EOPEN

opEn interOperable Platform for unified access and analysis of Earth observatioN data

H2020-776019

D6.1

System Requirements and Architecture

Dissemination level:	Public
Contractual date of delivery:	31/10/2018
Actual date of delivery:	05/11/2018 (First issue) 20/01/2020 (Second issue)
Work package:	WP6 System Development and Integration
Task:	T6.1 Systems Requirements and Architecture
Type:	Report
Approval Status:	Approved
Version:	2.1
Number of pages:	50
Filename:	D6.1-System Requirements and Architecture v2.1.docx

Abstract

The main purpose of this document is to describe the technical specifications, including the System and the Application Requirements, as stemming from D2.1 and D2.2, and foreseen in the GA, in addition to the Architectural Design of the EOPEN Platform.

The content of this document is the result of the analysis of the needs in terms of infrastructure and capabilities on three levels: the core building blocks and the extended features of the platform and the Use Case driven user requirements.

Version 2 includes two new chapters: Chapter 4 is focussed on the 1st and the 2nd integrated prototype modules, workflows and capabilities; Chapter 5 provides an overview of the Pilot Use Cases and of the platform features exploited for their implementation. Both chapters have been added also in reply to comments received at the first review about the system IT and PUC architecture.

The information in this document reflects only the author's views and the European Community is not liable for any use that may be made of the information contained therein. The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

co-funded by the European Union

History

Version	Date	Reason	Revised by	Approved By
1.0	13-Nov-2018	First release	Authors	Leslie Gale
2.1	20-Jan-2020	Second release	Authors	Leslie Gale

Author list

Organisation	Name	Contact Information
SpaceApps	Bernard Valentin	bernard.valentin@spaceapplications.com
SpaceApps	Leslie Gale	leslie.gale@spaceapplications.com
SpaceApps	Hakim Boulahya	hakim.boulahya@spaceapplication.com
AAWA	Michele Ferri	Michele.ferri@abdve.it
AAWA	Daniele Norbiato	Daniele.norbiato@abdve.it
AAWA	Francesco Zaffanella	Francesco.Zaffanella@abdve.it
NOA	Anestis Trypitsidis	Vsito@noa.gr
Serco	Tudor Pettengell	Tudor.Pettengell@serco.com
Serco	Laurence Marzell	Laurence.Marzell@serco.com
Serco	Gabriella Scarpino	Gabriella.Scarpino@serco.com
KU-eGISRS	Woo-Kyun Lee	leewk@korea.ac.kr
KU-eGISRS	Wona Lee	supeanut@naver.com
NOA	Anestis Trypitsidis	atrypitsdis@noa.gr
NOA	Vassilis Sitokonstantinou	Vsito@noa.gr
NOA	Charalampos (Haris) Kontoes	kontoes@noa.gr
FMI	Amara Graps	graps@psi.edu
FMI	Petteri Karsisto	petteri.karsisto@fmi.fi

Executive Summary

This deliverable presents the result of the analysis of Pilot Use Cases to extract and define the requirements for the EOPEN platform and its applications.

The analysis takes into account the roles of the actors and stakeholders within EOPEN for which the Pilot Use Cases have been specified. In particular the objective of the requirements is to meet the needs of the *Users* with the support of the *Service Providers* who operate in the appropriate application domain. The *Service Providers* who are responsible for implementing the services to support the PUCs rely on the support from the *Data and Information Experts*. These interpret the EO and other data streams to provide the data supporting the analysis and decision making processes of the PUCs. The *Platform Experts* take the combined processing and operational needs to create the EOPEN platform core services.

PUCs have been written without consideration of the roles and the EOPEN platform concept that distinguishes between core requirements specific to the platform. In the analysis the requirements have been categorised into three groups:

1. Application Requirements, to be implemented by the Service Providers
2. Extension requirements to be implemented by the Data and Information Experts
3. Core Requirements to be implemented by the Platform Experts.

A provisional description of the distributed architecture is presented. In particular, it foresees the deployment of the core EOPEN Platform components in Space Applications Services' infrastructure, and the dynamic deployment of processing services in SpaceApps, HLRS (HPC) and DIAS (including ONDA) environments. This is further described in section 3.1.

This document contains:

- Chapter 1, an introduction to the document
- Chapter 2, the technical specifications comprising:
    - Core Requirements
    - Extension Requirements
    - Application Requirements
- Chapter 3, hardware and software architecture of the core platform
- Chapter 4, description of the integrated prototypes
- Chapter 5, description of the pilot use cases

Abbreviations and Acronyms

ARD	Association of public service broadcasters in Germany
ARD	Analysis Ready Data
BMCO	Broadcast Mobile Convergence
COM	Current Operating Model
CES	Community Environmental Support
CPU	Central Processing Unit
CSW	Catalogue Service for the Web
DAML	DARPA Agent Markup Language
DID	Digital Item Definition
DII	Digital Item Identification
DRM	Digital Rights Management
EBU	European Broadcast Union
EO	Earth Observation
ETSI	European Telecommunications Standards Institute
EWS	Early Warning System
GA	Grant Agreement
GIS	Geographical Information System
GPU	Graphics Processing Unit
HPC	High Performance Computing
HPDA	High Performance Data Analytics
ICF	Informed consents form
IS	Information sheet
IEEE	Institute of Electrical and Electronics Engineers
IP	Integrated Project
IPTC	International Press Telecommunications Council
ISO	International Organization for Standardization
IST	Information Society Technologies
JDIG	Joint Decision & Information Governance
JPEG	Joint Photographic Experts Group
JSON	JavaScript Object Notation
KR	Key Result
MAF	Multimedia Application Format
ML	Machine Learning
MoSCoW	Must have, Should have, Could have, Won't have (this time)
MPEG	Moving Picture Experts Group
NEC	Non-European Countries
NITF	News Industry Text Format
NLP	Natural Language Processing
NN	Neural Network
NoE	Network of Excellence
OGC	Open Geospatial Consortium
OWL	Ontology Web Language
OWL-QL	Ontology Web Language Query Language
OWL-DL	Ontology Web Language Description Language

PUC	Pilot Use Case
R2RML	RDB to RDF Mapping Language
RDF	Resource Definition Framework
REST	Representational State Transfer
RSS	Really Simple Syndication
SCP	Secured Copy
SPARQL	SPARQL Protocol and RDF Query Language
SR	System Requirement
STREP	Specific Targeted Research Projects
SWE	Snow Water Equivalent
TB	Technical Baseline
TOM	Target Operating Model
UR	User Requirement
VM	Virtual Machine
W3C	World Wide Web Consortium
WCS	Web Coverage Service
WFS	Web Feature Service
WMS	Web Map Service
WN	Worker Node
WPS	Web Processing Service
XML	eXtensible Markup Language
SWRL	Semantic Web Rule Language

Table of Contents

1 INTRODUCTION 9

1.1 Background and Context 9

1.2 Reference Documents 10

2 TECHNICAL SPECIFICATIONS 11

2.1 MoSCoW Analysis 11

2.2 Core Requirements (KR15) 12
2.2.1 General 12
2.2.2 User Management, Privacy and Security 12
2.2.3 Interoperability 13
2.2.4 Storage 13
2.2.5 User Communication 13
2.2.6 Application Setup 14
2.2.7 Application Execution 14

2.3 Extension Requirements 15
2.3.1 General 15
2.3.2 Social Media Management (KR12) 15
2.3.3 Event Detection (KR02) 16
2.3.4 Concept Extraction (KR03) 16
2.3.5 Community Detection (KR07) 17
2.3.6 Similarity Fusion (KR04) 17
2.3.7 Weather Data Management (KR13A) 17
2.3.8 In-situ data integration and modelling (KR13B) 18
2.3.9 Change Detection in EO Data (KR01) 20
2.3.10 Knowledge / Linked Data Management (KR08, KR09) 21
2.3.11 Clustering of EO and non-EO images (KR05, KR06) 21
2.3.12 Visualisation (KR14) 22
2.3.13 Reporting 23

2.4 Application Requirements / Coverage of the User Requirements 24
2.4.1 Applications 24
2.4.2 Flood Monitoring 24
2.4.3 Flood Damage Estimation 26
2.4.4 Crop Monitoring 26
2.4.5 Climate Monitoring 27
2.4.6 Water Level Monitoring 27
2.4.7 Snow and Ice Monitoring 28
2.4.8 Road Condition Monitoring 29
2.4.9 Task Management 29
2.4.10 Weather Monitoring 29

3 ARCHITECTURE OF THE EOPEN PLATFORM 31

3.1 Hardware Infrastructure 31

3.2 Software Architecture 32
3.2.1 Cloud-based Processing 33
3.2.2 OGC WPS-Based Interface to Processes 35
3.2.3 HPC and Big Data Process Executor 35
3.2.4 Inter-Environment Communication 37

4 INTEGRATED EOPEN PLATFORM 38

4.1 Introduction 38

4.2 Software Architecture 38

4.3 First EOPEN Prototype (D6.3, M18) 39
4.3.1 Overview 39
4.3.2 Integrated Modules 40
4.3.3 Available Workflows 41
4.3.4 Visualisation Capabilities 42

4.4 Second EOPEN Prototype (D6.4, M26) 42
4.4.1 Overview 42
4.4.2 Integrated Modules 43
4.4.3 Available Workflows 43
4.4.4 Visualisation Capabilities 44

4.5 Final System (D6.5, M33) 44

5 PILOT USE CASES 45

5.1 Infrastructure 45

5.2 Validation 45

5.3 PUC1 "Flood Risk Assessment and Prevention" Architecture 45

5.4 PUC2 "Food Security through EO Datasets" Architecture 48

5.5 PUC3 "Monitoring Climate Change through EO" Architecture 50

1 INTRODUCTION

1.1 Background and Context

The EOPEN platform will support interoperability at data, processing and use levels. Conceptually 4 user roles of the EOPEN platform can be identified:

- The Users, consumers of the services and the end users as defined in the Use Cases.
- The Service Providers, who own the domain knowledge of the Users and develop the Services for the Pilot Use Case. They deliver applications that are hosted by EOPEN and made available for inclusion in workflows using steps to ingest, prepare and make available one or more applications as a process provided by the EOPEN core capabilities.
- The Data and Information Experts who will use their expertise to implement algorithms and processes to analyse, process and extract information needed by the Service Providers from data sources such as the Copernicus data and social media streams. They deliver the extensions to EOPEN for the extended processing capabilities that can be used by all users of EOPEN in building their workflows.
- The Platform Experts, whose task is to provide the EOPEN Platform delivering the components for the core service capabilities to ensure that a core infrastructure is developed that guarantees interoperability and provides services to develop processing chains that:
  a. make use of core service capabilities, extended processing capabilities provided by the Data and Information Experts;
  b. implement the processing workflows of the Service Providers that possibly could include Service Provider provided applications (algorithms, scripts);
  c. execute the processing services.

The System Requirements of the EOPEN Platform will take into account the high-level needs:

1. to be safe and secure to use.
2. to be interoperable with several execution environments (in particular public clouds and HPC environments) and with third-party systems.
3. to allow deploying custom modules and integrating them within processing chains.
4. to allow enriching the platform with application specific algorithms.
5. to allow building domain-specific applications using a combination of core and generic components.

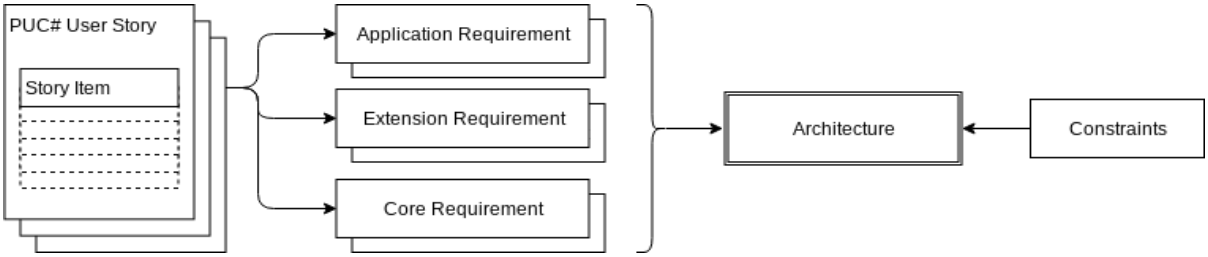
User requirements are expressed in terms of User Stories for each of the Pilot Use Cases (PUCs). They have been written without consideration of the EOPEN platform concept that distinguishes between:

1. Core requirements, being that part of the platform that provides the tools to deploy and execute custom processing chains (High level requirement 3);
2. Extension requirements for the Extended Capabilities being algorithms and libraries that capture expert knowledge needed to extract information from EO product and social media streams for instance to deliver information needed by users in their

applications. The extended capabilities, in general, map to the Key Results described in the EOPEN "Description of Work" and the EOPEN Grant Agreement (GA) that will be further elaborated in Work Packages 3, 4 and 5;

3. Domain and user specific aspects that will define the application requirements specific to a PUC.

The user stories of each pilot use case have been analysed. Each entry in the user stories has been analysed and derived into one or more technical requirements that are then allocated to the Core Requirements, Extended Capabilities or Application Requirements.



The system requirements apply to both the core platform and the generic libraries that will be available to build user applications. They are specified in the next chapter.

1.2 Reference Documents

Reference	Title
GA	Grant Agreement number 776019 – EOPEN
D1.3	EOPEN Self-Assessment Plan
D1.4	EOPEN Data Management Plan
D2.1	EOPEN Use Case Design Report
D2.2	EOPEN User Requirements
D3.1	EO data acquisition from the Collaborative Ground Segment and quality control
D4.1	Change detection techniques in Earth Observation
D5.1	The EOPEN ontology and semantic reasoning support
D7.1	Pilots Implementation and 1st Prototype Evaluation Report
D9.1	NEC - Requirement No. 1

2 TECHNICAL SPECIFICATIONS

The EOPEN Platform provides core components that allow orchestrating the execution of processes. The platform includes built-in processes that allow performing generic tasks such as sending email notifications and registering a product in a catalogue. Common libraries and toolboxes are also available in the core platform that allow developing custom algorithms and import them as processes.

In the course of EOPEN, the platform will be enriched by higher-level processes and libraries, some of which have been described as "Key Results" for EOPEN in the Grant Agreement (GA). Examples include event detection in social media data, machine learning, etc. These modules are implemented independently, integrated in the platform, and made available to application developers.

Application-specific processes implement the functions required by the target application. They are implemented by the application developers with or without using the available built-in and the higher-level libraries.

Applications themselves are specified by combining built-in, high-level and application modules within one or more workflows. The workflows specify the modules to be executed, their dependencies and the data flow.

The next sections document the core, the extension and the application requirements.

2.1 MoSCoW Analysis

Similarly to the User Requirements, they are based on (see document D2.2 section 3.1.2), the Technical Requirements specified in this chapter use the MoSCoW prioritisation technique. In short, this technique helps disambiguating the priority, or importance level, of a requirement.

The uppercase letters in “MoSCoW” stand for, in decreasing order of priority: *Must have*, *Should have*, *Could have*, and *Won’t have (this time)*.

- Must have requirements are considered critical and skipping them renders a system capability unusable.
- Should have requirements are considered important as skipping them impacts a capability (e.g. in terms of quality of the output or available options). they are not critical as there may be another way to satisfy these requirements.
- Could have requirements are desirable (“nice to have”). They could improve the user experience, for example, but do not impact the capabilities of the system. They are typically implemented if time permits.
- Won’t have (this time) requirements have the lowest priority. They have no impact on the system capabilities and on the user experience. Because of this they may be skipped or worked on only after the more important features have been implemented.

A detailed explanation of this technique is provided by the Agile Business Consortium:
https://www.agilebusiness.org/page/ProjectFramework_10_MoSCoWPrioritisation

2.2 Core Requirements (KR15)

The following requirements apply to the core components of the platforms. They have been organized according to key topics: user management and security, interoperability, usability, storage and visualization capabilities, etc.

The requirements for implementation and testing purposes have been categorised as follows:

Requirement Categories	
FC	Functional Requirement
IF	Interface Requirement
PF	Performance Requirement
RE	Resource Requirement
SE	Security Requirement
DI	Design and Implementation Requirement
CO	Configuration and Delivery Requirement

2.2.1 General

General requirements are associated with top level requirements derived from the overall concepts of EOPEN.

General	Category	Description	Key Req.
CR1.01	DI	The System must include an EOPEN Platform	KR14

2.2.2 User Management, Privacy and Security

User management, privacy and security requirements are defined to establish procedures, software and infrastructure to ensure that the platform and its users is protected from unauthorised access, accidental revealing of privileged information and breaches by those with ill-intent.

GDPR related issues are addressed in D9.1 “NEC - Requirement No. 1”.

ID	Category	Description	Key Req.
CR2.01	FC	The EOPEN Platform must provide single-sign on capability	
CR2.02	FC	The EOPEN Platform must be able to store user credentials	
CR2.03	FC	The EOPEN Platform must be able to assign one or more roles to individual users	
CR2.04	FC	The EOPEN Platform must be able to link individual users to one or more projects	
CR2.05	SE	The EOPEN Platform must be able to authenticate the users based on their credentials	
CR2.06	SE	The EOPEN Platform must be able to restrict access to specific resources based on the user role(s) and project(s)	
CR2.07	FC	The EOPEN Platform must allow users inspect the personal data stored in the system	
CR2.08	SE	The EOPEN Platform must restrict the access to user personal data to the concerned user and the administrators	

CR2.09	FC	The EOPEN Platform may use cookies in Web applications for managing the user sessions	
CR2.10	SE	The EOPEN Platform may use cookies to collect statistical data in an anonymous manner	

2.2.3 Interoperability

EOPEN seeks to demonstrate interoperability at 3 levels:

1. Data – data sources shall be usable across multiple processing environments
2. Processing – the platform shall not limit the choice of programming languages and packages used. It shall be possible to build heterogeneous processing chains. Effectively the platform shall be (user) application agnostic.
3. Processing resources – a processing chain should be executable on a platform of choice. There are some limitations. For instance, algorithms making use of parallelised processing such as Hadoop/Spark or features of HPC will need processing platforms able to support such specific needs. But in general processing chains should be interoperable across processing platforms (Cloud). I.e. EOPEN shall be processing platform agnostic.

ID	Category	Description	Key Req.
CR3.01	IF	The EOPEN Platform must be able to serve the stored data through standard programming interfaces (APIs)	

2.2.4 Storage

In EOPEN it shall be possible to match the storage needs of the processing services resulting from the workflows and their processing chains.

ID	Category	Description	Key Req.
CR4.01	FC	The EOPEN Platform must allow persisting data generated by the deployed sub-systems	
CR4.02	FC	The EOPEN Platform must include a service capable of storing geospatial data	
CR4.03	IF	The EOPEN Platform geospatial database must be accessed through standard interfaces	

2.2.5 User Communication

Communication towards the users on the status and progress of the execution of services running on the platform is essential for the user to manage the operational workflows and for use in the user decision making process.

ID	Category	Description	Key Req.
CR5.01	FC	The EOPEN Platform must allow the application modules send notifications programmatically	
CR5.02	PF	The EOPEN Platform should be able to issue notifications to the users maximum	

		10 minutes after an event has been detected	
CR5.03	FC	The EOPEN Platform must be able to ingest and manage notifications issued from outside the platform	
CR5.04	FC	The EOPEN Platform should be able to issue notifications with attached payloads	

2.2.6 Application Setup

An application is a generic concept referring to any module developed external to the EOPEN platform (i.e. it is not a Core Service) that a user wishes to include into a processing chain to create a processing service. It also refers to algorithms developed by the Data and Information Experts. The application in this case can be provided as a process or incorporated as a library.

ID	Category	Description	Key Req.
CR 6.01	FC	The EOPEN Platform must be able to store application metadata	
CR 6.02	FC	The EOPEN Platform must allow authorized users to import user algorithms	
CR 6.03	FC	The EOPEN Platform must allow importing user algorithms implemented in different languages	
CR 6.04	FC	The EOPEN Platform must be able to package algorithms within Docker containers	

2.2.7 Application Execution

EOPEN needs to be able to manage (interoperability) the diverse nature of the processing needs of an application. In general this will be a case of matching the processing requirements (language and library dependencies), the implementation form (GPU, HPC, Cloud) as well as constraints such as the need for Spark, Hadoop, etc. against the list of resources included in the EOPEN ecosystem.

ID	Category	Description	Key Req.
CR 7.01	FC	The EOPEN Platform must allow executing algorithms that require a GPU to run properly	
CR 7.02	FC	The EOPEN Platform must allow executing algorithms that require a High Performance Computing environment to run properly	
CR 7.03	FC	The EOPEN Platform must be able to deploy Docker containers to selected cloud processing platforms	
CR 7.04	FC	The EOPEN Platform must be able to execute containerized algorithms deployed within selected cloud processing platforms	
CR 7.05	FC	The EOPEN Platform should be able to deploy Docker containers to an HPC environment[1]	

[1] The priority has been turned from "must" to "should", as there are other means to deploy and execute algorithms on an HPC.

CR 7.06	FC	The EOPEN Platform must be able to execute algorithms deployed within an HPC environment	
CR 7.07	FC	The EOPEN Platform must be able to transfer the input data from one processing environment to another depending on the needs	
CR 7.08	FC	The EOPEN Platform must be able to determine the target deployment environment based on the algorithms resources needs	

2.3 Extension Requirements

2.3.1 General

The decision to include a requirement as an extension to the available built-in functions, libraries and shared modules of EOPEN is a choice that will depend very much upon the type of processing required, i.e. if it is a requirement that needs the Data and Information Experts then it will become an Extension Requirement.

A general although not strict rule is that if a module may be implemented in such a way that it is applicable to several application domains it should be considered as an extension. In that case, care must be taken to anticipate as much as possible any possible use by providing a means (usually through parameterisation) to control its behaviour.

The following common needs have been identified:

EOPEN must include a module capable of ingesting and managing weather data
EOPEN must include a module capable of ingesting and managing social media data
EOPEN must include a module capable of ingesting and managing satellite imagery
EOPEN must include a module capable of ingesting and managing linked data
EOPEN must include a module capable of ingesting and managing data issued by AAWA in NRT

2.3.2 Social Media Management (KR12)

The Social Media Crawlers module will ingest social media information into the system, based on predefined search options that concern specific topics. Then, a classification methodology will estimate the relevancy of the data according to the specified topics. In addition, the module will respond to queries of the EOPEN platform by filtering appropriately the collected data.

The following requirements formalize the features that should be supported by the Social Media Crawlers module.

The module must be able to ingest social media content into the System
The module must be able to collect social media content relating to predefined keywords, bounding boxes, and accounts
The module must be able to identify Twitter images related to road network
The module must be able to request a semantic analysis in order to estimate the location of social media content, when not available
The module must be able to augment social media content with estimated location

The module must be able to perform classification techniques in order to determine whether social media content is relevant or not to predefined topics
The module must be able to respond to search queries by the platform with filtered social media content in a GIS layer

2.3.3 Event Detection (KR02)

The Event Detection module will monitor streams of social media content and weather forecast data, aiming to detect peaks of the incoming flow or extreme values, respectively. Uncommon observations will be translated to events, along with an estimation of the location and the description of each event. Events will be served as GIS layers.

The following requirements formalize the features that should be supported by the Event Detection module.

The module must be able to access the relevant social media content available in the System
The module must be able to detect events in the social media flow
The module must be able to detect events in the weather forecast data
The module could be able to fuse information from social media and weather forecast data
The module must be able to estimate the location of an event
The module must be able to detect the top keywords that describe an event
The module could be able to categorize the events as common or uncommon/exceptional
The module must be able to serve the detected events in GIS layers
The module could be able to communicate with the platform for sending email notifications

2.3.4 Concept Extraction (KR03)

The Concept Extraction module will access social media content and will detect concepts from their textual as well as visual information. Extracted concepts will be added to the metadata of each social media post and will also be used to estimate the relevancy of a post to a given concept.

The following requirements formalize the features that should be supported by the Concept Extraction module.

The module must be able to access the social media content available in the System
The module must be able to extract concepts from textual and visual information of social media
The module must be able to determine whether social media content is relevant or not to a given concept
The module must be able to augment the social media content with extracted concepts

2.3.5 Community Detection (KR07)

The Community Detection module will access social media content or user activity inside the EOPEN platform and will identify user communities and influencers. The module will have the flexibility to adapt to different relationships between users and the results will be served to the EOPEN user interface.

The following requirements formalize the features that should be supported by the Community Detection module.

The module must be able to access the social media content available in the System
The module must be able to access the user activity in the System
The module must be able to detect user communities through their interaction in social media
The module must be able to detect the most influencing users
The module must be able to be adapted to use various types of interaction between users
The module must be able to serve the detected communities and most influencing users to the platform

2.3.6 Similarity Fusion (KR04)

The Similarity Fusion module will access satellite imagery and social media content and will fuse different types of information (visual, textual, spatiotemporal) in order to retrieve the results most similar to a multimodal query. All retrieved results will be forward to the EOPEN interface for visualization.

The following requirements formalize the features that should be supported by the Similarity Fusion module.

The module must be able to access the satellite imagery available in the System
The module must be able to access the social media content available in the System
The module must be able to retrieve information similar to a multimodal query
The module must be able to combine visual, textual, and spatiotemporal similarities
The module must be able to return the retrieved results to the platform

2.3.7 Weather Data Management (KR13A)

The Weather Data Management module will access weather data services and retrieve data to EOPEN platform. The module will perform requests to select web services and will extract the data and metadata from the response. The data will then be converted to data format(s) used internally within the EOPEN platform in communications between modules. The converted data will be stored within the EOPEN platform. The stored data will be used as an input for other modules and Key Results (e.g. KR02, KR13B). End users will be served with weather data via User Interface modules (KR14).

The module must be able to access data over HTTP
The module must be able to access WFS 2.0 Service
The module must be able to perform WFS Simple Profile compliant queries
The module must be able to extract data from service response
The module must be able to extract metadata from service response
The module must be able to convert the data to data format(s) used by the platform, if necessary
The module must be able to store the data to the platform, to be used as an input for other modules and as a cache
The module must be able to store the metadata to the platform
The module must be able to access the stored data in the platform
The module must be able to modify or remove the data generated by the module from the platform, if necessary

2.3.8 In-situ data integration and modelling (KR13B)

(AAWA) AMICO Flood Forecasting system is an integrated hydrologic-hydraulic model that based on weather forecasts and ground sensors readings can provide expected water levels over sections of the Bacchiglione river. AMICO stores all datasets into an ORACLE Database; the database can be inquired by specific queries. AMICO can be accessible via an HTTP-enabled service. EOPEN Platform should provide an integration tool to inquiry AMICO DB and extract forecasts. This service must run every time a forecast will be available (2 to 3 hours).

AAWA will provide to EOPEN platform the results from AMICO flood forecasting system. AMICO Flood forecasting system will ingest weather forecast from EOPEN platform and from local authorities; all forecasts will be stored into AMICO database. Every time a weather forecast is available AMICO runs a flood forecast. AMICO outputs are basically water levels on sections; the platform should read once all river sections and all thresholds (3 for each session corresponding to 1st, 2nd and 3rd crisis level). The platform should read AMICO water levels for each section and check thresholds; every time the water level exceeds a threshold the platform should show a warning (at now AMICO interface show a green warning when levels exceed the 1st threshold, a yellow warning when they exceed the 2nd threshold, and a red warning when they exceed the 3rd threshold and a particular warning when water level exceeds the river banks). In addition to it, a two-dimensional version of AMICO is being developed which will provide a flood extension map whenever river banks are exceeded – such a product has been preliminarily defined in D1.4.

2.3.8.1 Umbrella application of Sentinel hubs (KR10)

Searching of sentinel data is often a complicated process due to the different data and the different performance on the distributed data hubs. Thus, there is the need of an umbrella data hub that brings them all together. This single point hub, deployed and running on the EOPEN platform, will provide access to Sentinel1, Sentinel2, Sentinel3 and Sentinel 5p

metadata by connecting in the back end to a number of the available Sentinel hubs (core hubs and 23 National Collaborative Ground Segments) and serving the results to the users via an application programming interface (API).

Most of these data hubs are using data hub system (DHuS) that allows users to access the data via their own computer programs, scripts or client applications. API Query is based on OpenSearch protocol, which is a collection of technologies that allow publishing of search results in a format suitable for syndication and aggregation. Ultimately, this umbrella application gives the potential for linking federated Copernicus Sentinels Hubs, access to a single data hub instead of searching the appropriate hub for the user's needs, access to all Sentinel mission data and better performance on downloading products, as product will be chosen from the most appropriate data hub based on integrity, speed and availability tests.

The application must be able to connect to several data hubs
The application must be able to search data hubs using filters and criteria
The application must be able to store metadata coming from each data hub to a database.
The application must be able to test downloading speed of a data hub
The application must be able to check the integrity of a file
The application must be able to check the availability of a data hub
The application must be able to display search result lists
The application must be able to search data hubs periodically
The application must be able to display search results in a structured format.
The application must be able to allow users to filter search results
The application should be able to have a rolling archive policy

2.3.8.2 Demand driven EO data provision application (KR10)

This process is based on specific user demand and contains not only searching functionalities, but also the downloading and pre-processing of data. A typical search form will be used so that the user can set their specific needs according to some parametric, user-defined criteria. In particular, the system will systematically download specific Sentinel data needed for pilot implementation by filtering based on sensor, product, location, time, frequency, etc. Moreover, a suite of remote sensing tools will be provided as modules on the EOPEN platform in order to do some basic pre-processing of the Sentinel data (e.g. generate NDVI Time Series). The processing modules will be based on general concepts by abstracting common properties of data. Thus, they can be reused for creating another automated process, without the need of building a new module. Finally, new products will be generated from this workflow. These products can be accessed from the users via an API.

The application must allow all users to define search criteria
The application must be able to search data hubs using users' criteria
The application must be able to provide an API for automatically downloading products based on user defined

criteria
The application must be able to save products on a specific repository
The application must be able to check the integrity of each product
The application must be able to choose product from the most appropriate data hub
The application must be able to provide systematic generation of pre-processed products
The application would be able to provide image mosaicking and atmospheric correction tools, if applicable
The application would be able to prepare data for time series analysis (i.e. image stacking), if applicable
The application should be able to generate remote sensing indices
The application must be able to allow users to filter search results through the API
The application must display the metadata of the generated results in a structured format through the API

2.3.8.3 Metadata generation of EO imagery

Externally link the EOPEN platform with the well-established DKAN based regional data hub of GEOCRADLE (http://datahub.geocradle.eu/), which is operated by NOA. Create a separate EOPEN branch under the data hub's thematic groups to provide access to the Use Case related datasets, products and services developed by the EOPEN network.

Ultimately, DKAN is a complimentary offering to CKAN (Comprehensive Knowledge Archive Network, a web-based open source management system for the storage and distribution of open data) in the effort to make data more open and accessible. The hub in which EOPEN datasets and products will be published is a community-driven, free and open source open data platform that gives organizations and individuals ultimate freedom to publish and consume structured information. It treats data as content, facilitating the subsequent publication, management, and maintenance of these, no matter the administration team, its size and level of technical expertise.

The application must be able to publish all added value products produced by the Use Cases in the regional data hub (RDH) of GEOCRADLE
The application must guarantee that the datasets fulfil specific requirements for data and metadata format

2.3.9 Change Detection in EO Data (KR01)

The Change Detection module will access any satellite imagery available and will monitor the changes regarding water on the surface of sequential images, in order to detect flooding incidents and create a mask of the flooded area. Moreover, this module will detect if a road is passable or not because of a flood. All outcomes will be produced as GIS layers.

The following requirements formalize the features that should be supported by the Change Detection module.

The module must be able to access the satellite imagery available in the System

The module must be able to access the satellite imagery available on DIAS infrastructure
The module must be able to derive data from Sentinel-1 and/or Sentinel-2 satellite imagery, given a specific query
The module must be able to determine whether or not a delimited area is flooded
The module must be able to produce a mask of the flooded area
The module must be able to determine whether or not a road is passable due to floods
The module could be able to estimate the damages caused by floods in amount of damaged buildings
The module must be able to serve the data it produces using GIS layers

2.3.10 Knowledge / Linked Data Management (KR08, KR09)

This module will capture a) metadata coming from other services, e.g. meteorological and climate information, b) analysis results of EOPEN modules, e.g. detected events and topics from social media, and c) EOPEN-specific domain knowledge, concepts and correlations pertinent to the use cases, e.g. alerts, contextual relations, etc. It will also define the mapping rules for transforming incoming data into linked data populating the underlying KB. Finally, it will develop intelligent knowledge-driven reasoning and data integration and query mechanisms for deriving implicit correlations from the data stored in the KB.

The module must be able to access the semantic knowledge structures and vocabularies needed to capture information and data pertinent to EOPEN
The module must be able to access to a triple store (KB)
The module must be able to populate the KB with analysis results
The module must be able to enable general purpose rule-based reasoning
The module must be able to determine the location of social media content based on text analysis results
The module must be able to process incoming queries and semantically retrieve data from the KB
The module must be able to provide the necessary Web interfaces (Web APIs) for other modules to invoke specific functions

2.3.11 Clustering of EO and non-EO images (KR05, KR06)

The clustering module will ease the retrieval of relevant information by grouping similar artefacts into representative groups. Specifically, clustering will help EOPEN stakeholders to reduce the time spent to search for interesting information in vast amounts of data coming from both EO and non-EO data. The module will support both clustering of textual content, in particular social media content, as well as images (e.g. satellite images). However, there is no single solution available to perform clustering on both textual and image data, so that different approaches have to be implemented to achieve satisfactory results.

In EOPEN, the main textual content is coming from social media sites such as Twitter. As a result, it is required that this module is able to retrieve new data in a streaming fashion to allow for near real-time analytics. Taking Twitter as an example, clusters of tweets shall be

identified talking about specific topics. Since clustering algorithms such as k-Means typically just group similar items, dedicated topic modelling algorithms such as Latent Dirichlet Allocation shall be applied to assign to each group a descriptive label. In order to demonstrate the value of clustering to stakeholders, a suitable visualization will be selected to let users navigate through clusters and interactively explore individual tweets assigned to each cluster. Finally, the most important topics or keywords for each group will be shown to the users, so that they can immediately get a feeling for the underlying content.

The clustering module will support clustering of imagery data in a similar way as described above.

Since clustering large amounts of textual and image data is compute intensive, the computation will be performed on USTUTT's HPDA infrastructure using state-of-the-art data analytics frameworks such as Apache Spark.

The module will be tightly integrated with the EOPEN platform in order to provide clustering as a task to arbitrary datasets as long as they conform to a predefined data structure as input. Interfaces to retrieve social media content and non-EO data will be implemented.

The module must be able to access the social media content available in the System
The module must be able to detect clusters of non-EO data
The module must be able to be executed on Big Data HPC infrastructure
The module must be able to retrieve the most frequent words in a cluster
The module must be able to serve the detected clusters to the platform

2.3.12 Visualisation (KR14)

The visualisation module will provide a homogeneous and interactive Web-based user interface to the end-users. The interface will visually integrate generic graphical components that allow displaying the information managed within the EOPEN Platform in the most appropriate and convenient manner. Navigation between those components will also be provided where applicable. For example, a user must be able to navigate from a placemark on a map to a panel that will provide all the details about the related feature.

Envisaged graphical components include a map viewer, a time series visualizer, a tag clouds viewer, search result lists, image gallery viewers, a notification client tool, as well as different types of dynamic and interactive graphs. The information displayed includes geospatial data (incl. GIS layers) accessed via standard interfaces (OGC WMS, WFS, WCS), linked data and social media data, time series and notifications.

The following requirements formalize the features that should be supported by the visualisation module.

The EOPEN Platform must include a Web-based user interface (WebUI)
The EOPEN Platform WebUI must be able to display GIS layers in a generic Map Viewer

The EOPEN Platform WebUI must allow users select the layers to be displayed in the Map Viewer
The EOPEN Platform WebUI must allow users select an area of interest from a list of pre-defined AOIs
The EOPEN Platform WebUI must be able to filter the displayed information based on the selected AOI
The EOPEN Platform WebUI could allow users save the current map viewer pan and zoom configuration under a custom name
The EOPEN Platform WebUI could allow users select a saved pan and zoom configuration and re-apply it to the map viewer
The EOPEN Platform WebUI could allow users delete a stored map viewer pan and zoom configuration
The EOPEN Platform WebUI must be able to display time series data graphically and in an interactive manner
The EOPEN Platform WebUI should allow users request for statistical data to be derived from the displayed time series
The EOPEN Platform WebUI must be able to display galleries of images
The EOPEN Platform WebUI must be able to display search result lists
The EOPEN Platform WebUI must be able to display different types of dynamic and interactive graphs
The EOPEN Platform WebUI must be able to display the notifications issued or received by the platform
The EOPEN Platform WebUI must allow users send notifications to selected users and groups
The EOPEN Platform WebUI must allow users select the type and the severity of the notifications to be displayed
The EOPEN Platform WebUI must be able to display the data served from within the platform services
The EOPEN Platform WebUI must allow users navigate between the graphical components by following links
The EOPEN Platform WebUI must be able to display linked data in a structured manner
The EOPEN Platform WebUI must allow users request for data to be processed in the platform
The EOPEN Platform WebUI must allow users export the displayed information in a format usable off-line
The EOPEN Platform WebUI should allow users export the displayed data in a structured format
The EOPEN Platform WebUI should be able to display two or more datasets simultaneously
The EOPEN Platform WebUI should allow users select the datasets to display in a directly accessible menu
The EOPEN Platform WebUI must be reachable from different client platforms
The EOPEN Platform WebUI must be multi-user

2.3.13 Reporting

Reporting is needed from two sources.

1. From the core services providing information on use, results of processing and available resources

2. From user defined applications providing the relevant information needed for the service providers and the end-users of the services.

Both can be automated depending on the requirements of the users as defined in the PUCs.

This will need to be addressed during the PUC implementations.

2.4 Application Requirements / Coverage of the User Requirements

For clarity, application requirements express the needs of the users in terms of capabilities implemented by the Service Providers. Applications possibly combined with core and extensions into one or more workflows provide services.

When necessary, the description of the Pilot Use Case in [D2.1] is used to disambiguate the URs. The SRs and the architectural design (see next chapter) is the reference baseline for developing the EOPEN platform and implementing the PUCs.

In the course of the EOPEN project, and when they become available, the COM [D2.3] and the TOM [D2.4] reports become inputs to drive the design and the implementation of the EOPEN platform.

2.4.1 Applications

Based upon the User Requirements described in D2.2, the following applications aim to be delivered for the EOPEN platform:

AP1	EOPEN Must provide a Flood monitoring application
AP2	EOPEN Should provide a Flood Damage estimation application
AP3	EOPEN Must provide a Crop monitoring application
AP4	EOPEN Must provide a Climate monitoring application
AP5	EOPEN Could provide a Water level monitoring application
AP6	EOPEN Must provide a Snow and ice monitoring application
AP7	EOPEN Could provide a Road condition monitoring application
AP8	EOPEN Could provide a Task management application
AP9	EOPEN Must provide a Weather monitoring application

2.4.2 Flood Monitoring

The Flood monitoring application incorporates both the prediction of imminent flood events as well as an overview of activity during a flood event. The satisfaction of the following requirements depends on the AAWA implementation of the application specific modules:

AP1.1	The Application must be able to access the weather data available in the system
AP1.2	The Application must be able to access the satellite imagery available in the system
AP1.3	The Application must be able to access the in-situ data available in the system
AP1.4	The Application must be able to access the social media content available in the system

AP1.4.1	The Application must be able to determine whether social media content is relevant to the current flooding event
AP1.4.1.1	The Application should be able to link social media content to location of origin
AP1.5	The Application must be able to ingest data from predefined external data sources
AP1.5.1	The Application must be able to ingest predictions and recommendations information issued from AAWA in NRT
AP1.5.1.1	The Application must be able to serve the information issued from AAWA through a programming interface in NRT
AP1.5.2	The Application must be able to ingest and manage warning notifications issued from AAWA in NRT
AP1.5.2.1	The Application must be able to send the warning notifications received from AAWA in NRT
AP1.5.3	The Application must be able to ingest warning notifications issued from the Civil protection leader in NRT
AP1.5.3.1	The Application must be able to send the warning notifications received from the Civil protection leader in NRT
AP1.5.4	The Application must be able to ingest and manage safety information issued from the civil protection department in NRT
AP1.5.4.1	The Application must be able to serve the safety information through a programming interface in NRT
AP1.6	The Application must be able to predict an imminent flood event
AP1.6.1	The Application must be able to determine whether or not a delimited area will be flooded
AP1.6.1.1	The Application must be able to determine the extent in a delimited area that will be flooded based on [data-for-predicting-floods] data
AP1.6.1.2	The Application must be able to determine when a delimited area will be flooded based on [data-for-predicting-floods] data
AP1.6.2	The Application must be able to assess the severity of a predicted flood event
AP1.6.3	The Application must be able to send a warning notification within 10 minutes after the prediction
AP1.7	The Application must be able to provide an overview of currently flooded areas
AP1.7.1	The Application must be able to estimate water levels during a flood event
AP1.7.1.1	The Application must be able to serve the estimated water levels during a flood event through a programming interface
AP1.8	The Application should be able to allow users to upload pictures
AP1.8.1	The Application should be able to allow users to link pictures to locations
AP1.9	The Application could be able to ingest and manage task information related to the current incident
AP1.10	The Application must be able to store recently produced data
AP1.11	The Application must allow users to retrieve saved data

2.4.3 Flood Damage Estimation

The Flood Damage Estimation application automates the flood damage estimation process utilising satellite observation and in-situ data, creating a final report that can be forwarded on to the relevant predefined insurance companies.

Note: The estimation of the damages caused by floods, demanded to AAWA, requires additional EO data analyses based on data obtained by AAWA which are not analysed in the foreseen application modules – it is up to AAWA to extract the needed information. EOPEN provides the data, computation resources and support.

AP2.1	The Application must be able to access the satellite imagery available in the system
AP2.2	The Application must be able to access the in-situ data available in the system
AP2.3	The Application should allow users to upload pictures
AP2.4	The Application must be able to estimate the damages caused by floods in amount of damaged buildings
AP2.5	The Application must be able to distinguish buildings depending on their usage
AP2.6	The Application should allow users to create a flood damage report
AP2.6.1	The Application should allow users to edit the flood damage report
AP2.6.2	The Application should allow users to save the report
AP2.6.3	The Application should allow users to export the report
AP2.6.3.1	The Application could allow users to automate workflows to forward the report onto pre-defined contacts

2.4.4 Crop Monitoring

The Crop monitoring application provides users tools to assess the extent and type of crops within a given area, the health and status of crops and crop yield predictions

AP3.1	The Application must be able to access the weather data available in the system
AP3.2	The Application must be able to access the satellite imagery available in the system
AP3.3	The Application must be able to access the in-situ data available in the system
AP3.4	The Application must be able to access the social media content available in the system
AP3.4.1	The Application must be able to determine whether social media content is relevant to Food Security/ Crop Monitoring
AP3.4.2	The Application must be able to determine the location social media content originates from
AP3.5	The Application must be able to derive crop growth information from available satellite imagery
AP3.5.1	The Application must be able to determine the current phenology phase of the crops
AP3.5.2	The Application should be able to determine the current health of the crops
AP3.5.2.1	The application could be able to determine the current density of the vegetation
AP3.6	The Application must be able to format the information it generates within GIS layers
AP3.6.1	The Application must be able serve the GIS layers it produces through a standard protocol

AP3.7	The Application could be able to ingest data from external sources
AP3.8	The Application must be able to derive crop yield estimates from satellite imagery and external services
AP3.8.1	The Application could be able to estimate available crop yield in kg/ha
AP3.9	The Application should be able to persist historical data and serve them through GIS + raster layers
AP3.10	The Application must be able to derive data from Sentinel-1 and/or Sentinel-2 satellite imagery
AP3.11	The Application could be able to generate and serve data at parcel level

2.4.5 Climate Monitoring

The Climate monitoring application provides users an overview of seasonal conditions affected by climate change

AP4.1	The Application must be able to access the climatic / weather data available in the system
AP4.2	The Application must be able to access the satellite imagery available in the system
AP4.3	The Application must be able to access the in-situ data available in the system
AP4.4	The Application must be able to access the social media content available in the system
AP4.4.1	The Application must be able to determine whether social media content is relevant to climate monitoring
AP4.5	The Application should be able to produce the current measurements of the North Atlantic index
AP4.6	The Application should be able to produce historical North Atlantic Index measurements
AP4.7	The Application should be able to estimate the dates for yearly spring frost-heave
AP4.8	The Application should be able to produce historical spring-frost heave data
AP4.9	The Application should be able to predict freeze-thaw cycles within a geographic area
AP4.10	The Application should be able produce historical freeze-thaw cycle information
AP4.11	The Application should be able to predict areas with more frequent warm and wet periods
AP4.12	The Application should be able to record produced data
AP4.13	The Application should be able to serve recorded data
AP4.14	The Application must be able to serve the data it produces using GIS layers

2.4.6 Water Level Monitoring

The Water level monitoring application provides users information on current, historic and predicted water levels.

The satisfaction of the following application requirements depends on the AAWA implementation of the specific modules. It is not strictly related to the platform development although AAWA is willing to generate a product freely available from the platform, namely the "EOPEN AAWA AMICO Early Warning System Flood Forecast" (ref. AA_EWS_FF in D1.4).

AP5.1	The Application must be able to access the weather data available in the system
AP5.2	The Application must be able to access the satellite imagery available in the system
AP5.3	The Application must be able to access the in-situ data available in the system
AP5.4	The Application should be able to determine current groundwater levels
AP5.5	The Application must be able to determine current river water levels
AP5.6	The Application should be able to determine current sea levels
AP5.7	The Application could be able to predict areas at risk of future flooding
AP5.8	The Application must allow users to access historic water level data
AP5.9	The Application should provide water level predictions
AP5.10	The Application must be able to serve the data it produces using GIS layers

2.4.7 Snow and Ice Monitoring

The Snow and ice monitoring application provides users with an overview of snow, ice and ground frost information

AP6.1	The Application must be able to access the weather data available in the system
AP6.2	The Application must be able to access the satellite imagery available in the system
AP6.3	The Application must be able to access the in-situ data available in the system
AP6.4	The Application must be able to access the social media content available in the system
AP6.4.1	The Application must be able to determine whether social media content is relevant to Snow and Ice monitoring
AP6.5	The Application must be able to serve snow coverage data
AP6.5.1	The Application must be able to provide the current levels of snow coverage
AP6.5.2	The Application could be able to predict short-term snow coverage
AP6.5.3	The Application should be able to predict future average levels of snow coverage
AP6.5.4	The Application should be able to predict areas that may be more vulnerable to heavy snowfall
AP6.6	The Application should be able to serve ice coverage information
AP6.6.1	The Application should be able to identify areas that are currently covered by ice
AP6.6.2	The Application should be able to identify areas at imminent risk of dangerous ice conditions
AP6.6.3	The Application should be able to predict the areas more prone to icy conditions in the future
AP6.7	The Application should be able to identify the areas covered by snow with ice underneath
AP6.8	The Application should be able to predict areas where ice is more likely to form under snow
AP6.9	The Application should be able to serve ground frost coverage information
AP6.9.1	The Application should be able to identify current areas with ground frost
AP6.9.2	The Application should be able to predict short-term ground frost coverage
AP6.10	The Application should be able to record produced data
AP6.11	The Application should be able to serve recorded data

AP6.12	The Application should be able to serve historic snow and ice data by location
AP6.13	The Application must be able to serve the data it produces using GIS layers

2.4.8 Road Condition Monitoring

The road condition monitoring application provides users with information on the quality and usability of the road networks

AP7.1	The Application must be able to access the weather data available in the system
AP7.2	The Application must be able to access the satellite imagery available in the system
AP7.3	The Application must be able to access the in-situ data available in the system
AP7.4	The Application must be able to access the social media content available in the system
AP7.4.1	The Application must be able to determine whether social media content is relevant to Road Conditions
AP7.5	The Application could be able to measure the current road conditions for the wider road network
AP7.6	The Application could be able to predict road quality / deterioration
AP7.7	The Application must be able to serve the data it produces using GIS layers
AP7.8	The Application could be able to detect whether a road is passable or not

2.4.9 Task Management

The Task management application allows users to view and manage tasking information related to their role

AP8.1	The Application could allow users with certain permissions to issue tasking information
AP8.1.1	The Application could allow users with certain permissions to assign tasking information to users
AP8.1.2	The Application could allow users to assign tasking information by location
AP8.2	The Application could allow users to view relevant tasking information
AP8.3	The Application could allow users to update relevant tasking information
AP8.4	The Application could allow users to mark tasks as completed
AP8.5	The Application could send tasking information notifications to users
AP8.6	The Application could allow users to upload additional documentation to task information

2.4.10 Weather Monitoring

The weather monitoring application serves the ingested meteorological information in a format that compliments the other applications, including risk and event detections

AP9.1	The Application must be able to access the weather data available in the system
AP9.1.1	The Application could be able to serve weather data from multiple sources in NRT

AP9.2	The Application must be able to ingest and manage predicted weather incident information in NRT
AP9.3	The Application must be able to serve the predicted weather incident information through a programing interface
AP9.3.1	The Application must be able to filter the information served through its programming interface based on area and time of interest
AP9.4	The Application must be able to assess the risk related to meteorological data in NRT
AP9.5	The Application must be able to serve the risk assessment information through a programming interface in NRT
AP9.6	The Application must be able to produce information on the current weather
AP9.6.1	The Application must include current air temperature data
AP9.6.2	The Application must include current precipitation data
AP9.6.3	The Application could include ground soil temperature data
AP9.7	The Application should be able to record weather information
AP9.8	The Application should be able to serve the recorded weather information
AP9.9	The Application should be able to produce short term weather predictions
AP9.10	The Application could be able to produce long term weather predictions
AP9.11	The Application must be able to serve the data it produces using GIS layers

3 ARCHITECTURE OF THE EOPEN PLATFORM

This chapter documents the logical and physical architectures of the EOPEN Platform. In particular, we are describing the modules/services that will be available in the platform to allow users to access and use services and to allow Service Providers and Data and Information Experts to ingest and make available their applications and capabilities to EOPEN platform users. This section therefore does not address any Pilot Use Case aspects for the very simple reason that the EOPEN platform must be user application and cloud provider agnostic to be useable by future platform users. The development and inclusion of the applications is described in chapters 4 and 5.

3.1 Hardware Infrastructure

One of the key objectives of EOPEN is to develop and integrate technologies that provide interoperability between components and systems. This is not limited to software libraries and applications but also applies to the hardware and in particular to the main technologies available for providing online, shareable processing resources.

In this respect, we can distinguish between the different forms of processing environments:

- Servers managed individually or within a pool (e.g. using a Xen hypervisor) within the organisation infrastructure. This requires non-neglectable manual effort to deploy additional resources to cope with increasing processing needs.
- Servers integrated into an organisation private cloud. Should the hardware resources be sufficient to meet the processing needs, this allows scaling up and down the available virtual resources with minimal effort. The cost lies mostly in the initial investment in hardware purchase and setup. The running cost is limited.
- Servers deployed into a public cloud: This provides the biggest flexibility and scalability as the hardware resources of public cloud providers are usually very large. There is no initial cost for hardware purchase and setup. The running costs are however higher as running time is rented to the provider, which is also responsible for maintaining the infrastructure. To lower this running cost servers in the virtual environment may be stopped and even destroyed when they are not required anymore.
- Servers belonging to a High-Performance Computing (HPC) environment: This setup is less flexible as there is no such virtual environment that can be easily accessed and customized.

In EOPEN, we aim at giving each environment type introduced above a role in the overall hardware infrastructure and integrate them according to their best use in the platform.

The overall hardware architecture is depicted in Figure 1, below. This contains the following elements:

- The ASB Controller Node is an individually manager server. It runs software components that provide the management interfaces and are responsible for the management of the platform, the orchestration of the processing chains.
- The Worker Nodes located in the top-right corner of the figure run in a private cloud. There is a limited number of them but they can be configured to meet very specific needs such as include GPUs.

- The Worker Nodes located in the bottom-left of the figure run in a public cloud. Many instances can be started to meet the processing needs. Also, when deployed in an environment such as a DIAS, a fast, local access to Copernicus data is available.
- On the right of the public cloud is represented an HPC environment. This is divided in two main parts: the Login Node where the user or the client applications lands when connecting to the environment, and the Compute Nodes where the processing is performed when initiated using ad-hoc HPC-specific tools.

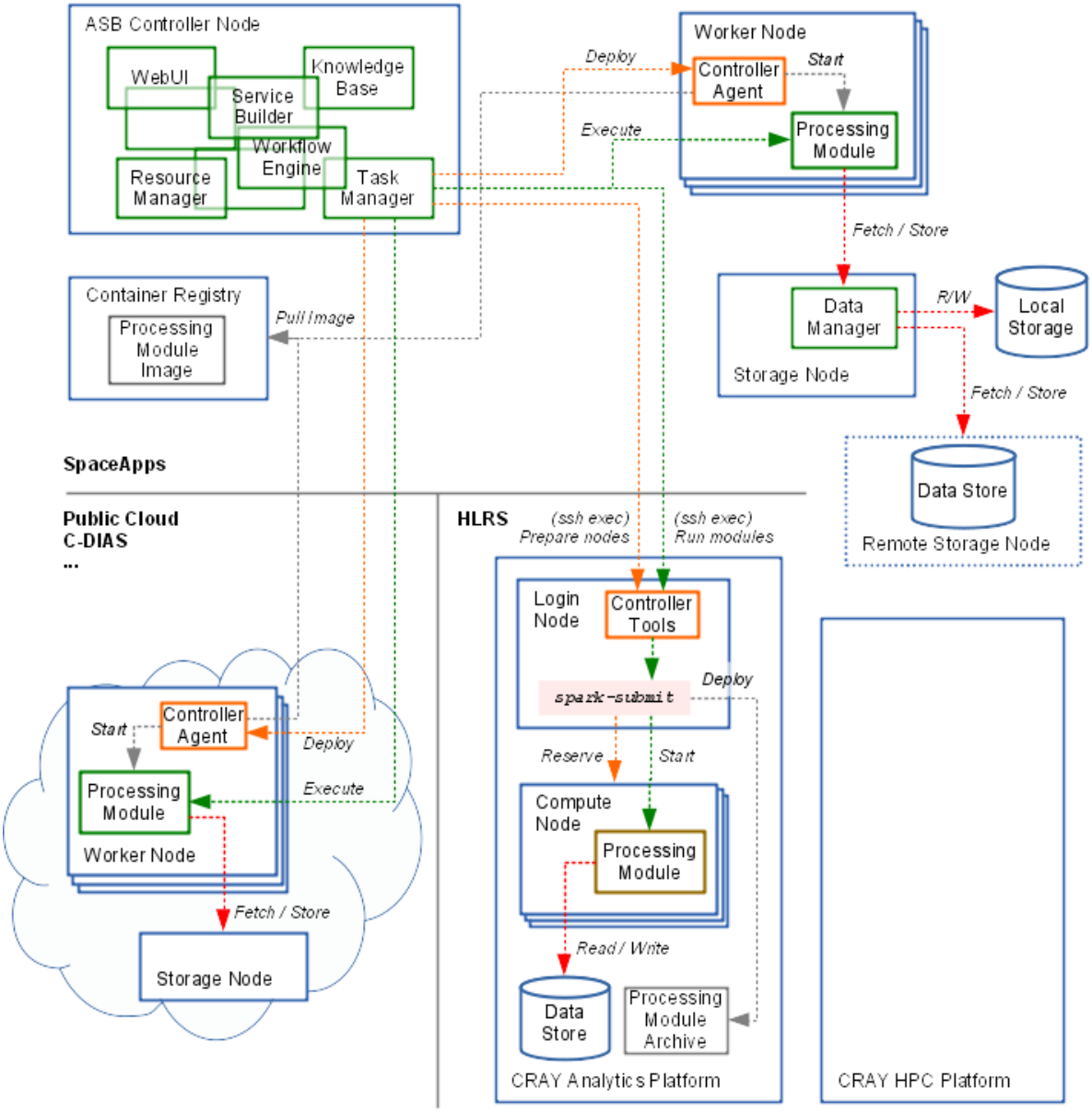


Figure 1 – High Level Architecture

3.2 Software Architecture

This section describes the software (frameworks, applications and libraries) that will be developed and integrated to form the EOPEN Platform.

The ASB Core Components are deployed before the platform may be used. The dynamicity in the platform lies in the processing services that are deployed on-the-fly when an execution is requested. This feature allows adapting the processing capabilities of the platform to the changing needs. The following mechanism is used:

- The Workflow Engine receives Workflow Execution Requests from the Service Builder. The execution of each workflow task is initiated as soon as all its input parameters are available. The task execution requests are transmitted to the Task Manager.
- The Task Manager communicates with the Resource Manager to obtain the reference to the appropriate (deployable) process image.
- The Resource Manager searches for the process image in container registries and provides back the necessary information to the Task Manager.
- The Task Manager selects the processing environment for execution of the workflow task (processing module) and for Cloud based processing a Virtual Machine available in the Cloud Environment. For each processing platform specific needs will have to be implemented in a back-end service. Generically speaking for cloud environments the Task Manager Agent (which runs in the VM) fetches the image from the Container Registry and deploys it as a new Processing Service. For HPC processing a trade-off is to be performed for the best and most appropriate API. Currently the selected API is Cloudify. As a first proof-of-concept command line tools such as `scp` and `ssh` will be used for the 1st release of EOPEN.
- The Task Manager executes the process in the Processing Service.
- When the process is complete, the Task Manager retrieves the outputs and un-deploys the Processing Service.
- The Workflow Engine, which in the meantime polls the task status, obtains the task outputs and move on with the next task(s) in the workflow.

Since the components executed asynchronously many processes may be executed simultaneously.

3.2.1 Cloud-based Processing

The platform has been developed in such a manner that the location and nature of the actual execution environment(s) are hidden from all but two of the core components: the Task Manager and the Resource Manager. All the other components are unaware of where and how the processes are executed.

This makes it possible to deploy the software in a so-called "stand-alone" mode where all the components as well as the processing chains are deployed and run on a single host. This setup is in particular convenient for developing and testing algorithms on small amount of data.

In "cloud" mode, the process execution environment is a Private or Public Cloud, where resources are scalable. The available processing resources, in the form of Virtual Machines, are automatically detected and used to deploy processes.

The "cluster" mode is a restricted "cloud" mode in which the available processing resources are considered as fixed, with no possibility to scale up or down the cluster.

Figure 2, below, presents the different modes and the possible deployments configurations.

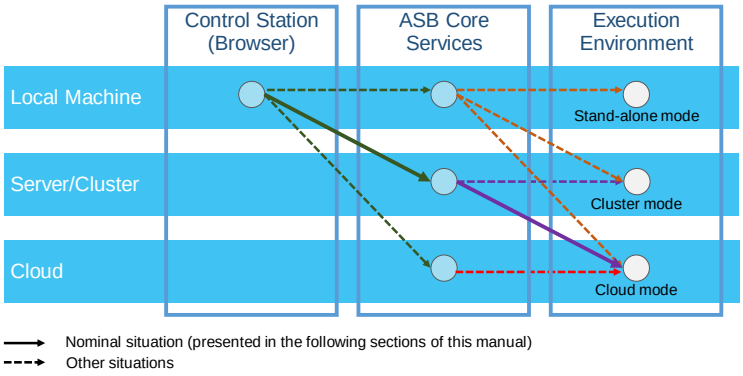


Figure 2 – ASB Execution Modes

A Docker solution is used because it provides a flexible and efficient means to dynamically deploy/un-deploy software services, and to run these with good performances.

In particular, containers have the big advantage to run directly on top of the hosting system, sharing the kernel, low-level libraries, as well as hardware resources such as CPUs, memory and disk. On the contrary, Virtual Machines run on top of a hypervisor, require a full Operating System to be installed in the virtual environment, and have reserved hardware resources.

Containerization is used both for developing the ASB Core Components (each component running in its own container), and for packaging the processes of the processing chains.

Figure 3 below presents the difference between Virtual Machines managed by a hypervisor, and Docker Containers, directly managed by the Host OS via the Docker Engine.

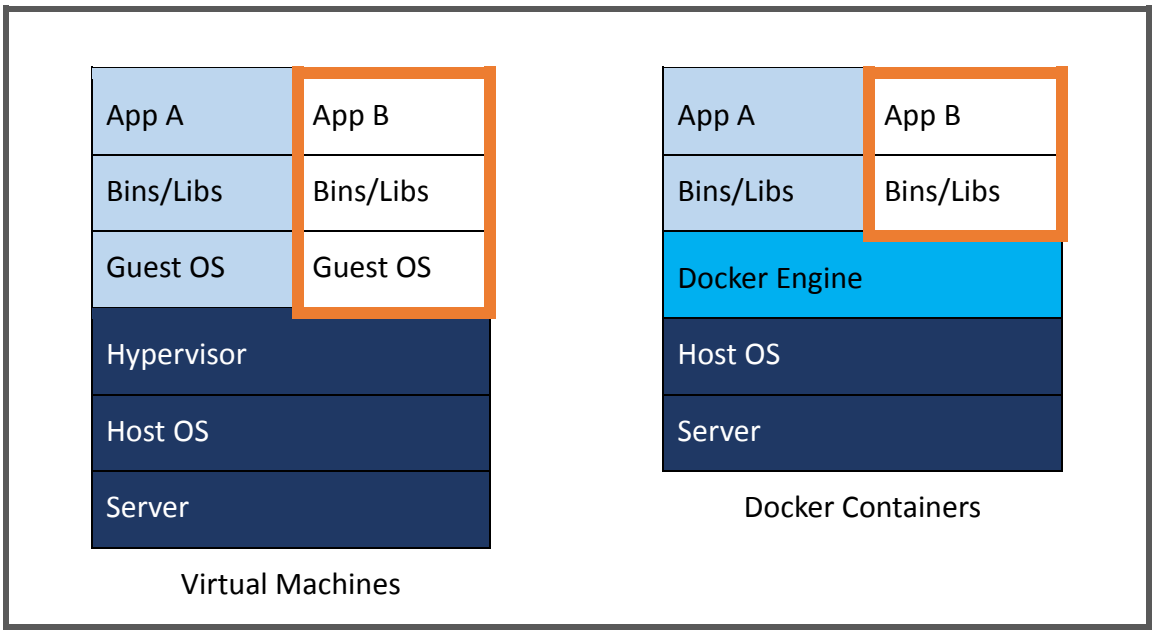


Figure 3 – Virtual Machines VS Docker Containers[2]

[2] Source: https://www.docker.com/whatisdocker/

Each of these components is packaged within a Docker container. A virtual network is configured in such a manner that the components behave as if they were connected on the same subnet, even if they are deployed on different hosts.

3.2.2 OGC WPS-Based Interface to Processes

The OGC Web Processing Service (WPS) standard offers a simple method for finding, accessing, and using all kinds of calculation models. Being an OGC standard, it is mostly used in geospatial infrastructures however nothing in the specification prevents using it in other contexts.

The WPS interface standardizes the way processes and their inputs/outputs are described, how a client can request the execution of a process, and how the outputs from a process are handled. WPS uses standard HTTP and XML as a mechanism for describing processes and the data to be exchanged.

The main advantage of WPS is interoperability of network-enabled data processing. It hides the underlying processes specificities by exposing a common standard interface.

The ASB software components as well as the processes available for composing processing chains are packaged within Docker containers. In each of the process containers a WPS-compliant service is deployed that acts as an adaptation layer for the actual executable process ("inner process"). The inner process typically implements a transformation or extraction algorithm.

This layered structure is necessary to integrate in ASB processes that have been implemented without taking the ASB design into account. The adaption layer allows, for example, invoking remotely a native executable file that was initially meant to be executed from a command line interface.

When a process execution request is received (from the Task Manager), the processing service transforms the input parameters into a representation understandable by the inner process sitting behind it. The processing service then initiates the execution of the inner process providing it with the appropriate inputs.

The processing service keeps track of the running inner process instances. Each instance is given a unique identifier that is provided back to the Task Manager and may be used to query its status and its outputs upon completion.

3.2.3 HPC and Big Data Process Executor

USTUTT acts primarily as an infrastructure provider in EOPEN, and thus is offering services and computing power to partners. Both a HPC system – a Cray XC40[3] having a peak performance of up to 7.42 PetaFLOPs – and a dedicated Big Data system, a Cray Urika-GX[4], are accessible to either perform embarrassingly parallel simulations or high performance data analytics (HPDA). Submitting jobs to both systems currently requires the usage of

[3] https://www.hlrs.de/systems/cray-xc40-hazel-hen/

[4] https://www.hlrs.de/systems/cray-urika-gx/

different resource and scheduling managers: TORQUE/PBS[5] for HPC and Mesos/Marathon[6] for HPDA. Both managers require expert knowledge to be efficiently used for resource allocation.

Furthermore, there exists---at least in the case of USTUTT's HPDA system---currently no means of interacting directly with the system other than direct command line access via a secure SSH connection. Disadvantages of using SSH also include no support for job monitoring and missing workflow management capabilities (e.g. uploading and downloading data before and after job execution).

As a result, EOPEN aims to provide access to USTUTT's infrastructure through the EOPEN platform via a single management interface. For this purpose, the orchestration tool Cloudify[7] is currently proposed. Cloudify provides access to both HPC and HPDA systems at USTUTT. The implementation of two additional plugins will enable Cloudify to manage HPC and HPDA resources in remote infrastructures. The core of HPC plugin was designed and implemented in the MSO4SC[8] project, but is limited to the SLURM resource manager not supported by USTUTT. Thus, the HPC plugin was further extended by USTUTT in the CoeGSS[9] project to support TORQUE/PBS. In EOPEN, we will continue the development of the HPC plugin and also start the implementation of an HPDA plugin from scratch to support deployment of Apache Spark applications using Mesos/Marathon.

Having both plugins, submitting a new job with either the Cloudify HPC or HPDA plugin comprises just a few steps. The most important step is the definition of a so-called blueprint. Each blueprint contains a mandatory OASIS TOSCA[10] file describing the application's topology. This file conveys information about the application's components, interrelationships, as well as instructions for installation, configuration, monitoring, and clean up. In addition to the TOSCA file, the blueprint may include auxiliary scripts (e.g., for bootstrapping and reverting the application), as well as additional data files. The blueprint is then uploaded to the Cloudify Manager running at USTUTT. Finally, end users will be able to run jobs on the defined target systems using Cloudify's API.

The EOPEN platform, specifically the ASB framework will interact with Cloudify in order to access infrastructure at USTUTT. Any future systems deployed at USTUTT are also planned to have support for deployment via Cloudify. Another advantage of using Cloudify is that any remote target system, e.g. a public Cloud provider such as Amazon AWS, can be selected.

---

[5] Garrick Staples. 2006. TORQUE resource manager. In Proceedings of the 2006 ACM/IEEE conference on Supercomputing (SC '06). ACM, New York, NY, USA, Article 8. DOI: https://doi.org/10.1145/1188455.1188464

[6] Benjamin Hindman, Andy Konwinski, Matei Zaharia, Ali Ghodsi, Anthony D. Joseph, Randy Katz, Scott Shenker, and Ion Stoica. 2011. Mesos: a platform for fine-grained resource sharing in the data center. In *Proceedings of the 8th USENIX conference on Networked systems design and implementation* (NSDI'11). USENIX Association, Berkeley, CA, USA, 295-308.

[7] https://cloudify.co/

[8] MSO4SC project: Mathematical Modelling, Simulation and Optimization for Societal Challenges with Scientific Computing) / http://mso4sc.eu.

[9] CoeGSS project: Centre of Excellence for Global System Sciences / http://www.coegss.eu.

[10] OASIS TOSCA Simple Profile in YAML Version 1.2: Committee Specification, 354 p., 2017.

3.2.4 Inter-Environment Communication

3.2.4.1 Public Clouds

ASB implements a back-end service to access processing environments. The BaaS hides all provider specific interfacing needs. Several public clouds are already offered from ASB. A pre-analysis of DIAS platforms has been performed and for the EOPEN selected DIAS platform a Back-end will be implemented

3.2.4.2 Big Data Environment

See section 3.2.3.

3.2.4.3 Data Transfer Module (KR16)

The data transfer module is required to provide a secure and fast data transfer between different sites in EOPEN. The module will rely on well-known protocols such as GridFTP[11], a "high-performance, secure, reliable data transfer protocol optimized for high-bandwidth wide-area networks." A limitation with respect to optimizing data transfer bandwidth is the fact that tools need to be installed and setup on both sites---at the source of data and the target system, where the data should be copied. Thus, EOPEN cannot improve data transfer speeds using parallel downloading tools like GridFTP when the source platform is not customizable. This is, for instance, the case for the Copernicus Open Access Hub, which uses predefined APIs to access data. Still, we can improve data transfer speed between sites managed and operated directly by EOPEN partners. Specifically, we are talking of connections between the data acquisition platform developed in WP3 and USTUTT's infrastructure, and between the ASB framework and USTUTT's infrastructure. In both cases, a client and server need to be installed, so that data can be transferred in a secure and fast way.

As a result, the data transfer module comprises the setup and configuration of parallel data transfer tools such as GridFTP between sites operated by EOPEN. For instance, a GridFTP server will be included in the data acquisition platform and a respective GridFTP client is running at USTUTT's infrastructure to retrieve data fast and efficient. Moreover, USTUTT will also setup a GridFTP server to allow for fast result retrieval from remote sites.

[11] http://toolkit.globus.org/toolkit/docs/latest-stable/gridftp/

4 INTEGRATED EOPEN PLATFORM

4.1 Introduction

Three versions of the integrated EOPEN platforms will be delivered in the course of the project. The first two versions are referred to as the first and the second prototypes and are meant to address a subset of the system requirements and of the needs of the use cases. The third version is the Final System. All three versions will be the result of the work on Task 6.2 "EOPEN System Integration" (M5-M33, March 2018-July 2020).

The first integrated prototype (D6.3) is due at M18 of the EOPEN project, that is in April 2019, the second prototype (D6.4) at M26 (December 2019), and the final system (D6.5) at M33 (July 2020).

Next section describes the overall software architecture. This architecture applies to all the versions and all the use cases that will be implemented in the platform.

The remaining of the chapter describes the modules and workflows integrated and configured in each version of the platform. It currently describes the first prototype and partially the second. It will be updated when new releases will be made available.

4.2 Software Architecture

The EOPEN platform is domain and application agnostic. The imported extension modules and the configured workflows determine which particular use cases are supported.

Figure 4, below, depicts the main constituents of the platform:

- The Web-based Developer Portal integrates the tools that allow module or application developers import their algorithms, configure the workflows and test their execution. The Developer Portal will be fully described in a User Manual delivered together with the EOPEN Platform.
- The Web-based User Portal allows end-users execute workflows and visualise the products and data available in the platform. As shown, the User Portal includes a series of Dashboards. Concretely, there are pre-defined dashboards but the users will also be able to create and configure their own custom dashboard pages. The features implemented in this component are further described in sections 4.3.4, 4.4.4. The User Portal will be fully described in a User Manual delivered together with the EOPEN Platform.
- The Core Components, already introduced in section 3.2, support the various core capabilities of the platform.
- The Container Registry keeps the packaged version of the imported processing modules. At execution time, modules are fetched from that registry.
- The Processes are the imported algorithms which, at workflow execution time, are dynamically deployed in remote worker nodes, executed, and then removed when their execution is complete. Their existence in the architecture is thus limited to their execution time (transient).
- The Services are manually deployed and are used by the transient processes. Some are also used by the User Portal for visualisation purpose. The services are persistent, which means they stay alive as long as they are not stopped explicitly.

- The Datastore is the location where input and output files are stored. It is centralised at first (all worker nodes reading and writing files in the same mounted file system). Eventually it will become decentralised, which will allow distributed processes to access files in their own environment.
- Data Providers are external to the EOPEN platform. They are used to obtain the data required by the applications and use cases. The supported providers and the downloaded data depend on the extension modules that are executed in the platform.

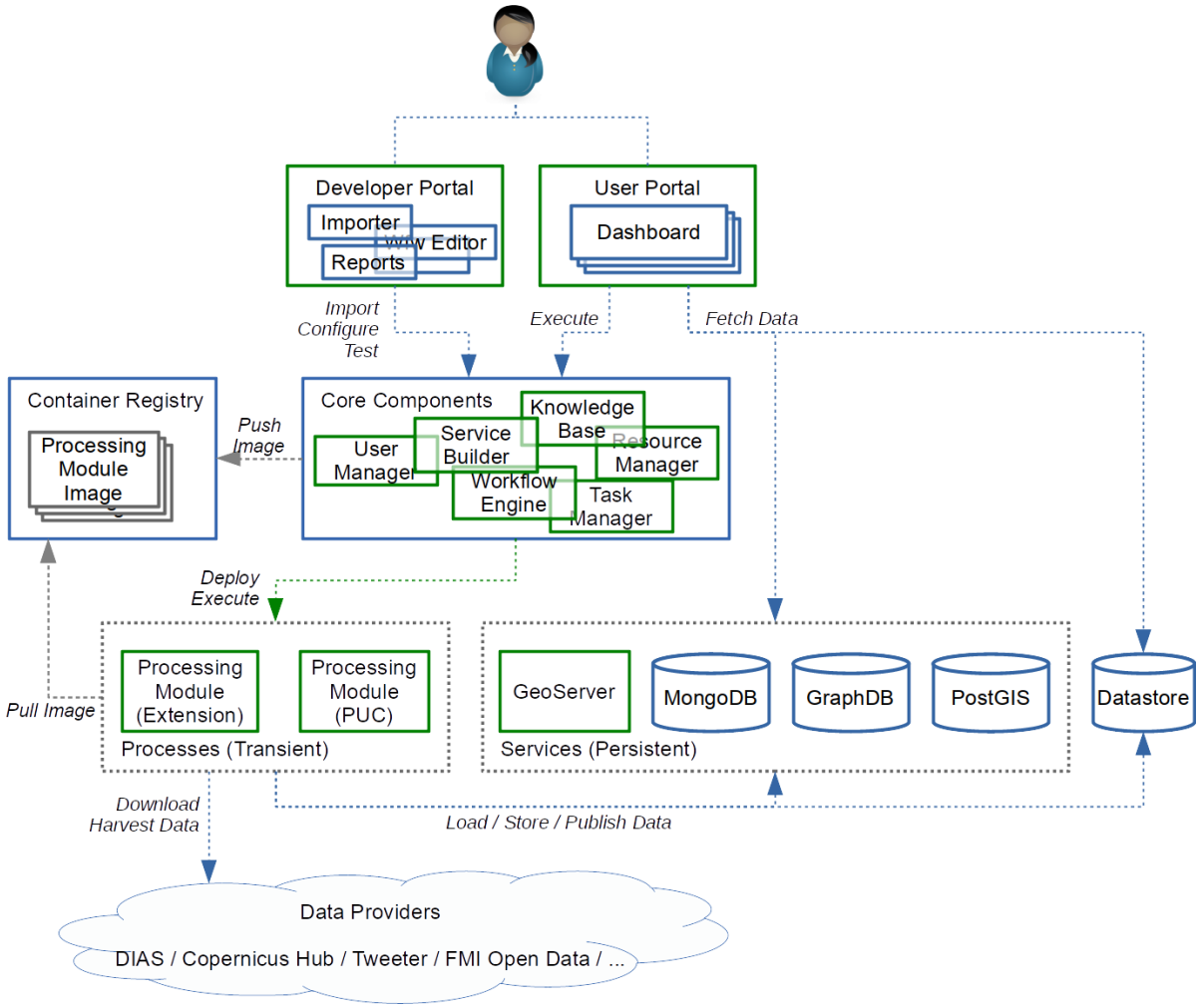


Figure 4 – EOPEN Platform Software Architecture

4.3 First EOPEN Prototype (D6.3, M18)

4.3.1 Overview

The First EOPEN Prototype is the result of the work on a variety of tasks at different levels and of different nature, which can be summarised as follows:

- Adaptation and implementation in the ASB Framework of new features required by the EOPEN Platform [D6.2] such as new orchestration capabilities and the support of GPUs-enabled processes.
- Initial support for the dynamic deployment and execution of processes in DIAS environments (ONDA and WEkEO in this version).

- Initial support for remote execution of processes in the HPC environment via Cloudify.
- Integration of processing modules implemented in Application Work Packages (WP3, WP4 and WP5), Pilot Work Package (WP7), and creation of processing chains.

4.3.2 Integrated Modules

This section introduces the modules that have been integrated in the first prototype. Integrated modules are the result of the work performed in other Work Packages. When appropriate, we are referring to the deliverables that describe these modules with more details.

4.3.2.1 Umbrella Application of Sentinel Hubs (KR10, NOA, Task 3.1, D3.1, M18)

The Umbrella Application of Sentinel Hubs implements the requirements included in section 2.3.8.1, page 18. The application is constituted of one database, which collects the Sentinel scenes metadata, and the following three maintenance processes:

- The **Hubs** process fetches the new products metadata from the configured Sentinel hubs and store these in the EO products database. This is automatically executed every 4 hours (00:00 CET, 04:00, and so on) by the EOPEN Platform built-in scheduler.
- The **Scores** process queries the configured Sentinel Hubs and determines which of these provides best performances in terms of availability, speed and integrity. The database is updated to reflect the new scores, whenever they change. This process is automatically executed once per hour, starting at 00:30 CET.
- The **Delete Products** process is in charge of removing from the database the EO products metadata that are not present in the Sentinel hubs anymore. This process is automatically executed once a day, at 01:00 CET.

The database is implemented by PostgreSQL with the PostGIS extension and runs as a service in the platform. The three processes introduced above have been implemented as Python scripts, and imported and configured in the platform using tools provided with the platform, including the process import tool, the workflow editor, and the scheduler.

Design and implementation details of the Umbrella Application of Sentinel Hubs (KR10A) may be found in deliverable D3.1.

4.3.2.2 Demand Driven EO-Data Provision Application (KR10B)

As described in section 2.3.8.2, page 19, the objective of this application is to provide the means to discover and download the available Sentinel data.

The application has been implemented as a service and two processes:

- The **Umbrella Application API** provides an HTTP-based discovery service on top of the EO Product Database. It has been deployed and started manually in the platform and it remains accessible to any process that needs to discover and download Sentinel data.
- The **Discovery** process is a module that communicates with the Umbrella Application API for identifying the Sentinel scenes that match search criteria. Obtained information includes the scenes metadata and download URL.
- The **Download** process is a module that receives download URLs and proceed with the actual retrieval of the scene files from the associated Sentinel hub.

The **Discovery** process and the **Download** process have been imported in the EOPEN platform using the developer user interface. They may be integrated in custom processing

chains (processor workflows). An example workflow that uses these processes is described in Figure 6, below.

4.3.2.3 Water Bodies Detection Modules (KR01, CERTH, Task 4.1, D4.1, M1-M33)

Two processes are being implemented by CERTH to generate water body masks derived from Sentinel-1 and Sentinel-2 products, respectively.

The first process uses images based on selected Sentinel-1 VV or VH band. It receives as inputs a list with multiple product filenames, an AOI polygon and optionally the desired bands to process (VV is default). It generates a GeoTIFF (raster) file and stores the product metadata in MongoDB. Note: A new version of this process integrated in the second prototype also outputs a Shapefile (vector) version of the same product (see section 4.4.2.1).

The second processes receives a list of Sentinel-2 products as input, it generates a water mask in GeoTIFF for each scene, and stores the metadata in MongoDB.

4.3.2.4 Social Media Crawling (KR12, CERTH, Task 3.2, D3.3, M33)

Several services have been implemented to perform the following operations:

Crawling and filtering of the tweets and storage in MongoDB (in collections depending on the target use cases).

Information about social media crawling may be found in section 10.2 of D3.1.

4.3.2.5 Event Detection in Social Media Data Module (KR02, CERTH, Task 4.2, D4.4, M3-M32)

The event detection module detects peaks in the daily number of tweets over a month. This allows for example detecting when the amount of tweets related to floods in a given region increases suddenly.

Information about event detection in social medial may be found in section 3.2 of D4.1.

4.3.2.6 Data Clustering Module (KR06, CERTH, Task 4.4, D4.2, M1-M36)

Apply a clustering mechanism to perform topic detection in Twitter records and return the results in JSON format.

Information about the module may be found in section 3.1 of D4.1.

4.3.2.7 Community Detection Module (KR07, CERTH, Task 4.5, D4.4, M33)

The community detection module analyses the links between the tweets ("re-tweets") and groups the tweet users into communities.

4.3.2.8 Generation of Linked Open Data (KR09, CERTH + SpaceApps, Task 5.2, D5.3, M7-M33)

A process has been implemented that allows converting structured data (including JSON, XML, and Shapefile) into RDF. NKUA's free and open-source tool GeoTriples is used to perform this conversion. The process receives as inputs the name of the input and output files, as well as the name of the files that contain the mapping rules and the applicable RDF namespaces.

4.3.3 Available Workflows

The main workflows available in the first prototype of the EOPEN Platform are the following:

- Workflows in charge of maintaining the catalogue of the Umbrella Application up to date.
- Workflow executing the crawling and analysis of social media data.

- Workflow chaining the event detection in social data module and the sending of the result by email and instant messaging.
- Workflow applying the water body detection algorithms on Sentinel-1 and Sentinel-2 products.
- Proof-of-concept workflow that triggers the execution of a process in an HPC environment (using the Cloudify interface) and outputs the results.

The workflows include the possibility to send notifications to users either by email or instant messaging (Mattermost). Workflows that generate products in GeoTIFF may publish these products as GIS layers through GeoServer.

4.3.4 Visualisation Capabilities

The first release of the End-User Web Portal includes the following pages and capabilities:

- A persistent menu bar gives access to the different pages of the portal. This includes an option list which allows selecting the language in which the interface must be displayed. Preliminary support for English, Italian, Finnish and Korean is provided.
- The Home page welcomes the visitor and explains the purpose of the portal.
- The Social Media page includes a form for searching for tweets, a result list showing the matching tweets and an interactive map where selected tweets may be geo-localised.
- The Map page includes an interactive map which allows visualising the products published as GIS layers through the integrated GeoServer. For example, water body masks encoded in GeoTIFF are published through GeoServer and visible on the map.
- The "About" page provides more technical information about the portal and the third party software libraries that are integrated.
- Other pages are included but only provide mockups of what will be implemented in the next releases.

4.4 Second EOPEN Prototype (D6.4, M26)

4.4.1 Overview

The following main activities have been performed in the core EOPEN Platform during the second period:

- Worker Nodes have been setup in the five DIAS environments. Currently four out of five are operational and may be used to deploy and execute individual processes. The fifth one (WEkEO) is in the works.
- The platform allows the workflow developers to constrain the individual process instances in the workflows to be deployed and executed in specific remote environments.
- The Developer Portal has been reworked to provide a more integrated and modern user interface and an enhanced user experience.
- The User Portal has been extended with the possibility for the users to create custom dashboard by selecting visual component and configuring their layout. The Web-based free open-source software OpenSphere has also been integrated to allow visualising GIS layers in a viewer backed by a community of developers.

In addition to this, in the second EOPEN Platform prototype, existing modules have been enhanced and new ones have been implemented, as described in the following section.

4.4.2 Integrated Modules

4.4.2.1 Enhancement: Water bodies detection modules

The two modules (applying to Sentinel-1 and Sentinel-2 respectively) have been extended to allow generating water body masks both in GeoTIFF and in shapefile. The shapefile version is expected to occupy less space on disk for the same level of details. Both GeoTIFF and Shapefiles may be published as GIS layers through GeoServer.

4.4.2.2 Location detection in social media module

This module analyses the text of the collected tweets, searches for toponyms, and applies geocoding to resolve the place names into coordinates. These coordinates may then be used to geo-localise the tweets on a map.

4.4.2.3 Rice paddy fields detection module

This module implements an algorithm required to implement PUC 2 on food security.

At first, algorithms and workflows have been implemented to train a Neural Network algorithm to obtain a model that can be re-applied on Sentinel-1 and Sentinel-2 scenes.

The model has been created and imported as a process. It may thus be used to generate rice paddy fields within workflows.

4.4.2.4 Publication of Shapefiles

The process that was used to publish GeoTIFF products as layers in GeoServer is extended to allow publishing Shapefiles as well.

4.4.2.5 Downloader and Harvesters

Processes have been implemented to allow automating the download of up-to-date data from remote providers to the EOPEN Platform.

Generic downloaders are meant to be used regularly to download the most recent version of a product, such as the most recent weather forecast data from FMI.

Generic harvesters are meant to be used to harvest in one operation historical data.

4.4.3 Available Workflows

The following workflows have been created or enhanced in the second integrated prototype of the EOPEN Platform:

- Workflow for training the rice paddy fields detection model.
- Rice paddy fields detection workflow which includes the publication of the generated products on GeoServer.
- Downloaders and harvesters scheduled to run at a fixed interval for ingesting remote data into the EOPEN Platform. These are used, for example, to download HIRLAM data from FMI and to harvest GlobSnow SWE L3A Products and SMOS Freeze-Thaw products.
- Test workflow which distributes the execution of process instances among the available DIAS environments.

4.4.4 Visualisation Capabilities

As mentioned in the overview section, the User Portal has been enhanced and extended. On the one hand, the users may now create and configure their custom dashboard pages, and on the other hand the third-part free open-source tool OpenSphere has been integrated to bring a more mature GIS Viewer into the Web portal.

4.5 Final System (D6.5, M33)

This section will describe the modules and workflows configured in the final release of the EOPEN Platform. It will be completed my M33.

5 PILOT USE CASES

5.1 Infrastructure

All three pilot use cases will be deployed and run on the same hardware infrastructure. It could be that specific use cases have specific requirements, such as a need to run processes in one of the HPC environments, but this has no impact on the general setup. The infrastructure will be created to support the current (and potentially future) use cases.

In general, the Worker Nodes are interchangeable. It does not matter which WN runs a specific process or service and the selection of a target WN is performed dynamically and automatically by the system. WNs are not fully interchangeable, though, as they may have different hardware characteristics (such as including a GPU). The hardware requirements of the processes are taken into account for selecting a target WN.

Figure 1 on page 32 provides a high-level view on the infrastructure put in place to support the development, execution and evaluation of the three PUCs, implying the generation of derived products as described in D1.4.

5.2 Validation

The EOPEN Platform may be compared to a generic operating system which integrates tools that allow developers import, configure and run their applications. The implemented EOPEN Platform will be assessed during the project using the strategies and against the criteria described in section 5.6 of D1.3 Self-Assessment Plan.

The services and processes that will be implemented and integrated in the EOPEN Platform have all different characteristics, apply on different input types and rely on different technologies, usually depending on specific constraints and objectives. For example, the social media analysis services use Natural Language Processing (NLP) whereas the rice paddy fields classification algorithm makes use of Machine Learning (ML) techniques.

There is thus no single strategy to validate the various modules that will be integrated in the EOPEN Platform for implementing the pilot use cases.

The validation of each algorithm is under the responsibility of the team in charge of its development. The validation techniques and results will be included in the deliverables of WP3, WP4 and WP5. Deliverable D7.1 also documents how each PUC is implemented and tested, as well as the expected performance of each Key Result.

The built-in modules provide generic functions such as downloaders, transformers, publishers. They are not meant to modify the data themselves and have thus no impact on the validity of the products generated by the extensions and use case specific modules.

5.3 PUC1 "Flood Risk Assessment and Prevention" Architecture

PUC1 makes use of social media data, tweets in particular, to detect the potential start of a flood event, and to monitor the progress. To accomplish this, tweet messages are collected and analysed to determine whether they relate to a flood event or not.

A potential flood event is detected when there is a sudden rise of the amount of positively matching tweets. At that time, the users are notified using the available channels (including email and instant messaging), and a workflow is executed for extracting waterbody masks

from the most recently sensed Sentinel-1 and Sentinel-2 scenes. These EOPEN products correspond to the Water Presence Maps (WPM) described in D1.4.

Both collected tweets and waterbody masks are published through a GIS server and made visible in the EOPEN user portal.

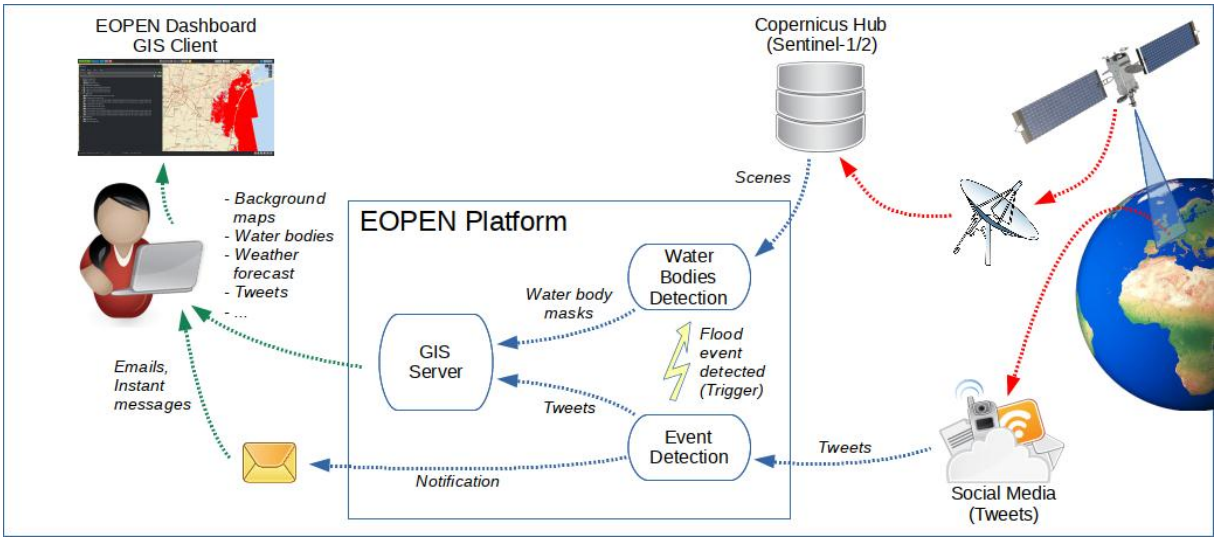


Figure 5 – PUC1 Overview

The following EOPEN built-in resources and extensions are involved in this scenario:

- Integrated databases PostgreSQL/PostGIS, MongoDB and GraphDB
- Integrated GIS server GeoServer
- The Social Media Crawling services and modules for collecting, analysing and storing the tweets data.
- The Umbrella Application (service) which collects and stores the Sentinel products metadata and exposes a discovery interface.
- Umbrella Application client modules for discovering and downloading the Sentinel products to be processed
- Water Body Mask detection module
- Built-in modules for sending emails and instant (Mattermost) messages
- Built-in module for publishing tweets data in GeoServer
- Built-in module for publishing GeoTIFF images as raster layers in GeoServer
- Built-in module for publishing Shapefiles as vector layers in GeoServer

In addition, the end-user portal is used to visualise:

- The instant messages
- Tweets data (searchable) in an ad-hoc dashboard
- Tweets data and water body masks in a Web-based GIS client

The following flowcharts depict the modules and the data flows used in this PUC:

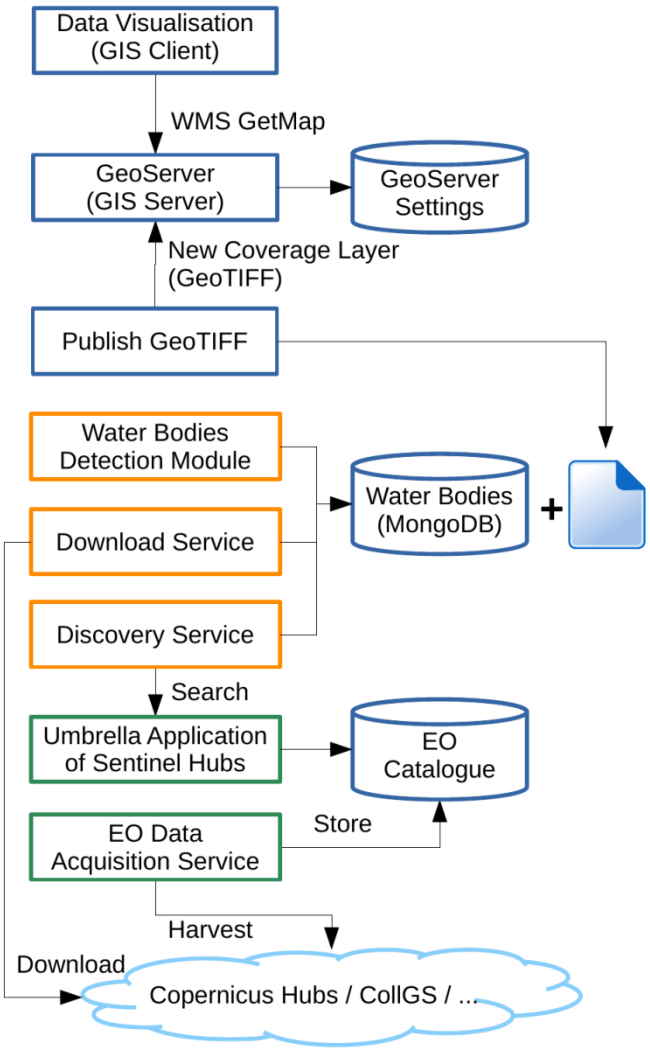


Figure 6 – Social Water Bodies Detection and Visualisation Flowchart

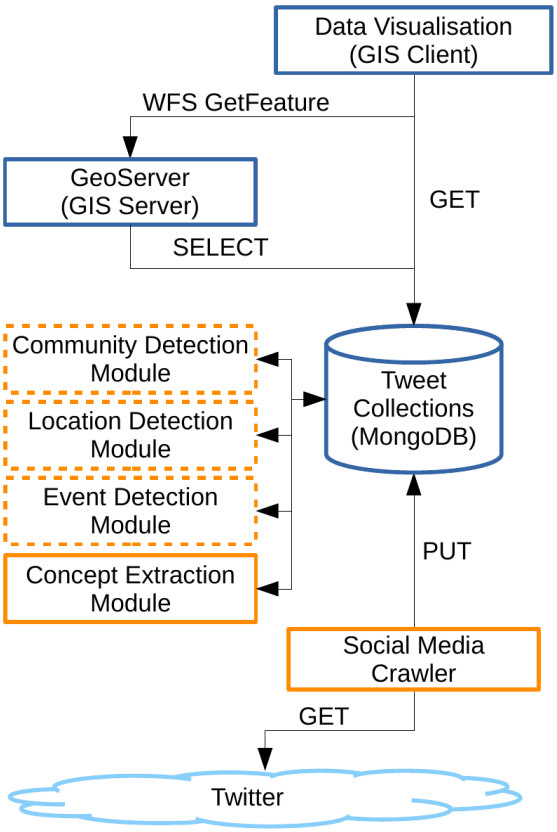


Figure 7 – Social Media Acquisition and Visualisation Flowchart

More detailed information about this PUC can be found in EOPEN D7.1 and in particular in section 5.2 dedicated to this use case. For the updated list of products consumed and generated in this PUC please see the living document D1.4 “EOPEN Data Management Plan”.

5.4 PUC2 "Food Security through EO Datasets" Architecture

As can be seen on Figure 8, PUC 2 implements two main services:

1. A rice paddy fields classification service for generating the EOPEN product Paddy Rice Mapping (ref. PRM in D1.4) and
2. A rice yield estimation service for generating the EOPEN product Rice Yield Estimation (ref. RYE in D1.4).

A third (not currently represented) EOPEN product will be the Rice (crop) Status Indicator (ref. RSI in D1.4) which will satisfy several requirements as listed in section 2.4.4.

The services make use of Machine Learning and Deep Learning techniques and will apply on Sentinel-1 and Sentinel-2 products discovered using the Umbrella Application. In addition to Sentinel data, the algorithms also use Land Use/Land Change products as well as meteorological data.

Both services require the training of dedicated models. The training of the rice paddy fields classification model will require a node with a GPU. The training of the rice yield estimation model will require much more data (timeseries) and thus computing power. This will be performed in HLRS' HPDA environment.

When executed on Sentinel products, the models generate products in GeoTIFF which are then published on the integrated GIS server. The products are thus visible in GIS client integrated in the EOPEN user portal. Emails and instant messages are issued each time a new product is generated.

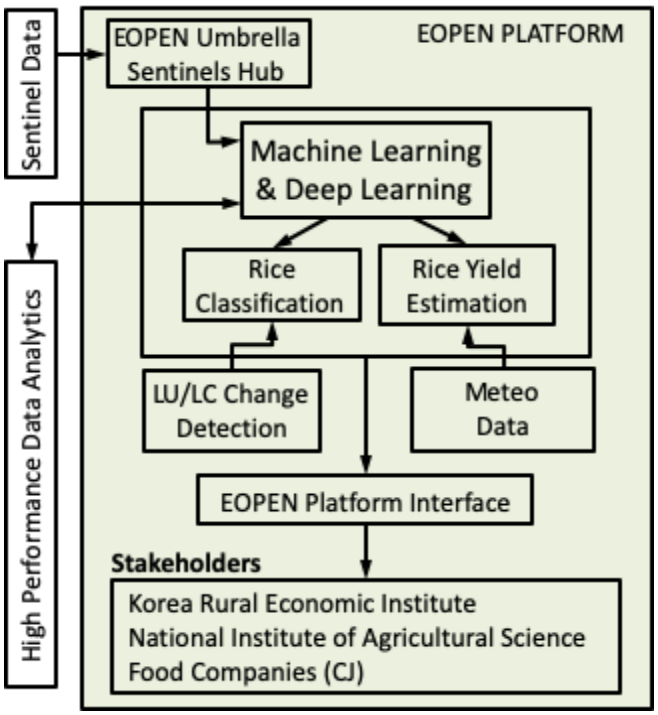


Figure 8 – PUC2 Overview

The following EOPEN built-in resources and extensions are involved in this scenario:

- Integrated database PostgreSQL/PostGIS (Sentinel metadata)
- Integrated GIS server GeoServer (published GeoTIFFs)
- The Umbrella Application (service) which collects and stores the Sentinel products metadata and exposes a discovery interface.
- Umbrella Application client modules for discovering and downloading the Sentinel products to be processed
- Rice paddy fields classification module
- Rice yield estimation module
- Built-in modules for sending emails and instant (Mattermost) messages
- Built-in module for publishing GeoTIFF images as raster layers in GeoServer

In addition, the end-user portal is used to visualise:

- The instant messages
- Rice paddy fields and rice yield estimations in a Web-based GIS client

More detailed information about this PUC can be found in EOPEN D7.1 and in particular in section 5.3 dedicated to this use case. For the updated and exhaustive list of products consumed and generated in this PUC please see the living document D1.4 “EOPEN Data Management Plan”.

5.5 PUC3 "Monitoring Climate Change through EO" Architecture

The objective of PUC 3 is to support the analysis of the impact of climate change to the reindeer herding livelihoods as well as for the infrastructure and transportation management in the Finnish region of Lapland. The main stakeholders are the reindeer herders, reindeer researchers and the Finnish Transport Infrastructure Agency (FTIA).

The herders' livelihood depends directly on the environmental and seasonal variation in snow cover, snow depth, temperature and onset of snow melting. Herders need to be able to adapt to sudden changes impacting reindeer pastures and to cope with feeding problems.

In this context, both historical and current data products will be harvested, transformed and ingested in the EOPEN platform and will be published through standard interfaces (GIS layers). The data may thus be visualised in an integrated manner on the EOPEN user portal.

The data products used in this use case include the following:

- Historical data of both air temperature and snow depth in Finland to estimate trends in local areas.
- Weather forecasts produced every 6 hours by FMI.
- Continuous data on the land surface temperature through the Sentinel LST product and on the snow cover and thawing conditions through the FMI GlobSnow and Freeze/Thaw products for monitoring purposes.

In addition, tools to plot the data and estimate both temporal/areal statistics will be provided and integrated in the EOPEN user portal. Together, these data and tools enable the stakeholders and users to mitigate their problems and better adapt to climate change.

The following EOPEN built-in resources and extensions are involved in this scenario:

- Integrated GIS server GeoServer (published GeoTIFFs)
- Extension modules harvesting data products (e.g. from FMI) at regular interval
- Built-in modules for sending emails and instant (Mattermost) messages
- Built-in modules for publishing GeoTIFF images and NetCDF files as raster layers in GeoServer

In addition, the end-user portal is used to visualise:

- The instant messages
- Any published GIS layer (weather forecast, land surface temperature, etc.) in a Web-based GIS client
- Time series (e.g. from NetCDF files) on interactive graphs.

More detailed information about this PUC can be found in EOPEN D7.1 and in particular in section 5.4 dedicated to this use case. For the updated and exhaustive list of products consumed and generated in this PUC please see the living document D1.4 “EOPEN Data Management Plan”.