this document Section 8.5:

- Name: generateCapabilityToken (authtoken,subject,resource,action)
- Expected output: Capability Token (a signed JSON document)

- Error messages: When a CT cannot be obtained, the "Can't generate capability token" message is received, (see 8.5.1.2).
- Data model used: each of the parameters received by this function are strings

The description of these parameters is:

Name	generateCapabilityToken()	
Property	**Type**	**Description**
authtoken	String	The authentication token obtained from the Identity Management
subject	String	The subject of the authorisation query
resource	String	The resource intended to be accessed
action	String	The operation mode: GET, POST, PUT, PATCH or DELETE

The next sample call:

generateCapabilityToken(“04c5b070-4292-4b3f-911b-“,user@dem.com,”ngsi-ld/v1/entities”,”GET”)

is included in the access to the resource in the corresponding repository. Before performing the authorisation process, the authentication process must already have been done as it generates an Authentication Token that must be included in the authorization calls to be validated.

The authorisation process, in a more detailed view, is based on a technology called Distributed Capability-Based Access Control (DCapBAC), which decouples the traditional XACML framework in two phases, one for receiving the authorisation (represented by the receipt of the Capability Token) and a second one for accessing the information repository, where a Policy Enforcement Point (PEP) Proxy first validates the received Capability Token and, in case of a positive answer, it continues acting just as a mere intermediary with the information repository. Additionally, it interacts with other resource repositories placed in both DEH and BSE so that the access can always be controlled.

The authorisation enabler depends on the resources repositories being addressed, since they must incorporate the Capability Token for the corresponding queries so that the PEP Proxy can validate them.

The Metrics module provides support for storing and managing data coming from DEH Clients. DEH Clients will send the data to the Metrics module, after which they will be processed and stored inside Resource Registry DB. This data will be accessible by DEH users through DEH Dashboard in order to track the consumption history of their DEH Resources. Metrics module provides support for showing metrics on different levels, from all users DEH Resources, specific DEH Resource, or on container level.

The DEH Discovery Management module works together with the Resource Registry Management module presented in the previous subsection. In fact, for the most part, it piggybacks upon the information provided by that module in order to accomplish its task. Its task is to get requests for specific types of DEMETER enabled entities through the DEH interface (from users/stakeholders) and, afterwards, to query the Resource Management component (of the Resource Registry Management) in order to discover the resources matching the user request that the user is allowed to access. Access control is being enforced by the data access features of the Resource Registry making certain that when the list of available resources is returned as result of a user query, this list this does not contain any resources whose access policies would prohibit usage and discovery by the specific user.

Therefore, the main feature of this component is to translate the data between the appropriate DEH Interface used by stakeholders in order to discover resources by placing the appropriate filters and the registry management component, and then to place the appropriate query through the relevant resource management interfaces. Once this information is returned to the module, then it is processed and encoded appropriately to be shown through the DEH interface to the end user. It will be possible to change the ordering of the resources returned based on criteria such as price, quality, availability, platform or device used (if appropriate) just to mention a few; it is possible to combine these criteria together as well. It will also be possible to keep track of the other queries made by the end user in order to provide resources which are compatible with the previous queries and resources sought previously by the same user in the same session.

Finally, in a second development stage for this component, we will aim to store and reprocess the past queries submitted by users. This would allow us to inform users regarding new resources available which match their past queries for example, or to find alternate similar resources/enablers in case the user wants to replace an enabler current used in her app with a new one. To accommodate such tasks, we will need to periodically rerun queries submitted by end users. This means that such information needs to be stored. If the resource registry cannot accommodate the storage of such information, then most likely this will require use of a simple database separate from the Resource Registry DB, inside the DEH Discovery Management module, in order to store this data. Any relevant resources which are then discovered will be logged, so that when the user reuses the DEH, this information will be available to them.

The DEH Compatibility Checker module capabilities are more valued in the case of using Resource Registry Management (RRM) through API's. Using Resource Registry Management (RRM) through UI, the validation of data is handled in DEH Dashboard. This module checks whether information like the status, maturity level, accessibility, and localisation is correct in the process of registering a new DEH Resource. If the information is correct, a new DEH Resource will be created, if not the user will get information about his mistake, and advice for possible values.

9.3 Process View

The process view covers the internal dynamic aspect of a DEH with a focus on the runtime behaviour and describes processes inside it and interaction between them. For the purpose of representing the process view, UML Activity and Sequence diagrams will be used.

Activity diagrams will depict the activities flow and how they are coordinated inside DEH Components, while Sequence Diagrams will focus on requests and messages that the same components are sharing among them in order to execute the process.

9.3.1 DEH Authentication & Authorisation

The diagram in Figure 41 depicts the sequence in the interaction between DYMER and components inside the Security Block from the moment the user enters his authentication credentials up to the moment where those credentials are validated alongside with access policies.

In the rest of this section, each diagram will imply that this part is passed, so each request from DYMER will contain an X-AUTH-TOKEN.

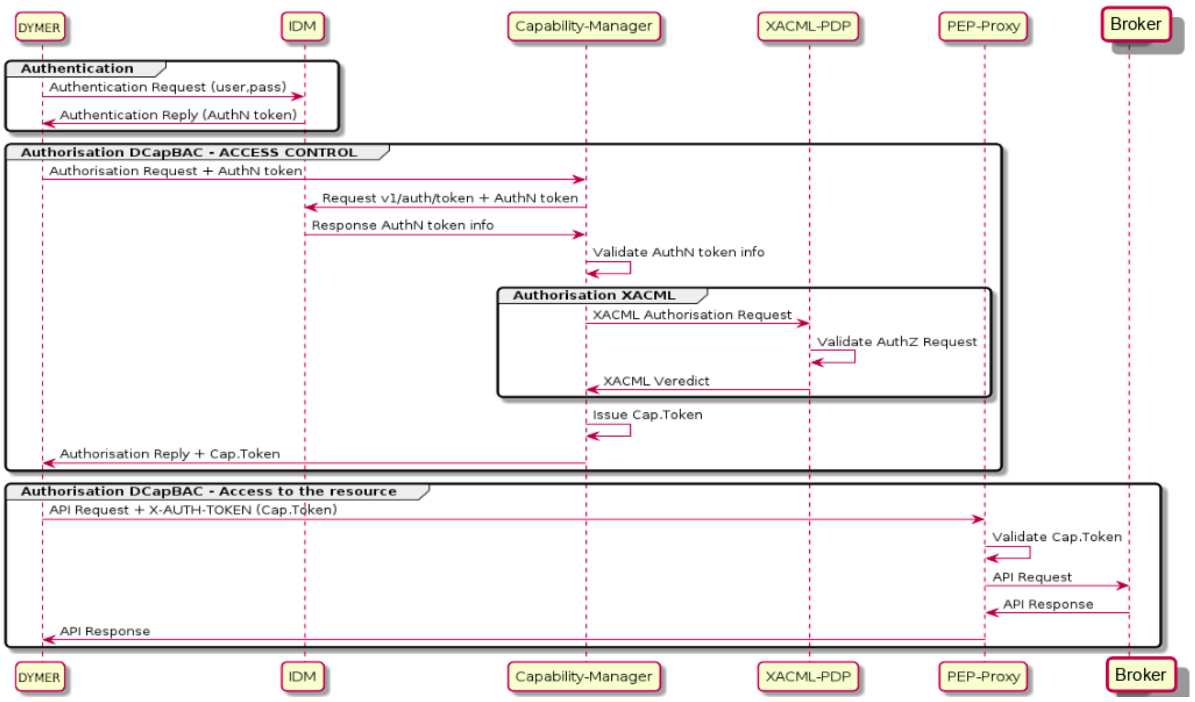

DYMER
IDM
Capability-Manager
XACML-PDP
PEP-Proxy
Broker
Authentication
Authentication Request (user,pass)
Authentication Reply (AuthN token)
Authorisation DCapBAC - ACCESS CONTROL
Authorisation Request + AuthN token
Request v1/auth/token + AuthN token
Response AuthN token info
Validate AuthN token info
Authorisation XACML
XACML Authorisation Request
Validate AuthZ Request
XACML Veredict
Issue Cap.Token
Authorisation Reply + Cap.Token
Authorisation DCapBAC - Access to the resource
API Request + X-AUTH-TOKEN (Cap.Token)
Validate Cap.Token
API Request
API Response
API Response

Figure 41: DEMETER Enabler Hub Authentication and Authorisation sequence diagram

9.3.2 DEH Resource Registry Management

This subsection covers the interaction between internal DEH components in a process of Registering and Discovery Resources in DEH.

9.3.2.1 Activity Diagrams

The diagram in Figure 42 depicts the interaction between three DEH components that are involved in a process of registering a new resource:

- DYMER - DEH Dashboard component where developers are creating a new resource.
- Access Control Server - component which checks the access policies of a user.
- Resource Registry Management - component which manages all the processes related to the DEMETER resources.

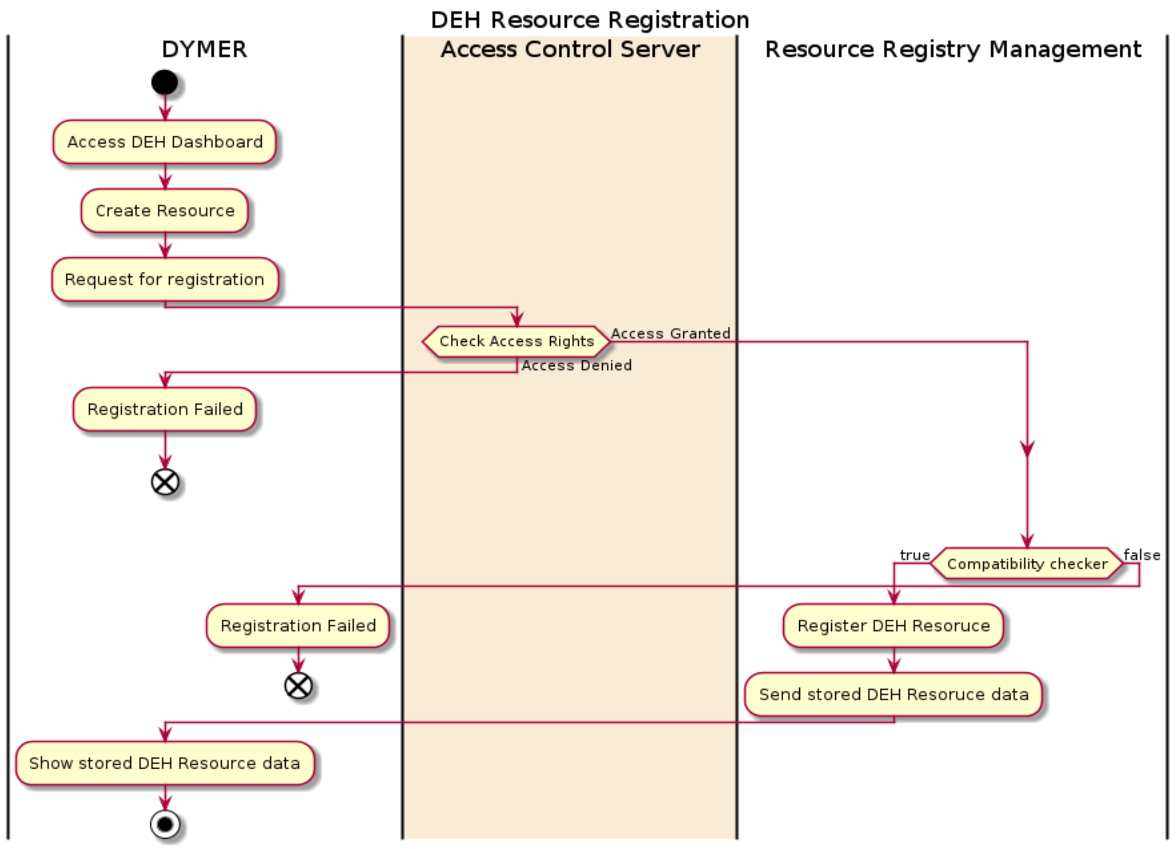


Figure 42: DEH Resource Registry activity diagram

To create a new resource in DEH, a DEH user must first authenticate and access the DEH Dashboard or use Resource Registry Management (RRM) API's: subsequently a request for registering a DEH Resource can be sent. Before the request arrives at the Resource Registry Management, Access Control Server validates the user's access policies. If the user has the rights to create a new resource, the request will be forwarded to it, if not, the user will get an error. The Resource Registry management component first validates input date and based on validation, the registry will be saved as DEH Resource or not. If the DEH Resource is stored, the Resource Registry Management component will return stored information about new the DEH Resource.

The diagram in Figure 43 depicts the interaction between three DEH components that are involved in a process of discovering resources:

- DYMER - DEH Dashboard component where user is searching for a resource.
- Access Control Server - component which checks the access policies of a user.
- Resource Registry Management - component which manages all the processes related to the DEMETER resources.

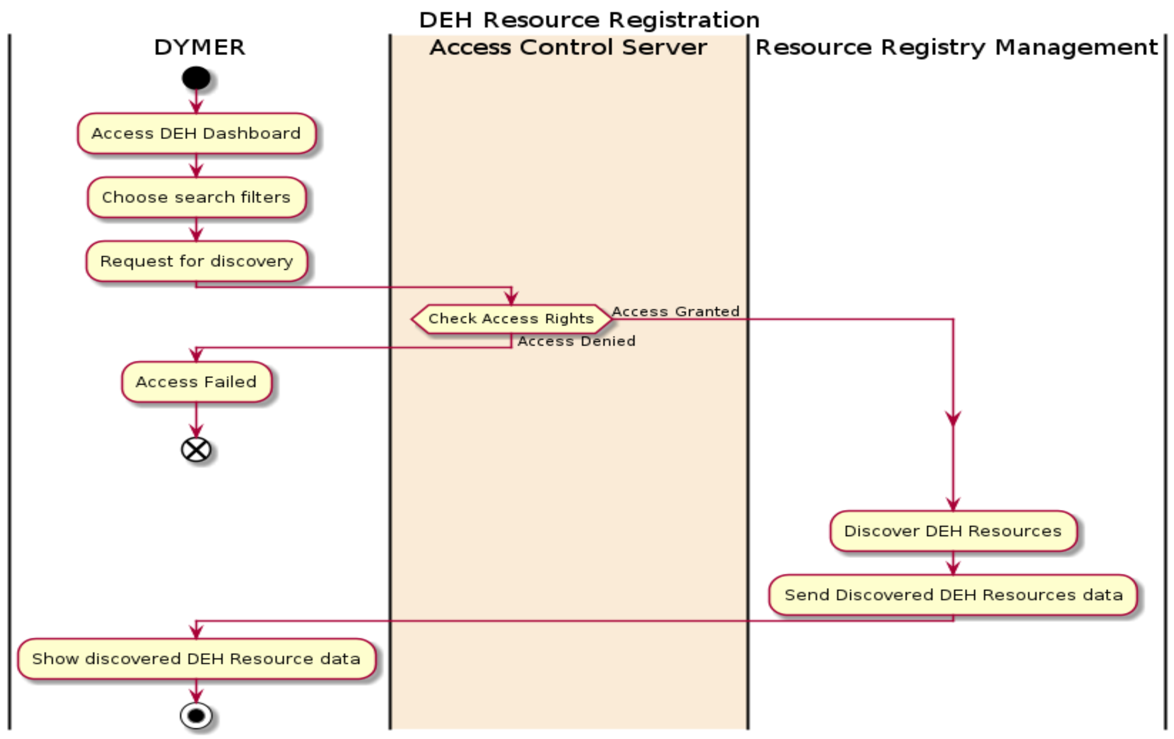


Figure 43: DEH Resource Discovery activity diagram

To discover specific DEH Resources, a DEH user must first authenticate and access the DEH Dashboard or use Resource Registry Management (RRM) API’s: subsequently a request to discover DEH Resources can be sent. After that user chooses specific search filters and sends a discovery request. Before the request arrives at the Resource Registry Management, Access Control Server validates the user’s access policies. If the user has the correct rights, the request will be forwarded to it, if not, the user will get an error. If a request is forwarded, the Resource Registry Management inside Discovery Module filters all DEH Resources by requested search criteria, and sends the data to the DEH user, after which those data is displayed.

9.3.2.2 Sequence Diagrams

The diagram in Figure 44 depicts the sequence interaction between main components which are involved in a process of creating a new resource, as depicted in Figure 42 DEH Resource Registry Activity diagram:

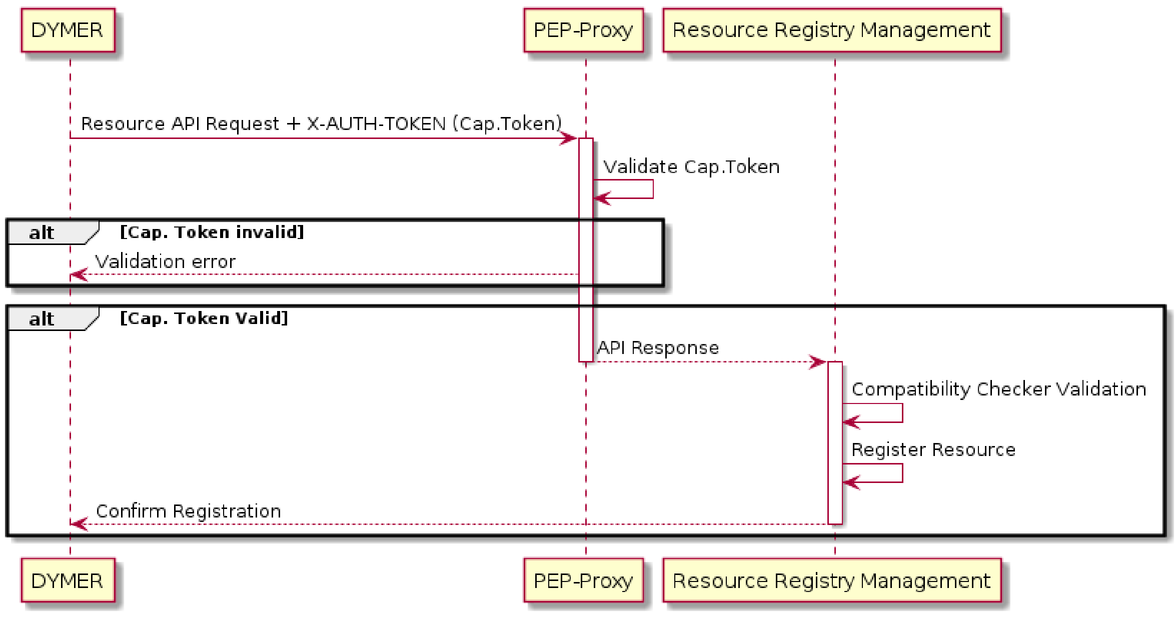


Figure 44: DEH Resource Registry sequence diagram

DYMER sends a request with X-AUTH-TOKEN in the header parameter and the content related to the resource in the body to the PEP-Proxy, which is a module inside the Access Control Server (ACS), to validate a request. Based on the Capability Token in the header, the PEP-Proxy is able to determine if a user can register a new Resource or not. In the case of invalid Capability Token, the PEP-Proxy returns an error to DYMER. If Capability token is valid, the PEP-Proxy forwards a request to the Resource Registry Management component which validates the compatibility of the new resource using DEH Compatibility Checker module. If the resource is compatible, the Resource Registry Management component will store the resource as a new DEH Resource in the DEMETER Resource Registry and send a response to DYMER with confirmation about storing and data related to the new DEH Resource.

Figure 45 shows the sequence diagram that depicts interaction between the main components which are involved in a process of discovering a resource depicted in Figure 43 DEH Resource Discovery Activity diagram:

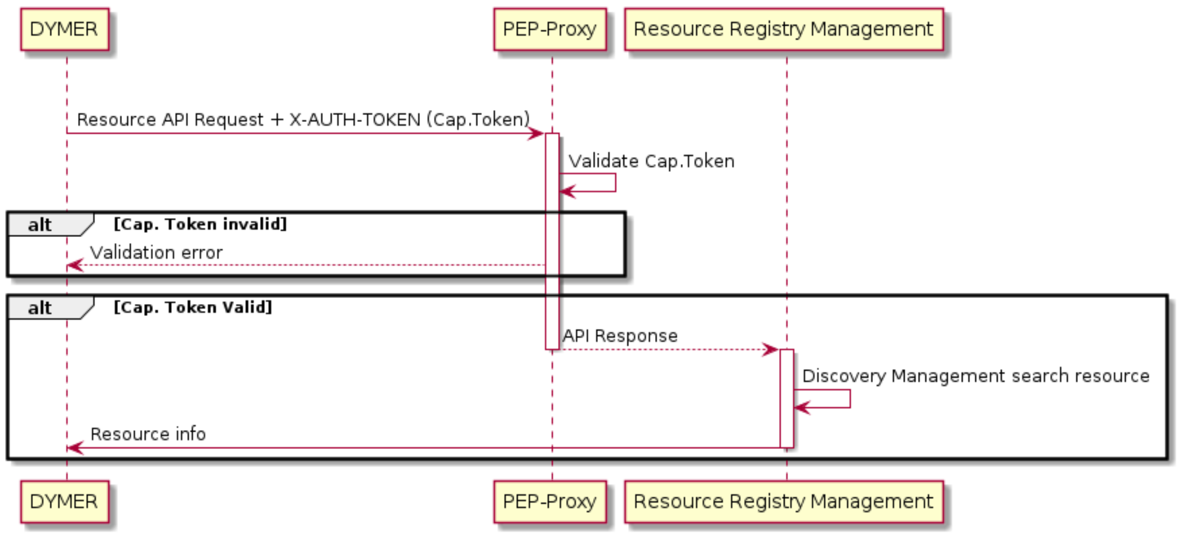

DYMER
PEP-Proxy
Resource Registry Management
Resource API Request + X-AUTH-TOKEN (Cap.Token)
Validate Cap.Token
alt
[Cap. Token invalid]
Validation error
alt
[Cap. Token Valid]
API Response
Discovery Management search resource
Resource info
DYMER
PEP-Proxy
Resource Registry Management

Figure 45: DEH Resource Discovery sequence diagram

The DYMER sends a request with the Capability Token in the X-AUTH-TOKEN header parameter, and queries for discovery as URL parameters to the PEP-Proxy which is a module inside Access Control Server (ACS) to validate a request. Based on the Capability Token in the header, the PEP-Proxy is able to determine if a user can discover resources or not. In the case of invalid Capability Token, PEP-Proxy returns an error to DYMER; if Capability token is valid, PEP-Proxy forwards request to the Resource Registry Management component. DEH Discovery Management module discovers DEH Resources based on passed queries and returns them to the DYMER application.

9.4 Interfaces

The data model presented in the following tables contains a list of properties that identify a resource within the DEH; however, any additions to the models presented below are not excluded in terms of new properties or new types of resources that could emerge during the developments or integrate new requirements in terms of requesting data that DEH will have to support. Consequently, these definitions may change; any updates on the DEH data model will be reported in D3.5.

9.4.1 DEH Resource Data Model

Table 16: DEH Resource Data Model

Name	*DEH Resource:* ***Component, Device, Service, Dataset, Platform***	
Property	**Type**	**Description**
UID	Alphanumeric	Resource unique identifier
Name	String	Resource name
Type	String	Resource type
Category	Arrays	Resource category
Description	String	Resource description
Endpoint	String	Resource endpoint
Status	Integer	Resource status
Version	String	Resource version
Maturity Level	Integer	Resource maturity level
Owner	String	Resource owner
Author	Arrays	Resource author details (username, email)
Tags	Arrays	Resource tag
Attachments	MultipartFile	Resource attachment
Images	MultipartFile	Resource images
Rating	Double	Resource rating
Localisation	Arrays	Resource localisation (geo- points)
Accessibility	Integer	Resource accessibility (0 = Public, 1=Private)
Created	Timestamp	Resource creation date and time
Last Update	Timestamp	Resource last update date and time
Dependencies	Arrays	Resource dependencies (with other resources)
Access controls policies	Arrays	Resource ACL
URL	String	URL for downloading/streaming data or entity
Downloads history	Map	Resource downloads history

Table 17: DEH Attachment and image Data model

Name	*Attachment/Image*	
Property	**Type**	**Description**
ID	Alphanumeric	Attachment unique identifier
Original name	String	Attachment name on local storage
File name	String	Attachment name in DB
Chunk size	Integer	Attachment chunk size in DB
Length	String	Attachment length
Upload date	Timestamp	Attachment upload date and time
Content Type	String	Attachment content type

Table 18: DEH Metrics Data Model

Name	*Metrics*	
Property	**Type**	**Description**
ID	Alphanumeric	Metrics unique identifier
rrmId	Alphanumeric	DEH Resource ID
name	Alphanumeric	DEH Resource name
owner	Alphanumeric	DEH Resource owner
numberOfInstances	Integer	Number of deployed DEH Resource instances
Containers	Arrays	Metrics data of deployed DEH Resource instances

Table 19: DEH Metrics Data Data Model

Name	*Metrics Data*	
Property	**Type**	**Description**
Container ID	Alphanumeric	Container ID of running resource
Uptime	Long	Container uptime
Hostname	Alphanumeric	Hostname
IP	Alphanumeric	IP address
Image	Alphanumeric	Docker image name of resource
BSE ID	Alphanumeric	Resource BSE ID
RRM_ID	Alphanumeric	Resource DEH ID
Last updated	Timestamp	Metrics last update date and time
CPU consumption	Arrays	Resource CPU consumption in container (timestamp, percentage)
Memory consumption	Arrays	Resource Memory consumption in container (timestamp, percentage)

9.4.2 Description of API's

The DEH API described in the templates below represent a first version (v1) of the DEH API software stack that will be produced during the project. These services or preliminary set of API's could be integrated with other services that will become necessary due to changed conditions or application needs or simply to update or improve the existing ones. Consequently, these refinements and new API definitions may change; any updates on the DEH API will be reported in D3.5.

9.4.2.1 DEH Dashboard (DYMER Core API)

Title	*Get entities*
URL:	
/api/entities/api/v1/entity/_search	
Headers	
enctype: "multipart/form-data	
Method	
POST	
URL Params	
Required:	
N/A	N/A
Optional:	
N/A	N/A
Data Params	
Required:	

`{`
`        "bool": {`
`                "must": [`
`                        {`
`                        "term": {`
`                        "category": "open”`
`                                }`
`                        }`
`                ]`
`        }`
`}`
`}`
	query: JSON object with the desired query

Optional:	
N/A	*N/A*

Success response	

200
Content: {
"success": true,
"message": "",
"data": [ ],
"extraData": {}
}
	success: true if there no errors
	message:message form the server e.g., Resource List or Empty List
	data: JSON array of objects
	extraData: optional JSON object which contains extra information e.g., Logs, data etc.

Error response	

404
Content: {
"success": false,
"message": "",
"data": [ ],
"extraData": {}
}
	success: false if there is an error
	message:message form the server e.g., Resource List or Empty List
	data: empty JSON array
	extraData: optional JSON object which contains extra information e.g., Logs, data etc.

Sample call	
N/A	
Notes	
N/A	

Title	*Get File by ID*
URL:	
api/entities/api/v1/entity/content/:fileid	
Headers	
mimeType: "text/html"	
contentType: "text/html"	
Method	
GET	
URL Params	
Required:	
fileid = [integer]	*ID of the file contained in the json object*
Optional:	
N/A	N/A
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	
200 Content: { file }	*Response contains file*
Error response	
404	*Resource not found*
Sample call	
N/A	
Notes	
N/A	

Title	*Create Entity*
URL:	
/api/entities/api/v1/entity/	
Headers	
enctype: "multipart/form-data;"	
Method	
POST	
URL Params	
Required:	
N/A	N/A
Optional:	
N/A	N/A
Data Params	
Required:	

`{`
`    "instance":{`
`    "index":"demeterproduct",`
`    "type":" demeterproduct "`
`    },`
`    "data":{`
`        ….`
`        "properties":{}`
`        }`
`}`
	instance: json object defining the type and index of entity
	
	---:key:value/jsonobject/jsonarray that define details of the entity
	
	properties: json object defining the properties of the entity

Optional:	
N/A	*N/A*
Success response	

200
`Content: {`
`       "success": true,`
`       "message": "",`
`       "data": [    ],`
`       "extraData": {}`
`}`
	success: true if there no errors
	message:message form the server e.g., "Entity successful created"
	data: json array of objects
	extraData: optional json object which contains extra information e.g., Logs, data etc.

Error response	

404
`Content: {`
`       "success": false,`
`       "message": "",`
`       "data": [    ],`
`       "extraData": {}`
`}`
	success: false if there is an error
	message:message form the server e.g., "Entity could not be created"
	data: empty json array
	extraData: optional json object which contains extra information e.g., Logs, data etc.

Sample call	
N/A	
Notes	
N/A	

Title	*Edit Entity*
URL:	
/api/entities/api/v1/entity/:id	
Headers	
enctype: "multipart/form-data;"	
Method	
PUT	
URL Params	
Required:	
id=[integer]	*ID of the entity to update*
Optional:	
N/A	N/A
Data Params	
Required:	

`{`
`    "instance":{`
`    "index":"demeterproduct",`
`    "type":" demeterproduct "`
`    },`
`    "data":{`
`        ….`
`        "properties":{}`
`        }`
`}`
	instance: json object defining the type and index of entity
	
	---:key:value/jsonobject/jsonarray that define details of the entity
	
	properties: json object defining the properties of the entity

Optional:	
N/A	*N/A*
Success response	

200
`Content: {`
`       "success": true,`
`       "message": "",`
`       "data": [    ],`
`       "extraData": {}`
`}`
	success: true if there no errors
	message:message form the server e.g., "Entity successful updated"
	data: json array of objects
	extraData: optional json object which contains extra information e.g., Logs, data etc.

Error response	

404
`Content: {`
`       "success": false,`
`       "message": "",`
`       "data": [    ],`
`       "extraData": {}`
`}`
	success: false if there is an error
	message:message form the server e.g., "Entity could not be updated"
	data: empty json array
	extraData: optional json object which contains extra information e.g., Logs, data etc.

Sample call	
N/A	
Notes	
N/A	

Title	*Delete Entity*
URL:	
/api/entities/api/v1/entity/:id	
Headers	
enctype: "multipart/form-data;"	
Method	
PUT	
URL Params	
Required:	
id=[integer]	*ID of the entity to update*
Optional:	
N/A	N/A
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	

200
`Content: {`
`"success": true,`
`"message": "",`
`"data": [ ],`
`"extraData": {}`
`}`
	success: true if there no errors
	message:message form the server e.g., "Entity successful deleted"
	data: json array of objects
	extraData: optional json object which contains extra information e.g., Logs, data etc.

Error response	

404
`Content: {`
`"success": false,`
`"message": "",`
`"data": [ ],`
`"extraData": {}`
`}`
	success: false if there is an error
	message: message from the server e.g., "Entity could not be deleted"
	data: empty json array
	extraData: optional json object which contains extra information e.g., Logs, data etc.

Sample call	
N/A	
Notes	
N/A	

9.4.2.2 Resource Registry Management

Title	*Create new DEH resource*
URL:	
api/v1/resources	
Headers	
Required:	
x-subject-token	Authentication token obtained from IDM
x-auth-token	Capability token obtained from Capability Manager
Method	
POST	
URL Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Data Params	
Required:	
name=[String]	*resource name*
type=[String]	*resource type*
status=[Integer]	resource status
Optional:	
category=[Arrays]	*resource categories*
description=[String]	*resource description*
endpoint=[String]	*resource endpoint*
version=[String]	*resource version*
maturityLevel=[Integer]	*resource maturity level*
tags[Arrays]	*Resource tags*
localisation=[Arrays]	*resource localisation geo-points*
url=[String]	*URL for downloading/streaming data or entity*
accessibility=[Integer]	*resource accessibility*
attachmentFIle = [MultipartFile]	*resource Attachments*
Success response	

200
`Content: {`
`"success": true,`
`"message": "DEH Resource successfully created",`
`"data": {`
`"uid": "606ef80c6393cd08312af39a",`
`"name": "Test Resource 1",`
`"type": "Service",`
`"category": [`
`"Business application"`
`],`
`"description": "Lorem ipsum dolor sit amet",`
`"endpoint":`
`"https://jsonplaceholder.typicode.com/comments",`
`"status": 1,`
	success: true if there no errors
	message: message form the server e.g., DEH Resource successfully created
	data: JSON object with DEH Resource data
	extraData: optional JSON object which contains extra information e.g., Logs, data etc.

```
"version": "1",
"maturityLevel": 1,
"owner": "0a1d17fe-a4bd-4aa8-9054-
998a2018ae70",
"author": {
"username": "mstojanovi",
"email": "marko.stojanovic@eng.it"
},
"tags": [
"Docker",
"AI"
],
"attachment": [
{
"id": "606ef80c6393cd08312af398",
"originalName": "Screenshot-
20210324142051-1514x422.png",
"fileName": "Screenshot-20210324142051-
1514x422.png",
"chunkSize": 261120,
"length": 49301,
"uploadDate": "2021-04-
08T12:33:16.476+00:00",
"contentType": "image/png"
}
],
"rating": 0.0,
"localisation": [
{
"x": 1.0,
"y": 1.0,
"coordinates": [
1.0,
1.0
],
"type": "Point"
}
],
"accessibility": 0,
"createAt": "2021-04-08T14:33:16.523",
"lastUpdate": "2021-04-08T14:33:16.523",
"dependencies": [
"docker"
],
"accessControlPolicies": [
"read"
],
"url": "www.google.com",
"billingInformation": [],
"downloadsHistory": {}
},
```

`"extraData": null` `}`	
Error response	
`Content: {` `"success": false,` `"message": "",` `"data": [],` `"extraData": {}` `}`	*success: false if there is an error* *message:message form the server e.g., Resource Not Found* *data: null* *extraData: optional JSON object which contains error specific information like type of error, exact error message, timestamp, called path*

Sample call

```
curl --location --request POST 'https://acs.bse.h2020-demeter-
cloud.eu:1029/api/v1/resources' \
--header 'x-auth-token: $TOKEN_VALUE' \
--header 'x-subject-token: $TOKEN_VALUE'\
--form 'name="Test Resource 1"' \
--form 'type="Service"' \
--form 'category="Business application"' \
--form 'description="Lorem ipsum dolor sit amet"' \
--form 'endpoint="https://jsonplaceholder.typicode.com/comments"' \
--form 'status="1"' \
--form 'version="1"' \
--form 'maturityLevel="1"' \
--form 'tags="Docker, AI"' \
--form 'attachmentFile=@"/home/mare/Downloads/Screenshot-20210324142051-1514x422.png"' \
--form 'localisation="1,1"' \
--form 'accessibility="0"' \
--form 'dependencies="docker"' \
--form 'accessControlPolicies="read"' \
--form 'url="www.google.com"'
```

Notes

N/A

Title	*Get All DEH Resources*
URL:	
api/v1/resources	
Method	
GET	
Headers	
Required:	
x-subject-token	Authentication token obtained from IDM
x-auth-token	Capability token obtained from Capability Manager
URL Params	
Required:	
N/A	*N/A*
Optional:	
pageNumber[Integer]	*value used for server pagination; it represents page number (first page 0)*
pageSize[Integer]	*value used for server pagination, it represents number of DEH Resources on page(e.g., 20)*
sortBy[String]	*value used for server pagination; it represents attribute of DEH Resource on which sorting will be applied (e.g.*
sortingOrder[String]	*value used for server pagination, it represents sorting order (e.g., ASC, DESC)*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	
200 `Content: { "success": true, "message": "DEH Resource found", "data": {[ALL DEH RESOURCE OBJECTS]}, "extraData": null }`	*success: true if there no errors* *message: message form the server e.g., DEH Resource found* *data: JSON Array with DEH Resource objects data* *extraData: optional JSON object which contains extra information e.g., Logs, data etc.*
Error response	
`Content: { "success": false, "message": "", "data": [], "extraData": {} }`	*success: false if there is an error* *message:message form the server e.g., Resource Not Found* *data: null* *extraData: optional JSON object which contains error specific information like type of error, exact error message, timestamp, called path*
Sample call	
curl --location --request GET 'https://acs.bse.h2020-demeter-cloud.eu:1029/api/v1/resources' \ --header 'x-auth-token: $TOKEN_VALUE' \ --header 'x-subject-token: $TOKEN_VALUE'	
Notes	
N/A If values related to server pagination are not inserted, default ones are:	

pageNumber: 0, pageSize: 20, sortBy: name, sortingOrder: ASC.

Title	*Get DEH Resource by UID*
URL:	
api/v1/resources/{uid}	
Method	
GET	
Headers	
Required:	
x-subject-token	Authentication token obtained from IDM
x-auth-token	Capability token obtained from Capability Manager
URL Params	
Required:	
uid= alphanumeric]	*DEH resource UID*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	

200
Content: {
"success": true,
"message": "DEH Resource found",
"data": {
"uid": "604a30b6d0ba181e42ed1bdf",
"name": "Test Resource ",
"type": "Service",
"category": [
"Business Applications"
],
"description": "Lorem ipsum dolor sit amet",
"endpoint":
"https://jsonplaceholder.typicode.com/comments",
"status": 1,
"version": "1.0",
"maturityLevel": 1,
"owner": "0a1d17fe-a4bd-4aa8-9054-998a2018ae70",
"author": {
"username": "mstojanovi",
"email": "marko.stojanovic@eng.it"
},
"tags": [
"Applications"
],
"attachment": [],
	success: true if there no errors
	message: message form the server e.g., DEH Resource found
	data: JSON object with DEH Resource data
	extraData: optional JSON object which contains extra information e.g., Logs, data etc.

```
"rating": 0.0,
"localisation": [
{
"x": 44.8155452518633,
"y": 20.4585588390912,
"coordinates": [
44.8155452518633,
20.4585588390912
],
"type": "Point"
}
],
"accessibility": 1,
"createAt": "2021-03-11T16:01:10.801",
"lastUpdate": "2021-03-11T16:13:00.519",
"dependencies": [
"Resource 2"
],
"accessControlPolicies": [
"read"
],
"url": "www.google.com",
"billingInformation": [],
"downloadsHistory": {
"2021-03-23": 1,
"2021-03-24": 1,
"2021-04-08": 1
}
},
"extraData": null
}
```

Error response

```
Content: {
        "success": false,
        "message": "",
        "data": [],
        "extraData": {}
}
```

success: false if there is an error
message:message form the server e.g., Resource Not Found
data: null
extraData: optional JSON object which contains error specific information like type of error, exact error message, timestamp, called path

Sample call

```
curl --location -g --request GET 'https://acs.bse.h2020-demeter-
cloud.eu:1029/api/v1/resources/{uid}' \
--header 'x-auth-token: $TOKEN_VALUE' \
--header 'x-subject-token: $TOKEN_VALUE'
```

Notes

N/A

Title	*Get DEH Resources matching multiple criteria*
URL:	
/api/v1/resources/search?<query with the data, e.g., type etc.>	
Headers	
Required:	
x-subject-token	Authentication token obtained from IDM
x-auth-token	Capability token obtained from Capability Manager
Method	
GET	
URL Params	
Required:	
N/A	*N/A*
Optional:	
uid	*resource uid*
name	resource name
type	resource type
category	resource category
tags	resource tags
status	resource status
rating	*resource rating*
maturityLevel	*resource maturity level*
localisationDistance	*nearby resources from user location*
createAt	*resource creation timestamp*
lastUpdate	*resource last update timestamp*
pageNumber[Integer]	*value used for server pagination; it represents page number (first page 0)*
pageSize[Integer]	*value used for server pagination, it represents number of DEH Resources on page(e.g., 20)*
sortBy[String]	*value used for server pagination; it represents attribute of DEH Resource on which sorting will be applied (e.g.*
sortingOrder[String]	*value used for server pagination, it represents sorting order (e.g., ASC, DESC)*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	

200
`Content: {`
`"success": true,`
`"message": "DEH Resource found",`
`"data": {[ALL DEH RESOURCE OBJECTS]},`
`"extraData": null`
`}`
	success: true if there no errors
	message: message form the server e.g., DEH Resource found
	data: JSON Array with DEH Resource objects data
	extraData: optional JSON object which contains extra information e.g., Logs, data etc.

Error response	

`Content: {`
`"success": false,`
	success: false if there is an error

`"message": "",`
`"data": [],`
`"extraData": {}`
`}`
	message:message form the server e.g., Resource Not Found
	data: null
	extraData: optional JSON object which contains error specific information like type of error, exact error message, timestamp, called path

Sample call	

`curl --location --request GET 'https://acs.bse.h2020-demeter-cloud.eu:1029/api/v1/resources/search?name=test&type=service' \`
`--header 'x-auth-token: $TOKEN_VALUE' \`
`--header 'x-subject-token: $TOKEN_VALUE' \`	

Notes	
For localisationDistance you need to insert your current coordinates (x,y) and preferred distance from it in metres, e.g., (44.819419, 20.461853, 1000). Range filtering is available for rating, maturityLevel, createAt and lastUpdate. In order to use it, insert search criteria twice, e.g., (rating=1&rating=5). If values related to server pagination are not inserted, default ones are: pageNumber: 0, pageSize: 20, sortBy: name, sortingOrder: ASC.	

Title	*Update existing DEH resource*
URL:	
api/v1/resources/{uid}	
Headers	
Required:	
x-subject-token	Authentication token obtained from IDM
x-auth-token	Capability token obtained from Capability Manager
Method	
PUT	
URL Params	
Required:	
uid=[String]	*DEH resource uid*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
name[String]	*resource name*
type[String]	*resource type*
description=[String]	*resource description*
endpoint[String]	*Resource endpoint*
status=[String]	*resource status*
category=[Arrays]	*resource categories*
maturityLevel=[Integer]	*resource maturity level*
localisation=[Arrays]	*resource localisation geo-points*
tags=[Arrays]	*Resource tags*
url=[String]	*URL for downloading/streaming data or entity*
accessibility=[Integer]	*resource accessibility*
dependencies=[Arrays]	*resource dependencies*
attachmentFIle = [MultipartFile]	*resource Attachments*
Success response	

200
Content: {
"success": true,
"message": "DEH Resource successfully updated",
"data": {
"uid": "606ef80c6393cd08312af39a",
"name": "Test Resource 1",
"type": "Service",
"category": [
"Business application"
],
"description": "Lorem ipsum dolor sit amet",
"endpoint":
"https://jsonplaceholder.typicode.com/comments",
"status": 1,
"version": "1",
"maturityLevel": 1,
"owner": "0a1d17fe-a4bd-4aa8-9054-998a2018ae70",
"author": {
"username": "mstojanovi",
"email": "marko.stojanovic@eng.it"
},
"tags": [
"Docker",
"AI"
],
"attachment": [
{
"id": "606ef80c6393cd08312af398",
"originalName": "Screenshot-20210324142051-1514x422.png",
"fileName": "Screenshot-20210324142051-1514x422.png",
"chunkSize": 261120,
"length": 49301,
"uploadDate": "2021-04-08T12:33:16.476+00:00",
"contentType": "image/png"
}
],
"rating": 0.0,
"localisation": [
{
"x": 1.0,
"y": 1.0,
"coordinates": [
1.0,
1.0
	success: true if there no errors
	message: message form the server e.g., DEH Resource successfully updated
	data: JSON object with DEH Resource data
	extraData: optional JSON object which contains extra information e.g., Logs, data etc.

`],`
`"type": "Point"`
`}`
`],`
`"accessibility": 0,`
`"createAt": "2021-04-08T14:33:16.523",`
`"lastUpdate": "2021-04-08T14:33:16.523",`
`"dependencies": [`
`"docker"`
`],`
`"accessControlPolicies": [`
`"read"`
`],`
`"url": "www.google.com",`
`"billingInformation": [],`
`"downloadsHistory": {}`
`},`
`"extraData": null`
`}`	

Error response	

`Content: {`
`        "success": false,`
`        "message": "",`
`        "data": [],`
`        "extraData": {}`
`}`
	success: false if there is an error
	message:message form the server e.g., Resource Not Found
	data: null
	extraData: optional JSON object which contains error specific information like type of error, exact error message, timestamp, called path

Sample call	

`curl --location --request PUT 'https://acs.bse.h2020-demeter-cloud.eu:1029/api/v1/resources/{uid}' \`
`--header 'x-auth-token: $TOKEN_VALUE' \`
`--header 'x-subject-token: $TOKEN_VALUE'\`
`--form 'name="Resource Test Edit"' \`
`--form 'type="Service"' \`
`--form 'category="Business Application"' \`
`--form 'description="Edit description"' \`
`--form 'endpoint="www.edit.com"' \`
`--form 'status="1"' \`
`--form 'version="1"' \`
`--form 'maturityLevel="1"' \`
`--form 'tags="ML, BigData"' \`
`--form 'attachmentFile=@"/home/mare/Pictures/Screenshot-20210401122805-1347x2357.png"' \`
`--form 'localisation="1,1"' \`
`--form 'accessibility="0"' \`
`--form 'dependencies="java"' \`
`--form 'url="www.edit.com"'`	

Notes	
N/A	

Title	*Delete DEH resource*
URL:	
api/v1/resources/{uid}	
Headers	
Required:	
x-subject-token	Authentication token obtained from IDM
x-auth-token	Capability token obtained from Capability Manager
Method	
DELETE	
URL Params	
Required:	
uid=[integer]	*DEH resource uid*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	

200
`Content: {`
`"success": true,`
`"message": "DEH Resource successfully deleted",`
`"data": null,`
`"extraData": "Successfully deleted resource with uid: 606ef80c6393cd08312af39a"`
`}`
	success: true if there no errors
	message: message form the server e.g., DEH Resource successfully deleted
	data: null
	extraData: optional JSON object which contains extra information about deleted resource etc.

Error response	
Content: { "success": false, "message": "", "data": [], "extraData": {}	*success: false if there is an error* *message:message form the server e.g., Resource Not Found* *data: null* *extraData: optional json object which contains error specific information like type of error, exact error message, timestamp, called path*
Sample call	

`curl --location -g --request DELETE 'https://acs.bse.h2020-demeter-cloud.eu:1029/api/v1/resources/{uid}' \`
`--header 'x-auth-token: $TOKEN_VALUE' \`
`--header 'x-subject-token: $TOKEN_VALUE'`	

Notes	
N/A	

Title	*Rate DEH resource*
URL:	
api/v1/resources/{uid}/rate	
Headers	
Required:	
x-subject-token	Authentication token obtained from IDM
x-auth-token	Capability token obtained from Capability Manager
Method	
POST	
URL Params	
Required:	
uid	*DEH Resource UID*
Optional:	
N/A	*N/A*
Data Params	
Required:	
rating[Dobule]	*DEH rating score*
Optional:	
N/A	*N/A*
Success response	

200
`Content: {`
`"success": true,`
`"message": "DEH Resource successfully rated",`
`"data": null,`
`"extraData": null`
	success: true if there no errors
	message: message form the server e.g., DEH Resource successfully rated
	data: null
	extraData: optional JSON object which contains extra information about deleted resource etc.

Error response	
Content: { "success": false, "message": "", "data": [], "extraData": {}	*success: false if there is an error* *message:message form the server e.g., Resource Not Found* *data: null* *extraData: optional json object which contains error specific information like type of error, exact error message, timestamp, called path*
Sample call	

`curl --location -g --request POST 'https://acs.bse.h2020-demeter-cloud.eu:1029/api/v1/resources/{uid}/rate' \`
`--header 'x-auth-token: $TOKEN_VALUE' \`
`--header 'x-subject-token: $TOKEN_VALUE' \`
`--header 'Content-Type: application/json' \`
`--data-raw '4.0'`	

Notes	
N/A	

Title	*Get Attachment by ID*
URL:	
api/v1/attachments/{id}	
Headers	
Required:	
x-subject-token	Authentication token obtained from IDM
x-auth-token	Capability token obtained from Capability Manager
Method	
GET	
URL Params	
Required:	
id=[integer]	*Attachment id*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	

200
`Content: {`
`"success": true,`
`"message": "Attachment found",`
`"data": {`
`"id": "605cbaff43c6a118630419e6",`
`"originalName": "Screenshot-20210304140849-1395x160.png",`
`"fileName": "Screenshot-20210304140849-1395x160.png",`
`"chunkSize": 261120,`
`"length": 26648,`
`"uploadDate": "2021-03-25T16:31:59.669+00:00",`
`"contentType": "image/png"`
`},`
`"extraData": null`
`}`
	success: true if there no errors
	message: message form the server e.g., Attachment found
	data: JSON object with attachment data
	extraData: optional JSON object which contains extra information about deleted resource etc.

Error response	

Content: {
`"success": false,`
`"message": "",`
`"data": [],`
`"extraData": {}`
	success: false if there is an error
	message:message form the server e.g., Resource Not Found
	data: null
	extraData: optional json object which contains error specific information like type of error, exact error message, timestamp, called path

Sample call	
curl --location -g --request GET 'https://acs.bse.h2020-demeter-cloud.eu:1029/api/v1/attachments/{id}' \ --header 'x-auth-token: $TOKEN_VALUE' \	

--header 'x-subject-token: $TOKEN_VALUE'
Notes
N/A

Title	*Get Attachment content by ID*
URL:	
api/v1/attachments/content/{uid}	
Headers	
Required:	
x-subject-token	Authentication token obtained from IDM
x-auth-token	Capability token obtained from Capability Manager
Method	
GET	
URL Params	
Required:	
id=[integer]	*Attachment id*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	
200 Content: Attachment content/stram	Picture, video, doc file.
Error response	

`Content: {`
`"success": false,`
`"message": "",`
`"data": [],`
`"extraData": {}`
	success: false if there is an error
	message:message form the server e.g., Resource Not Found
	data: null
	extraData: optional json object which contains error specific information like type of error, exact error message, timestamp, called path

Sample call	

`curl --location -g --request GET 'https://acs.bse.h2020-demeter-cloud.eu:1029/api/v1/attachments/content/{id}' \`
`--header 'x-auth-token: $TOKEN_VALUE' \`
`--header 'x-subject-token: $TOKEN_VALUE'`	

Notes	
N/A	

Title	*Get All Metrics of users DEH Resources*
URL:	
api/v1/metrics	
Headers	
Required:	
x-subject-token	Authentication token obtained from IDM
x-auth-token	Capability token obtained from Capability Manager
Method	
GET	
URL Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	

200
`Content: {`
`"success": true,`
`"message": "Users metrics found",`
`"data": {`
`[`
`   {`
`      "rrm_id":1,`
`      "name":"resource_1",`
`      "numberOfInstances":3`
`   },`
`   {`
`      "rrm_id":2,`
`      "name":"resource_2",`
`      "numberOfInstances":5`
`   },`
`   {`
`      "rrm_id":3,`
`      "name":"resource_3",`
`      "numberOfInstances":4`
`   }`
`]`
`},`
`"extraData": null`
`}`
	success: true if there no errors
	message: message form the server e.g., Users metrics found
	data: JSON object with attachment data
	extraData: optional JSON object which contains extra information about deleted resource etc.

Error response	

Content: {
`        "success": false,`
`        "message": "",`
`        "data": [],`
`        "extraData": {}`
	success: false if there is an error
	message:message form the server e.g., Resource Not Found
	data: null
	extraData: optional json object which contains error specific information like type of error, exact error message, timestamp, called path

Sample call
curl --location -g --request GET 'https://acs.bse.h2020-demeter-cloud.eu:1029/api/v1/metrics' \ --header 'x-auth-token: $TOKEN_VALUE' \ --header 'x-subject-token: $TOKEN_VALUE'
Notes
N/A

Title	*Get Metrics by DEH Resource ID*
URL:	
api/v1/metrics/rrmid/{rrm_id}	
Headers	
Required:	
x-subject-token	Authentication token obtained from IDM
x-auth-token	Capability token obtained from Capability Manager
Method	
GET	
URL Params	
Required:	
rrm_id=[Alphanumeric]	*Resource RRM id*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	

200
`Content: {`
`"success": true,`
`"message": "Metrics for Resource with  id 605f580337801e241cf995ec  found",`
`"data":`
` {`
`   "rrm_id":"605f580337801e241cf995ec",`
`   "name":"estimate-animal-welfare-condition",`
`   "numberOfInstances":3,`
`   "containers":[`
`      {`
 	
`"containerId":"9c84b82b1e3f2a4517161b39710275be2dffecd74e934171f090e2c10ce10388",`
`         "uptime":428457,`
`         "hostname":"0c9ff4f3419b",`
`         "ip":"172.17.0.22",`
`         "image":"estimate-animal-welfare-condition",`
`         "bseId":"DEMETER:BSE-ID:(estimate-animal-welfare-condition9)-9eb2face-45a4-416f-8241-dd8a33cc74cb",`
`         "rrmId":"605f580337801e241cf995ec",`
`         "lastupdated":"2021-04-07 14:34:56 UTC+0000",`
`         "cpuConsumption":[`
	success: true if there no errors
	message: message form the server e.g., Users metrics found
	data: JSON object with attachment data
	extraData: optional JSON object which contains extra information about deleted resource etc.

```
            {
               "date":"2021-04-07 14:48:54
UTC+0000",

"cpuPercentPeak":1.2174489775561097
            },
            {
               "date":"2021-04-08 15:14:13
UTC+0000",

"cpuPercentPeak":11.2167604739336493
            },
            {
               "date":"2021-04-09 17:22:13
UTC+0000",

"cpuPercentPeak":33.9167604739336493
            }
         ],
         "memoryConsumption":[
            {
               "date":"2021-04-07 14:34:56
UTC+0000",

"memoryPercentPeak":3.2411014738789543
            },
            {
               "date":"2021-04-08 15:25:06
UTC+0000",

"memoryPercentPeak":33.3027923153840795
            },
            {
               "date":"2021-04-09 15:25:06
UTC+0000",

"memoryPercentPeak":63.3027923153840795
            }
         ]
      }
   ]
}
"extraData": null

}
```

Error response

Content: { "success": false, "message": "", "data": [], "extraData": {}	*success: false if there is an error* *message:message form the server e.g., Resource Not Found* *data: null* *extraData: optional json object which contains error specific information like type of error, exact error message, timestamp, called path*

Sample call

```
curl --location -g --request GET 'https://acs.bse.h2020-demeter-
cloud.eu:1029/api/v1/metrics/{rrm_id}' \
--header 'x-auth-token: $TOKEN_VALUE' \
--header 'x-subject-token: $TOKEN_VALUE'
```

Notes
N/A

Title	*Get Metrics by Container ID*
URL:	
api/v1/metrics/containerid/{container_id}	
Headers	
Required:	
x-subject-token	Authentication token obtained from IDM
x-auth-token	Capability token obtained from Capability Manager
Method	
GET	
URL Params	
Required:	
container_id=[Alphanumeric]	*Container ID*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	

200
`Content: {`
`"success": true,`
`"message": "Metrics for Resource with  id 605f580337801e241cf995ec  found",`
`"data":  ,"extraData": null}`
	success: true if there no errors
	message: message form the server e.g., Users metrics found
	data: JSON object with attachment data
	extraData: optional JSON object which contains extra information about deleted resource etc.

Error response	

`Content: {`
`"success": false,`
`"message": "",`
`"data": [],`
`"extraData": {}`
	success: false if there is an error
	message:message form the server e.g., Resource Not Found
	data: null
	extraData: optional json object which contains error specific information like type of error, exact error message, timestamp, called path

Sample call	

`curl --location -g --request GET 'https://acs.bse.h2020-demeter-cloud.eu:1029/api/v1/metrics/{container_id}' \`
`--header 'x-auth-token: $TOKEN_VALUE' \`
`--header 'x-subject-token: $TOKEN_VALUE'`	

Notes	
N/A	

Title	*Create new Metrics data*
URL:	
api/v1/metrics	
Headers	
Required:	
x-subject-token	Authentication token obtained from IDM
x-auth-token	Capability token obtained from Capability Manager
Method	
POST	
URL Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	

200
`Content: {`
`"success": true,`
`"message": "Metrics successfully stored",`
`"data": null,`
`"extraData": null`
	success: true if there no errors
	message: message form the server e.g., DEH Resource successfully rated
	data: null
	extraData: optional JSON object which contains extra information about deleted resource etc.

Error response	

`Content: {`
`"success": false,`
`"message": "",`
`"data": [],`
`"extraData": {}`
	success: false if there is an error
	message:message form the server e.g., Resource Not Found
	data: null
	extraData: optional json object which contains error specific information like type of error, exact error message, timestamp, called path

Sample call	
N/A	
Notes	
N/A	

9.5 Technologies and implementation details

This section summarises the technologies, tools and frameworks linked to the implementation of DEH. Table 20 summarises resources linked to the DEH Dashboard:

Table 20: DEH Dashboard linked resources

AngularJS	https://docs.angularjs.org/api
NodeJS	https://nodejs.org/en/docs
ElasticSearch	https://www.elastic.co/guide/index.html
MongoDB	https://docs.mongodb.com
Express.js	https://expressjs.com/en/guide/routing.html

DEH Dashboard has two parts, the Front-End part developed in AngularJS which is SPA (Single Page Application) framework and the back-end part developed in Node.js, which is JavaScript runtime environment that executes JavaScript code outside a web browser, where Express.js is used for routing. For the purpose of storing templates, services and forms MongoDB which is a NoSQL document-oriented database is used. Elasticsearch is used for the purpose of storing entities, their indexing, and fast searching.

Table 21 summarises resources linked to the DEH Resource Registry Management:

Table 21: DEH Resource Registry Management linked resources

Spring Boot	https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/
MongoDB	https://docs.mongodb.com/
Spring Data MongoDB	https://docs.spring.io/spring-data/mongodb/docs/current/reference/html
Swagger	https://swagger.io/docs/
Lombok	https://projectlombok.org/features/all
Apache Maven	https://maven.apache.org/guides/index.html

The DEH Resource Registry Management is developed using Spring Boot which is an industry-standard Java-based Framework, with Maven as a build automation tool. For the purpose of storing the data related to resources, MongoDB, which is a NoSQL document-oriented database, was used. As additional modules that are used within Spring eco-system, we can mention Spring Data MongoDB which is used to make it easier to manage data in MongoDB. Lombok is used to reduce the boilerplate code related to Entities, while Swagger is used for the purpose of documenting and testing REST API's.

DEH Discovery Management, the design and operations of which are briefly described in section 9.2.2, is essentially based on data provided by the DEH Resource Registry Management components. This essentially means that the same technologies used in the aforementioned module will be sufficient for the development of the DEH Discovery Management as well. These cover all the key requirements for resource discovery as laid out by the requirements analysis. For the second phase of the development of this component, we may need to develop some additional support tools which would probably require the use of a simple database separate from the Resource Registry DB (easily covered by MongoDB) in order, e.g., to store and reprocess the past queries submitted by users.

10 Core Enablers for Integration

10.1 Functional Interoperability Enabler (FIE)

10.1.1 Functionality description

Functional Interoperability core Enabler (FIE) is a core DEMETER enabler and can be regarded as the client-side of the Brokerage Service Environment. This Enabler provides all the services of the BSE to the rest of the DEMETER modules and enablers (Core and Advanced), and also to the Consumer's application. It serves as a wrapper for the Registration, Discovery, and Provisioning services offered by the BSE, but also offers the compatibility check feature, i.e., a compatibility check of a service to be registered, with the BSE data model. Despite initially envisioned being deployed on the consumers' application premises, aiming to ease its use remove complexity from the consumer, the Functional Interoperability (core) enabler is bundled with the BSE module and provides its functionality along with the BSE module.

10.1.2 Interaction with other Enablers

This Enabler offers an HTTPS API to any other Enabler (Core and Advanced) that needs to make use of the provided services.

10.1.3 Dependencies on other Core/Advanced Enablers

This Enabler depends on the Access Control Enabler (also a Core DEMETER Enabler) as it requires the access token (authentication/authorisation) to successfully communicate with the BSE. It also depends on the Semantic Interoperability Core Enabler (WP2) and the Agricultural Information Model (AIM) related functionality that this Enabler offers.

10.1.4 Deployment considerations

As mentioned earlier, this core enabler is bundled with the BSE; hence, there is no action needed from a user's point of view. The deployment considerations are the ones described for the deployment of BSE on-premises.

10.1.5 Technical description

This section formally describes features/characteristics of this Enabler.

10.1.5.1 API and Data model

In section 7.4, the BSE API and data models were described. As already mentioned, the FIE enabler serves as a wrapper of the services offered by BSE, providing the compatibility check functionality on top of them (triggered during a resource registration). Therefore, the API and data model described in that section partially covers FIE's API and data model; below, we describe the additional endpoint and data model that concern only the FIE.

Table 22: Functional Interoperability enabler CompatibilityChecker data model

Name	CompatibilityChecker Data Model	
Property	Type	Description
applicationCategory	string	*Type of service*
description	string	*Description of the service*
version	number	*service's version*
deh_id	string	*Service DEMETER HUB unique identifier*
featureList	String array	*service's feature list*
dataEncryption	boolean	*Whether data are encrypted or not*
authentication	boolean	*Whether the user needs to authenticate*
conditionsOfAccess	string	*Process to authenticate user*
timeRequired	integer	*max seconds between when a user makes a request and system response*
quota	string	*the maximum number of requests that can be done to the service*
offers	number	*cost of the service*
TermsOfService	string	*License model of the service*
usageInfo	string	*Link to extra information about the service*
provider	string	*The name of the service provider*
spatial	string	*country where the service is hosted*
aggregateRating	integer	*The score representing the service 1-5*
apiModel	Object	*Service's API description data model*

Table 23: Functional Interoperability compatibility endpoint

Title	Check data model compatibility
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
https://bse.h2020-demeter-cloud.eu/api/BSE/compatibility	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Optional:	
Data Params This field holds the body payload of a post request.	
Required:	
data	CompatibilityChecker Data Model
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	
201 Content: { }	Compatibility check was successful
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	

404	Not found
403	Not authorised
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

10.1.5.2 Use cases / Data flow

In this section, we present the sequence diagram (Figure 46) and the activity diagrams (Figure 47, Figure 48,) that depict how FIE achieves the registration, discovery and provisioning of a resource. In this process the security module, FIE, and BSE are involved.

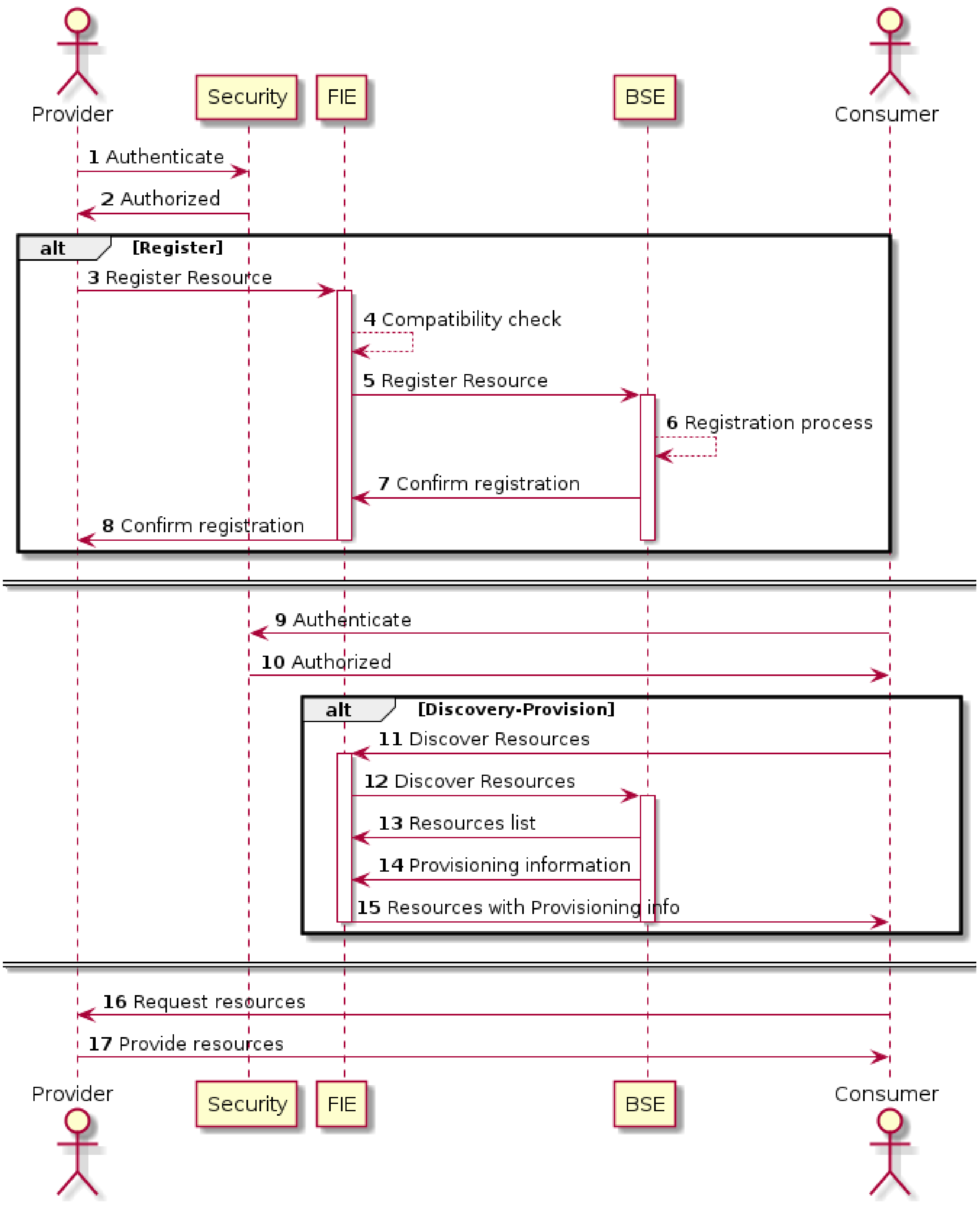


Figure 46: Functionality Enabler (FI) Sequence diagram

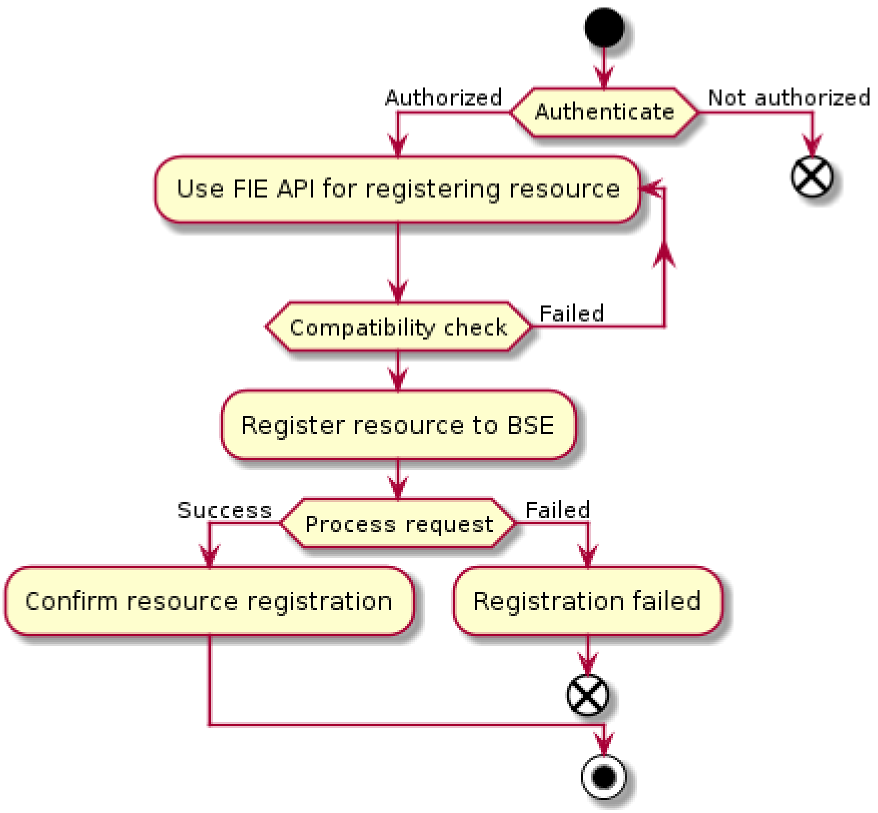


Figure 47: FI Activity diagram - Provider

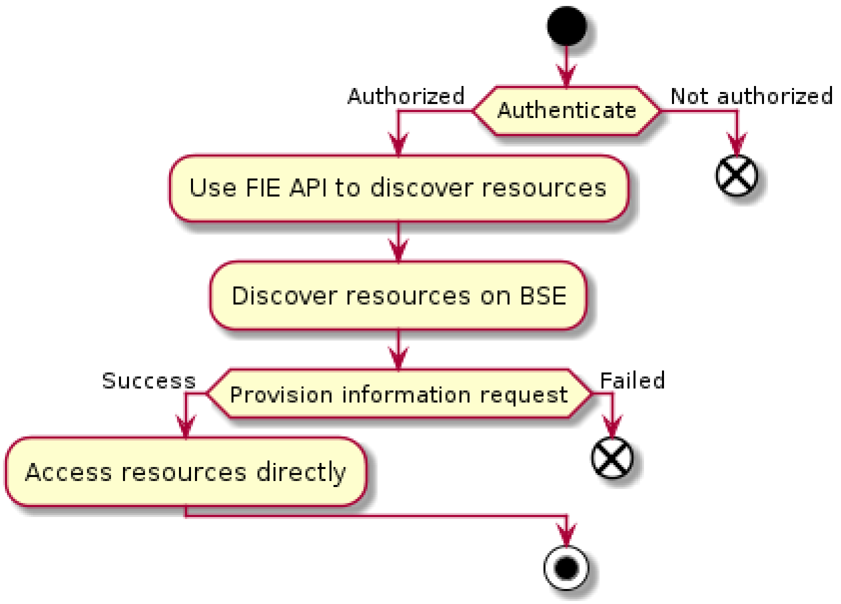

Authenticate
Authorized
Not authorized
Use FIE API to discover resources
Discover resources on BSE
Provision information request
Success
Failed
Access resources directly

Figure 48: FI Activity diagram - Consumer

10.1.5.3 Deployment

In this section, we describe the deployment process for the FI enabler using a Docker-compose script and the deployment execution commands. As described above, FIE is bundled with the BSE module. Therefore, the deployment process described concerns not only the FIE enabler, but the BSE/FIE bundle. In addition to the BSE/FIE bundle, taking into consideration the dependency of the bundle to the Access Control core enabler, the deployment process also includes the deployment of the PEP-Proxy submodule (described in section 8.6). These modules/enablers along with ACS, form a self-contained, full-featured, and secure (authentication/authorisation) full deployment of DEMETER's Brokerage Service Environment.

```
version: '3.4'

services:
  fie:
    image: registry.gitlab.com/demeterproject/wp3/bse/bse:bse_web_1.3.2
    restart: always
    command: ["sh", "/home/app/web/entrypoint.sh"]
    volumes:
      - staticfiles:/home/app/web/staticfiles
    expose:
      - 8000
    env_file:
      - ./common/.env.prod
    depends_on:
      - db
    networks:
      - bse_network

  pepproxybse:
    image:
registry.gitlab.com/demeterproject/wp3/se/components/pep_proxy:1.0
    logging:
      driver: "json-file"
      options:
        max-size: "10m"
        max-file: "3"
    expose:
        - 1027

    depends_on:
      - fie

    volumes:
      - ./etc/letsencrypt/fullchain.pem:/opt/PEP-Proxy/certs/fullchain-
demeter.pem  # volumes to import certificates to container
      - ./etc/letsencrypt/privkey.pem:/opt/PEP-Proxy/certs/privkey-
demeter.pem

    env_file:
      - ./common/.env.prod.pepproxy

    networks:
      - bse_network

    restart: unless-stopped

  nginx:
    image: registry.gitlab.com/demeterproject/wp3/bse/bse:bse_nginx_1.1
    volumes:
      - staticfiles:/home/app/web/staticfiles
      - ./fullchain.pem:/etc/letsencrypt/live/bse.h2020-demeter-
cloud.eu/fullchain.pem
      - ./privkey.pem:/etc/letsencrypt/live/bse.h2020-demeter-
cloud.eu/privkey.pem
    ports:
      - 80:80
      - 443:443
    depends_on:
      - fie
      - pepproxybse
    networks:
      - bse_network

  db:
    image: postgres:12.0-alpine
    volumes:
      - postgres_data:/var/lib/postgresql/data/
    env_file:
      - ./common/.env.prod.db
    networks:
      - bse_network

  consul:
    image: registry.gitlab.com/demeterproject/wp3/bse/bse:bse_consul_1.0
    expose:
      - 8500
      - 8600/udp
    restart: always
    command: agent -server -bind 0.0.0.0 -client 0.0.0.0 -bootstrap-expect 1
-ui -config-file=/consul/config/consul-config.json -data-dir=/consul/data
    volumes:
      - ./common/consul/config/consul-config.json:/consul/config/consul-
config.json
      - ./common/consul/data:/consul/data
    networks:
      - bse_network

networks:
  bse_network:
    driver_opts:
      com.docker.network.bridge.name: br_bse
```

Deploy by:

```
sudo docker-compose up -d
```

10.1.5.4 Configuration parameters

Configuration parameter	Value	Type	Description
`pep_protocol`	Defined by user	String	HTTP (SSL is served through the reverse proxy service, i.e., nginx)
`target_protocol`	Defined by user	String	HTTP (SSL is served through the reverse proxy service, i.e., nginx)
`target_host`	Defined by user	String	FIE service name, e.g., `fie`
`target_port`	Defined by user	Number	FIE service port e.g., 8000
`target_API`	Defined by user	String	BSE API title, e.g., `DemeterBSE`
`PEP_ENDPOINT`	Defined by user	String	BSE endpoint, e.g., `https://bse.h2020-demeter-cloud.eu:443`

10.2 Access Control Enabler

10.2.1 Authentication Security Enabler

10.2.1.1 Functionality description

The Security Authentication Enabler library provides the DEMETER components and the pilots developers with an abstract way to access to the Authentication OAuth 2.0 functionalities exposed by the DEMETER Authentication component REST API.

This library provides the following functions:

- Authentication by username and password
- Refresh authentication
- Revoke authentication token

10.2.1.2 Interaction with other Enablers

The Security Authentication Enabler may need to interact with the Communication and Networking Enabler to obtain a secured communication channel to perform the authentication functionalities.

This enabler will also provide to the Security Authorisation Enabler(s) the authentication token needed to perform authorisation functionalities.

10.2.1.3 Dependencies on other Core/Advanced Enablers

The functionalities provided by the security enablers (e.g., https communication, authentication and authorisation tokens) will be used by the other Core/Advanced Enablers and other DEMETER components in order to obtain a secured communication channel and get direct access to DEMETER resources. Therefore, the security enablers do not have any dependencies with other Enablers or DEMETER components.

10.2.1.4 Deployment/Development considerations

The authentication security enabler will be provided as a dynamic library, initially for both Windows and Linux Operating Systems.

This dynamic library can be used in different programming languages and frameworks.

10.2.1.5 Technical description

This section formally describes features/characteristics of the authentication Enabler.

Functions and Data model

The following functions are provided by this dynamic library in order to obtain, refresh and revoke authentication tokens:

Title	*Create token with Username and Password*
Function 1 *This field holds the name of the function used and the required (and optional) parameters*	
get_authentication_token(username, password)	
Output *This field holds the type of the output expected*	
Authentication token (string) and expiration (time/date)	
Params *This field holds the parameters (if any). Separated based on the fields below into <u>required</u> and <u>optional</u>.*	
Required:	
username=[string]	*String with the username to log in*
Required:	
password=[string]	*String with the password to log in*
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	
0: Output parameters ----------------- token object: string - json json string object with Authentication_Toekn field, Authentication_Token_expires_at field and the details of the response to the query Example: Authentication_Token:04c5b070-4292-4b3f-911b- Authentication_Token_expires_at:"2018-03-20T15:05:35.697Z"	*Authentication token and its expiration date to be used with following authentication/authorisation functions.*
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
-1, "Invalid client: client is invalid"	*There has been a time out event while connecting to Keyrock Server*
-1, "Invalid grant: user credentials are invalid"	*The username or password provided doesn´t match any registered user in Keyrock*
Sample call *This field holds a possible sample call to the described function*	
get_authentication_token ("user.example@example.com", "password1234")	
Notes *This field holds any additional helpful info related to the function described.*	

Title	*Refresh token*
Function 1 *This field holds the name of the function used and the required (and optional) parameters*	
refresh_authentication_token(authentication_token)	
Output *This field holds the type of the output expected*	
Authentication token (string) and expiration (time/date)	
Params *This field holds the parameters (if any). Separated based on the fields below into* *<u>required</u>* *and* *<u>optional</u>.*	
Required:	
authentication =[string]	*String with the authentication token*
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	
Authentication_Token: 65c6b870-3535-6b4f-345b-34a345f3ac7f Authentication_Token_expires_at:"2018-03-20T15:05:35.697Z"	*New authentication token and its expiration date to be used with following authentication/authorisation functions.*
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
-1, '{"400":"Invalid grant: refresh token is no longer valid"}'	*The token provided is no longer valid, therefore, a new authentication token is not provided.*
Sample call *This field holds a possible sample call to the described function*	
refresh_authentication_token(65c6b870-3535-6b4f-345b-34a345f3ac7f)	
Notes *This field holds any additional helpful info related to the function described.*	

Title	*Revoke token*
Function 1 *This field holds the name of the function used and the required (and optional) parameters*	
revoke_authentication_token(authentication_token)	
Output *This field holds the type of the output expected*	
Params *This field holds the parameters (if any). Separated based on the fields below into* *<u>required</u>* *and* *<u>optional</u>.*	
Required:	
authentication =[string]	*String with the authentication token*
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	
0	*Success response for token deletion.*
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
1, '{"400":"Invalid grant: refresh token is no longer valid"}'	*The token provided is no longer valid.*
Sample call *This field holds a possible sample call to the described function*	
revoke_authentication_token ("65c6b870-3535-6b4f-345b-34a345f3ac7f")	
Notes *This field holds any additional helpful info related to the function described.*	

Use cases / Data flow

The following figure depicts the sequence diagrams for *get_authentication_token*, *refresh_authentication_token* and *revoke_authentication_token* functions.

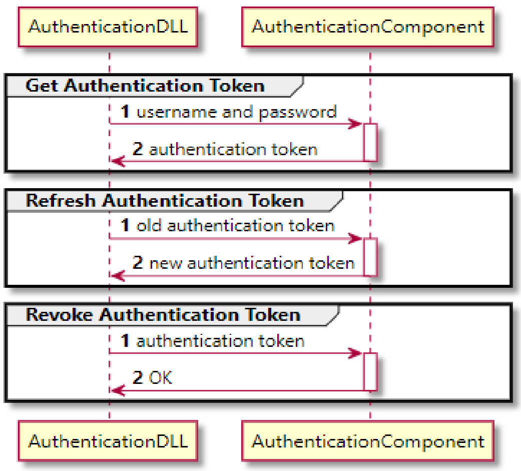


Figure 49: Authentication function sequence diagrams

The functions obtain the parameters and send it to the Authentication Component endpoint, where is processed and a response is provided, either with a new authentication token (*authentication_token*, *refresh_authentication_token*) or with the confirmation that an authentication token has been revoked (*revoke_authentication_token*).

Deployment

The library needs to be imported in the programming language of choice, and the function imported. The following examples show how to import them for several well-known and widely used programming languages such as Python, Java, and C#:

```
Python:
from demeter_authentication import login_with_password, login_with_client_credentail, refresh_token
authentication_token, expire_at = login_with_password("user1@example.com","password123")


Java:
import static demeter_authentication.*;
authentication_token, expire_at = login_with_password("user1@example.com","password123")


C#:
using demeter_authentication;
authentication_token, expire_at = login_with_password("user1@example.com","password123")
```

Configuration Parameters

The following configurations parameters are need for the library to access to the DEMETER Authentication Component:

Configuration parameter	Value	Type	Description
KEYROCK_URL	URL	String	Keyrock Endpoint

10.2.2 Authorisation Security Enabler

10.2.2.1 Functionality description

The authorisation enabler provides a solution for controlling the access to the resources stored in an information repository. It is based on a technology called Distributed Capability-Based Access Control, which basically decouples the traditional XACML framework, into two phases: one for receiving the authorisation, which is represented by the receipt of an authorisation token called Capability Token, and a second one for accessing the information repository where basically, the user/service inserts the previous Capability Token in the corresponding query so that a Policy Enforcement Point Proxy (PEP_Proxy) could check if the query matches the content of the Capability Token. In case of a positive answer, the PEP_Proxy acts as a mere intermediary between the user/service and the information repository.

10.2.2.2 Interaction with other Enablers

This enabler interacts with the authentication enabler. Before performing the authorisation process, the authentication one must be carried out. After this authentication phase, an authentication token is generated, and this token must be present in the authorisation requests. This way, the authorisation enabler interacts with the authentication enabler in order to validate this token.

Additionally, this enabler interacts with other resource repositories placed in both BSE and DEH so that the access to the different resource repositories can be controlled. So far, the current implementation depends on NGSI or NGSI-LD resource repositories.

10.2.2.3 Dependencies on other Core/Advanced Enablers

The authorisation enabler depends on the resource repository to be addressed by user/services, since they must incorporate the Capability Token to the corresponding queries so that the PEP_Proxy would be able to validate them.

10.2.2.4 Deployment/Development considerations

The authorisation enabler comprises different sub-components, nevertheless, only the endpoint for the Capability Manager is provided to the other components. For this reason, it can be accessed by following a specific REST API. Additionally, a java library (jar) has been developed to make it easier to interact with the corresponding servers. As this library uses JAVA, it can be run on different OS's.

10.2.2.5 Technical description

This section formally describes features/characteristics of an Enabler.

Functions and Data model

Data model used: each of the parameters received by this function are strings.

- Function details:
  - Name: generateCapabilityToken("authtoken","subject","resource","action")

- Expected output: CapabilityToken. A signed JSON document.
- Error messages: Error connecting to the Authorisation server.

Data models used by the functions/methods shall be described in tables:

Table 24: Authorisation Enabler Data Model Information

Name	*Authentication Enabler Data Model*	
Property	**Type**	**Description**
Authtoken	String	The token obtained from the Identity Management
Subject	String	The subject of the authorisation query
Resource	String	The resource intended to access
action	String	The operation mode: GET, POST, PUT, PATCH or DELETE

Title	*Generate Capability Token*
Function 1 *This field holds the name of the function used and the required (and optional) parameters*	
generateCapabilityToken *(authtoken,subject,resource,action)*	
Output *This field holds the type of the output expected*	
Authorisation token (json document)	
Params *This field holds the parameters (if any). Separated based on the fields below into required and optional.*	
Required:	
authtoken=[alphanumeric]	*Alphanumeric string with the authentication token*
Required:	
subject=[alphanumeric]	*Alphanumeric string with the subject of the authorisation request*
Required:	
Resource=[alphanumeric]	*Alphanumeric string identifying the resource to be accessed*
Required:	
Action=[alphanumeric]	*Alphanumeric string corresponding to the operation to be performed: GET, POST, PUT, PATCH or DELETE*
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	

`Authorisation token:`
`{`
`   “id”: “7g3vfT_q9vTL2aQ4”,`
`   “ii”: 1415174237,`
`   “is”: “issuer@um.es”,`
`   “su”:`
`“zNwS5FetB4rwzSKsWwSBAxm5wDa=JgLjHU8zSnmeSFQgSG9Hh`
`dsJrE8=”,`
`   “de”:`
`“coap://sensortemp.floor1.computersciencefaculty.u`
`m.es”,`
`   “si”:`
`“SbUudG4zuXswFBxDeHB87N6t9hR=PBQqCN3gpu7nSkuPzDk7k`
`aR3dq1=”,`
`   “ar”: [`
`        {`
`                “ac”: “GET”,`
`                “re”: “temperature”,`
`                “f”: 1,`
`                “co”: [`
`                        {`
`                        “t”: 5,`
`                        “v”: 25,`
`                        “u”: “Cel”,`
	Authorisation token.

`},`
`{`
`“t”: 6,`
`“v”: 20,`
`“u”: “Cel”,`
`}`
`]`
`}`
`],`
`“nb”: 1415174237,`
`“na”: 1415175381`
`}`	

Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
Java jar: -1, “connection timeout” REST API: 408, “request timeout”	*There has been a time out event while connecting to Authorisation Server*
Sample call *This field holds a possible sample call to the described function*	
generateCapabilityToken(“04c5b070-4292-4b3f-911b-“,jamartinez@odins.es,”ngsi-ld/v1/entities”,”GET”)	
Notes *This field holds any additional helpful info related to the function described.*	

Use cases / Data flow

Authorisation DCapBAC access control flow is presented in Figure 50 below.

UML Sequence diagram(s)

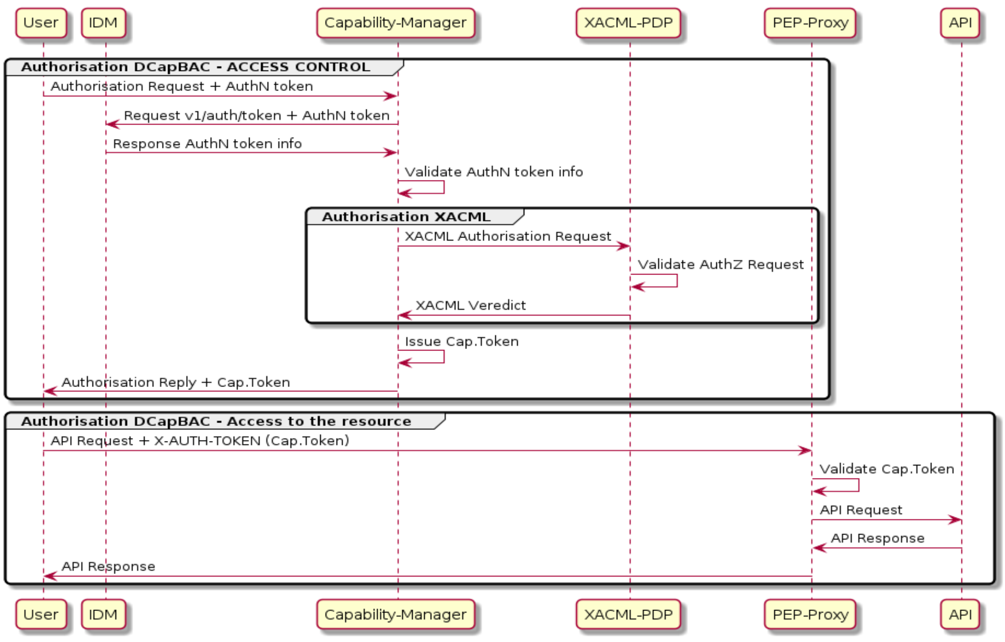

User
IDM
Capability-Manager
XACML-PDP
PEP-Proxy
API
Authorisation DCapBAC - ACCESS CONTROL
Authorisation Request + AuthN token
Request v1/auth/token + AuthN token
Response AuthN token info
Validate AuthN token info
Authorisation XACML
XACML Authorisation Request
Validate AuthZ Request
XACML Veredict
Issue Cap.Token
Authorisation Reply + Cap.Token
Authorisation DCapBAC - Access to the resource
API Request + X-AUTH-TOKEN (Cap.Token)
Validate Cap.Token
API Request
API Response
API Response

Figure 50: Authorisation DCapBAC access control sequence diagram

Deployment

The following shows how to import the demeter_authorisation library and an example of a call to generateCapabilityToken function.

```
i.e java:
import demeter_authorisation;
…
capToken = generateCapabilityToken("04c5b070-4292-4b3f-911b-",jamartinez@odins.es,"ngsi-
ld/v1/entities","GET");
```

Configuration Parameters

The following configurations parameters are need for the library to access to the DEMETER Authorisation enabler:

Configuration parameter	Value	Type	Description
AUTHORISATION_URL	URL	String	Authorisation Endpoint

10.2.3 Communications and Networking Enabler: TLS/DTLS

10.2.3.1 Functionality description

This module provides confidentiality properties to a client-server communication, to prevent unauthorised readings or alterations by malicious users.

10.2.3.2 Interaction with other Enablers

This enabler will be integrated with the authentication enabler and, in general, with other security enablers. An authentication phase is mandatory to guarantee confidentiality aspects in a secure system, in fact these security components should be considered as a single element in the system.

10.2.3.3 Dependencies on other Core/Advanced Enablers

This enabler only depends on the others security enablers in the suite, such as authentication enabler, authorisation enabler, etc.

10.2.3.4 Deployment/Development considerations

The module will be implemented with OpenSSL, which is a well-known toolkit written in C that provides several libraries and API's to perform some cryptographic tasks. OpenSSL supports several operating systems, e.g., Linux, Windows, OS X, iOS, Android, etc, with some different platforms, such as Intel, ARM, X32, etc.

10.2.3.5 Technical description

This module implements TLS/DTLS protocols providing confidentiality. Thanks to this, an HTTPS communication between client and server will be established. OpenSSL is a powerful and open-source solutions that provide an SSL/TLS toolkit and a cryptographic library. This toolkit implements all the features required by a secure communication over a computer network, in particular the module ensures that the information is not made available to unauthorised entities, preventing both readings and modifications.

The confidentiality is implemented through a secure communication channel and session keys that make the information that is exchanged private. These security aspects are possible with algorithms that cypher the data in a proper manner, so that its reading is possible only for the entities which are in possession of the right keys.

Functions and Data model

Since the OpenSSL interface is a shell command line through which the user runs the commands for the machine, there are no functions or data model to describe.

UML Activity diagram(s)

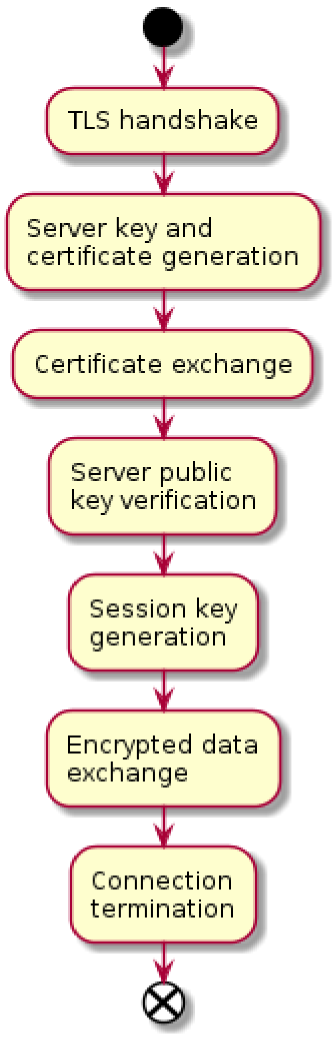


Figure 51: OpenSSL TLS/DTLS activity diagram

UML Sequence diagram(s)

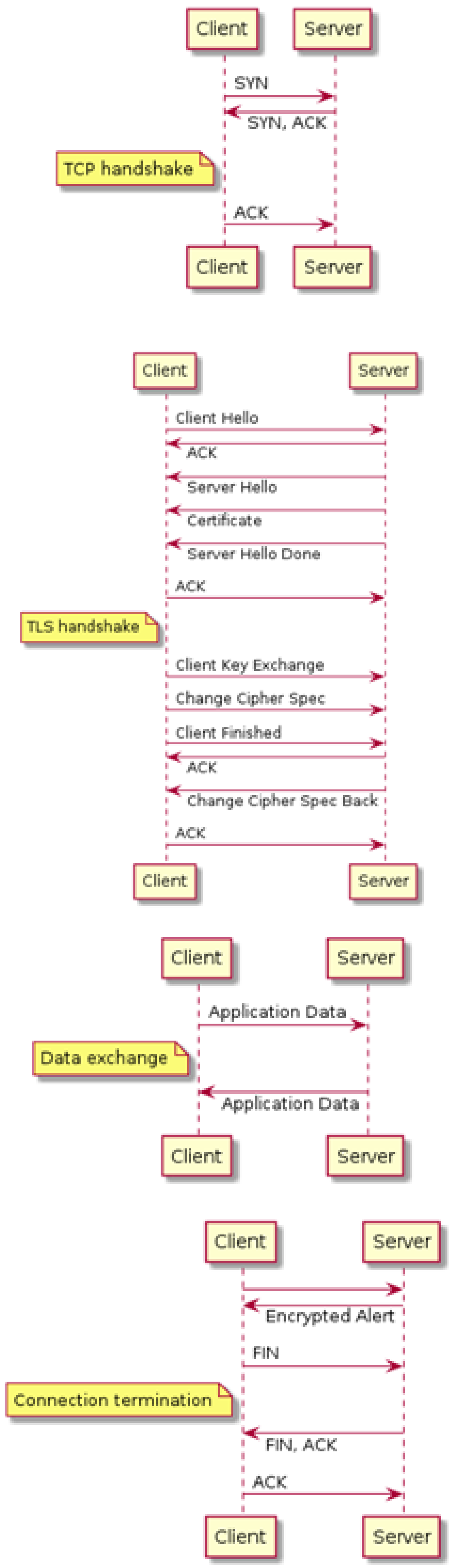


Figure 52: OpenSSL TLS/DTLS sequence diagram

Deployment

After downloading the OpenSSL master sources, to configure its library the toolkit uses a custom build system. Once configured, it is necessary to run a make command to build the library.

Configuration Parameters

This enabler does not have configuration parameters.

10.2.4 Communications and Networking Enabler: JSON/XML Encryption

10.2.4.1 Functionality description

Encryption and decryption of JSON and XML.

10.2.4.2 Interaction with other Enablers

This core enabler does not interact with any other enabler.

10.2.4.3 Dependencies on other Core/Advanced Enablers

This core enabler does not depend on any other enabler.

10.2.4.4 Deployment/Development considerations

Python library dependent on the following libraries: objcrypt, json, pyDes

```
make clean
make all
```

10.2.4.5 Technical description/information

The functions in this library make use of existing external libraries to encrypt and decrypt JSON and XML objects.

Functions and Data model

Data models used by the functions/methods shall be described in tables:

Table 25: Encryption enabler, json data model

Name	JSON	
Property	**Type**	**Description**
N/A	JSON	JSON data model

Table 26: Encryption enabler, XML data model

Name	XML	
Property	**Type**	**Description**
N/A	XML	XML data model

Title	*Encrypt_json*
Function 1 *This field holds the name of the function used and the required (and optional) parameters*	
encrypt_json(json_to_encrypt, key, labels=None)	
Output *This field holds the type of the output expected*	
Encrypted JSON (str)	
Params *This field holds the parameters (if any). Separated based on the fields below into <u>required</u> and <u>optional</u>.*	
Required:	
json_to_encrypt	JSON object to encrypt
key	Alphanumeric string containing the password for the encryption
Optional:	
labels	Name of the labels separated by ";", string. If none, select all
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
{"test": "X5rP1dfame+/5UIW35kmzoISBYOlZ4KjklL7qTTMBcSKA9lpCOZkD7lVgOWk1hWY"}	Encrypted JSON
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
-1, "connection timeout"	
Sample call *This field holds a possible sample call to the described function*	

Sample_object={
'test': 'test value'
}
encrypt_json(sample_object, "test")	

Notes *This field holds any additional helpful info related to the function described.*	

Title	*Decrypt_json*
Function 2 *This field holds the name of the function used and the required (and optional) parameters*	
decrypt_json(json_to_decrypt, key)	
Output *This field holds the type of the output expected*	
Decrypted JSON (dict)	
Params *This field holds the parameters (if any). Separated based on the fields below into <u>required</u> and <u>optional</u>.*	
Required:	
json_to_decrypt	JSON string to decrypt
Required:	
key	Alphanumeric string containing the password for the decryption
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
{'test': 'test value'}	Decrypted JSON
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
{'test': ''}	The password is wrong and the decryption failed

Sample call *This field holds a possible sample call to the described function*	

encrypted_json={"test": "X5rP1dfame+/5UIW35kmzoISBYOlZ4KjklL7qTTMBcSKA9lpCOZkD7lVgOWk1hWY"}
encrypt_json(encrypted_json, “test”)	

Notes *This field holds any additional helpful info related to the function described.*	

Title	*Encrypt_XML*
Function 3 *This field holds the name of the function used and the required (and optional) parameters*	
encrypt_xml(xml_to_encrypt, key, labels=None)	
Output *This field holds the type of the output expected*	
Encrypted XML (bytes)	
Params *This field holds the parameters (if any). Separated based on the fields below into <u>required</u> and <u>optional</u>.*	
Required:	
xml_to_encrypt	XML string to encrypt
Required:	
key	Alphanumeric 8 characters string containing the password for the encryption
Optional:	
labels	Name of the labels separated by ”;”, string. If none, select all
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
'\xfca8\x8f\xdc\xffO\n\xee\xaed\xbd\x89\xe7\xd7y\x19\xfe\xee\xa6\xa8\xb9PI\xacG-\xcd\x15ASn\xe4Yd\xaeZ#\x04G\xd2\xcb\x91|\xb4\x07\x94"z\xe5\n!\x94\xa3\x03N~Z\x19^\xa4a\xc7x\x95x\x91\xde\xc3e\'o\xb1L\xf1V\xfe\x1c\x19\xa5'	Encrypted XML
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
ValueError: Invalid DES key size. Key must be exactly 8 bytes long.	The password is too short or too long
Sample call *This field holds a possible sample call to the described function*	

data = '''
<?xml version="1.0"?>
  <test>
    <title>Sample text</title>
  </test>
'''
encrypt_xml(data, "test1234")	

Notes *This field holds any additional helpful info related to the function described.*	

Title	*Decrypt_XML*
Function 4 *This field holds the name of the function used and the required (and optional) parameters*	
decrypt_xml(xml_to_decrypt, key)	
Output *This field holds the type of the output expected*	
Decrypted XML (bytes)	
Params *This field holds the parameters (if any). Separated based on the fields below into <u>required</u> and <u>optional</u>.*	
Required:	
xml_to_decrypt	XML bytes to decrypt
Required:	

key	Alphanumeric 8 characters string containing the password for the decryption
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
b'\n<?xml version="1.0"?>\n <test>\n <title>Sample text</title>\n </test> \n'	Decrypted XML
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
TBD	The password is wrong and the decryption failed
ValueError: Invalid DES key size. Key must be exactly 8 bytes long.	The password is too short or too long
Sample call *This field holds a possible sample call to the described function*	

data = b'\xfca8\x8f\xdc\xffO\n\xee\xaed\xbd\x89\xe7\xd7y\x19\xfe\xee\xa6\xa8\xb9PI\xacG-\xcd\x15ASn\xe4Yd\xaeZ#\x04G\xd2\xcb\x91|\xb4\x07\x94"z\xe5\n!\x94\xa3\x03N~Z\x19^\xa4a\xc7x\x95x\x91\xde\xc3e\'o\xb1L\xf1V\xfe\x1c\x19\xa5'
decrypt_xml(data, "test1234")	

Notes *This field holds any additional helpful info related to the function described.*	

UML Activity diagram(s)

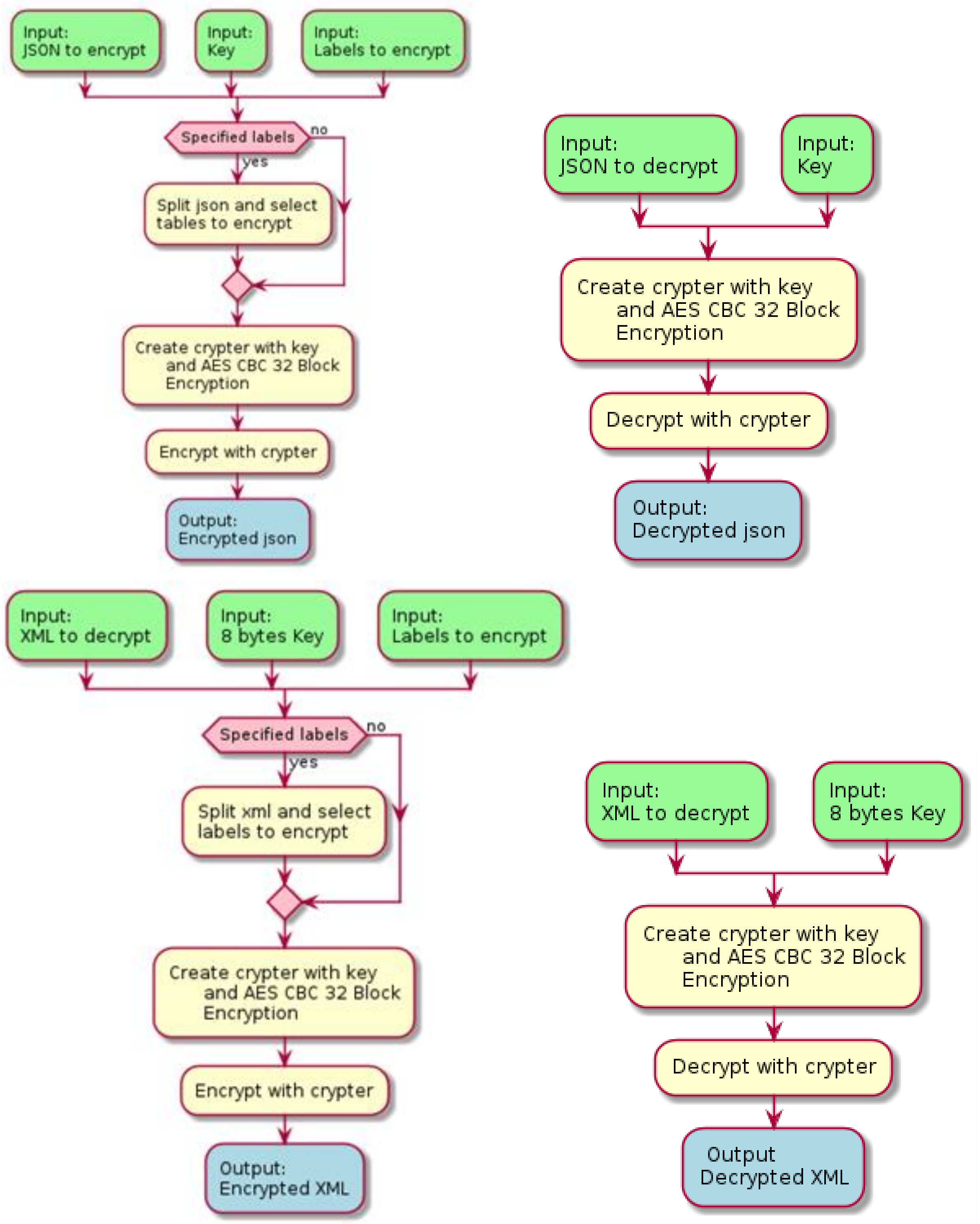


Figure 53: XML encryption and decryption activity diagram

UML Sequence diagram(s)

This enabler does not involve two agents; hence, when the library is imported in the application code, it becomes a part of it.

Deployment

```
gcc compiler:

make clean

make all
```

Configuration Parameters

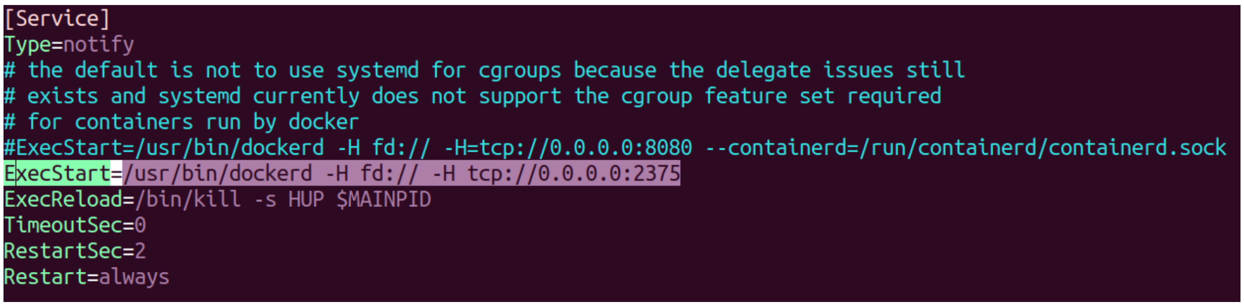
This enabler does not have configuration parameters.

10.3 DEH Client Enabler

The DEH Client Enabler Module is a core component of DEH (Demeter Enabler Hub) architecture, this module enables us to discover, monitor, and generate resource consumption metrics of DEH Service containers deployed on a Docker Host. The metrics thus generated are periodically updated to the DEH Resource Registry Management (RRM) and the metrics data is visualised with the DEH Dashboard. Here Docker Host represents a host or server running docker daemon service hosting Docker Containers and a resource represents DEH Core Modules deployed as Docker Containers i.e., DEH Service Containers on a Docker Host. This communication between DEH Client Module and Docker Host happens over a secured channel.
To encapsulate DEH Client Module as a flexible, lightweight, portable, across various environments (e.g., dev/test/qa/prod), and self-sufficient/ self-contained solution, the same is deployed as Docker. This would enable an external party to deploy DEH Client solution as a stand-alone Docker Container and start monitoring containers on any Docker Host hosting DEH Service Containers.
This section outlines the how DEH Client Module work, its functionality, interaction with other DEH Core Enablers with an architecture and data flow diagram, deployment considerations, and finally data model, exposed API’s, and usage.

10.3.1.1 How DEH Client works

This section describes DEH Client Module design and underlying technology. By default, Docker runs through a non-networked UNIX socket. It can optionally be made to communicate using an HTTP socket. The Docker Engine API is an HTTP API served by the Docker Engine. So, everything the Docker Client CLI can do can also be done with the API and this API can be accessed from a remote Docker Client CLI. It is this API service (Docker Engine Remote API) that DEH Client Module uses to communicate with the Docker Engine and have full access over Docker Daemon service running on Docker Host. This enables the module to monitor containers running on a remote Docker Host and generate resource consumption metrics at runtime. Thus, enabling modularity and flexibility by not having to bundle DEH Client solution with all possible Docker Host. So, the user can configure to listen to any Docker Host and runtime and start generating metrics.
The screenshot below shows how to enable Docker Engine to communicate over HTTP socket. Updated /lib/systemd/system/docker.service configuration is shown below:



```
[Service]
Type=notify
# the default is not to use systemd for cgroups because the delegate issues still
# exists and systemd currently does not support the cgroup feature set required
# for containers run by docker
#ExecStart=/usr/bin/dockerd -H fd:// -H=tcp://0.0.0.0:8080 --containerd=/run/containerd/containerd.sock
ExecStart=/usr/bin/dockerd -H fd:// -H tcp://0.0.0.0:2375
ExecReload=/bin/kill -s HUP $MAINPID
TimeoutSec=0
RestartSec=2
Restart=always
```

Figure 54: DEH Client Docker Engine API

Figure 54 above shows how to enable Docker Engine to communicate over HTTP socket. Update /lib/systemd/system/docker.service configuration with -H tcp://0.0.0.0:2375, and the Daemon will open a TCP socket listening on 2375 TCP port for all system interfaces and for any remote Docker Client to have access to.

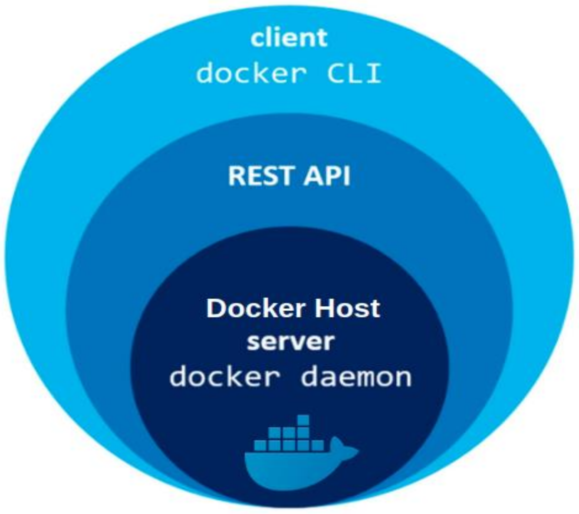


Figure 55: DEH Client Docker Engine Remote API.

Figure 55 depicts the Docker Engine Remote API layer.

To secure communication between DEH Client and Docker Host a configuration is needed. DEH Client uses Remote Docker Engine API to communicate with Docker Host. At this point, the API is public and available to anyone. This is not a good idea, especially if this Engine API has to be exposed over the internet.

For secured communication, users can configure both the Docker Client CLI (DEH Client Module) and the Docker Engine daemon to require TLS for authentication. So, all communications between the Docker Engine CLI - DEH Client and the Docker Engine daemon are accompanied by a signed trusted digital certificate.

How to secure communication between DEH Client and Docker Host

This section outlines how a secured communication channel is established between DEH Client Module and Docker Host using TLS Authentication.

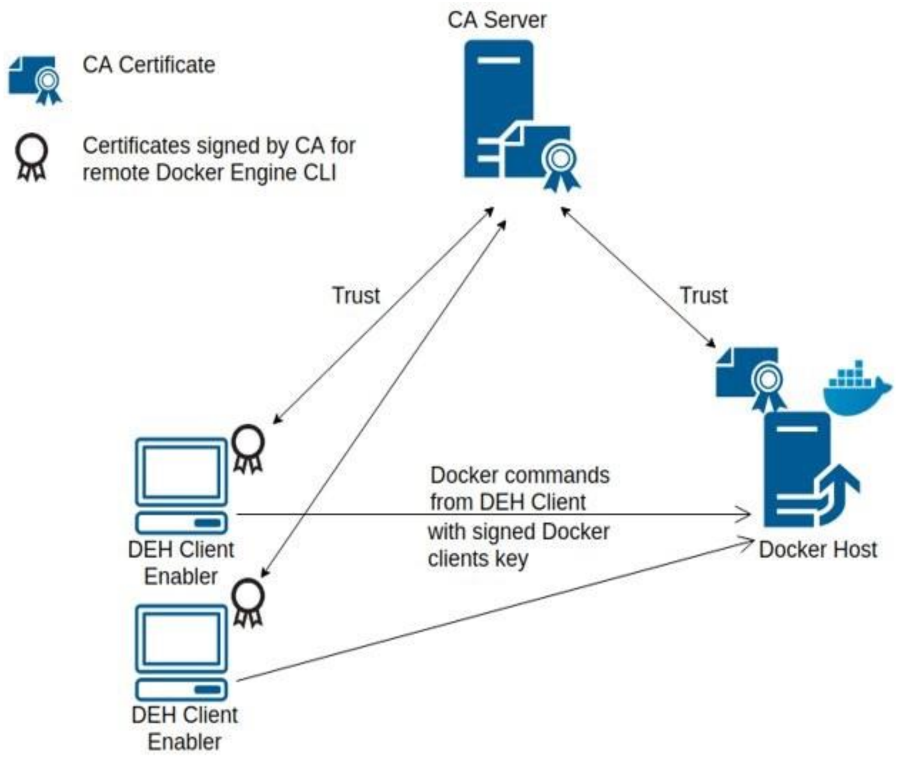

CA Server
CA Certificate
Certificates signed by CA for remote Docker Engine CLI
Trust
Trust
Docker commands from DEH Client with signed Docker clients key
DEH Client Enabler
DEH Client Enabler
Docker Host

Figure 56: DEH Client and Docker Host Secured Communication.

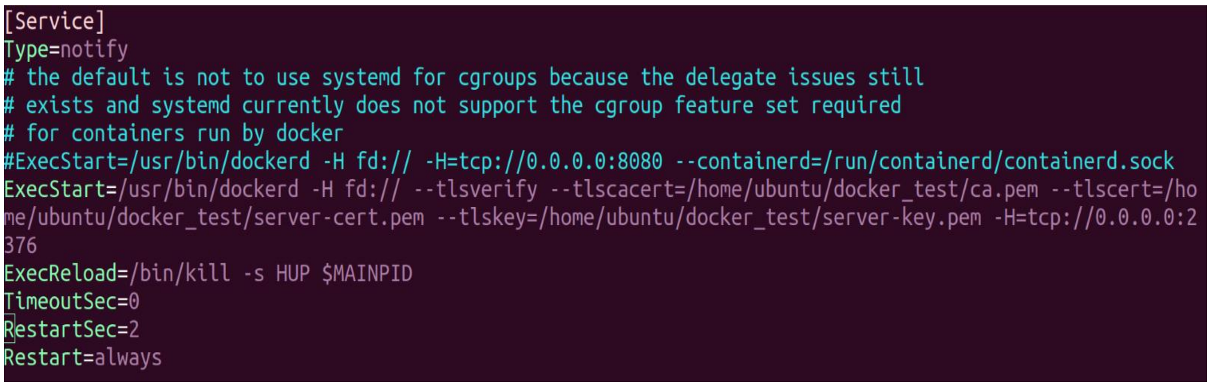
Figure 56 above depicts how trust is established between DEH Client and Docker Host.

Configure Docker Host: In order to make Docker Daemon reachable through the network in a safe manner, secured TLS connection must be configured using self-signed certificates with OpenSSL package and TLS must be enabled by specifying the tlsverify flag and pointing Docker's tlscacert flag to a trusted CA certificate as shown below. To Make Docker Daemon accept only connections from clients providing a certificate trusted by our CA and over an assigned port, run Docker daemon with --tls flag along with CA issued certificate as below..

```
$ dockerd --tlsverify --tlscacert=Cert/ca.pem --tlscert=Cert/server-cert.pem --tlskey=Cert/server-key.pem -H=0.0.0.0:<<Port Number>>
```

Sample Docker Host configuration as shown in the below screenshot.

```
vi /lib/systemd/system/docker.service
```

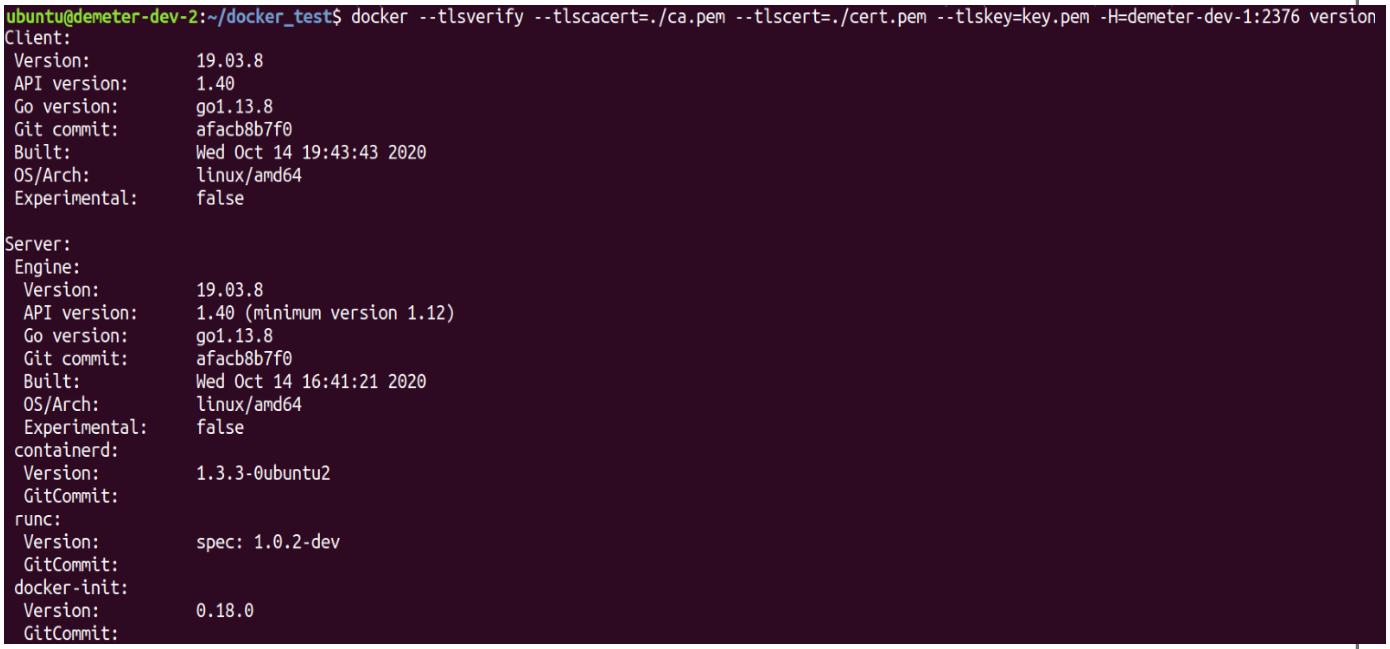


```
[Service]
Type=notify
# the default is not to use systemd for cgroups because the delegate issues still
# exists and systemd currently does not support the cgroup feature set required
# for containers run by docker
#ExecStart=/usr/bin/dockerd -H fd:// -H=tcp://0.0.0.0:8080 --containerd=/run/containerd/containerd.sock
ExecStart=/usr/bin/dockerd -H fd:// --tlsverify --tlscacert=/home/ubuntu/docker_test/ca.pem --tlscert=/ho
me/ubuntu/docker_test/server-cert.pem --tlskey=/home/ubuntu/docker_test/server-key.pem -H=tcp://0.0.0.0:2
376
ExecReload=/bin/kill -s HUP $MAINPID
TimeoutSec=0
RestartSec=2
Restart=always
```

Figure 57: DEH Client - Configure Docker Host.

On Client to access remote Docker Daemon, the Docker Engine CLI (DEH Client) must provide its digital certificate for Docker engine daemon to accept incoming commands from it.

$ docker --tlsverify --tlscacert=Cert/ca.pem --tlscert=Cert/cert.pem --tlskey=Cert/key.pem -H=$DOCKER_DAEMON_HOST:2376 version



```
ubuntu@demeter-dev-2:~/docker_test$ docker --tlsverify --tlscacert=./ca.pem --tlscert=./cert.pem --tlskey=key.pem -H=demeter-dev-1:2376 version
Client:
 Version:           19.03.8
 API version:       1.40
 Go version:        go1.13.8
 Git commit:        afacb8b7f0
 Built:             Wed Oct 14 19:43:43 2020
 OS/Arch:           linux/amd64
 Experimental:      false

Server:
 Engine:
  Version:          19.03.8
  API version:      1.40 (minimum version 1.12)
  Go version:       go1.13.8
  Git commit:       afacb8b7f0
  Built:            Wed Oct 14 16:41:21 2020
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.3.3-0ubuntu2
  GitCommit:
 runc:
  Version:          spec: 1.0.2-dev
  GitCommit:
 docker-init:
  Version:          0.18.0
  GitCommit:
```

Figure 58: DEH Client CLI Interaction with Remote Docker Daemon Docker Host.

The above screenshot depicts how Docker Client CLI can access Docker Daemon service running on remote Docker Host.

- With Docker Daemon running with tlsverify flag set with a trusted signed digital certificate, Docker Host Daemon will accept connections from clients authenticated by a certificate signed by that CA. Thus, trust is established between Docker Host and Client, and flow is outlined as below:
- Docker daemon (Docker Host) configured to allow connections from clients with a trusted certificate.
- DEH Client sends a request to access container information along with certificate.

- So, all communications between the DEH Client and the Docker Host must be accompanied via a signed trusted digital certificate.
- Once secured communication is established DEH Client can assess the information of Containers running on Docker Host.

10.3.1.1.1 Functionality description

This section outlines the core functionalities of DEH Client Module. The features list as follows. Resource consumption metrics.

Resource consumption metrics:
DEH Client Module's core functionality is to monitor DEH Service Containers deployed and running on Docker Host, generate resource consumption metrics, and report back with metrics data to DEH RRM. The following attributes are tracked as part of metrics data for all the running containers.

- Volume: Memory Usage & CPU Usage.
- Time Usage: Duration of running containers. (Container uptime)

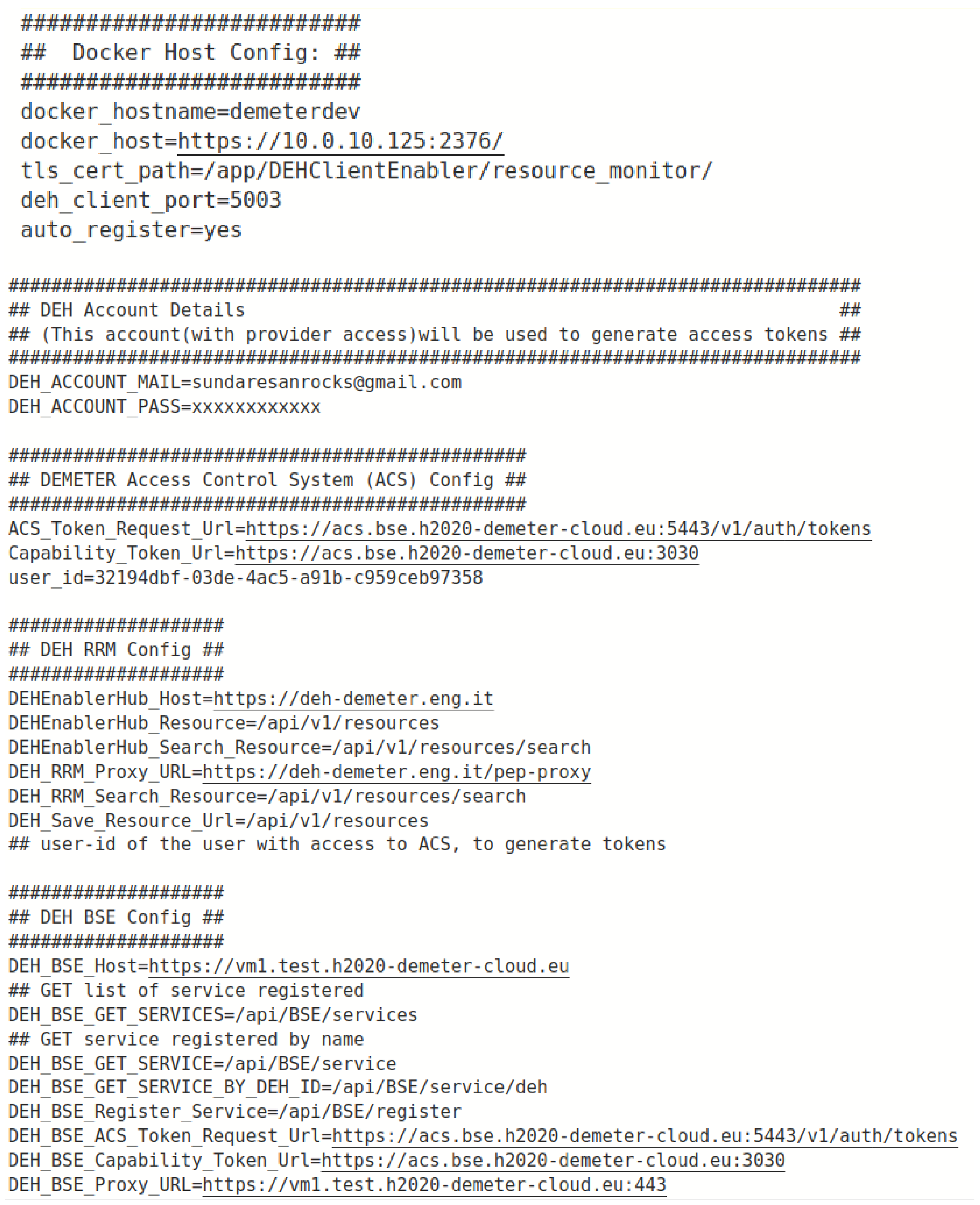
Auto registration of container information with DEH RRM and BSE:

This section outlines another core functionality of DEH Client Module which is to establish an interface with RRM (DEH Resource Registry Management) and BSE (Brokerage Service Environment). As part of DEH resource registry policy, all DEH Service Container deployed & their metadata needs to be registered with DEH RRM and BSE. This is done manually using respective API's or from DEH Dashboard which is tedious and time-consuming.
For reliability, seamless operational flow and to reduce time on the manual effort needed in gathering metadata on running containers and register with DEH RRM & BSE, a key feature DEH Client Module offers is to expose a configurable option (flag auto_register=yes) for the user to enable Client Module to auto-registration of container information. With this option enabled, DEH Client when discovering any new container, automatically attempts to registers the same with DEH RRM & BSE internally using corresponding API's. Refer to Resource Discovery functionality on how DEH Client Modules can discover containers automatically when deployed.

Currently, DEH Client Module interacts with DEH RRM & BSE API for a couple of reasons.

- GET request to DEH RRM & BSE, to Request for resource registration data if the container is already registered. Their corresponding registration IDs will be included in the metrics report sent to RRM.
- POST request to DEH RRM & BSE, if auto_register = yes gather container information and register. Once registration is successful their corresponding registration IDs will be included in the metrics report sent to RRM.

For more configurability and to give users more control over the application, DEH Client exposes an environment file where users can configure DEH Client to use cloud instances of DEH RRM and BSE or use their instance (running as Docker Containers).



```
##########################
##  Docker Host Config: ##
##########################
docker_hostname=demeterdev
docker_host=https://10.0.10.125:2376/
tls_cert_path=/app/DEHClientEnabler/resource_monitor/
deh_client_port=5003
auto_register=yes

#############################################################################
## DEH Account Details                                                     ##
## (This account(with provider access)will be used to generate access tokens ##
#############################################################################
DEH_ACCOUNT_MAIL=sundaresanrocks@gmail.com
DEH_ACCOUNT_PASS=xxxxxxxxxxxx

################################################
## DEMETER Access Control System (ACS) Config ##
################################################
ACS_Token_Request_Url=https://acs.bse.h2020-demeter-cloud.eu:5443/v1/auth/tokens
Capability_Token_Url=https://acs.bse.h2020-demeter-cloud.eu:3030
user_id=32194dbf-03de-4ac5-a91b-c959ceb97358

####################
## DEH RRM Config ##
####################
DEHEnablerHub_Host=https://deh-demeter.eng.it
DEHEnablerHub_Resource=/api/v1/resources
DEHEnablerHub_Search_Resource=/api/v1/resources/search
DEH_RRM_Proxy_URL=https://deh-demeter.eng.it/pep-proxy
DEH_RRM_Search_Resource=/api/v1/resources/search
DEH_Save_Resource_Url=/api/v1/resources
## user-id of the user with access to ACS, to generate tokens

####################
## DEH BSE Config ##
####################
DEH_BSE_Host=https://vm1.test.h2020-demeter-cloud.eu
## GET list of service registered
DEH_BSE_GET_SERVICES=/api/BSE/services
## GET service registered by name
DEH_BSE_GET_SERVICE=/api/BSE/service
DEH_BSE_GET_SERVICE_BY_DEH_ID=/api/BSE/service/deh
DEH_BSE_Register_Service=/api/BSE/register
DEH_BSE_ACS_Token_Request_Url=https://acs.bse.h2020-demeter-cloud.eu:5443/v1/auth/tokens
DEH_BSE_Capability_Token_Url=https://acs.bse.h2020-demeter-cloud.eu:3030
DEH_BSE_Proxy_URL=https://vm1.test.h2020-demeter-cloud.eu:443
```

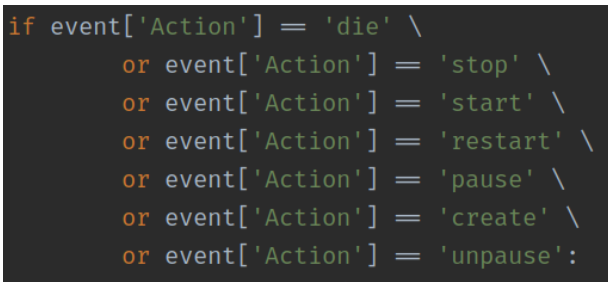
Figure 59: DEH Client Configurable Parameters.

Figure 59 depicts a set of configurable parameters including Docker Host, auto_register (which will enable or disable auto registration with DEH RRM & BSE), ACS, DEH RRM & BSE instances, and their corresponding methods.

Resource Discovery:

Another interesting feature/component of the DEH Client is Resource Discovery. Once secured communication is established between DEH Client and remote Docker Host, DEH Client Module can monitor resource information and Docker Events on Docker Host.

Resource discovery components constantly monitor for docker events on Docker Host, this enables the client to automatically detects any new containers deployed and adds the same to the monitoring list. For effective utilisation of resources, the client only monitors and generates resource consumption metrics for those running containers and sends data to RRM periodically. The following Figure 60 depicts the list of Docker Events monitored.



```
if event['Action'] == 'die' \
        or event['Action'] == 'stop' \
        or event['Action'] == 'start' \
        or event['Action'] == 'restart' \
        or event['Action'] == 'pause' \
        or event['Action'] == 'create' \
        or event['Action'] == 'unpause':
```

Figure 60: DEH Client Monitored Docker Events.

DEH Client also exposes an API that will enable users to list DEH Service Containers deployed on Docker Host by placing the appropriate filters. The filters include:

- Filter by container name, string pattern match.

```
GET
Request:  "http://172.17.0.5:5003/api/v1/DEHClientEnabler/ResourceConsumption/get_resources_filter?name=estimate"
Response :
[
  "estimate-animal-welfare-condition-ws01",
  "estimate-animal-welfare-condition-ws1",
  "estimate-animal-welfare-condition-demo11"
]
```

- Filter by containers current status (running, paused, or exited).

```
GET
Request:  "http://172.17.0.5:5003/api/v1/DEHClientEnabler/ResourceConsumption/get_resources_filter?status=running"
Response :
[
  "estimate-animal-welfare-condition-ws01",
  "dehclient_ws01",
  "estimate-animal-welfare-condition-ws1",
  "estimate-animal-welfare-condition-demo11",
  "authorisation_capabilitymanager_1"
]
```

- Filter by Ancestor (Instance of a specific image):

```
GET
Request:  "http://172.17.0.5:5003/api/v1/DEHClientEnabler/ResourceConsumption/get_re
sources_filter?ancestor=demeterengteam/estimate-animal-welfare-condition:candidate"
Response :
[
  "estimate-animal-welfare-condition-ws01",
  "estimate-animal-welfare-condition-ws1",
  "estimate-animal-welfare- condition-ws2
]
```

10.3.1.1.2 Interaction with other Enablers

This section elaborates on DEH Client Module's interaction with other DEH Core Enablers with architecture and UML sequence diagrams.

DEH Client Module interacts with three core components of DEH namely DEMETER Access Control System (ACS - Access Control Enable), DEH RRM (Resource Registration Management), and BSE (Brokerage Service Environment). Communication between these is established through their respective application programming interfaces (API's). For DEH Client Module, the purpose of interaction currently is limited to resource registry management.

Identity Manager i.e., DEMETER Access Control System (ACS - Access Control Enable) is used to ensure authentication and authorisation for Resource Registration Management component i.e., DEH RRM & BSE API's. This Security Authentication Enabler is needed to perform the authentication functionalities and ensure interaction with other modules over a secured communication channel.

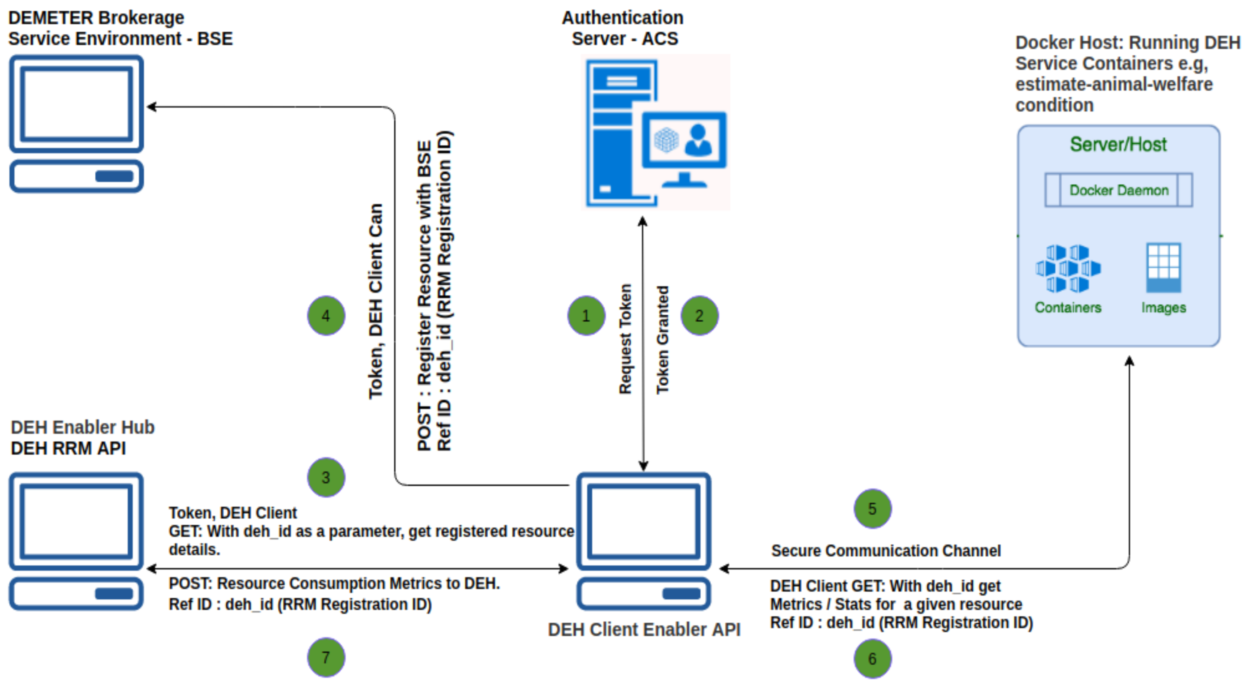

DEMETER Brokerage Service Environment - BSE
Authentication Server - ACS
Docker Host: Running DEH Service Containers e.g, estimate-animal-welfare condition
Server/Host
Docker Daemon
Containers
Images
4
Token, DEH Client Can
POST : Register Resource with BSE Ref ID : deh_id (RRM Registration ID)
1
Request Token
2
Token Granted
DEH Enabler Hub DEH RRM API
3
Token, DEH Client GET: With deh_id as a parameter, get registered resource details.
POST: Resource Consumption Metrics to DEH. Ref ID : deh_id (RRM Registration ID)
7
DEH Client Enabler API
5
Secure Communication Channel
DEH Client GET: With deh_id get Metrics / Stats for a given resource Ref ID : deh_id (RRM Registration ID)
6

Figure 61: DEH Client Architecture Diagram.

Figure 61 below depicts an architecture diagram illustrating DEH Client Modules' interaction with other DEH core enablers/ modules.

This section elaborates on data flow and interaction between DEH Client and other DEH Core Enablers:

- Step1: DEH Client requests for authentication Token from ACS (Access Control System) to have API level access to BSE and RRM.
- Step2: Authentication Server: Token granted. (Client can use Token to communicate with DEH Core Enablers - RRM and BSE).
- Step3: DEH Client, with valid Token and deh_id, get registered resource details from DEH RRM using RRM API GET call. Reference ID: uid/ deh_id.
- Step4: DEH Client, with valid Token and deh_id, register resource with BSE using BSE API POST call. Reference ID: uid/ deh_id.
- Step5: DEH Client: Communicate with Docker Host over a secured channel.
- Step6: DEH Client: For resource requested, generate Resource Consumption Metrics/Stats. Reference ID: uid/ deh_id.
- Step7: DEH Client: Reports backs to DEH RRM with the Resource Consumption Stats periodically. Reference ID: uid/ deh_id.

DEH Client Module - UML sequence is as follows:

Figure 62 below depicts UML sequence diagram showing interactions between DEH Client components.

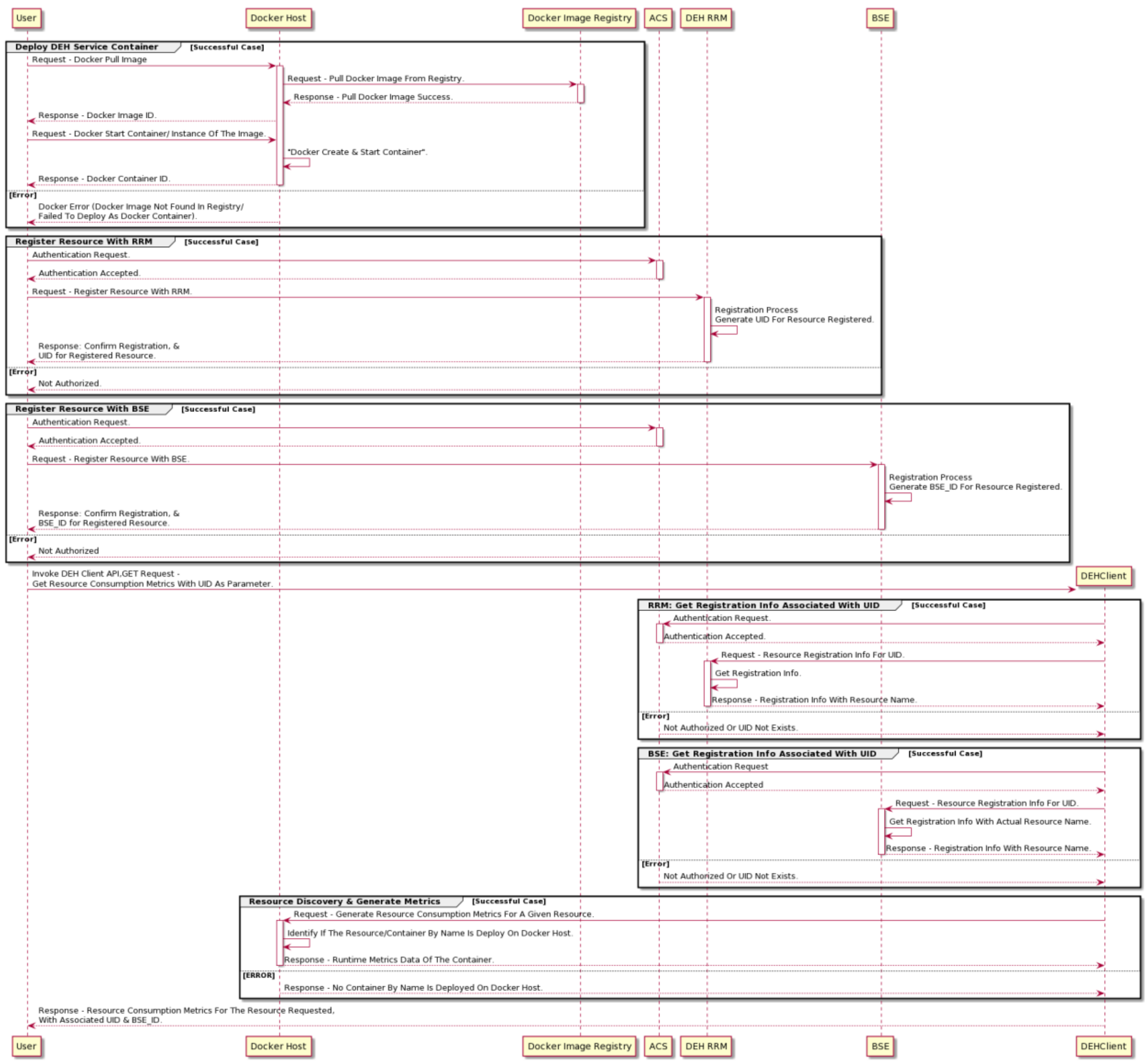

User
Docker Host
Docker Image Registry
ACS
DEH RRM
BSE
Deploy DEH Service Container
[Successful Case]
Request - Docker Pull Image
Request - Pull Docker Image From Registry.
Response - Pull Docker Image Success.
Response - Docker Image ID.
Request - Docker Start Container/ Instance Of The Image.
"Docker Create & Start Container".
Response - Docker Container ID.
[Error]
Docker Error (Docker Image Not Found In Registry/ Failed To Deploy As Docker Container).
Register Resource With RRM
[Successful Case]
Authentication Request.
Authentication Accepted.
Request - Register Resource With RRM.
Registration Process Generate UID For Resource Registered.
Response: Confirm Registration, & UID for Registered Resource.
[Error]
Not Authorized.
Register Resource With BSE
[Successful Case]
Authentication Request.
Authentication Accepted.
Request - Register Resource With BSE.
Registration Process Generate BSE_ID For Resource Registered.
Response: Confirm Registration, & BSE_ID for Registered Resource.
[Error]
Not Authorized
Invoke DEH Client API.GET Request - Get Resource Consumption Metrics With UID As Parameter.
DEHClient
RRM: Get Registration Info Associated With UID
[Successful Case]
Authentication Request.
Authentication Accepted.
Request - Resource Registration Info For UID.
Get Registration Info.
Response - Registration Info With Resource Name.
[Error]
Not Authorized Or UID Not Exists.
BSE: Get Registration Info Associated With UID
[Successful Case]
Authentication Request
Authentication Accepted
Request - Resource Registration Info For UID.
Get Registration Info With Actual Resource Name.
Response - Registration Info With Resource Name.
[Error]
Not Authorized Or UID Not Exists.
Resource Discovery & Generate Metrics
[Successful Case]
Request - Generate Resource Consumption Metrics For A Given Resource.
Identify If The Resource/Container By Name Is Deploy On Docker Host.
Response - Runtime Metrics Data Of The Container.
[ERROR]
Response - No Container By Name Is Deployed On Docker Host.
Response - Resource Consumption Metrics For The Resource Requested, With Associated UID & BSE_ID.
User
Docker Host
Docker Image Registry
ACS
DEH RRM
BSE
DEHClient

Figure 62: DEH Client UML Sequence Diagram

The below section describes the sequence flow in detail with corresponding action and response. Each action needs to be performed by ICT expert to be complete. An action corresponds with an operation using a DEH Client tool; the ICT expert at the end of each action will receive a confirmation message by the Module if the specific action gives him a good result.

1. Pilot Action (step 1) " Action Docker configuration":
   Set-up: On DockerHost, Request pull image from the registry
   (e.g., demeterengteam/estimate-animal-welfare-condition:candidate)
   Response: Image pulled successfully to Docker Host.

2. Pilot Action (step 2) " Action Docker configuration":

Set-up: On DockerHost, Create/ Start a container (Instance of Image).
(e.g., container name: estimate-animal-welfare-pilot1)
Response: Response: Container successfully created/ started on Docker Host.

3. Pilot Action (step 3) " Action Registry Management":
   Set-up: Register container details with DEH RRM.
   Response:
   RRM, with valid container information registers and responds with a unique identifier i.e., UID. This uid will be used as a unique identifier of a container across DEH. Also, the Pilot uses the uid to track or generate container resource consumption metrics. e.g., container estimate-animal-welfare-pilot1 is registered with DEH RRM.

4. Pilot Action (step 4) :" Invoke DEH Client API": To generate metrics of the given container.
   Set-up: Issue DEH Client API GET Request with UID as parameter to generate resource consumption metrics.

DEH Client Internal flow: Internal DEH Client Event.
DEH Client Internal flow below represents the internal sequence of events triggered by Step 4 of Pilot action. Each event sequence is associated with an expected outcome, any event failure will be reported back with appropriate error as a response for Pilot Action 4.

- Event sequence 1) DEH Client Action: Request for Authorisation Token from ACS to access other DEH module i.e., DEH RRM & BSE.
  Expected Outcome: ACS, for authorised user request from DEH Client generates and issues token to interact with DEH RRM and BSE.
  The client uses this token for further communicate with DEH RRM & BSE API's.

- Event sequence 2) DEH Client: With valid token and UID, request DEH RRM for associated container information.
  Expected Outcome: With valid UID, RRM responds backs with the associated container information.
- Event sequence 3) DEH Client: With valid Token and UID, request BSE for associated container information.
  Expected Outcome: With valid UID, BSE responds backs with the associated container information.
- Event sequence 4) DEH Client: Communicate with Docker Host over a secured channel, does resource discovery i.e.,
  check if the container by name is hosted on the configured Docker Host.
  Expected Outcome: For resource requested, generate Resource Consumption Metrics.
- Event sequence 5)DEH Client: Reports backs with the Resource Consumption Stats.
  Reference ID: uid.

Response: (Response for Setup Pilot Action 4)
RRM, with valid container information registers and responds with a unique identifier i.e., UID. This uid will be used as a unique identifier of a container across DEH. Also, the Pilot uses the uid to track or generate container resource consumption metrics. e.g., container estimate-animal-welfare-pilot1 is registered with DEH RRM.

10.3.1.1.3 Deployment considerations

This section outlines the DEH Client module's configurable parameters, pre-requisite user actions and step-by-step instruction on deploying DEH Client module as a Docker Container.
The container image of DEH Client module will be available via DEMETER's Image Registry. It is freely available to all DEMETER consortium and the requirements are minimal, i.e., OS capable of running docker containers, docker service up and running. Registry and Image details below:

Registry: registry.gitlab.com/demeterproject/wp3/demeterenablerhub/dehclient:v1

Image : dehclient:v1

Configuration Parameters:
The below table represents the configurable parameters (.env file). These are mandatory configuration parameters and needs to update in environment file before deploying DEH Client Module as a container.

Table 27: DEH Client Module Configurable Parameters

Configuration parameter	Value	Type	Description
docker_hostname	Defined by user.	String	Docker Host Name
docker_host	Defined by user.	Number	Docker Host Engine API URL. e.g., https://10.0.10.125:2376/
deh_client_port	Defined by user.	Number	DEH Client Port
auto_register	Defined by user.	String	Flag (yes/no) tells Modules to auto-register container data with DEH RRM & BSE
DEH_ACCOUNT_MAIL	Defined by user.	String	DEH account (with provider access) email, will be used to generate access ACS tokens.
DEH_ACCOUNT_PASS	Defined by user.	String	DEH account (with provider access) password, will be used to generate access ACS tokens.
DEHEnablerHub_Host	Defined by user.	String	DEH RRM endpoint, can be configured to use the cloud instance or user instance e.g., https://deh-demeter.eng.it
ACS_Token_Request_Url	Defined by user.	String	Identity Manager - ACS instance e.g., https://acs.bse.h2020-demeter-cloud.eu:5443/v1/auth/tokens
Capability_Token_Url	Defined by user.	String	Capability Manager endpoint e.g., https://acs.bse.h2020-demeter-cloud.eu:3030
DEH_RRM_Proxy_URL	Defined by user.	String	DEH RRM pep proxy url e.g., https://deh-demeter.eng.it/pep-proxy
user_id	Defined by user.	String	DEH account user-id, account with provider access and will be used to

Configuration parameter	Value	Type	Description
			generate access tokens.
DEH_BSE_Host	Defined by user.	String	BSE endpoint, can be configured to use the cloud instance or user instance e.g., https://vm1.test.h2020-demeter-cloud.eu
DEH_BSE_ACS_Token_Request_Url	Defined by user.	String	Identity Manager - ACS instance e.g., https://acs.bse.h2020-demeter-cloud.eu:5443/v1/auth/tokens
DEH_BSE_Capability_Token_Url	Defined by user.	String	Capability Manager endpoint e.g., https://acs.bse.h2020-demeter-cloud.eu:3030
DEH_BSE_Proxy_URL	Defined by user.	String	BSE pep proxy url e.g., https://vm1.test.h2020-demeter-cloud.eu:443

Also refer to Figure 59 DEH Client Configurable Parameters, for a sample .env file with possible parameters.

Pre-Requirements:

Below are the pre-requirements to be satisfied before running DEH Client Module in a Pilot environment as a docker container.

- DEH Client: Docker pull image DEH Client from registry.
- .env: Update the environment file with related attributes/ configurable parameter as shown in Figure 59, before starting DEH Client.
- Docker Host: For secured communication, configure Docker daemon to only accept connections from clients providing a trusted certificate and port. These are self-signed certificates generated using openssl package.
  Note: Please refer to the Securing Docker Engine API[15] document (accessible only for DEMETER consortium at the moment) for detailed info on how to generate the certificates and configure
- DEH Enabler/s or Containers running on Docker Host.
- User Account: With privileges to get ACS tokens & access BSE & RRM API's.
- DEH RRM & BSE: Either use corresponding cloud or local instances.

Deployment Instructions:

The following section outlines step by step installation instructions for the DEH Client Module.

- Login into the registry

  $ sudo docker login registry.gitlab.com

- Provide your credentials

[15] Securing Docker Engine API: https://gitlab.com/demeterproject/wp3/demeterenablerhub/dehclient/-/blob/master/Securing_Docker_Engine_API.pdf

Successful Login Response: Login Succeeded

- Pull the image from the registry.

  $ docker image pull
  registry.gitlab.com/demeterproject/wp3/demeterenablerhub/dehclient:v1

- Create container, supplying env variables as a file:

  $ docker create --name <<"ContainerName">> --env-file <<"env file name">>
  registry.gitlab.com/demeterproject/wp3/demeterenablerhub/dehclient:v1

- Copy relevant Client certificates to the container before starting the container.

  Note: Please refer to the document "Securing Docker Engine API.pdf" to establish a secure connection between Docker Client & Docker Host.

  https://gitlab.com/demeterproject/wp3/demeterenablerhub/dehclient/-/blob/master/Securing_Docker_Engine_API.pdf

  $ docker cp <<"CertPath">>/ca.pem
  <<"ContainerName">>:/app/DEHClientEnabler/resource_monitor

  $ docker cp <<"CertPath">>/key.pem
  <<"ContainerName">>:/app/DEHClientEnabler/resource_monitor

  $ docker cp <<"CertPath">>/cert.pem
  <<"ContainerName">>:/app/DEHClientEnabler/resource_monitor

- Start container (-i if you want to use interactive mode)

  $ sudo docker start -i <<"ContainerName">>.

10.3.1.1.4 Technical description

This section summarises the technologies, tools and frameworks linked to the implementation of DEH Client Module.

The following table summarises resources linked to the DEH Client Module

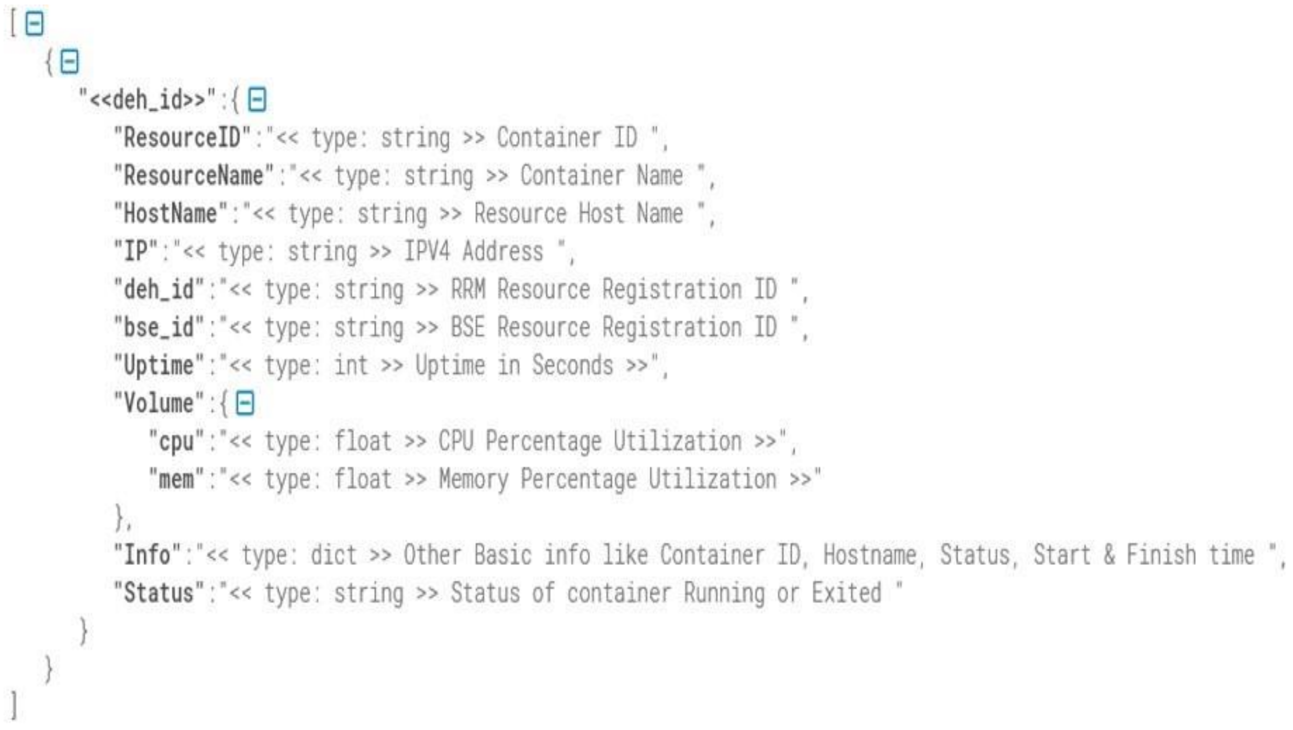
Flask web framework	https://pypi.org/project/Flask/
Docker	https://docs.docker.com/get-docker/

10.3.1.1.5 API and Data model

This section below describes the Data Model used with DEH Client Model.

Table 28: DEH Client Module Resource Consumption Metrics Data Model

Name	DEH Client generated resource consumption metrics data model	
Property	Type	Description
_id	string	RRM Resource Registration ID
ResourceID	string	Container ID
ResourceName	string	Container Name
HostName	string	Container Host Name
IP	string	Container IPV4 address
deh_id	string	RRM Resource Registration ID
bse_id	string	BSE Resource Registration ID
Uptime	integer	Uptime in Seconds
Volume	dictionary	cpu & mem utilisation data.
Info	dictionary	Other Basic info like Container ID, Hostname, Status, Start & Finish time
Status	string	Current status of container like Running or Exited.



```
[⊟
  {⊟
    "<<deh_id>>":{⊟
      "ResourceID":"<< type: string >> Container ID ",
      "ResourceName":"<< type: string >> Container Name ",
      "HostName":"<< type: string >> Resource Host Name ",
      "IP":"<< type: string >> IPV4 Address ",
      "deh_id":"<< type: string >> RRM Resource Registration ID ",
      "bse_id":"<< type: string >> BSE Resource Registration ID ",
      "Uptime":"<< type: int >> Uptime in Seconds >>",
      "Volume":{⊟
        "cpu":"<< type: float >> CPU Percentage Utilization >>",
        "mem":"<< type: float >> Memory Percentage Utilization >>"
      },
      "Info":"<< type: dict >> Other Basic info like Container ID, Hostname, Status, Start & Finish time ",
      "Status":"<< type: string >> Status of container Running or Exited "
    }
  }
]
```

Figure 63: DEH Client Module Resource Consumption Metrics Data Model

Figure 63 depicts the DEH Client Module's generated Resource Consumption Metrics data model.

API description : Resource Consumption Metrics

The below table gives an overview of DEH Client Modules exposed API's, which enables users to generate runtime resource consumption metrics of resources/containers deployed on a configured Docker Host.

Title	GET resource consumption metrics by resource/ container name or uid (RRM registration ID).
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
/api/v1/DEHClientEnabler/ResourceConsumption/individual/metrics	
Method This field holds the type of the Method used.	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required: name or uid <<User can generate metrics using container name or by using uid (RRM registration ID)>>	
name Or uid	Name of the resource / container deployed on the docker host, for which resource consumption metrics are to be generated. Uid : unique registration id generated for resources registered with DEH RRM
Optional:	
NA	NA
Data Params: NA	
Required:	
NA	NA
Optional:	
NA	NA
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	

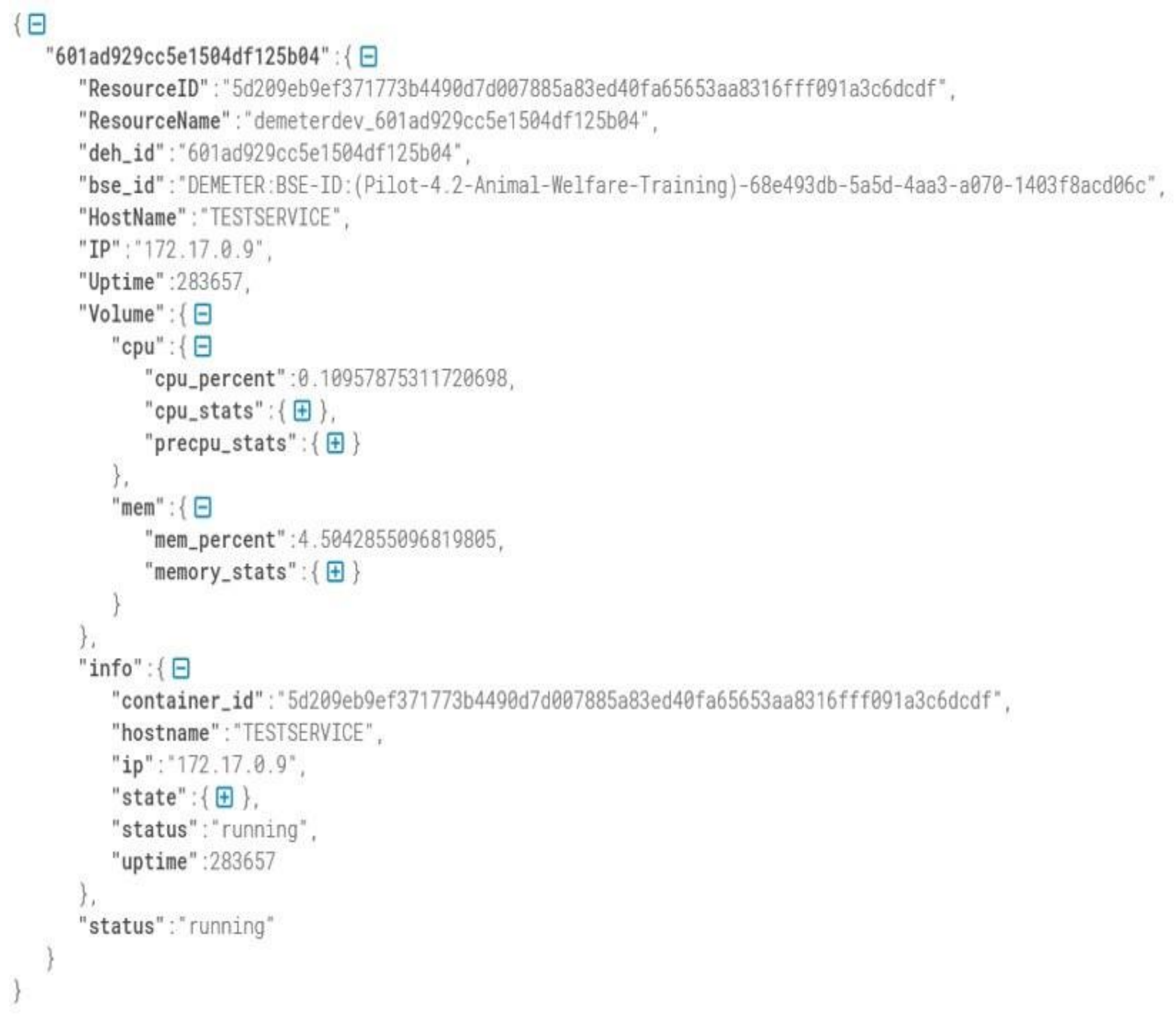
200
Content: {
 "**605f580337801e241cf995ec**": {
 "**BSE_ID**": "DEMETER:BSE-ID:(estimate-animal-welfare-condition9)-9eb2face-45a4-416f-8241-dd8a33cc74cb",
 "HostName": "9c84b82b1e3f",
 "IP": "172.17.0.22",
 "RRM_ID": "605f580337801e241cf995ec",
 "ResourceID": "9c84b82b1e3f2a4517161b39710275be2dffecd74e934171f090e2c10ce10388",
 "ResourceName": "estimate-animal-welfare-condition9",
 "ServiceID": "",
 "Uptime": 428457,
 "Volume": {
 "cpu": { "cpu_percent": 0.07610460784313726,
 "cpu_stats": {
 "cpu_usage": {
 "percpu_usage": [98684140988, 100850488718, 101182150986, 99365285059],
 "total_usage": 400082065751,
	DEH Clients Steps:
	Case 1: GET Request with name
	User request for resource consumption metrics of a resource/ container deployed on configured Docker Host, via DEH Client GET metrics request.
	DEH Client communicates with Docker Host over a secured channel and check if the given resource / container by name is deployed on the Docker Host.
	If resource / container found, generate resource consumption metrics.
	Case 2: GET Request with UID (RRM registration ID)
	Pre-requirement: User manually deploys a container on the Docker Host and registers the same with RRM via DEH Dashboard or RRM API's.
	Note: in this case the container name (for which you intend to generate resource consumption metrics) should match with the name used to register with RRM.

```
        "usage_in_kernelmode": 160100000000,
        "usage_in_usermode": 195890000000
      },
      "online_cpus": 4,
      "system_cpu_usage": 89341210230000000,
      "throttling_data": {
        "periods": 0,
        "throttled_periods": 0,
        "throttled_time": 0
      }
    },
    "precpu_stats": {
      "cpu_usage": {
        "percpu_usage": [ 98683938352,
          100850488718, 101181577355,
          99365285059 ],
        "total_usage": 400081289484,
        "usage_in_kernelmode": 160100000000,
        "usage_in_usermode": 195890000000
      },
      "online_cpus": 4,
      "system_cpu_usage": 89341206150000000,
      "throttling_data": {
        "periods": 0,
        "throttled_periods": 0,
        "throttled_time": 0
      }  }  },
  "mem": {
    "mem_percent": 2.987636866289207,
    "memory_stats": {
      "limit": 6234546176,
      "max_usage": 274284544,
      "stats": {
        "active_anon": 91013120,
        "active_file": 77824,
        "cache": 253952,
        "dirty": 0,
        "hierarchical_memory_limit":
9223372036854772000,
        "hierarchical_memsw_limit": 0,
        "inactive_anon": 91189248,
        "inactive_file": 126976,
        "mapped_file": 270336,
        "pgfault": 1119129,
        "pgmajfault": 35046,
        "pgpgin": 435666,
        "pgpgout": 391159,
        "rss": 182521856,
        "rss_huge": 0,
        "total_active_anon": 91013120,
        "total_active_file": 77824,
```
	User request for resource consumption metrics for a resource/ container deployed on configured Docker Host, via DEH Client GET metrics request.
	DEH Client communicates with DEH RRM & fetches associated resource details like resource / container name from RRM.
	DEH Client communicates with Docker Host over a secured channel and checks if the obtained container by name is deployed on the Docker Host.
	If resource / container is found, generate resource consumption metrics.

```
        "total_cache": 253952,
        "total_dirty": 0,
        "total_inactive_anon": 91189248,
        "total_inactive_file": 126976,
        "total_mapped_file": 270336,
        "total_pgfault": 1119129,
        "total_pgmajfault": 35046,
        "total_pgpgin": 435666,
        "total_pgpgout": 391159,
        "total_rss": 182521856,
        "total_rss_huge": 0,
        "total_unevictable": 0,
        "total_writeback": 135168,
        "unevictable": 0,
        "writeback": 135168
       },
       "usage": 186265600
      } } },
   "info": {
    "container_id":
"9c84b82b1e3f2a4517161b39710275be2dffecd74e9341
71f090e2c10ce10388",
    "hostname": "9c84b82b1e3f",
    "ip": "172.17.0.22",
    "state": {
     "Dead": false,
     "Error": "",
     "ExitCode": 0,
     "FinishedAt": "0001-01-01T00:00:00Z",
     "OOMKilled": false,
     "Paused": false,
     "Pid": 251254,
     "Restarting": false,
     "Running": true,
     "StartedAt": "2021-03-27T16:02:41.819820345Z",
     "Status": "running"
    },
    "status": "running",
    "uptime": 428457
   },
   "last_updated": "2021-04-01 15:03:27 UTC+0000",
   "status": "running"
  }
}
```

Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
501 { "Reason" : "Resource/ Container with name: <<Container Name>> not hosted on Docker Host: <<Docker Host Name>>"} 404	501 --> When DEH Client is not able find the requested resource/ container by name deployed with Docker Host. 404 --> When DEH Client fails to

{ "Reason1" : "Resource with uid/ deh_id :<<uid>> not registered with DEH RRM, Please register the resource with RRM via DEH dashboard or RRM API's to generate metrics.", "Reason2":"Possibly, failed to communicate with DEH RRM. Please check."}	communicate with Docker, possibly misconfiguration. Or When DEH Client could not find uid with RRM or uid is invalid.
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
Case 1: GET request, request for resource consumption metrics by name. Request: curl -X GET http://<<DEH Client Container IP>>:<<PORT>>/api/v1/DEHClientEnabler/ResourceConsumption/individual/metrics?name=<<Resource / Container Name>>. Case 2: GET request, request for resource consumption metrics by uid. curl -X GET http://<<DEH Client Container IP>>:<<PORT>>/api/v1/DEHClientEnabler/ResourceConsumption/individual/metrics?uid=<<uid/ deh_id - RRM Registration ID>>.	
Notes This field holds any additional helpful info related to this endpoint.	



```
{⊟
   "601ad929cc5e1504df125b04":{⊟
      "ResourceID":"5d209eb9ef371773b4490d7d007885a83ed40fa65653aa8316fff091a3c6dcdf",
      "ResourceName":"demeterdev_601ad929cc5e1504df125b04",
      "deh_id":"601ad929cc5e1504df125b04",
      "bse_id":"DEMETER:BSE-ID:(Pilot-4.2-Animal-Welfare-Training)-68e493db-5a5d-4aa3-a070-1403f8acd06c",
      "HostName":"TESTSERVICE",
      "IP":"172.17.0.9",
      "Uptime":283657,
      "Volume":{⊟
         "cpu":{⊟
            "cpu_percent":0.10957875311720698,
            "cpu_stats":{⊞},
            "precpu_stats":{⊞}
         },
         "mem":{⊟
            "mem_percent":4.5042855096819805,
            "memory_stats":{⊞}
         }
      },
      "info":{⊟
         "container_id":"5d209eb9ef371773b4490d7d007885a83ed40fa65653aa8316fff091a3c6dcdf",
         "hostname":"TESTSERVICE",
         "ip":"172.17.0.9",
         "state":{⊞},
         "status":"running",
         "uptime":283657
      },
      "status":"running"
   }
}
```

Figure 64: DEH Client Module Generated Real Time Resource Consumption Metrics Data

Figure 64 depicts DEH Client Module generated resource consumption metrics data of a given container. It showcases only values which are being used at this point in time by other DEH submodules (DEH RRM).

Resource Discovery:
The below table gives an overview of DEH Client Modules exposed API's, which enables users to discover resources/containers deployed on a configured Docker Host.

Title	Resource Discovery, get list of Containers matching filter.
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
/ResourceConsumption/get_resources_filter	
Method This field holds the type of the Method used.	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required: name or status or ancestor	

name Or Status Or ancestor	With filter by name, List of all Containers matching name pattern. Parameter: name (search pattern i.e., part of Container name) With filter by status, List of all Containers with given status. Parameter: status (running, exited, restarting, paused) With filter by ancestor, Get List of Containers which are instances of the given image. Parameter: ancestor (image name or id)
Optional:	
NA	NA
Data Params: NA	
Required:	
NA	NA
Optional:	
NA	NA
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200	
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	404 --> When DEH Client fails to communicate with Docker, possibly misconfiguration.
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that it is clear and easy to read by the interested parties.	
Case 1: GET request, Get list of Containers matching filter by name Request : curl -X GET http://<<DEH Client Container IP>>:<<PORT>>/api/v1/DEHClientEnabler/ResourceConsumption/get_resources_filter?name=<<Part Of Container Name>> Response: [<<List of all container names matching the filter criteria>>] **Case 2:** GET request, Get list of Containers matching filter by Status curl -X GET http://<<DEH Client Container IP>>:<<PORT>>/api/v1/DEHClientEnabler/ResourceConsumption/get_resources_filter?status=<<running || paused|| stopped>> Response: [<<List of all container names matching the filter criteria>>] **Case 3:** GET request, Get list of Containers matching filter by ancestor Request: curl -X GET http://<<DEH Client Container IP>>:<<PORT>>/api/v1/DEHClientEnabler /ResourceConsumption/get_resources_filter?ancestor=<<Image Name>> Response: [<<List of all container names matching the filter criteria>>]	
Notes This field holds any additional helpful info related to this endpoint.	

11 Infrastructure and Tools

In the context of DEMETER's Reference Implementation, the project's private infrastructure and toolbox have been setup and configured to assist the project operations, from code sharing and development to issue reporting and DevOps operations. The sections below describe the tools and infrastructure that have been used in DEMETER so far.

11.1 Tools in DEMETER

This environment comprises of several tools, which are described below.

11.1.1 Version control

GitLab has been selected to be used for managing source code and version control in DEMETER. GitLab is an open-source code management system based on Git, which includes a user management part that can be hosted online. DEMETER's code repository is using GitLab's online version where several private repositories have been created following the structure indicated by the partners involved. The group functionality offered by GitLab allows for code isolation, hence, to better accommodate privacy and IPR concerns among the consortium, subgroups have been defined where access is only granted to partners directly involved to the related component and task. In cases where public repositories are required, e.g., for public components, according to the Description of Action (DoA) commitments, source code that will be made public, and will, of course, be subject to licensing terms and conditions as agreed between the partners involved. GitLab provides the ability to allow access to external parties.

In addition, GitLab offers pipelines for automated integration and deployment processes. Pipelines describe sets of sequential continuous integration (CI) and continuous delivery (CD) operations. In this course, CI pipelines include code building followed by automated unit and integration tests while CD pipelines deploy the code to different environments, for reviewing purposes, for actual user testing (staging environment) and, finally, for production use (production environment). The above is depicted in Figure 65.

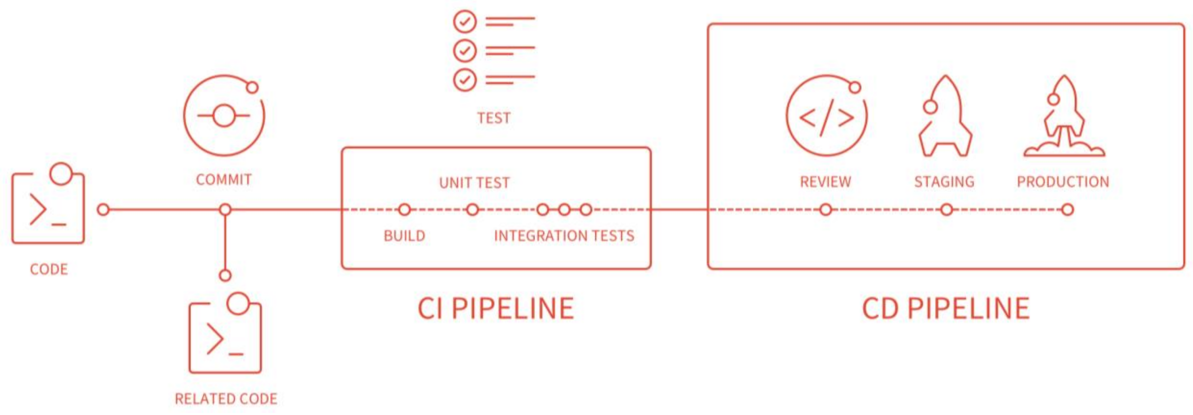


Figure 65: CI/CD operations at work

11.1.2 CI/CD

Continuous Integration (CI) is a developer practice to keep a target system up and running by making small incremental changes and integrating them frequently (usually at least daily)) on the mainline using appropriate tools supporting automation with lots of automated tests. This enables teams to work on shared code and increases the visibility into the development and quality of the system. By referring to a developer practice, Continuous Integration (CI) typically expects developers to implement Test-driven development (TDD) with constant refactoring practice. When a developer is unit-test-driving his code, he ensures that his local copy is always working.

Continuous Deployment (CD) refers to the automated deployment of new -release- versions of a system to the production environment. Following the continuous integration process, as described above, when a system reaches a maturity level (as indicated by specific, predefined criteria), the CD takes care of updating an existing running version of the system automatically, minimising downtime.

Combined, CI/CD is a pipeline that gets new developments and provides an updated running version of a system hosted in a predefined environment.

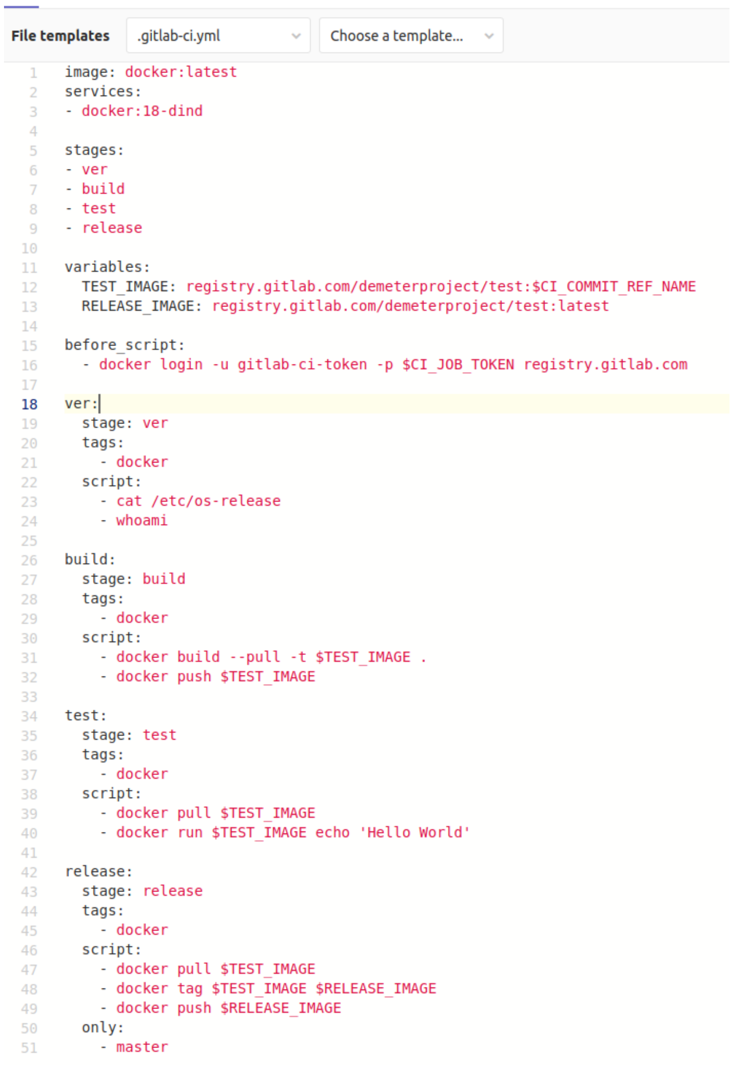
11.1.2.1 CI/CD pipelines

As mentioned above, Gitlab's CI/CD framework uses pipelines to automate integration and deployment processes. Such a pipeline is depicted in Figure 65.

Pipelines are defined and described in script files (.gitlab-ci.yml files, an example available in Figure 66), each of them representing a "job", including various pipelines organised in stages. Each job is assigned to a Gitlab runner to be executed. Gitlab runners are merely (client) Gitlab services that run on private or public infrastructure, connect to a public or private Gitlab instance, and execute the jobs described in the job files (building, testing, deploying). Runners execute the jobs in Docker containers while they also run as Docker containers, hence, in DEMETER we are using a Docker-in-Docker paradigm for the Runners we use. This enables us to achieve higher utilisation of our cloud infrastructure resources. Upon the execution of the jobs, GitLab offers a reporting tool to the developers to inspect all job stages.

Core functionalities of GitLab's CI/CD framework are listed below:

- Multiple projects are possible, grouped under groups and subgroups, allowing for organising source code and components according to the architectural blocks they belong to, or other criteria indicated by the partners.
- Private/public projects can be created, so components and source code can be publicly available if needed.
- GitLab provides branching, developing, testing, reviewing features to the development teams so that they can carry out their tasks in parallel before merging their work.
- Gitlab provides a private Image Registry where container images can be uploaded and used in e.g., Docker container deployments.

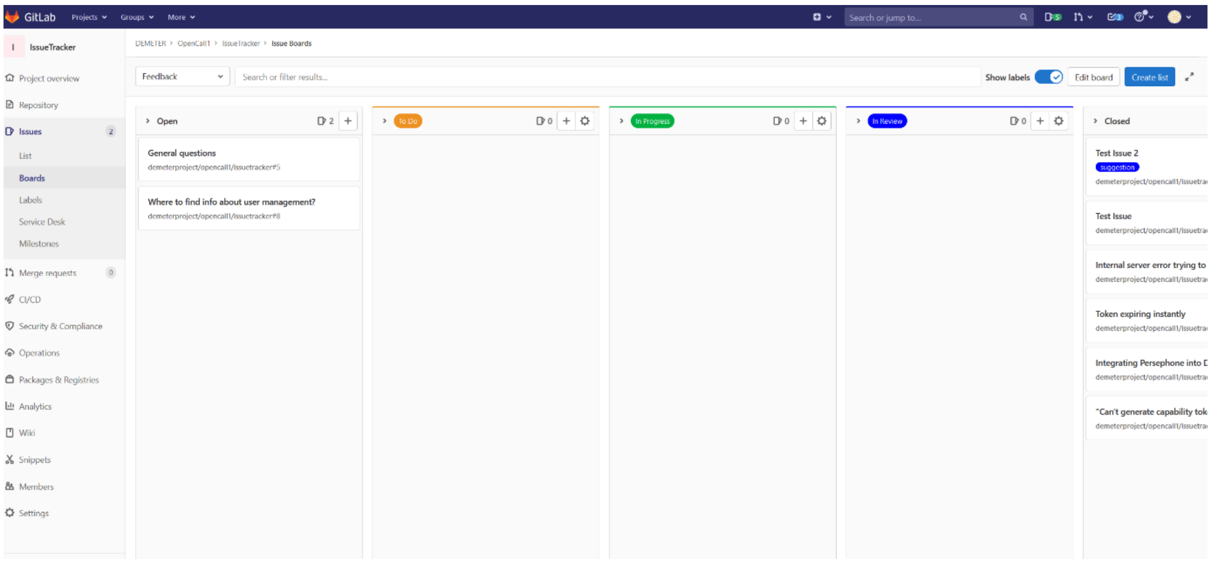


```yaml
image: docker:latest
services:
- docker:18-dind

stages:
- ver
- build
- test
- release

variables:
  TEST_IMAGE: registry.gitlab.com/demeterproject/test:$CI_COMMIT_REF_NAME
  RELEASE_IMAGE: registry.gitlab.com/demeterproject/test:latest

before_script:
  - docker login -u gitlab-ci-token -p $CI_JOB_TOKEN registry.gitlab.com

ver:
  stage: ver
  tags:
    - docker
  script:
    - cat /etc/os-release
    - whoami

build:
  stage: build
  tags:
    - docker
  script:
    - docker build --pull -t $TEST_IMAGE .
    - docker push $TEST_IMAGE

test:
  stage: test
  tags:
    - docker
  script:
    - docker pull $TEST_IMAGE
    - docker run $TEST_IMAGE echo 'Hello World'

release:
  stage: release
  tags:
    - docker
  script:
    - docker pull $TEST_IMAGE
    - docker tag $TEST_IMAGE $RELEASE_IMAGE
    - docker push $RELEASE_IMAGE
  only:
    - master
```

Figure 66: Example job file

11.1.3 Issue Trackers

GitLab offers an option to setup an issue tracker for projects. DEMETER leveraged on this functionality by using issue trackers at different levels. Since early in the project, Issue Trackers have been setup and configured for each of the technical WP (2/3/4), one for the pilots' WP, and one dedicated for the open call projects. In addition, to further improve visibility across inter-WP matters, issue trackers have been created at a coordination level, targeting more managerial aspects rather than technical issues. Figure 67 and Figure 68 show two different views of an issue tracker created for testing purposes: the Board view where the five "buckets" (named lists in GitLab's terminology) are depicted ("Open", "To Do", In Progress", "In Review", and "Closed"), and the List view where all non-closed issues are shown.



DEMETER > OpenCall1 > IssueTracker > Issue Boards

Open 2
- General questions
  demeterproject/opencall1/issuetracker#5
- Where to find info about user management?
  demeterproject/opencall1/issuetracker#8

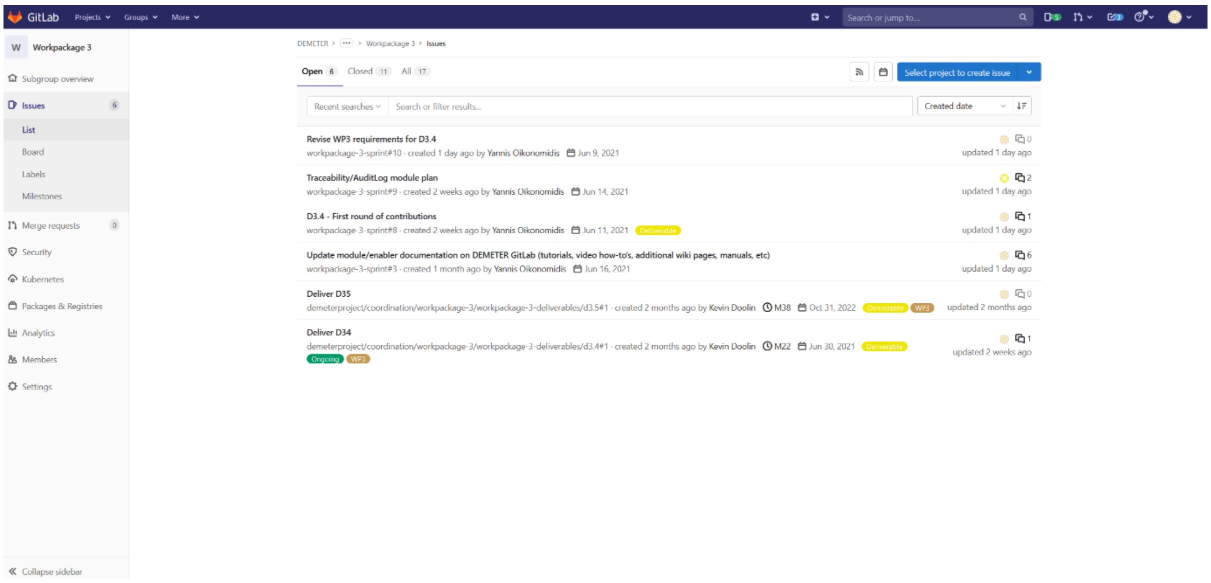
To Do 0

In Progress 0

In Review 0

Closed
- Test Issue 2
  suggestion
  demeterproject/opencall1/issuetra
- Test Issue
  demeterproject/opencall1/issuetra
- Internal server error trying to
  demeterproject/opencall1/issuetra
- Token expiring instantly
  demeterproject/opencall1/issuetra
- Integrating Persephone into D
  demeterproject/opencall1/issuetra
- "Can't generate capability tok
  demeterproject/opencall1/issuetra

Figure 67: Issue Tracker - Board view



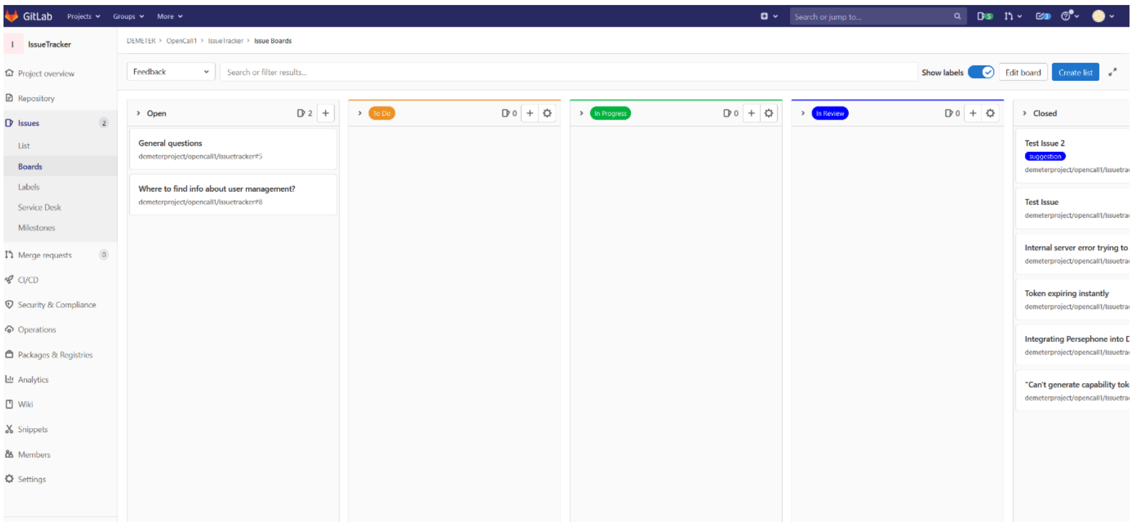
DEMETER > ••• > Workpackage 3 > Issues

Open 6 Closed 11 All 17

- Revise WP3 requirements for D3.4
  workpackage-3-sprint#10 · created 1 day ago by Yannis Oikonomidis Jun 9, 2021 — 0 — updated 1 day ago
- Traceability/AuditLog module plan
  workpackage-3-sprint#9 · created 2 weeks ago by Yannis Oikonomidis Jun 14, 2021 — 2 — updated 1 day ago
- D3.4 - First round of contributions
  workpackage-3-sprint#8 · created 2 weeks ago by Yannis Oikonomidis Jun 11, 2021 Deliverable — 1 — updated 1 day ago
- Update module/enabler documentation on DEMETER GitLab (tutorials, video how-to's, additional wiki pages, manuals, etc)
  workpackage-3-sprint#3 · created 1 month ago by Yannis Oikonomidis Jun 16, 2021 — 6 — updated 1 day ago
- Deliver D35
  demeterproject/coordination/workpackage-3/workpackage-3-deliverables/d3.5#1 · created 2 months ago by Kevin Doolin M38 Oct 31, 2022 Deliverable WP3 — 0 — updated 2 months ago
- Deliver D34
  demeterproject/coordination/workpackage-3/workpackage-3-deliverables/d3.4#1 · created 2 months ago by Kevin Doolin M22 Jun 30, 2021 Deliverable Ongoing WP3 — 1 — updated 2 weeks ago

Figure 68: Issue Tracker - List view

DEMETER GitLab offers an option to setup an issue tracker for projects. DEMETER leveraged on this functionality by using issue trackers at different levels. Since early in the project, Issue Trackers have been setup and configured for each of the technical WP (2/3/4), one for the pilots' WP, and one dedicated for the open call projects. In addition, to further improve visibility across inter-WP matters, issue trackers have been created at a coordination level, targeting more managerial aspects rather than technical issues. Figure 67 and Figure 68 show two different views of an issue tracker in DEMETER: the Board view where the five "buckets" (named lists in GitLab's terminology) are depicted ("Open", "To Do", In Progress", "In Review", and "Closed"), and the List view where all non-closed issues are shown.



DEMETER > OpenCall1 > IssueTracker > Issue Boards

Feedback

Open 2

General questions
demeterproject/opencall1/issuetracker#5

Where to find info about user management?
demeterproject/opencall1/issuetracker#8

To Do 0

In Progress 0

In Review 0

Closed

Test Issue 2
suggestion
demeterproject/opencall1/issuetra

Test Issue
demeterproject/opencall1/issuetra

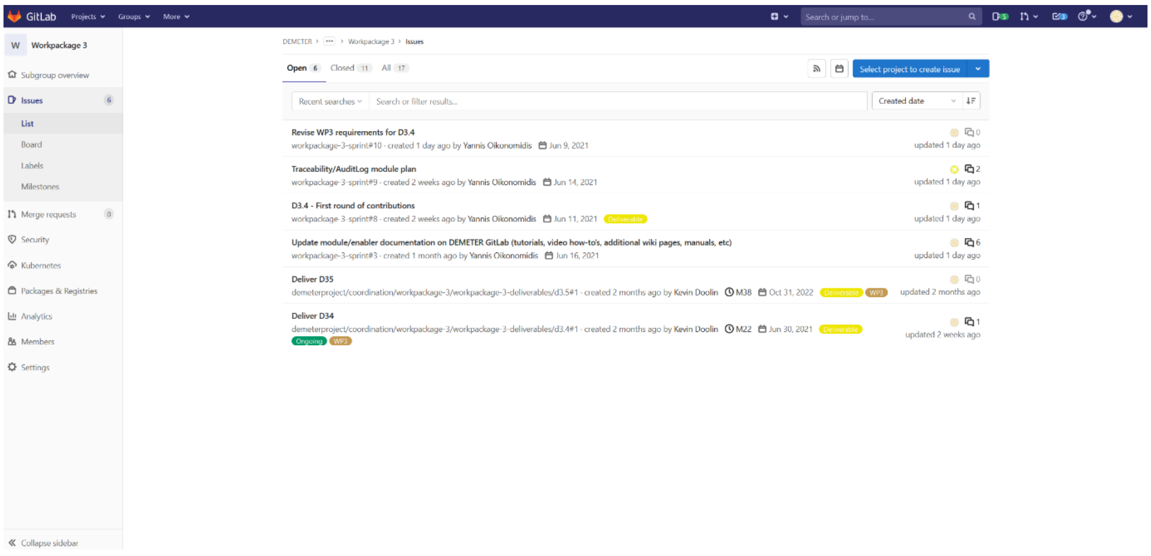
Internal server error trying to
demeterproject/opencall1/issuetra

Token expiring instantly
demeterproject/opencall1/issuetra

Integrating Persephone into D
demeterproject/opencall1/issuetra

"Can't generate capability tok
demeterproject/opencall1/issuetra

Figure 69: Issue Tracker - Board view



DEMETER > ••• > Workpackage 3 > Issues

Open 6 Closed 11 All 17

Revise WP3 requirements for D3.4
workpackage-3-sprint#10 · created 1 day ago by Yannis Oikonomidis Jun 9, 2021 — 0 — updated 1 day ago

Traceability/AuditLog module plan
workpackage-3-sprint#9 · created 2 weeks ago by Yannis Oikonomidis Jun 14, 2021 — 2 — updated 1 day ago

D3.4 - First round of contributions
workpackage-3-sprint#8 · created 2 weeks ago by Yannis Oikonomidis Jun 11, 2021 Deliverable — 1 — updated 1 day ago

Update module/enabler documentation on DEMETER GitLab (tutorials, video how-to's, additional wiki pages, manuals, etc)
workpackage-3-sprint#3 · created 1 month ago by Yannis Oikonomidis Jun 16, 2021 — 6 — updated 1 day ago

Deliver D35
demeterproject/coordination/workpackage-3/workpackage-3-deliverables/d3.5#1 · created 2 months ago by Kevin Doolin M38 Oct 31, 2022 Deliverable WP3 — 0 — updated 2 months ago

Deliver D34
demeterproject/coordination/workpackage-3/workpackage-3-deliverables/d3.4#1 · created 2 months ago by Kevin Doolin M22 Jun 30, 2021 Deliverable Ongoing WP3 — 1 — updated 2 weeks ago

Figure 70: Issue Tracker - List view

11.1.4 Requirements management and traceability

DEMETER needs to handle a large number of requirements (being a large integration project) and they are of a varying type (e.g., stakeholder needs and technical requirements) and origin (e.g., from pilots, state-of-art-analysis, MAA). To ensure proper requirements management throughout the project lifecycle which would be usable by many parties, a GitLab issues board was adapted to store them with their interdependencies. The following lists are available at the board:

List	Content	Maintainer (WP leader)
Project requirements	DEMETER project objectives and innovations	WP1
MAA needs	Stakeholder needs (from MAA)	WP7
WP5 requirements	User requirements from the WP5 pilots	WP5
WP2 requirements	WP2 technical requirements	WP2
WP3 requirements	WP3 technical requirements	WP3
WP4 requirements	WP4 technical requirements	WP4

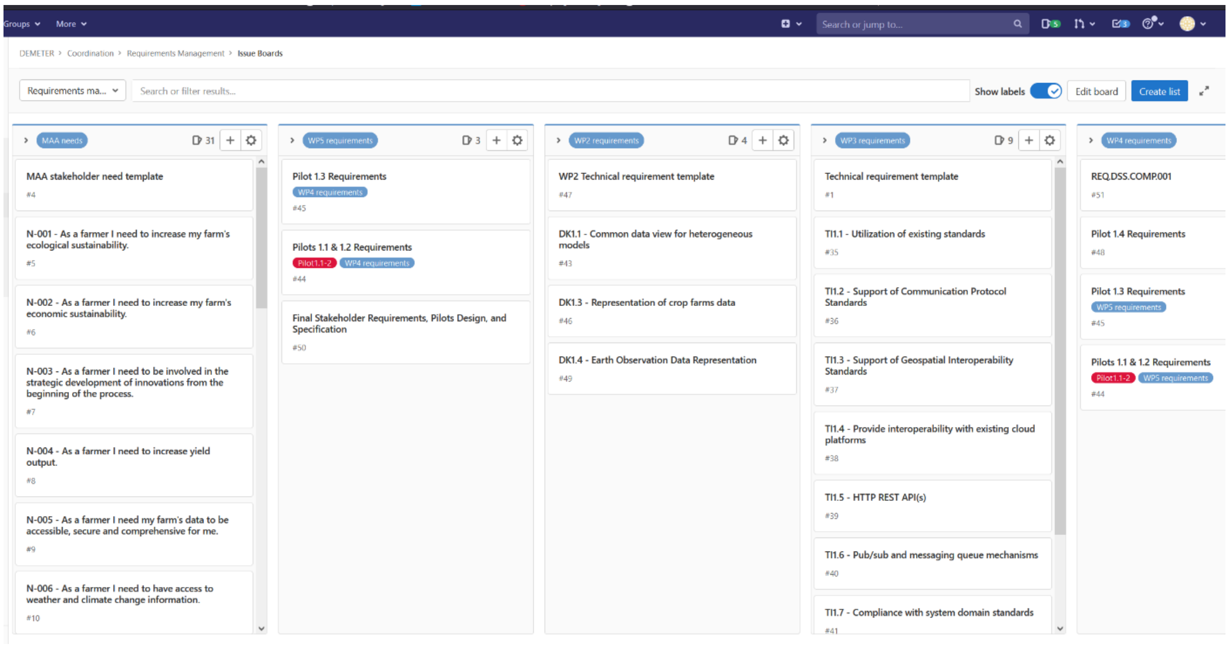
Each group (the maintainer in the table above) manages their list. A new requirement is not entered in a list unless pre-approved by its group.

The dependencies of a requirement are added by using the linked issues mechanism[16], to denote its relation to other ones in the same or other lists. Typically, the project requirements, MMA needs and WP5 requirements are linked to technical requirements in WP2-3-4, but more complex dependencies are also established. Dependencies to other items like state-of-the-art analysis are depicted within the requirement's description.

When a requirement needs to be updated, comments are added in its entry to mark the change. Each group periodically checks whether their requirements should be updated. In this process they check whether their linked requirements have changed and if their requirements need to be updated as a result.

The verification & validation reports (see next Section) record whether the technical requirements have been satisfied. The pilot evaluation documents record whether the pilot requirements have been addressed.

GitLab has been used widely in DEMETER for software management. In addition, the GitLab issues mechanism has been used for issue tracking (see for example the previous Subsection) and for overall project management. Therefore, this requirements management mechanism was selected as it was familiar with the partners (some of which may not be ICT experts) and it provided the basic functionalities for requirements management.


Groups
More
Search or jump to...
DEMETER > Coordination > Requirements Management > Issue Boards
Requirements ma...
Search or filter results...
Show labels
Edit board
Create list

MAA needs — 31

MAA stakeholder need template #4

N-001 - As a farmer I need to increase my farm's ecological sustainability. #5

N-002 - As a farmer I need to increase my farm's economic sustainability. #6

N-003 - As a farmer I need to be involved in the strategic development of innovations from the beginning of the process. #7

N-004 - As a farmer I need to increase yield output. #8

N-005 - As a farmer I need my farm's data to be accessible, secure and comprehensive for me. #9

N-006 - As a farmer I need to have access to weather and climate change information. #10

WP5 requirements — 3

Pilot 1.3 Requirements (WP4 requirements) #45

Pilots 1.1 & 1.2 Requirements (Pilot1.1-2, WP4 requirements) #44

Final Stakeholder Requirements, Pilots Design, and Specification #50

WP2 requirements — 4

WP2 Technical requirement template #47

DK1.1 - Common data view for heterogeneous models #43

DK1.3 - Representation of crop farms data #46

DK1.4 - Earth Observation Data Representation #49

WP3 requirements — 9

Technical requirement template #1

TI1.1 - Utilization of existing standards #35

TI1.2 - Support of Communication Protocol Standards #36

TI1.3 - Support of Geospatial Interoperability Standards #37

TI1.4 - Provide interoperability with existing cloud platforms #38

TI1.5 - HTTP REST API(s) #39

TI1.6 - Pub/sub and messaging queue mechanisms #40

TI1.7 - Compliance with system domain standards #41

WP4 requirements

REQ.DSS.COMP.001 #51

Pilot 1.4 Requirements #48

Pilot 1.3 Requirements (WP5 requirements) #45

Pilots 1.1 & 1.2 Requirements (Pilot1.1-2, WP5 requirements) #44

Figure 71: Requirements management in GitLab

[16] https://docs.gitlab.com/ee/user/project/issues/related_issues.htm

11.2 *Infrastructure*

DEMETER's Reference Implementation modules and enablers are hosted on DEMETER's infrastructure. DEH, BSE/FIE, and ACS instances are all deployed and accessible on DEMETER's cloud. That does not exclude the possibility for pilots to deploy ACS and BSE in their own premises, depending on their needs.

DEMETER's Infrastructure includes at the moment:

- 3 virtual machines (VMs) with 2vCPU, 4 GB RAM, 40 GB SSD
- 3 virtual machines (VMs) with 1vCPU, 2 GB RAM, 20GB SSD

All the VMs above are hosted on Hetzner Cloud, located in Germany. Depending on partners' needs, further resources might be allocated. In such case, this will be reported in D3.5 (M38).

To avoid technical incompatibilities among components and to ensure isolation, thus, increasing component security, all components are deployed as docker containers on DEMETER's cloud infrastructure.

The infrastructure hosts two environments: The Testing environment and the Production environment. The Testing environment contains module versions that have been developed and are available to be tested before releasing them. The Production environment contains the module versions that are available to the partners for using them in their pilots. The Production environment is being monitored using commercially proven tool (netdata[17]), a screenshot of which can be seen in Figure 72.

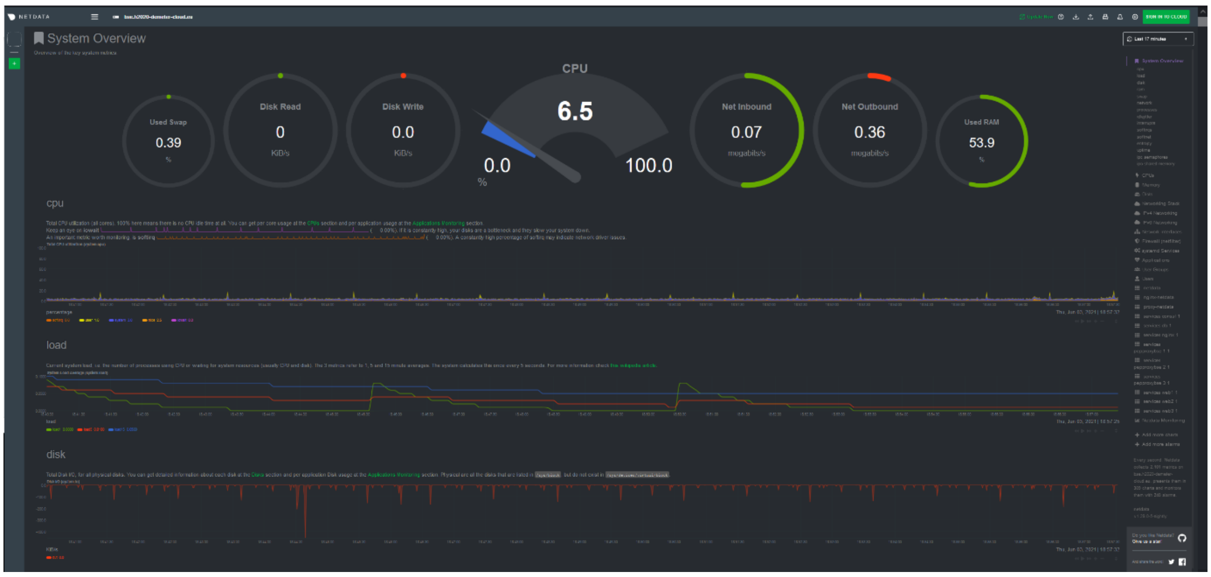


Figure 72: Monitoring Production environment

In DEMETER, a private CI/CD environment has been setup and is already being used by the consortium. DEMETER's CI/CD infrastructure comprises of the online repository (including the Image Registry) hosted on Gitlab's cloud and DEMETER's private cloud which is meant to host Gitlab Runner containers for the automated processes in CI/CD but also for the deployment of any DEMETER modules that are deployed on DEMETER's cloud. This infrastructure is not meant for pilot-

[17] https://www.netdata.cloud/

specific components as these components are deployed on pilots' premises. The CI/CD infrastructure setup is depicted in Figure 73.

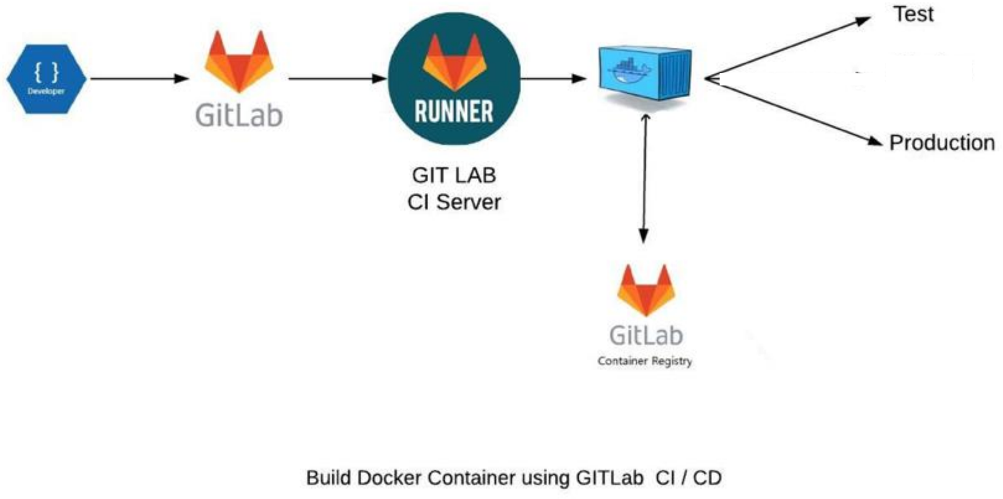
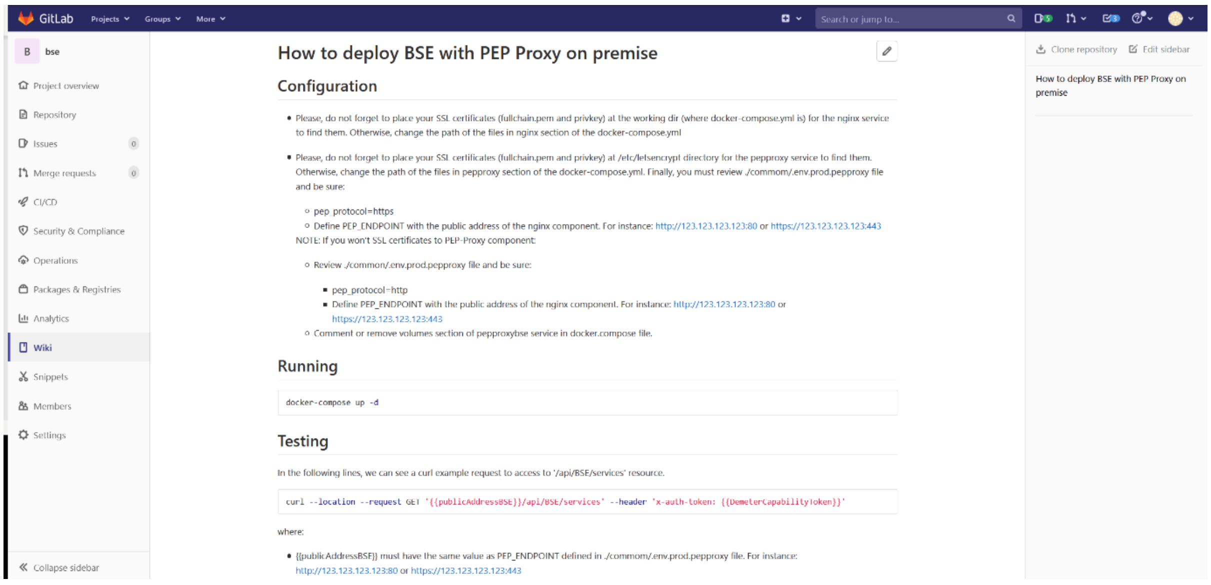


Figure 73: DEMETER CI/CD infrastructure

11.3 Internal Documentation

To assist DEMETER pilot partners to integrate with the core DEMETER modules and enablers, wiki pages and readme files have been created. This internal documentation targets developers and integrators, and includes information about the different modules and enablers, guidelines, and instructions, as well as how-to-use examples. Copies of these have also been provided to the open call projects. An example of a wiki page can be found in Figure 74 while an example of a README file can be found in Figure 75.



How to deploy BSE with PEP Proxy on premise

Configuration

- Please, do not forget to place your SSL certificates (fullchain.pem and privkey) at the working dir (where docker-compose.yml is) for the nginx service to find them. Otherwise, change the path of the files in nginx section of the docker-compose.yml
- Please, do not forget to place your SSL certificates (fullchain.pem and privkey) at /etc/letsencrypt directory for the pepproxy service to find them. Otherwise, change the path of the files in pepproxy section of the docker-compose.yml. Finally, you must review ./commom/.env.prod.pepproxy file and be sure:
  - pep_protocol=https
  - Define PEP_ENDPOINT with the public address of the nginx component. For instance: http://123.123.123.123:80 or https://123.123.123.123:443

  NOTE: If you won't SSL certificates to PEP-Proxy component:
  - Review ./common/.env.prod.pepproxy file and be sure:
    - pep_protocol=http
    - Define PEP_ENDPOINT with the public address of the nginx component. For instance: http://123.123.123.123:80 or https://123.123.123.123:443
  - Comment or remove volumes section of pepproxybse service in docker.compose file.

Running

```
docker-compose up -d
```

Testing

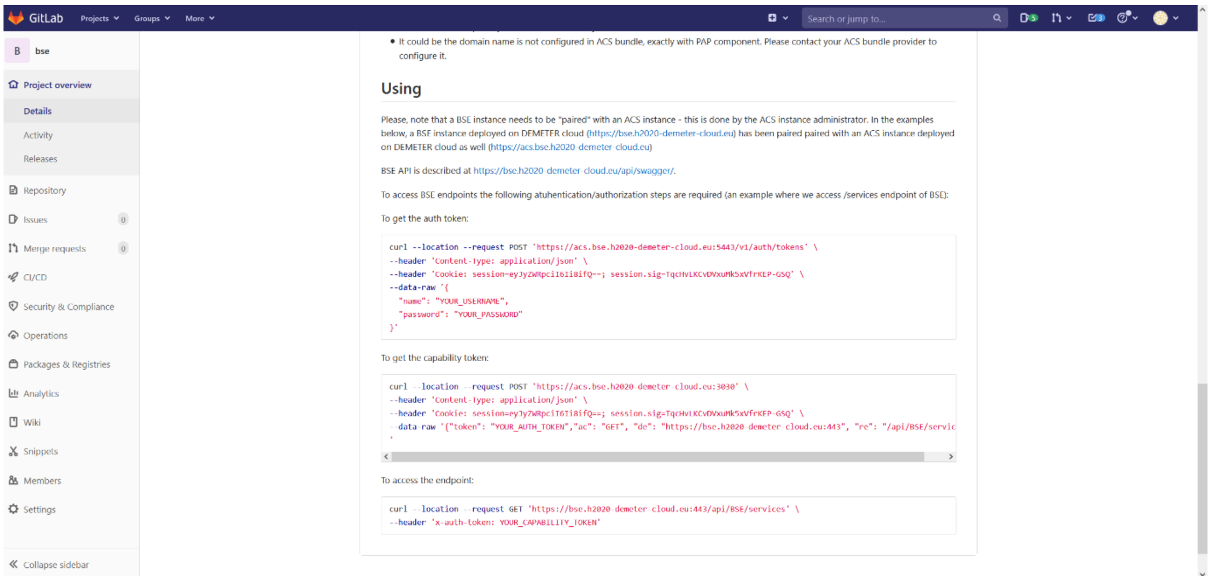
In the following lines, we can see a curl example request to access to '/api/BSE/services' resource.

```
curl --location --request GET '{{publicAddressBSE}}/api/BSE/services' --header 'x-auth-token: {{DemeterCapabilityToken}}'
```

where:

- {{publicAddressBSE}} must have the same value as PEP_ENDPOINT defined in ./commom/.env.prod.pepproxy file. For instance: http://123.123.123.123:80 or https://123.123.123.123:443

Figure 74: Wiki page in DEMETER GitLab



- It could be the domain name is not configured in ACS bundle, exactly with PAP component. Please contact your ACS bundle provider to configure it.

Using

Please, note that a BSE instance needs to be "paired" with an ACS instance - this is done by the ACS instance administrator. In the examples below, a BSE instance deployed on DEMETER cloud (https://bse.h2020-demeter-cloud.eu) has been paired paired with an ACS instance deployed on DEMETER cloud as well (https://acs.bse.h2020-demeter-cloud.eu)

BSE API is described at https://bse.h2020-demeter-cloud.eu/api/swagger/.

To access BSE endpoints the following atuhentication/authorization steps are required (an example where we access /services endpoint of BSE):

To get the auth token:

```
curl --location --request POST 'https://acs.bse.h2020-demeter-cloud.eu:5443/v1/auth/tokens' \
--header 'Content-Type: application/json' \
--header 'Cookie: session=eyJyZWRpciI6Ii8ifQ==; session.sig=TqcHvLKCvDVxuMk5xVfrKEP-GSQ' \
--data-raw '{
  "name": "YOUR_USERNAME",
  "password": "YOUR_PASSWORD"
}'
```

To get the capability token:

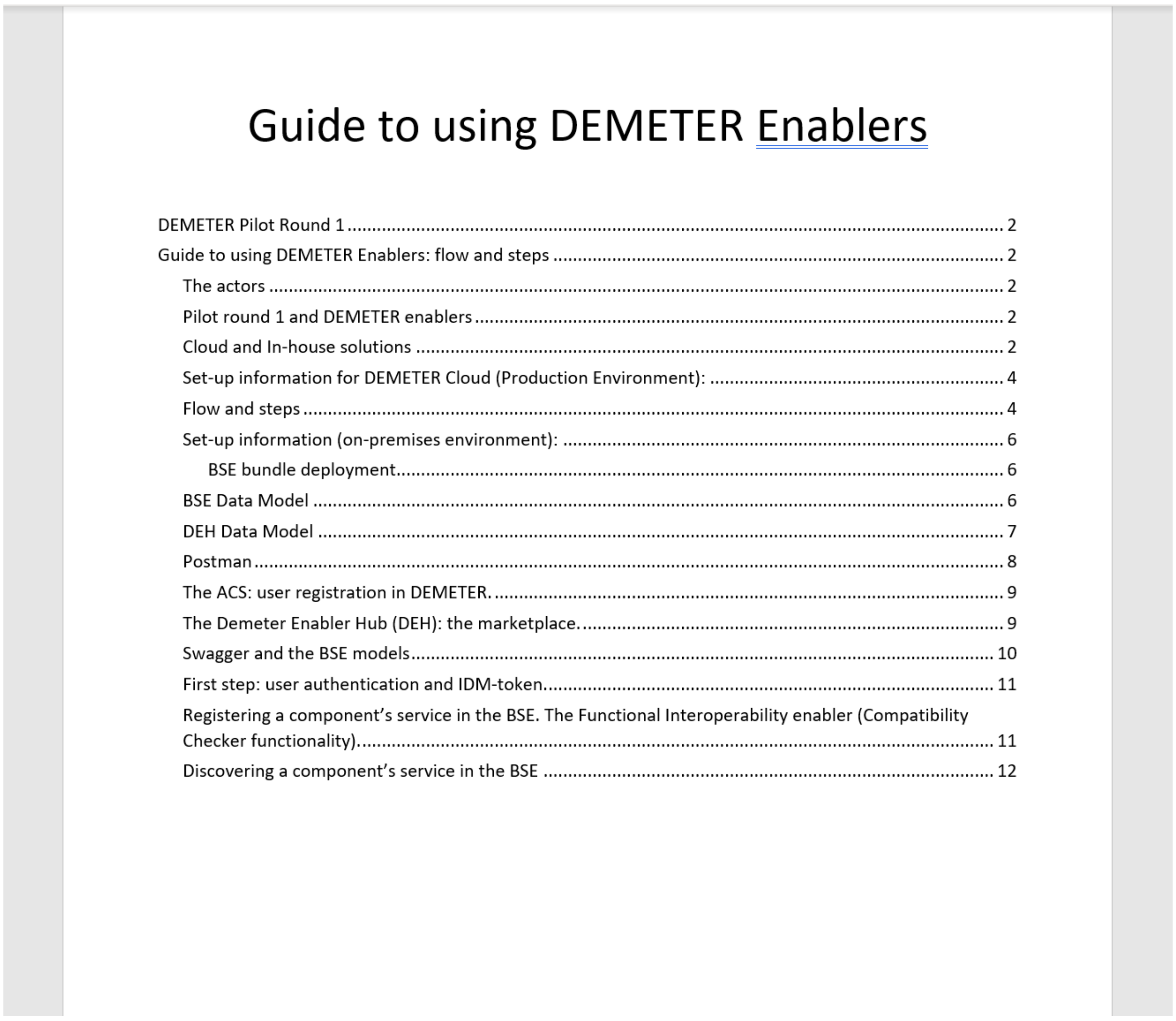
```
curl --location --request POST 'https://acs.bse.h2020-demeter-cloud.eu:3030' \
--header 'Content-Type: application/json' \
--header 'Cookie: session=eyJyZWRpciI6Ii8ifQ==; session.sig=TqcHvLKCvDVxuMk5xVfrKEP-GSQ' \
--data-raw '{"token": "YOUR_AUTH_TOKEN","ac": "GET", "de": "https://bse.h2020-demeter-cloud.eu:443", "re": "/api/BSE/servic
'
```

To access the endpoint:

```
curl --location --request GET 'https://bse.h2020-demeter-cloud.eu:443/api/BSE/services' \
--header 'x-auth-token: YOUR_CAPABILITY_TOKEN'
```

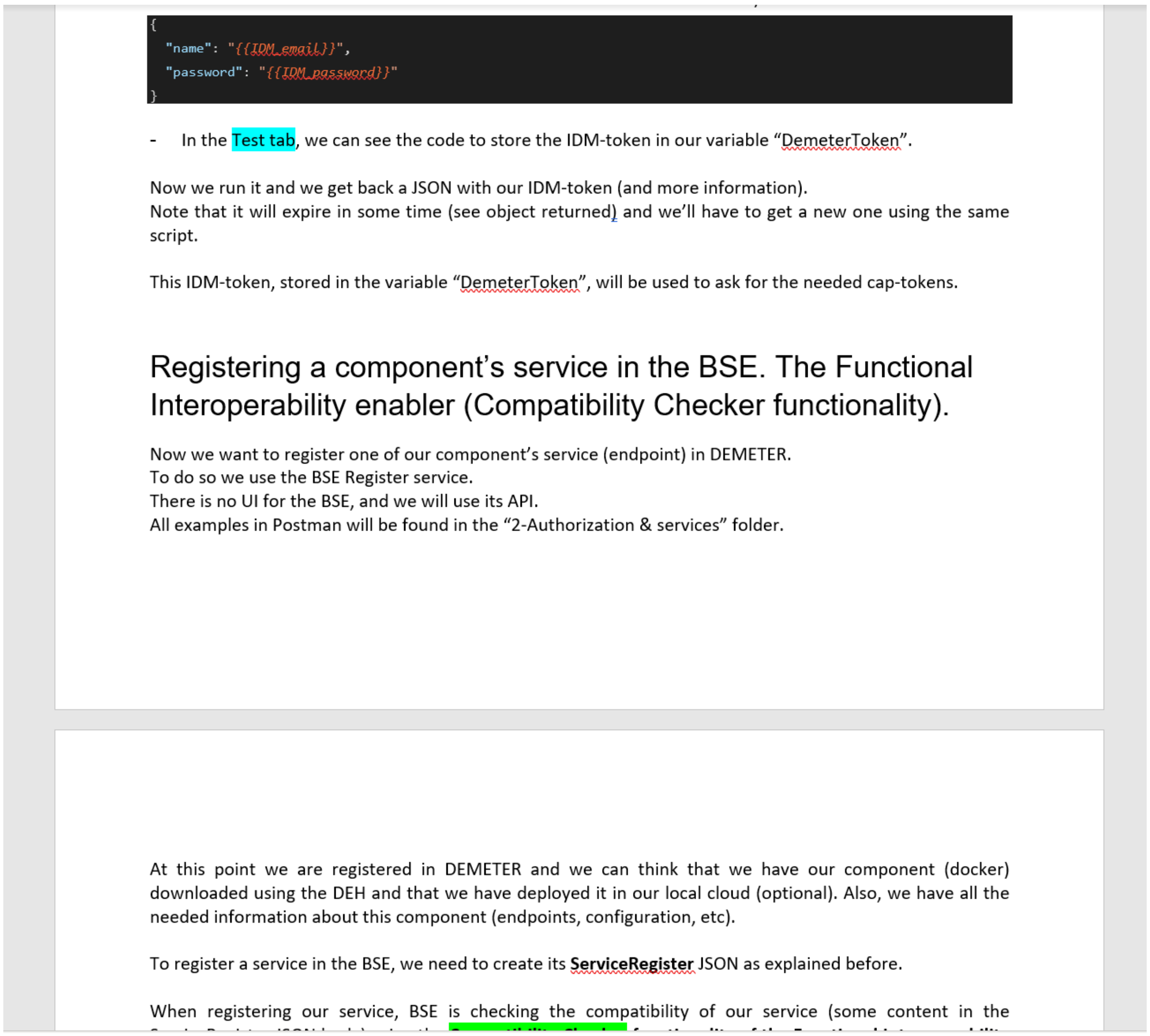
Figure 75: README file in GitLab

Besides wiki pages and readme files on GitLab, word documents have been used, too. In particular, documents that include information on important aspects of DEMETER, such as the AIM, the data model of the metadata of the BSE, have been created, stored on NextCloud, and made available for all partners in the consortium. In Figure 76 and Figure 77, one document that has been created and acts as a hands-on tutorial for the pilots and the open call projects is depicted through its table of contents and an example of its contents. This document, namely Integration Flow document has been mainly used for the integration of the several pilot and open call components with DEMETER core modules and enablers (excluding AIM which is documented in a different document).


Guide to using DEMETER Enablers

DEMETER Pilot Round 1 2
Guide to using DEMETER Enablers: flow and steps 2
The actors 2
Pilot round 1 and DEMETER enablers 2
Cloud and In-house solutions 2
Set-up information for DEMETER Cloud (Production Environment): 4
Flow and steps 4
Set-up information (on-premises environment): 6
BSE bundle deployment........ 6
BSE Data Model 6
DEH Data Model 7
Postman 8
The ACS: user registration in DEMETER. 9
The Demeter Enabler Hub (DEH): the marketplace. 9
Swagger and the BSE models........ 10
First step: user authentication and IDM-token........ 11
Registering a component's service in the BSE. The Functional Interoperability enabler (Compatibility Checker functionality)........ 11
Discovering a component's service in the BSE 12

Figure 76: Integration Flow document table of contents



```
{
  "name": "{{IDM_email}}",
  "password": "{{IDM_password}}"
}
```

- In the Test tab, we can see the code to store the IDM-token in our variable "DemeterToken".

Now we run it and we get back a JSON with our IDM-token (and more information).
Note that it will expire in some time (see object returned) and we'll have to get a new one using the same script.

This IDM-token, stored in the variable "DemeterToken", will be used to ask for the needed cap-tokens.

Registering a component's service in the BSE. The Functional Interoperability enabler (Compatibility Checker functionality).

Now we want to register one of our component's service (endpoint) in DEMETER.
To do so we use the BSE Register service.
There is no UI for the BSE, and we will use its API.
All examples in Postman will be found in the "2-Authorization & services" folder.

At this point we are registered in DEMETER and we can think that we have our component (docker) downloaded using the DEH and that we have deployed it in our local cloud (optional). Also, we have all the needed information about this component (endpoints, configuration, etc).

To register a service in the BSE, we need to create its **ServiceRegister** JSON as explained before.

When registering our service, BSE is checking the compatibility of our service (some content in the

Figure 77: Integration Flow document contents

On top of the documentation, explanatory videos which demonstrate how to use a particular module have been created and uploaded on GitLab and on NextCloud (DEMETER's online document cloud). So far, videos for the ACS, the DEH, and the BSE have been created, aiming to support developers and integrators, but also users (in the cases of DEH and ACS). In addition, the recorded sessions from the first Implementation Workshop (March 2021) have been made available to the consortium and the open call participants. Finally, developers were strongly advised to adopt Swagger for online documentation of the developed API's.

12 Verification and Validation Report

The Verification and Validation plan was described in the previous version of the document, D3.2, and aimed to ensure that DEMETER's components are successfully integrated and perform as they were prescribed during the design phase of the project. It described the process that DEMETER components need to follow in order to be tested properly and subsequently the process that documents and validates their functional and non-functional performance in stand-alone manner and as a part of a greater system (pilot). For completeness, we included the plan section in this document, too, and we added a new section for the report summary.

12.1 Verification Plan

The Verification Plan describes the process that DEMETER implementations have to follow in order to be able:

- to verify that each application offers the functionality that was envisioned provide during the design phase of each of the DEMETER platform's components, and
- to verify that the integration between each of the DEMETER platform's components has been successfully carried out.

This is realised through a set of Test Levels that can be executed upon them. Briefly, the purpose of these test is to verify that each component:

- Provides the required functionality that was designed for.
- Can Integrate successfully with the relevant DEMETER platform's components.
- forms a system that meets the required KPIs that were set.

Verification plan includes Test Levels, each of them addressing a different aspect of the verification process. These are described below.

According to the International Software Testing Qualifications Board's (ISTQB's) Agile Test Extension **Invalid source specified.** the following test levels can be defined:

Component testing (also known as unit, module, or program testing) searches for defects in, and verifies the function of, software modules programs, objects, classes, etc., that are separately testable. It may be done in isolation from the rest of the system, depending on the context of the development life cycle and the system. In the context of DEMETER platform development, separate component tests will be planned and executed in each technical Work package delivering DEMETER components. Such tests will facilitate the verification at component level (unit-test).

Integration testing evaluates the interfaces between components, interactions with different parts of a system and interfaces between systems. Systematic integration strategies may be based on system architecture (such as top-down and bottom-up), functional tasks, transaction processing sequences or some other aspect of the system or components. To ease fault isolation and detect defects early, integration should normally be incremental rather than "big bang".

System testing is concerned with the behaviour of a whole product. In system testing, the test environment should correspond to the final target or production environment as much as possible to minimise the risk of environment-specific failures not being found in testing. System testing may include tests based on risks and/or on requirements specifications, business processes, use cases, or other high-level text descriptions or models of system behaviour, interactions with the operating system, and system resources. In DEMETER, this level of testing should be scheduled prior to the pilot demonstrations and is expected to be facilitated by the DEMETER Pilot leaders.

Acceptance testing aims to establish confidence in the system, parts of the system or specific non-functional characteristics of the system. It is often the responsibility of the customers or users of a system; other stakeholders may be involved as well. Finding defects is not the main focus in acceptance testing. Acceptance testing may assess the system's readiness for deployment and use, although it is not necessarily the final level of testing. In DEMETER this level of testing is optional, since commercialisation of the DEMETER platform is not expected within the project timeframe.

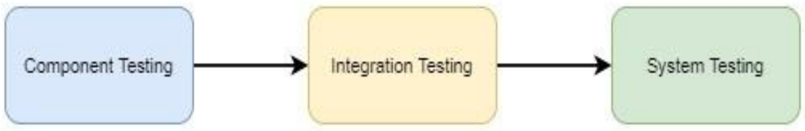


Figure 78: DEMETER's Component Test Levels

Figure 78 illustrates, in a summative manner, the Test Levels that each module and enabler has to be validated and verified against in the context of DEMETER project.

12.2 Validation Plan

The Validation Plan describes the process that DEMETER pilot partners have to follow in order to be able validate the various DEMETER modules and enablers that are being used in their pilots.

Storyboard based validation. The release-based validation will be performed by pilot owners. A specific release validation form, containing all the user requested functionalities per pilot, will be used to validate the DEMETER platform during the pilot demonstration phases. The validation procedure will produce a list of features not implemented, partially implemented, or fully implemented. The list of the incomplete features and an analysis of the issue(s) will be presented to the relevant DEMETER partners in order to assist them solving the problem(s).

Documentation should be done per feature/component validated. The documentation per feature/component will be collected and reviewed by the pilot owners, which will follow a documentation template. The missing documentation will be reported to the relevant DEMETER partners. The documentation provided to the pilot leaders will serve as a basis for the validation process. The relevant documentation description that is required by each module/enabler owner is described in the Annex C. After each pilot-based validation, the validated documentation will be incorporated into platform release package.

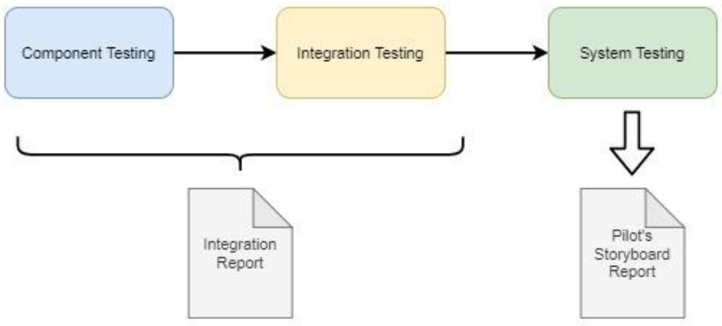


Figure 79: Verification and Validation process

Figure 79 above illustrates the Verification and Validation process that needs to be followed by each DEMETER's partner. Through Component and Integration Testing each partner compiles the

Integration report that summarises the testing results. Given on the component integration reports, Pilot leaders will compile the storyboard validation report that would verify that all the request functionalities, per pilot, have been realised.

Validate the Key Performance Indicators (KPI's). Each release will present a list of KPI's that will be validated using measurable characterisation, such as: not reached, partially reached, fully reached. A KPI validation report will be shared with technical partners to assist them reaching the goal. The KPIs are also included in the template that is described in Annex C.

12.3 Report collections status

As described in the previous two sections, the template presented in Annex C was filled in by module and enabler owners whose modules and enablers have been released during the past year. Given the public nature of this document, we cannot present extensive details of these documents; however, there are stored on DEMETER's online documentation platform (NextCloud) and are available to the EU. Summarising the status of the collection of the reports, by the submission of this deliverable we have already collected thirty-four (34) reports that fully cover WP3 modules and enablers, and partially cover WP2 and WP4 modules and (advanced) enablers. The collection is ongoing and as more modules and (advanced) enablers are being finalised, more reports will be delivered. These reports are available to the pilot partners for the final -user- validation phase.

13 Conclusions

This document accompanies and describes the Release 2 of the DEMETER components and tools that enable solution integration, interoperability with external platforms and deployment support for pilot cases. These updated components and tools are released in a scheme that:

- on one hand provide a **concrete** implementation to be used by the pilot applications and guide further development, and
- on the other hand, allows full **flexibility** for the application configuration and deployment to facilitate the highly different pilot needs and the various business models of the stakeholders.

This document also includes the updates on the requirements presented in the previous version of the deliverable while summarising their status.

It reflects work done in Tasks T3.2, T3.3, T3.4, T3.5 and T3.5 but it is based on work done in T3.1 (which produced D3.3 “DEMETER reference architecture - Release 2”) and utilises work done in WP2 (D2.3 “Common data models and semantic interoperability mechanisms – Release 2” and D2.4 “DEMETER Data and knowledge extraction tools – Release 2”). It is accompanied by a set of other deliverables derived in WP4 (D4.3 “Decision Support, Benchmarking and Performance Indicator Monitoring Tools - Release 2” and D4.4 “Decision Enablers, Advisory Support Tools and DEMETER Stakeholder Open Collaboration Space – Release 2”). All together they provide the second release of the DEMETER reference implementation and contribute to project Milestone 6 “DEMETER Enablers, Hub, Spaces and Applications Release 2”.

The key achievements of the D3.4 are the updated implementations and deployments of the BSE, ACS, DEH, Core Enablers for integration, along with the Verification and Validation activities over the available DEMETER enablers and modules.

This release is to be used in the coming months by the 20 DEMETER pilots and the Open Call projects to build and evaluate their DEMETER-enabled applications. Based on their feedback, the final version of the DEMETER Reference Implementation will be presented in D3.5 (M38). The work until the final release includes intermediate releases with improvements in DEH, BSE, and the core enablers. Also, focus will be on supporting the pilot partners by assisting during integration with the DEMETER modules and enablers, as well as providing additional or improving existing documentation.

Annex A Detailed Requirements

A.1. Technical and Syntactic Interoperability of pilot technologies/platforms

Requirement ID	TI1.1	Version	0.1	Last Update Date	27/01/2020
Title	Utilization of existing standards				
Description	DEMETER should utilize existing standards to provide interoperability. DEMETER should consider the following Internet Interoperability standards and adopt or build on top of the most appropriate/relevant: 1. ISO/IEC AWI 21823 2. ISO/IEC 29182 3. AIOTI WG03 4. FIWARE 5. FIWARE - NGSI 6. W3C 7. ETSI NGSI-LD 8. oneM2M				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control Innovation 8: Unified agriculture ontology				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				

Identified by Partner(s)	INTRA
Status	Proposed
Comments/Remarks	This requirement is not covered by any core module/enabler of WP3 (out of scope). It is more related to enablers developed in WP2 (AIM) and/or WP4.

Requirement ID	TI1.2	Version	0.1	Last Update Date	27/01/2020
Title	Support of Communication Protocol Standards				
Description	DEMETER solutions should support existing communication protocol standards: 1. OASIS (ISO/IEC 20802) - MQTT 2. NB-IoT 3. LoRA 4. ISO 11783 ISOBUS 5. AEF: EFDI 6. ISO 18000 (RFID) 7. SigFox 8. Cellular (3G, 4G, etc) 9. BLE 10. Bluetooth 11. Wi-Fi 12. IEEE 802.15.4				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control Innovation 7: Cost and power effective IoT data acquisition				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/a	Technology providers, Solution providers				

ctors	
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA, TECNALIA
Status	Rejected
Comments/Remarks	DEMETER does not target communication/networking protocols but higher layers, e.g., at an IoT platform level. It is, thus, rejected.

Requirement ID	TI1.3	Version	0.1	Last Update Date	27/01/2020
Title	Support of Geospatial Interoperability Standards				
Description	DEMETER solutions should support existing Geospatial Interoperability standards: 1. OGC WFS 2. OGC WMS 3. OGC WCS 4. OGC WPS 5. OGC Agriculture 6. OGC SWE 7. OGC POI				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control Innovation 4: Earth Observation data service Innovation 14: Smart fruit pesticides management				
Reference component(s)	TBD				
Reference	TBD				

technology(ies)	
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA
Status	Proposed
Comments/Remarks	This requirement is not covered by any core module/enabler of WP3 (out of scope). It is more related to enablers developed in WP2 (AIM) and/or WP4.

Requirement ID	TI1.4	Version	0.1	Last Update Date	27/01/2020
Title	Provide interoperability with existing cloud platforms				
Description	DEMETER solutions should provide interoperability with existing cloud platforms: 1. Azure 2. Proba-V 3. AVR Connect 4. Digital Ocean				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control				
Reference component(s)	BSE, ACS				
Reference technology(ies)	HTTP(S) APIs				
Involved stakeholders/a	Technology providers, Solution providers				

ctors	
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI1.5	Version	0.1	Last Update Date	27/01/2020
Title	HTTP REST API(s)				
Description	DEMETER should be able to connect to (external) platforms via REST API(s)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control				
Reference component(s)	BSE, FI, ACS				
Reference technology(ies)	HTTP(S)				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	INTRA				

Status	Fulfilled
Comments/Remarks	

Requirement ID	TI1.6	Version	0.1	Last Update Date	27/01/2020
Title	Pub/sub and messaging queue mechanisms				
Description	DEMETER should be able to connect to (external) platforms via pub/sub and messaging queue mechanisms				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control				
Reference component(s)	StreamHandler				
Reference technology(ies)	Apache Kafka				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	INTRA				
Status	Fulfilled				
Comments/Remarks					

Requirement ID	TI1.7	Version	0.1	Last Update Date	27/01/2020
Title	Compliance with system domain standards				

Description	DEMETER shall be designed in compliance with standards selected according to system domain, i.e., web standards, telecommunication standards, user interface standards, WCAG 2.1, etc.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.4, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	TECNALIA
Status	Proposed
Comments/Remarks	Not evaluated yet

Requirement ID	TI1.8	Version	0.1	Last Update Date	27/01/2020
Title	Data formats				
Description	DEMETER should offer APIs using common data formats such as JSON, JSON-LD, XML, or RDF				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models				

	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control
Reference component(s)	BSE, FIE, ACS, DEH, DEH Client
Reference technology(ies)	JSON
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	TECNALIA, INTRA
Status	Fulfilled
Comments/Remarks	

A.2. Environment for service discovery and provisioning

Requirement ID	TI2.1	Version	0.1	Last Update Date	27/01/2020
Title	Service description definition				
Description	DEMETER must propose a common service description definition to be used for registering and discovering services from different platforms, building on existing frameworks and standards (such as OASIS SOA-RM)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyze, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control Innovation 8: Unified agriculture ontology				

Reference component(s)	BSE, FIE, DEH
Reference technology(ies)	Service metadata model
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	UMU, INTRA, PSNC
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI2.2	Version	0.1	Last Update Date	27/01/2020
Title	Services provisioning maintaining data security and privacy				
Description	Services provided by DEMETER must implement security and privacy mechanisms to protect the data exchanged with other entities (services, users)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyze, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control Innovation 8: Unified agriculture ontology Innovation 9: Secure Agricultural data sharing services				
Reference component(s)	All WP3 modules/enablers				
Reference technology(ies)	ACS				

Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	UMU, PSNC
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI2.3	Version	0.1	Last Update Date	27/01/2020
Title	Services registration to DEMETER Enabler Hub				
Description	DEMETER-enabled services must be able to register to DEMETER Enabler Hub and the BSE and make themselves discoverable to consumers, i.e., other services or end users.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control Innovation 5: Farm enabler dashboards				
Reference component(s)	BSE, FIE, DEH				
Reference technology(ies)	HTTP(S) API				
Involved stakeholders/actors	Technology providers, Solution providers, Farmers				
Prerequisite(s)	None				
Type	Functional				

Priority Level	Mandatory
Identified by Partner(s)	UMU, PSNC
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI2.4	Version	0.1	Last Update Date	27/01/2020
Title	Services' categorization				
Description	DEMETER must provide a way to group services in categories (e.g., Farm monitoring, Supply chain monitoring, Milk quality, etc.)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control				
Reference component(s)	BSE, DEH				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	UMU, PSNC				
Status	Fulfilled				
Comments/Remarks					

A.3. Networking and Communication

Requirement ID	TI3.1	Version	0.2	Last Update Date	04/02/2020
Title	Secure transport layer (TLS, SSH, etc.)				
Description	The transport layer should be secure to protect communications				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 9: secure Agricultural data sharing services				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	VICOM				
Status	Fulfilled				
Comments/Remarks					

Requirement ID	TI3.2	Version	0.2	Last Update Date	04/02/2020
Title	GDPR technical requirements				
Description	DEMETER must comply with GDPR technical requirements				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				

Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 9: secure Agricultural data sharing services Innovation 11: data integration Innovation 20: product authentication/fraud detection
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers, farmers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	VICOM
Status	Proposed + Review
Comments/Remarks	

Requirement ID	TI3.3	Version	0.2	Last Update Date	04/02/2020
Title	Combination of physical/wireless communications and Internet backbone networks				
Description	DEMETER should combine the use of physical/wireless communications and Internet backbone networks, in order to enable data sharing, network and device management, cloud computations and storage.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: secure interoperability				

	Innovation 3: device automation and control Innovation 5: traceability Innovation 7: IoT data acquisition Innovation 9: secure Agricultural data sharing services Innovation 11: data integration Innovation 20: product authentication/fraud detection
Reference component(s)	-
Reference technology(ies)	-
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Desirable
Identified by Partner(s)	UMU
Status	Proposed + Review
Comments/Remarks	

Requirement ID	TI3.4	Version	0.2	Last Update Date	04/02/2020
Title	Control devices sharing information				
Description	DEMETER should provide the means to control IoT devices sharing information				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 3: device automation and control				

	Innovation 5: traceability Innovation 7: IoT data acquisition Innovation 9: secure Agricultural data sharing services Innovation 20: product authentication/fraud detection
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	UMU
Status	Proposed + Review
Comments/Remarks	

A.4. Security

Requirement ID	TI4.1	Version	0.2	Last Update Date	04/02/2020
Title	Attribute Based Access Control or Distributed Capabilities Access Control component				
Description	DEMETER should provide an Attribute Based Access Control or Distributed Capabilities Access Control component that can be integrated with trial site platforms and gateways				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 5: traceability Innovation 9: secure Agricultural data sharing services Innovation 20: product authentication/fraud detection				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	WIT				
Status	Proposed + Review				
Comments/Remarks					

Requirement ID	TI4.2	Version	0.2	Last Update Date	04/02/2020
Title	Authentication and authorization mechanisms for services, accessing				

	resources and information audit tools
Description	DEMETER must offer authentication and authorization mechanisms for using services and accessing resources as well as information audit tools
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T2.4, T3.4
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 5: traceability Innovation 9: secure Agricultural data sharing services Innovation 11: data integration Innovation 20: product authentication/fraud detection
Reference component(s)	ACS, AC Enabler
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	UMU
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI4.3	Version	0.2	Last Update Date	04/02/2020
Title	Data protection and privacy on software execution, network communications and integrated solution security				

Description	DEMETER will offer Data protection and privacy on software execution, network communications and integrated solution security
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T2.4, T3.4
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 5: traceability Innovation 9: secure Agricultural data sharing services Innovation 11: data integration Innovation 20: product authentication/fraud detection
Reference component(s)	ACS, AC Enabler
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	UMU
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI4.4	Version	0.2	Last Update Date	04/02/2020
Title	Identity management, access control and audit log				
Description	DEMETER must allow identity management, access control and audit log				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				

Relevant Task(s)	T2.4, T3.4
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 5: traceability Innovation 9: secure Agricultural data sharing services Innovation 20: product authentication/fraud detection
Reference component(s)	ACS, AC Enabler
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	UMU
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI4.5	Version	0.2	Last Update Date	04/02/2020
Title	Encrypted communications, integrity controls and electronic signature functionalities				
Description	DEMETER should offer encrypted communications, integrity controls and electronic signature functionalities				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				

Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 5: traceability Innovation 9: secure Agricultural data sharing services Innovation 20: product authentication/fraud detection
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	UMU
Status	Proposed + Review
Comments/Remarks	

A.5. Device/resource Management (including databases)

Requirement ID	TI5.1	Version	0.1	Last Update Date	27/01/2020
Title	Data storage systems access management				
Description	DEMETER should provide the means to manage access (CRUD operations) to data storage systems including semantic repositories				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 7: Cost- and power-effective IoT data acquisition				
Reference	TBD				

component(s)	
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	TECNALIA, INTRA
Status	Proposed
Comments/Remarks	This requirement is not covered by any WP3 module/enable but is more related to the Semantic repository specified in WP2

Requirement ID	TI5.2	Version	0.1	Last Update Date	27/01/2020
Title	Registration the capabilities of a resource				
Description	DEMETER should provide the means to register the capabilities of a resource (platform, thing, service) to the DEMETER Enabler Hub and the Brokerage Service Environment, thus being available to interested parties. Therefore, it will be able to make use of other Enablers registered in the Hub to enhance the resource's features				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	BSE, FIE, DEH				
Reference technology(ies)	HTTP(S) API				
Involved	Technology providers, Solution providers				

stakeholders/actors	
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	TECNALIA
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI5.3	Version	0.1	Last Update Date	27/01/2020
Title	Multiple devices bulk operations				
Description	DEMETER solutions should support multiple devices bulk operations (e.g., to be able to access a service offered by multiple devices)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by	UMU				

Partner(s)	
Status	Proposed
Comments/Remarks	

Requirement ID	TI5.4	Version	0.1	Last Update Date	27/01/2020
Title	Resource/device sharing rules				
Description	DEMETER solutions could specify rules (e.g., concurrent users, data limits, etc.) on the usage of shared resources/devices				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control Innovation 9: Secure Agricultural data sharing services				
Reference component(s)	BSE, DEH				
Reference technology(ies)	HTTP(S) API				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Optional				
Identified by Partner(s)	INTRA				
Status	Fulfilled				
Comments/Remarks					

A.6. Runtime environment, Deployment Management & Orchestration

Requirement ID	TI6.1	Version	0.1	Last Update Date	27/01/2020
Title	DEMETER Enablers deployment				
Description	DEMETER must make it possible for developers to deploy DEMETER Enablers on their premises				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	BSE, FIE, ACS, AC Enabler, DEH, DEH Client				
Reference technology(ies)	Docker				
Involved stakeholders/ actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				
Priority Level	Mandatory				
Identified by Partner(s)	INTRA				
Status	Fulfilled				
Comments/Remarks					

Requirement ID	TI6.2	Version	0.1	Last Update Date	27/01/2020
Title	DEMETER Enablers compliance				
Description	DEMETER should provide means to developers to verify that the enablers they implemented are DEMETER-compliant				
Relevant Pilot(s)	ALL				

Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.5, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	BSE, FIE
Reference technology(ies)	Compatibility check HTTP(S) API
Involved stakeholders/ actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	INTRA
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI6.3	Version	0.1	Last Update Date	27/01/2020
Title	DEMETER deployment tests				
Description	DEMETER could provide tests to verify that the deployment of an enabler has been successful (e.g. having an endpoint at the enabler side that will be used for testing its connectivity)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				

Reference component(s)	BSE, FIE
Reference technology(ies)	HTTP(S) API
Involved stakeholders/ actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	INTRA
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI6.4	Version	0.1	Last Update Date	27/01/2020
Title	DEMETER runtime environment agnostic				
Description	DEMETER enablers should be runtime environment agnostic (i.e., developers can develop an enabler in any runtime environment (be it Linux or Windows or others SO) and achieve DEMETER-compliance. Technologies that enable virtualization and allow applications to run independently of the environment must be guaranteed. Enablers should be able to be deployed in various environments				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	BSE, FIE				
Reference technology(ies)	Docker				

Involved stakeholders/ actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI6.5	Version	0.1	Last Update Date	27/01/2020
Title	Deployment process documentation				
Description	DEMETER should provide clear guidelines (e.g. reference documentation) for technology and solution providers in order to standardize the deployment process as much as possible in both development and production environments				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/ actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				
Priority Level	Mandatory				

Identified by Partner(s)	ENG
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI6.6	Version	0.1	Last Update Date	27/01/2020
Title	Deployment software life-cycle management				
Description	DEMETER should provide an adequate technology solution and suitable tools able to manage the entire software life-cycle management (from development to production environment)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/ actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				
Priority Level	Optional				
Identified by Partner(s)	ENG				
Status	Proposed + Reviewed				
Comments/Remarks					

Requirement ID	TI6.7	Version	0.1	Last Update Date	27/01/2020

Title	Deployment process security
Description	DEMETER should ensure a good level of security in the deployment process (e.g. the connections with DEMETER components for deployment purposes such as the AIS and the Enabler Hub should be secure)
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.3, T3.4, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 9: Secure Agricultural data sharing services
Reference component(s)	BSE, FIE, DEH, DEH Client, ACS
Reference technology(ies)	SSL
Involved stakeholders/ actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Non-Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Fulfilled
Comments/Remarks	

A.7. Service / application life-cycle management

Requirement ID	TI7.1	Version	0.1	Last Update Date	27/01/2020
Title	Service/application life-cycle management methodology				
Description	Each software component development in DEMETER should follow a service/application life-cycle management methodology (waterfall or agile)				

Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.3, T3.4, T3.5, T3.6
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	All WP3 modules/enablers
Reference technology(ies)	Agile
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Non-Functional
Priority Level	Optional
Identified by Partner(s)	ATOS
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI7.2	Version	0.1	Last Update Date	27/01/2020
Title	Technical requirements review				
Description	DEMETER technical requirements will be defined or reviewed for all the different software/services to be developed in the beginning of every iteration, and will be used for the development plan design as well as for the evaluation stages				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				

Reference component(s)	All WP3 modules/enablers
Reference technology(ies)	N/A
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Non-Functional
Priority Level	Optional
Identified by Partner(s)	ATOS
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI7.3	Version	0.1	Last Update Date	27/01/2020
Title	Components' testing				
Description	DEMETER components have to pass a set of tests to be defined at the beginning of the development phases in order to evaluate the results from those development phases.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	All WP3 modules/enablers				
Reference technology(ies)	Validation Reports				
Involved stakeholders/actors	Technology providers, Solution providers				

Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	ATOS
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI7.4	Version	0.1	Last Update Date	27/01/2020
Title	Development teams' communication				
Description	DEMETER component development teams will have fluid communication (at any level) to guarantee the correct development and integration of the different components involved in the project.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.4, T3.5, T3.6				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	All WP3 modules/enablers				
Reference technology(ies)	Biweekly and Ad-hoc Telcos, Slack, Emails, GitLab Issue Tracker				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				
Priority Level	Mandatory				
Identified by Partner(s)	ATOS				
Status	Fulfilled				
Comments/Remarks					

Requirement ID	TI7.5	Version	0.1	Last Update Date	27/01/2020
Title	Component maintenance				
Description	DEMETER components will be maintained by their corresponding developers to guarantee their correct functioning during the project.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.4, T3.5, T3.6				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	All WP3 modules/enablers				
Reference technology(ies)	N/A				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				
Priority Level	Mandatory				
Identified by Partner(s)	ATOS				
Status	Fulfilled				
Comments/Remarks					

Requirement ID	TI7.6	Version	0.1	Last Update Date	27/01/2020
Title	Service/application life-cycle management software suites				
Description	Service/application life-cycle management software suites will be used in DEMETER in order to ease the implementation of the life-cycle management methodology				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				

Relevant Task(s)	T3.2, T3.3, T3.4, T3.5, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Non-Functional
Priority Level	Optional
Identified by Partner(s)	ATOS
Status	Fulfilled
Comments/Remarks	

A.8. API's and Application development support

Requirement ID	TI8.1	Version	0.1	Last Update Date	27/01/2020	
Title	CRUD to HTTP methods mapping					
Description	DEMETER's API(s) should handle CRUD operations by proper mapping to HTTP methods which indicate the type of action to be performed on the resources.					
Relevant Pilot(s)	ALL					
Relevant Use Case(s)	ALL					
Relevant Task(s)	T3.2, T3.4, T3.5, T3.6					
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms					
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space					

Reference component(s)	BSE, FIE, DEH
Reference technology(ies)	HTTP(S) API
Involved stakeholders/ actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DNET, SIVECO
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI8.2	Version	0.1	Last Update Date	27/01/2020
Title	Proper HTTP response codes				
Description	DEMETER services should always return the right status codes. HTTP status codes are standardized codes which have various explanations in various scenarios.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	All WP3 modules/enablers				
Reference technology(ies)	HTTP(S) responses				
Involved stakeholders/ actors	Technology providers, Solution providers				

Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DNET, SIVECO
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI8.3	Version	0.1	Last Update Date	27/01/2020
Title	Searching, sorting, filtering, and pagination				
Description	DEMETER API(s) should support searching, sorting, filtering and pagination. GET requests over collection resources can potentially return a large number of items. Web API should limit the amount of data returned by any single request.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	BSE, FIE, DEH				
Reference technology(ies)	HTTP(S) API				
Involved stakeholders/ actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DNET, SIVECO				

Status	Fulfilled
Comments/Remarks	

Requirement ID	TI8.4	Version	0.1	Last Update Date	27/01/2020
Title	Stateless Authentication & Authorization				
Description	DEMETER API(s) should be stateless. Every request should be self-sufficient and must be fulfilled without knowledge of the prior request.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	ACS, AC Enabler				
Reference technology(ies)	TBD				
Involved stakeholders/ actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DNET, SIVECO				
Status	Fulfilled				
Comments/Remarks					

Requirement ID	TI8.5	Version	0.1	Last Update Date	27/01/2020
Title	Usage of Swagger for Documentation				
Description	DEMETER should use swagger for Documentation. Swagger is a widely used tool to document REST APIs that provides a way to explore the use				

	of a specific API. The Open API Initiative was created by an industry consortium to standardize REST API descriptions across vendors. As part of this initiative, the Swagger 2.0 specification was renamed the OpenAPI Specification (OAS) and brought under the Open API Initiative.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	FIE, BSE
Reference technology(ies)	Swagger
Involved stakeholders/ actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DNET, SIVECO
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI8.6	Version	0.1	Last Update Date	27/01/2020
Title	REST-based services				
Description	DEMETER should be able to support REST based web services. DEMETER Enablers should be able to consume and provide REST APIs				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.5, T3.6				

Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	All WP3 modules/enablers
Reference technology(ies)	HTTP(S) REST API
Involved stakeholders/ actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DNET, SIVECO
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI8.7	Version	0.1	Last Update Date	27/01/2020
Title	Access control mechanisms in API(s)				
Description	DEMETER should require access control mechanisms for the API(s) and allow access to the offered endpoints only to authorized users/clients (e.g. other DEMETER Enablers)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	All WP3 modules/enablers				
Reference	ACS				

technology(ies)	
Involved stakeholders/ actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI8.8	Version	0.1	Last Update Date	27/01/2020
Title	API and application documentation				
Description	DEMETER should provide documentation to assist developers in API and application development. Documentation (tutorials, videos, guidelines, code examples) should be available to all developers that would like to e.g. create a new DEMETER enabler.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control				
Reference component(s)	All WP3 modules/enablers				
Reference technology(ies)	GitLab README files, Wiki pages, Swagger				
Involved stakeholders/ actors	Technology providers, Solution providers				
Prerequisite(s)	None				

Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA
Status	Fulfilled
Comments/Remarks	

A.9. Enabler registration, discovery, provision, management, composition, accounting, billing

Requirement ID	TI9.1	Version	0.2	Last Update Date	04/02/2020
Title	DEH Resource registry				
Description	DEH should allow a provider to register offered services based on a registration description which DEH should provide. This could e.g. be in a NoSQL store to allow consumers more speed and flexibility in retrieving data.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 8: Unified agriculture ontology				
Reference component(s)	DEH – Resource Repository				
Reference technology(ies)	NoSQL data store				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by	ICCS				

Partner(s)	
Status	Fulfilled
Comments/Remarks	Seems ok (possible duplicate)

Requirement ID	TI9.2	Version	0.2	Last Update Date	04/02/2020
Title	Discovery Management				
Description	DEH web application should allow a consumer (end-users) to discover suitable resources (e.g. components, devices, services, data sources, platforms, etc.), returning correct outputs matching with their requirements (search API or tags). The discovery service should be based on a query, offering a wizard to help the user build her query.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Discovery Management				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	ICCS, ENG, DEH Survey				
Status	Fulfilled				
Comments/Remarks	Seems ok (1 comment and possible duplicate)				

Requirement ID	TI9.3	Version	0.2	Last Update Date	04/02/2020

Title	Query Management
Description	DEH should be able to temporarily store latest query of a consumer.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	Reference component module (or sub-module) in the DEMETER Architecture
Reference technology(ies)	DEH – Resource Registry Management
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	ICCS
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.4	Version	0.2	Last Update Date	04/02/2020
Title	Rate services				
Description	DEH should allow a consumer to rate the quality of the service it uses.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				

Reference component(s)	DEH – Resource Registry Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Fulfilled
Comments/Remarks	Seems ok (possible duplicate)

Requirement ID	TI9.5	Version	0.2	Last Update Date	04/02/2020
Title	Resource Access Control				
Description	DEH should mandate that access to its APIs will be secured through user authentication and access control. Furthermore, DEH should make the use of credentials between consumers and producers in order to authenticate the communication between them. Finally, this process should be reasonably simple for developers to use.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.4, T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Access Control				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				

Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Fulfilled
Comments/Remarks	Seems ok (possible duplicate)

Requirement ID	TI9.6	Version	0.2	Last Update Date	04/02/2020
Title	Query Management				
Description	DEH should always returns to the consumer the list of the most recent and updated resources available in the catalog after a resource discovery operation.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Optional				
Identified by Partner(s)	ICCS				
Status	Fulfilled				

Comments/Remarks	Seems ok

Requirement ID	TI9.7	Version	0.2	Last Update Date	04/02/2020
Title	Publish & Subscribe Notification				
Description	DEH should notify a consumer if a subscribed service is changed by its provider (e.g. service withdrawn, or conditions changed).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Optional				
Identified by Partner(s)	ICCS				
Status	Not implemented				
Comments/Remarks	Seems ok				

Requirement ID	TI9.8	Version	0.2	Last Update Date	04/02/2020
Title	Enablers Information Management				
Description	DEH should store information regarding the history of registration/provision and usage/consumption of enablers.				
Relevant Pilot(s)	ALL				

Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – Resource Registry Management, DEH Client Enabler
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.9	Version	0.2	Last Update Date	04/02/2020
Title	DEH Scalability & Availability				
Description	DEH could be scalable, allowing the increase of users (providers and/or consumers) it accommodates without impacting performance or availability.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				

Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Non-Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Fulfilled
Comments/Remarks	Seems ok (1 comment)

Requirement ID	TI9.10	Version	0.2	Last Update Date	04/02/2020
Title	Licensing				
Description	DEH should not be based on software that has IPR implications (or needs expensive licenses) thus blocking the provider/consumer ecosystem building.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				

Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.11	Version	0.2	Last Update Date	04/02/2020
Title	Data encryption in communications				
Description	DEH should ensure encrypted communication between the user and the web server that exposes its interfaces (web GUI). Furthermore, it should ensure that internal communication between its software components (and therefore its APIs) also transmits the data in an encrypted manner, especially when it comes to user sensitive data.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.4, T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Optional				
Identified by Partner(s)	ICCS + ENG				
Status	Fulfilled				
Comments/Remarks	(T3.4 to clarify) + (possible duplicate) + 1 comment				

Requirement ID	TI9.12	Version	0.2	Last Update Date	04/02/2020
Title	Service User Advisory				
Description	DEH could offer an “advisory” service in order to direct consumers towards contracting the appropriate services that they need.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.4, T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – User Account Control				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Optional				
Identified by Partner(s)	ICCS				
Status	Fulfilled				
Comments/Remarks	(T3.4 to clarify)				

Requirement ID	TI9.13	Version	0.2	Last Update Date	04/02/2020
Title	Accounting Management				
Description	DEMETER should provide accounting solution to allow users (consumers and producers) to create and send invoices to customers who purchase non-public resources available from the DEMETER Enabler Hub. An API framework could be provided for collecting accounting events, and also another API for users to interact with DEH				
Relevant Pilot(s)	ALL				

Relevant Use Case(s)	ALL
Relevant Task(s)	T3.4, T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH - Accounting
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	ICCS, ENG
Status	Not implemented
Comments/Remarks	(not to be implemented)

Requirement ID	TI9.14	Version	0.2	Last Update Date	04/02/2020
Title	Semantic Interoperability Framework				
Description	DEH must include advanced enablers for each device or external service that translates any data used by it to the Demeter AIM, when this data do not follow the AIM format natively; this ensure the necessary syntactic, semantic and technical interoperability of any Demeter-enabled applications.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				

Reference component(s)	DEH – Resource Registry Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.15	Version	0.2	Last Update Date	04/02/2020
Title	Application portability				
Description	It would be desirable for the hub app to be portable to other platforms such as iOS and android.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	TBD				
Reference technology(ies)	iOS				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				

Priority Level	Optional
Identified by Partner(s)	ICCS
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.16	Version	0.2	Last Update Date	04/02/2020
Title	System security services				
Description	Hub should consider the safety effects such as loss, damage or harm from an improper usage of the system, maintaining an expected integrity level. Also, there should be protection of the system from viruses, spyware, trojans and similar threats.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.4, T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				
Priority Level	Optional				
Identified by Partner(s)	ICCS				
Status	Not implemented				
Comments/Remarks	Check comment				

Requirement ID	TI9.17	Version	0.2	Last Update Date	04/02/2020
Title	System availability				
Description	DEH should guarantee response times for the app in the order of seconds, also considering the expected volume of request and use, even at peak times.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	ICCS, ENG				
Status	Fulfilled				
Comments/Remarks	Seems ok				

Requirement ID	TI9.18	Version	0.2	Last Update Date	04/02/2020
Title	External registration and provision management				
Description	Registration and provision management should be done from an external platform, IoT devices can be configured from external platform.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				

Relevant Task(s)	T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – Resource Registry Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Non-Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Fulfilled
Comments/Remarks	Check comment

Requirement ID	TI9.19	Version	0.2	Last Update Date	04/02/2020
Title	Data synchronization				
Description	DEH needs to support different technological solutions to allow resources registration, coming from the DEMETER Provider (each platform, thing, service or application which can be available as a resource) through an API-based framework to offer an entry point to the Platform for the applications and services that intend share and synchronize data.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management				

Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.20	Version	0.2	Last Update Date	04/02/2020
Title	Data federation				
Description	DEMETER needs to guarantee data federation techniques for API-based framework and technology tools to reduce data complexity.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T4.3				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	Adaptive Visualisation Framework				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				

Identified by Partner(s)	ENG
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.21	Version	0.2	Last Update Date	04/02/2020
Title	Technology specification				
Description	DEH should define general and high-level specification on technological composition of the DEH Resource Registry and the User Registry, or the main features to be supported.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry, User Registry				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	ENG				
Status	Fulfilled				
Comments/Remarks	Seems ok				

Requirement ID	TI9.22	Version	0.2	Last Update Date	04/02/2020

Title	DEH modules characteristic definition
Description	DEH should define in detail all the functions or at least the high-level definition of the main features that each module must support. The involved modules are: DEH Compatibility checker, Resource registry management, Resource access control, User Account Management, Discovery Management.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.1, T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – Resource Registry Management, DEH Compatibility Checker, Resource Access Control, User Account Control, Discovery Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI.23	Version	0.2	Last Update Date	04/02/2020
Title	Data management				
Description	DEH should offer means to have full control over all the services (such as compatibility checker, resource access controls, resource registry management, user account management), in order to get, add, update and delete their information (or entities).				

Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – Resource Registry Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.24	Version	0.2	Last Update Date	04/02/2020
Title	Data fusion				
Description	DEMETER needs to guarantee data fusion techniques for API-based framework and technology tools in order to produce a consistent data integration model (e.g. data coming from heterogeneous data sources).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.3				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				

Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.25	Version	0.2	Last Update Date	04/02/2020
Title	Monitoring & Audit				
Description	DEH management should offer means to monitor registered services and data sources workload.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management, Resource Access Control, User Account Control				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				

Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.26	Version	0.2	Last Update Date	04/02/2020
Title	Information Management				
Description	DEH should enable users (acting as DEMETER Providers) to register their offered resources (components, devices, services, data sources, platforms, etc.), recording attributes such as name, description, maturity level, tags, etc.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management				
Reference technology(ies)	Reference technology for the module or sub-module				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DEH Survey				
Status	Fulfilled				

Comments/Remarks	Seems ok

Requirement ID	TI9.27	Version	0.2	Last Update Date	04/02/2020
Title	Metadata collection				
Description	DEH should enable resources to be described by meta-data, which include the security and data usage policies applicable (provided by WP2).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.1				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DEH Survey				
Status	Fulfilled				
Comments/Remarks	Seems ok				

Requirement ID	TI9.28	Version	0.2	Last Update Date	04/02/2020
Title	Data Resource Definition				
Description	DEH should enable users to provide enablers either developed in the project or external ones (e.g. from third-party platforms).				
Relevant Pilot(s)	ALL				

Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – Resource Registry Management, Resource Access Control, User Account Control
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.29	Version	0.2	Last Update Date	04/02/2020
Title	Resource Management (CRUD operations)				
Description	DEH should enable users to add new resources anytime and edit them. It will be possible to see when the last edit related to the added resource was done.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.4, T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management, Resource Access Control				

Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.30	Version	0.2	Last Update Date	04/02/2020
Title	Web service interoperability				
Description	DEH should enable users to use web services or interoperability APIs (which use the HTTP protocol as data transport) to access the resources available to the DEH (USAGE API).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management, Resource Access Control, User Account Control				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				

Identified by Partner(s)	DEH Survey
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.31	Version	0.2	Last Update Date	04/02/2020
Title	Resource compatibility checker				
Description	DEH should enable users to integrate the available resources by allowing their compatibility checking (VALIDATION).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management, DEH Compatibility Checker				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DEH Survey				
Status	Fulfilled				
Comments/Remarks	Seems ok				

Requirement ID	TI9.32	Version	0.2	Last Update Date	04/02/2020
Title	Agriculture interoperability space resources				

Description	DEH should enable users to connect using a unique identifier their resources as part of the AIS.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – Resource Registry Management, User Account Control
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.33	Version	0.2	Last Update Date	04/02/2020
Title	Data Discovery Management				
Description	DEH should enable users to browse the DEH and to discover suitable resources matching challenge requirements (SOCS).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant	TBD				

Innovation(s)	
Reference component(s)	DEH – Resource Discovery Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.34	Version	0.2	Last Update Date	04/02/2020
Title	Rating service				
Description	DEH web application should enable users to rate used components, services or enablers.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management, DEH Dashboard				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				

Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.35	Version	0.2	Last Update Date	04/02/2020
Title	Resource download report				
Description	DEH should enable users to view download statistic statistics on registered components.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DEH Survey				
Status	Fulfilled				
Comments/Remarks	Seems ok				

Requirement ID	TI9.36	Version	0.2	Last Update Date	04/02/2020
Title	Collection of enablers system				
Description	DEH should enable the design of a system (collection) of enablers and services to help users (or developers) who fuse together such enablers in order to provide a whole system which can then be sold to other users (e.g. farmers).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				
Priority Level	Optional				
Identified by Partner(s)	DEH Survey - proposed features				
Status	Fulfilled				
Comments/Remarks	Seems ok				

Requirement ID	TI9.37	Version	0.2	Last Update Date	04/02/2020
Title	User profile management				
Description	DEMETER should offer user’s profile service management.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				

Relevant Task(s)	T3.4
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	ACS – Access Control Server
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	DEH Survey - proposed features
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.38	Version	0.2	Last Update Date	04/02/2020
Title	Responsive web GUI				
Description	DEH web application should be accessible via a web browser or smartphone/tablet, without requiring any client software installation.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				
Reference technology(ies)	TBD				

Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Fulfilled
Comments/Remarks	Seems ok (possible duplicate)

Requirement ID	TI9.39	Version	0.2	Last Update Date	04/02/2020
Title	User account management				
Description	DEH web application should have a user registration/login section.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – User Account Control				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DEH Survey				
Status	Fulfilled				

Comments/Remarks	Seems ok

Requirement ID	TI9.40	Version	0.2	Last Update Date	04/02/2020
Title	User private home page				
Description	DEH web application should have a home page with description of the main functionalities.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DEH Survey				
Status	Fulfilled				
Comments/Remarks	Seems ok				

Requirement ID	TI9.41	Version	0.2	Last Update Date	04/02/2020
Title	User registration web page				
Description	DEH web application should have a page to register new resources or edit the already registered ones.				
Relevant Pilot(s)	ALL				

Relevant Use Case(s)	ALL
Relevant Task(s)	T3.4, T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – User Account Control
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.42	Version	0.2	Last Update Date	04/02/2020
Title	Resources Management web page				
Description	DEH web application should have a page for each resource.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	TBD				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – User Account Control				
Reference	TBD				

technology(ies)	
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.43	Version	0.2	Last Update Date	04/02/2020
Title	Interoperability marketplace and catalogues solution				
Description	DEH web application should include the interaction with other initiatives which provide catalogues and marketplaces of solutions, as well as independent (INTEROPERABILITY).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management, Resource Access Control, User Account Control				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				

Identified by Partner(s)	DEH Survey
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	TI9.44	Version	0.2	Last Update Date	04/02/2020
Title	DEH solutions web page				
Description	DEH web application should have a page to register SOLUTIONS and associate to the solution a group of resources also their categories.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management, Resource Access Control, User Account Control				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DEH Survey - proposed features				
Status	Fulfilled				
Comments/Remarks	Seems ok				

Requirement ID	TI9.45	Version	0.2	Last Update Date	04/02/2020
Title	Team services				

Description	Localisation service will be needed in order to filter resources near user Consumer or Provider.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Optional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey - proposed features
Status	Fulfilled
Comments/Remarks	Seems ok

A.10. Stakeholder account management

Requirement ID	TI10.1	Version	0.1	Last Update Date	27/01/2020
Title	Stakeholder access				
Description	DEMETER should define different roles with which various stakeholders will get access to the system.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models				

	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	ACS, DEH
Reference technology(ies)	ACS
Involved stakeholder s/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner (s)	INTRA
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI10.2	Version	0.1	Last Update Date	27/01/2020
Title	Account management roles functionality				
Description	Different account management roles in DEMETER should correspond to different functionality				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	BSE, FIE, ACS, DEH				
Reference technology(ies)	ACS				

Involved stakeholders/ actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA
Status	Fulfilled
Comments/Remarks	

Requirement ID	TI10.3	Version	0.1	Last Update Date	27/01/2020
Title	Distinguishing a) internal and external stakeholders and b) primary and secondary stakeholders				
Description	DEMETER account management roles should distinguish between internal and external stakeholders, and between primary and secondary stakeholders				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/ actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				

Identified by Partner(s)	INTRA
Status	Not implemented
Comments/Remarks	No need for such categorization among the stakeholders emerged

Requirement ID	TI10.4	Version	0.1	Last Update Date	27/01/2020
Title	Stakeholders' categorization				
Description	DEMETER account management should categorize the following stakeholders into different role groups: 1. Provider, 2. Consumer, 3. Administrator				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	ACS, DEH				
Reference technology(ies)	ACS				
Involved stakeholders/ actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	INTRA				
Status	Fulfilled				
Comments/Remarks					

A.11. Monitoring, Awareness, Feedback

Requirement ID	TI11.1	Version	0.1	Last Update Date	27/01/2020
Title	Feedback from end-users				
Description	DEMETER should provide solutions to gather feedback from farmers and end-users				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.3, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 2: Stakeholder Open Collaboration Space				
Reference component(s)	DEMETER collaboration tool, DEH				
Reference technology(ies)	GitLab Issue tracker, DEH rating functionality				
Involved stakeholders/ actors	Technology providers, Solution providers, End-users, Farmers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	TECNALIA				
Status	Fulfilled				
Comments/Remarks					

Requirement ID	TI11.2	Version	0.1	Last Update Date	27/01/2020
Title	Upvoting mechanism				
Description	DEMETER should provide a way for users to upvote a service (introduce a popularity indicator)				

Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 2: Stakeholder Open Collaboration Space
Reference component(s)	DEH
Reference technology(ies)	DEH rating functionality
Involved stakeholders/ actors	Technology providers, Solution providers, End-users, Farmers
Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	INTRA
Status	Fulfilled
Comments/Remarks	

A.12. General Non-Functional Requirements

Requirement ID	GNFR.1	Version	0.2	Last Update Date	04/02/2020
Title	Business analytic data visualization suite				
Description	DEMETER needs to provide appropriate mechanism for the visualization of data coming from heterogeneous sources such as Big data systems, data warehouses, relational databases (or NoSQL) and web services that supply the data.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T4.3				

Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms O3: Empower the farmer, as a prosumer O6: Ease and streamline mechanisms for all stakeholders
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards Innovation 6: Performance evaluation of Decision Support Systems
Reference component(s)	BID (Business Intelligence Dashboard Tool)
Reference technology(ies)	KNOWAGE (Open Source Suite for any modern Business Analytics)
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Not implemented
Comments/Remarks	This is more related to WP4 and covered by KNOWAGE

Requirement ID	GNFR.2	Version	0.2	Last Update Date	04/02/2020
Title	Decision Support System Dashboards				
Description	DEMETER needs to provide user interfaces that enable users to interactively explore and analyze digital data, allowing them to effectively identify interesting patterns, infer correlations and causalities, and supports sense-making activities.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5, T4.3, T4.5				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				

	O3: Empower the farmer, as a prosumer O6: Ease and streamline mechanisms for all stakeholders
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards Innovation 6: Performance evaluation of Decision Support Systems
Reference component(s)	1. SOCS (Stakeholder Open Collaboration Space Implementation) 2. DEH (DEMETER Hub) 3. Adaptive Visualisation for Dashboard
Reference technology(ies)	1. KNOWAGE (Open Source Suite for any modern Business Analytics)
Involved stakeholders/actors	Technology providers, Solution providers, Farmer, Advisors, Researchers, Interest groups
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	GNFR.3	Version	0.2	Last Update Date	04/02/2020
Title	Web applications usability				
Description	DEMETER needs to ensure the usability feature of user interfaces to cover all aspects of the user's experience when interacting with the DEMETER data visualization tools and with its graphical interfaces or web GUI.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5, T4.3, T4.5				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models				
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards				
Reference component(s)	1. SOCS (Stakeholder Open Collaboration Space Implementation) 2. DEH (DEMETER Hub)				

	3. Adaptive Visualisation for Dashboard
Reference technology(ies)	1. KNOWAGE (Open Source Suite for any modern Business Analytics)
Involved stakeholders/actors	Technology providers, Solution providers, Farmer, Advisors, Researchers, Interest groups
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Fulfilled
Comments/Remarks	Seems ok (1 comment)

Requirement ID	GNFR.4	Version	0.2	Last Update Date	04/02/2020
Title	Web application stylesheet				
Description	DEMETER needs to guarantee an appropriate Look & Feel for its user interfaces (e.g. web GUI) so they satisfy the user needs both in terms of visual appearance (look) and that of user interaction (feel).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5, T4.3, T4.5				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models				
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards				
Reference component(s)	1. SOCS (Stakeholder Open Collaboration Space Implementation) 2. DEH (DEMETER Hub) 3. Adaptive Visualisation for Dashboard				
Reference technology(ies)	1. KNOWAGE (Open Source Suite for any modern Business Analytics)				
Involved stakeholders/actors	Technology providers, Solution providers, Farmer, Advisors, Researchers, Interest groups				
Prerequisite(s)	None				

Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Fulfilled
Comments/Remarks	Seems ok (1 comment)

Requirement ID	GNFR.5	Version	0.2	Last Update Date	04/02/2020
Title	Web application friendliness				
Description	DEMETER needs to guarantee the friendliness of user interfaces (e.g. web GUI) in order to ease the use of the provided features or visualizations. In order to cover this feature, the DEMETER user interfaces should satisfy the following criteria: a. Be intuitive, i.e. graphical interfaces should be well designed, b. Easy-to-navigate c. Easy to update and remove d. Should not require third-party installation software, e. Manage errors effectively (simply, the user should always be notified of the software malfunction through targeted alerts that show the user what is going on and indicate a possible solution to solve it).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5, T4.3, T4.5				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models				
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards				
Reference component(s)	1. SOCS (Stakeholder Open Collaboration Space Implementation) 2. DEH (DEMETER Hub) 3. Adaptive Visualisation for Dashboard				
Reference technology(ies)	1. KNOWAGE (Open Source Suite for any modern Business Analytics)				
Involved stakeholders/actors	Technology providers, Solution providers, Farmer, Advisors, Researchers, Interest groups				

Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Fulfilled
Comments/Remarks	Seems ok (1 comment)

Requirement ID	GNFR.6	Version	0.2	Last Update Date	04/02/2020
Title	Business analytic data visualization suite				
Description	DEMETER needs to guarantee valid approach in exploiting Big Data sources. This approach will need to support users browsing, understanding and discovering data insights. Nonetheless, Big data characteristics, such as volume, variety and velocity pose a number of challenges for visualization. Indeed, current visualization and exploration systems should effectively and efficiently handle the following aspects: a. Real-time Interaction. Efficient and scalable techniques should support the interaction with datasets, while maintaining the system response in the range of a few seconds, b. On-the-fly Processing. Support of on-the-fly visualizations over dynamic sets of volatile raw (i.e., not preprocessed) data, c. Visual Scalability. Provision of effective data abstraction mechanisms is necessary for addressing problems related to visual information overloading, d. User Assistance and Personalization. Encouraging user comprehension and offering customization capabilities to different user-defined exploration scenarios and preferences according to the analysis needs are important features.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5, T4.3				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms O3: Empower the farmer, as a prosumer O6: Ease and streamline mechanisms for all stakeholders				

Relevant Innovation(s)	Innovation 5: Farm enabler dashboards Innovation 6: Performance evaluation of Decision Support Systems
Reference component(s)	1. DEH (DEMETER Hub) 2. AIS (Agriculture Interoperability Space) 3. Adaptive Visualisation for Dashboard
Reference technology(ies)	1. KNOWAGE (Open Source Suite for any modern Business Analytics)
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Desirable
Identified by Partner(s)	ENG, ICE
Status	Fulfilled
Comments/Remarks	Seems ok

Requirement ID	GNFR.7	Version	0.2	Last Update Date	04/02/2020
Title	DSS dashboard outcomes data visualization				
Description	DEMETER needs to support the visualization needs in the outcomes of some DEMETER tasks such as data analytics (WP2), decision making services (T4.1), benchmarking techniques (T4.2) and workflows of enablers (T4.4): a. Regarding data analytics, the analytic services need visualization support to provide better understanding of data structures and meaningful insight i.e. the outcome of the data analytic services b. Regarding benchmarking techniques, DEMETER could benefit from a user dashboard (Web GUI) that allow companies to consult and compare the outcome of DEMTER DSS with other DSS and also provide guidance on the choice of different technologies c. Regarding decision making, DEMETER decision support services would need appropriate visualization support to present decision support to the users. The visualization can present decision support in the form of alerts, reports, comparisons, performance KPIs, historic analysis etc.				

	d. Regarding workflows of enablers, the decision support enablers will require visualisation to manage the data flows as well as reporting the outcomes of the actual processing taking place within certain enablers.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T4.1, T4.2, T4.3, T4.4
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms O3: Empower the farmer, as a prosumer O6: Ease and streamline mechanisms for all stakeholders
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards Innovation 6: Performance evaluation of Decision Support Systems
Reference component(s)	
Reference technology(ies)	KNOWAGE (Open Source Suite for any modern Business Analytics)
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG, ICE
Status	Proposed
Comments/Remarks	This is more related to WP4 and is part of KNOWAGE

Requirement ID	GNFR.8	Version	0.2	Last Update Date	04/02/2020
Title	DSS dashboard notification				
Description	DEMETER needs to guarantee a good standard in data visualization in a substantial number of Decision Support System (or optimize the existing DSS) in order to give suitable advice, recommendations and automated				

	actions to end-users (e.g. farmers, dairy companies). This will allow the visualizations to be used by other pilots and to increase the interoperability between the Pilots through the use of common visualizations that address their potential needs e.g.: a. DSS for cost optimization (e.g. linking economical aspects of wholesale and retail prices) b. DSS for production preferences (e.g. the use of non-chemical pesticides, attention to animal welfare and tracking, transparency to the consumers) c. DSS for cost/benefit analysis of field operations (irrigation/fertilization) d. DSS for optimization on irrigation/fertilization strategies, disease monitoring, yield analysis (e.g. the estimation of crop yield according to climate conditions) e. DSS for the forecasting of phytopathologies on crops f. DSS for animal welfare tracking g. DSS to support the farmers for live support of agricultural processes.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T4.1, T4.2, T4.3, T4.4
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms O3: Empower the farmer, as a prosumer O6: Ease and streamline mechanisms for all stakeholders
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards Innovation 6: Performance evaluation of Decision Support Systems
Reference component(s)	
Reference technology(ies)	KNOWAGE (Open Source Suite for any modern Business Analytics)
Involved stakeholders/actors	Technology providers, Solution providers, Farmer, Advisors, Researchers, Interest groups
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory

Identified by Partner(s)	ENG, ICE
Status	Proposed
Comments/Remarks	This is more related to WP4 and is part ofKNOWAGE

Requirement ID	GNFR.9	Version	0.2	Last Update Date	04/02/2020
Title	DSS Dashboard widget				
Description	1. DEMETER needs to guarantee a data visualization tool with the following features in terms of graphical widgets for displaying data (at least): b. Text c. Image d. Chart e. HTML f. Table g. DSS for animal welfare tracking h. DSS to support the farmers for live support of agricultural processes.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T4.3				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models				
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards				
Reference component(s)					
Reference technology(ies)	KNOWAGE (Open Source Suite for any modern Business Analytics)				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				

Identified by Partner(s)	ENG
Status	Proposed
Comments/Remarks	This is more related to WP4 and is part of KNOWAGE

Annex B DEMETER Enabler Template

B.1. Text information – metadata

We need this information as metadata, for BSE/DEH descriptions or other components

B.1.1. Functionality description

Describe the functionality of the enabler

B.1.2. Interaction with other Enablers

Describe how the Enablers functionality is combined with other Enablers. E.g. How will the Security Enabler be utilized by other Enablers? Will it be accessible by all Enablers or just by the Communication/Networking enabler for example?

B.1.3. Dependencies on other Core/Advanced Enablers

Describe any dependencies on other Core/Advanced Enablers. Which Enablers' APIs are implemented by this Enabler?

B.1.4. Deployment considerations

Describe consideration related to deployment, e.g., where the image will reside, how access will be provided, resources required, etc.

B.2. Technical description

This information formally describes features/characteristics of an Enabler

B.2.1. API and Data model

Describe the API and the Data model of the enabler in a technical way. E.g., a list of endpoints and their description using Swagger documentation, a list of topics to access in case of MQTT, NGSI-LD payload, etc.

Data models used by the APIs shall be described in tables:

Name	This field holds the name of the data model that is described in this table	
Property	Type	Description

Developers are strongly advised to adopt Swagger for online documentation of the developed APIs. If Swagger (or any other online documentation tool) is being adopted, additionally, provide here Swagger details for the online documentation.

(REST API)

Title	This field holds the description of the API
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off	

from this description and can be placed as a hypertext above the API template	
Method This field holds the type of the Method used	
GET | POST | DELETE | PUT	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
id=[integer]	parameter description
Optional:	
image_id=[alphanumeric]	parameter description
Data Params This field holds the body payload of a post request.	
Required:	
id=[integer]	parameter description
Optional:	
image_id=[alphanumeric]	parameter description
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	
200 Content: { }	response description
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	response description
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
Notes This field holds any additional helpful info related to this endpoint.	

(publish/subscribe API)

Title	This field holds the description of the API
URL: This field holds the relative URL to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
Topic	This field holds the topic identifier (uid/path/name)
Payload request/response	This field holds the format type payload of the message (e.g JSON, NGSI-LD)
Payload representation request/response: This field holds the structure of the payload used	

{
  "id": {string},
  "type": {object},
  "name": {string},
  "value": {string},
   "…" : "…"
}	

Payload Property description	Please write here the Data Model table that describe the payload

	properties
Required parameters (request)	This field holds the required parameters
Required parameters (response)	This field holds the required parameters
Success response This field holds the list of all possible successful responses	
Error response This field holds the list of all possible error responses	
Sample call This field holds a possible sample call to the described topic. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
Notes This field holds any additional helpful info related to this endpoint.	

B.2.2. **Use cases / Data flow**

Technically describe use cases of the enabler and the data flow using formal UML activity and sequence diagrams

UML Activity diagram(s)

Place activity diagram(s) here

UML Sequence diagram(s)

Place sequence diagram(s) here

UML Component diagram(s)

B.2.3. **Deployment**

Technically describe the deployment process for the enabler using a Docker-compose script and the deployment execution commands.

```
version: '3'
services:
  db:
     image: mysql
     container_name: mysql_db
     restart: always
     environment:
        - MYSQL_ROOT_PASSWORD="secret"
  web:
    image: apache
    build: ./webapp
    depends_on:
      - db
    container_name: apache_web
    restart: always
    ports:
      - "8080:80"
```

Deploy by:

```
sudo docker-compose up --build -d
```

B.2.4. **Configuration Parameters**

Describe all configuration parameters that can be provided by a user/developer (mandatory/optional). These could be defined as env vars in the docker-compose script provided above. Examples could be external component URLs, IPs, ports, SSL params

Configuration parameter	Value	Type	Description
MYSQL_ROOT_PASSWORD	secret	String	Your MySQL password

Annex C Module & Enabler Verification template

C.1. Module Validation Report

Table 29, below tabulates the general information of a DEMETER module/enabler that is deployed and validated. For more information regarding the validation process please refer to section **Error! Reference source not found.** of D3.2.

Table 29: Component's general description

Title	This field holds the name of the DEMETER module	WP	This field holds the WP that the module belongs
Module Identifier	This field holds a short identifier for the module (e.g., BSE)		
Description	This field holds the module's operation description		
Repository type	This field holds the repository type of the source code of the component. (e.g., Private, DEMETER GitLab)	Justification	This field holds the justification of the source code repository type selection
Repository URL	If the Repository type is in DEMETER's GitLab, then the absolute URL of the module's location must be filled in here		
Features' list	The list of the functionalities provided by the module		
Integration component list	This field holds the components list that this module interoperates and will integrate with		
Deployment location	This field holds deployment location (e.g., DEMETER cloud infrastructure, DEMETER's pilot infrastructure, proprietary location)		
Docker container location and size	If the component is containerized, then please fill in the location of the docker registry that resides and the size of the docker container		
Requirements	This field holds computational requirements for this module. Among others, you can describe here the CPU, RAM, STORAGE requirements of the component.		
Contact email	This field holds the email of the developer of the module.		

C.1.1. Module technical description

In this section, you can give a brief description of the module implementation and operation logic. Also, among others, you can include installation instructions, docker related info or any other helpful information. In most cases you shall elaborate a little bit if necessary, on the information that is described in source code repository (e.g., README file, etc.).

C.1.2. **Data Models and Interfaces**

In this section, you are encouraged to add/update the Interfaces (APIs) and used Data Models that are described in the module's relevant deliverable. For the sake of completeness, we append below the relevant templates.

Data Model description

Name	Data Model	
Property	Type	Description

API description

Title	This field holds the description of the API
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
Method This field holds the type of the Method used	
GET | POST | DELETE | PUT	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
id=[integer]	parameter description
Optional:	
image_id=[alphanumeric]	parameter description
Data Params This field holds the body payload of a post request.	
Required:	
id=[integer]	parameter description
Optional:	
image_id=[alphanumeric]	parameter description
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	
200 Content: { }	response description
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	response description
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
Notes This field holds any additional helpful info related to this endpoint.	

Publish/Subscribe API

Title	This field holds the description of the API
URL: This field holds the relative URL to the described API. For simplicity Root path can be cut off from	

this description and can be placed as a hypertext above the API template	
Topic	This field holds the topic identifier (uid/path/name)
Payload request/response	This field holds the format type payload of the message (e.g JSON, NGSI-LD)
Payload representation request/response: This field holds the structure of the payload used	
{ "id": {string}, "type": {object}, "name": {string}, "value": {string}, "…" : "…" }	
Payload Property description	Please write here the Data Model table that describe the payload properties
Required parameters (request)	This field holds the required parameters
Required parameters (response)	This field holds the required parameters
Success response This field holds the list of all possible successful responses	
Error response This field holds the list of all possible error responses	
Sample call This field holds a possible sample call to the described topic. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
Notes This field holds any additional helpful info related to this endpoint.	

C.1.3. **Functionality and Integration Tests**

Functionality tests verify that your module sufficiently covers the features needed, as expressed in previously defined requirements. Integration tests verify that your code works with external dependencies correctly. Tests shall be described and include/adhere the following categories:

1. **Functionality Tests:** Validating the functionality provided by the modules to be integrated. Test all the test cases for the chosen module. Provide in a tabulated format the summary of the functionalities provided by the module. Indicate, whether a specific functionality (feature) is fully implemented, partially implemented, or not implemented.

Test ID	**<MODULE_IDENTIFIER_FTCXX> (e.g., <BSE_FTC01>)**
Test description	
Related requirements	(Have a look at Annex A)
Feature(s) under test	(Related to the Module feature list)
Test environment	(e.g., lab, DEMETER cloud, pilot cloud, etc)
Dependencies	(Any dependencies on other modules, environments, etc)
Steps	1. Step 1 2. Step 2

	3. …
Pass criteria	(What qualifies this test as passed)

2. **Integration Tests:** Validating the interconnection between the modules to be integrated. Provide in a tabulated format the summary of the integration activities performed. Indicate whether an integration activity is successful, partially successful, not successful. You should also indicate the Pilot/infrastructure that where these tests were executed.

Test ID	**<MODULE_IDENTIFIER_ITCXX> (e.g., <BSE_ITC01>)**
Test description	
Module(s) under test	(Related to the Integration component list)
Module(s) under test environment	(e.g., DEMETER cloud, pilot cloud, etc)
Test environment	(e.g., lab, DEMETER cloud, pilot cloud, etc)
Dependencies	(Any dependencies on other modules, environments, etc)
Steps	1. Step 1 2. Step 2 3. …
Pass criteria	(What qualifies this test as passed)

3. **Security Tests:** Validating principles such as data integrity, user authorization where applicable etc. For example, testing module access through the ACS module.

Test ID	**<MODULE_IDENTIFIER_STCXX> (e.g., <BSE_STC01>)**
Test description	
Test environment	(e.g., lab, DEMETER cloud, pilot cloud, etc)
Dependencies	(Any dependencies on other modules, environments, etc. E.g., ACS module)
Steps	1. Step 1 2. Step 2 3. …
Pass criteria	(What qualifies this test as passed)

C.2. Validation summary

In this section, each partner shall include a summary of the performed functionality and integration tests for this module (how many tests per type have been performed, how many were passed, where there any failed tests, what was the coverage of the functionality, etc). Please also include any useful comments about the testing and the integration process. Integration is considered successful when:

- The previous descriptions have been submitted.
- Sufficient functionality and integration tests have been carried out providing adequate coverage of the functionality provided by each module and interconnection with other core DEMETER modules.

C.3. Module KPIs

In this section, each partner will need to include module KPIs (e.g., System performance), *if applicable*. This section **does not** concern business KPIs, but technical-oriented ones.

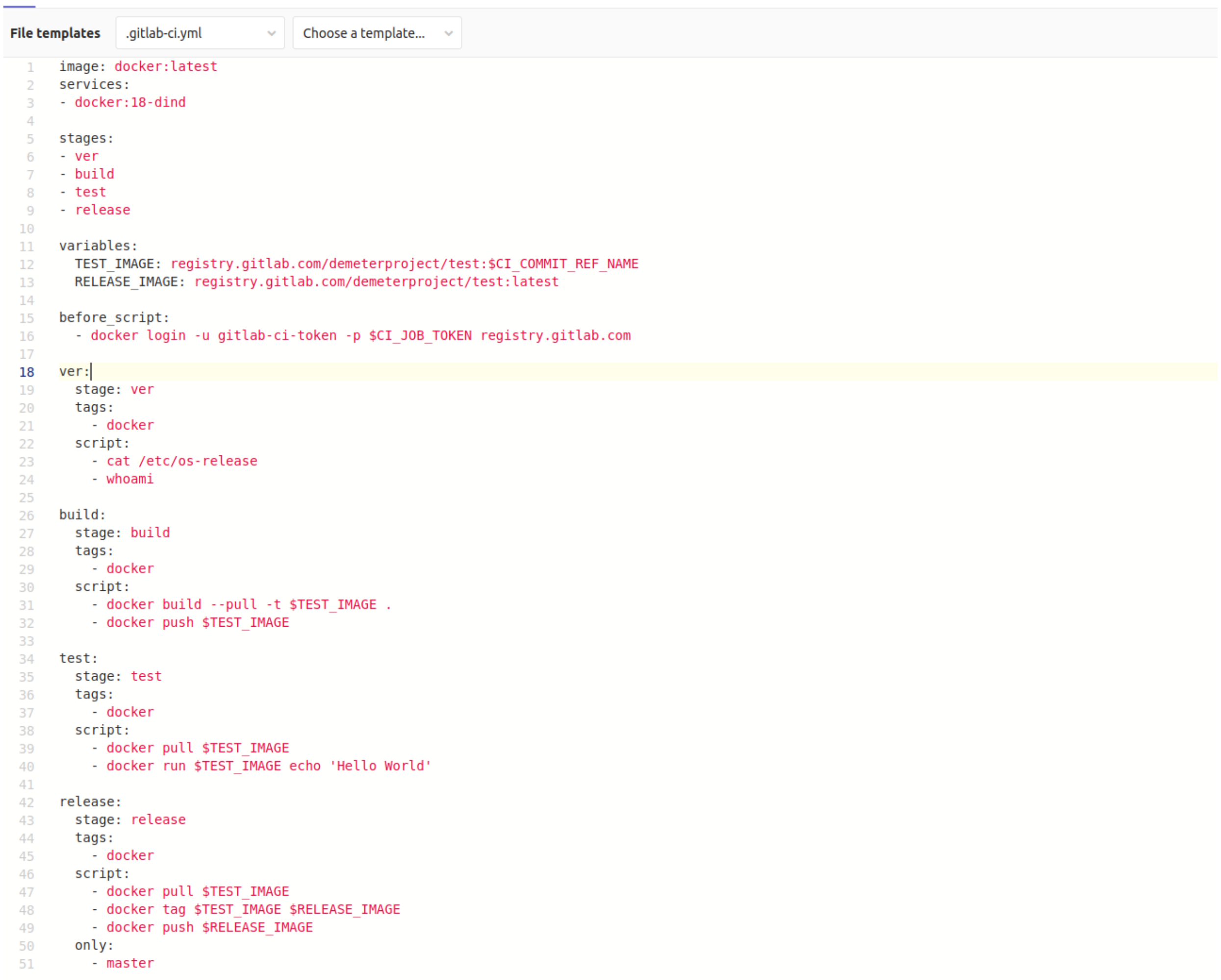
KPI	Name	Description	Metric	Method of measurement	Target	Result
System performance						
<KPI_MODULE_IDENTIFIER_XX> (e.g., KPI_BSE_01)	(the name of the KPI, e.g., Fetch services time)	(description of the KPI, e.g., time to fetch services data from registry)	(e.g., Time in seconds)	(method to measure if the KPI was reached, e.g., measure the total time required to process a request)	(KPI target, e.g., <1min)	**TO BE FILLED IN THE LAST DELIVERABLE**

C.4. DEMETER Cloud Testing environment

DEMETER's main modules and enablers can **optionally** be tested using DEMETER's CI/CD environment (described in D3.2 section 10).

IMPORTANT: **Regardless of whether DEMETER's CI/CD or on-premises environment is used for testing, it is important to use BSE testing environment for any integration tests, i.e, vm1.test.h2020-demeter-cloud.eu and not the production environment, as it might cause problems with actual users. So, bse.h2020-demeter-cloud.eu should NOT be used for testing.**

Gitlab's CI/CD framework uses pipelines to automate integration and deployment processes. Pipelines are defined and described in script files (.gitlab-ci.yml files, an example available in Figure 80), each of them representing a "job", including various pipelines organized in stages.



```yaml
image: docker:latest
services:
- docker:18-dind

stages:
- ver
- build
- test
- release

variables:
  TEST_IMAGE: registry.gitlab.com/demeterproject/test:$CI_COMMIT_REF_NAME
  RELEASE_IMAGE: registry.gitlab.com/demeterproject/test:latest

before_script:
  - docker login -u gitlab-ci-token -p $CI_JOB_TOKEN registry.gitlab.com

ver:
  stage: ver
  tags:
    - docker
  script:
    - cat /etc/os-release
    - whoami

build:
  stage: build
  tags:
    - docker
  script:
    - docker build --pull -t $TEST_IMAGE .
    - docker push $TEST_IMAGE

test:
  stage: test
  tags:
    - docker
  script:
    - docker pull $TEST_IMAGE
    - docker run $TEST_IMAGE echo 'Hello World'

release:
  stage: release
  tags:
    - docker
  script:
    - docker pull $TEST_IMAGE
    - docker tag $TEST_IMAGE $RELEASE_IMAGE
    - docker push $RELEASE_IMAGE
  only:
    - master
```

Figure 80: Example job file

Each job is assigned to a Gitlab runner to be executed. Gitlab runners are merely (client) Gitlab services that run on private or public infrastructure, connect to a public or private Gitlab instance, and execute the jobs described in the job files (building, testing, deploying). Runners execute the jobs in Docker containers while they also run as Docker containers, hence, in DEMETER we are using a Docker-in-Docker paradigm for the Runners we use. This enables as to achieve higher utilization of our cloud infrastructure resources. Upon the execution of the jobs, GitLab offers a reporting tool to the developers to inspect all job stages.