HiSea Deliverable 5.5

Service Routing and Deployment Prototype I

Work Package Number: 5

Work Package Title: Service Implementation

HISea Project Information	
Project full title	High Resolution Copernicus-Based Information Services at Sea for Ports and Aquaculture
Project acronym	HiSea
Grant agreement number	821934
Project coordinator	Dr. Ghada El Serafy
Project start date and duration	1st January, 2019, 30 months
Project website	https://hiseaproject.com/

Deliverable Information	
Work package number	5
Work package title	Service Implementation
Deliverable number	D5.5
Deliverable title	Service Routing and Deployment Prototype I
Description	Services of HiSea will be realized on top of Docker as described in T5.1. As such, they will be available as independent services ("micro services"), which can be used by the clients of T5.2. Task 5.4 will interconnect those micro services using a service routing approach, which will most likely be handled via a reverse proxy such as Traefik (https://traefik.io/). The component will ensure that requests from clients are reaching the correct services. The task will also shield all services via an HTTPS layer and it will allow the detection of attacks. Additionally, services will be monitored for their availability. As soon as a service becomes unavailable, new services instances will be deployed via T5.5 and team members will be notified. There will be two deliverables in total. This will be the first one.
Lead beneficiary	Ascora

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 821934

Lead Author(s)	Danny Pape
Contributor(s)	Daniel Wegmann (ASC)
Revision number	4
Revision Date	28 Feb 2020
Status (Final (F), Draft (D), Revised Draft (RV))	F
Dissemination level (Public (PU), Restricted to other program participants (PP), Restricted to a group specified by the consortium (RE), Confidential for consortium members only (CO))	PU

Document History

Revision	**Date**	**Modification**	**Author**
0.1	20.02.2020	Create Document Structure	Danny Pape
0.2	20.02.2020	Fill Content	Danny Pape
0.3	20.02.2020	First internal review	Daniel Wegmann
0.4	28.2.2020	Review by management	Bracha Ehrman

Approvals

	Name	**Organisation**	**Date**	**Signature (initials)**
Coordinator	Ghada El Serafy	Deltares	28 Feb 2020	GES
WP Leaders	Danny Pape	Ascora	28 Feb 2020	DP

Executive Summary

The deliverable 5.5 Service Routing and Deployment Prototype I describes and shows screenshots of the first prototype of the HiSea Platform infrastructure. At this stage, all services are containerized and except the mobile apps, all demonstrators are also containerized. For the Docker Environment, a server has been rented in order to deploy most components on it. Apart from some demonstrator, which are hosted on other servers from consortium partners, all backend services are running on the rented server. Even though DIAS providers were the first choice for using the HiSea platform, the implementation of the HiSea services could not be achieved at this stage of the project, because of lack of actuality and connection problems to provided virtual machines.

Next Actions have been identified to enrich the current infrastructure with tools to reroute web requests and monitor the health of Docker containers while they are running. A further challenge is to establish orchestration with Docker Kubernetes.

Table of Contents

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 821934

1 Scope

In order to setup a project infrastructure for the HiSea platform, due to independent technology selection of the created components, the consortium agreed that employing Docker[1] as the base technology is the best strategy for successful and sustainable deployment. Docker is a software technology which provides operating-system-level virtualization, also known as containers. Docker provides an additional layer of abstraction and automation of operating-system-level virtualization on Windows and Linux. On top of docker, several services can be installed to route requests to related containers, monitor running containers, etc. For this, services such as Traefik and Prometheus are selected to cover those needs. Another decisive advantage of Docker is that, by being independent of the selected technology, the partner is not obliged to select a specific technology for the development of the platform components.

Docker also provides the ability to have continuous integration among the components. Each time a component has been updated, the current running docker container is being exchanged for the newly created one.

The greatest benefit of Docker is the platform independency that makes the HiSea platform ready to be integrated into every system and particularly with the targeted DIAS environment which is compatible with Docker and can be used to host the entire HiSea platform.

2 Planned Activities

The originally planned activities until the current stage of the project were to provide a first platform infrastructure to interconnect the HiSea components as microservices. The communication flow will then be handled via Restful APIs. To reach this goal, each partner was advised to deliver his components as docker containers, so that the dockers can be integrated into the platform in a straightforward way. The following table summarizes the planned objectives and the achieved goals.

Planned Goals	**Achieved Goals**
Each component must be created as a docker container, to be integrated smoothly to the Hisea platform infrastructure.	Most of the components provide docker files and/or docker compose files, which enables to create docker images out of those files and run them subsequently as containers.
The configuration and structure of docker containers must ensure to a container-to-container communication.	Each docker container has an assigned port, which prevents collisions and enables to address them

[1] https://www.docker.com/

	uniquely. Furthermore, the APIs are restful and are used for communication across the components.
Docker containers should be always Up-to-date.	Containers have already a continuous integration activated. Each time, the component is updated in the code repository and rebuild there, the docker container at the server is exchanged with the newly created one.
The HiSea Platform should be hosted in DIAS	Due to several unforeseen circumstances, it was not possible to use the DIAS environment to host the platform at this stage of the project (a few more information on the reason why this is not being possible are described in section 3 of this deliverable)
Routing	Traefik should be used to control an uncomplicated routing of the requests. This is also used to accept https requests. This goal has not been achieved so far, because of permanent issues with the DIAS providers.
Orchestration of Docker containers	At this stage, Docker orchestration is only implemented in Docker Compose files. This will be changed in the future by using Docker Kubernetes.

3 Progress

In the following sections, the infrastructure is detailed. The infrastructure was also successfully demonstrated at the 1st review of the project duration.

3.1 Docker containerisation

Except for the mobile prototypes of the dashboard and the alarm app, all services and prototypes are running as Docker containers. The list of the containerized applications follows.

- Demonstrators
  - HiSea Dashboard (web)
  - MSFD-eutro Model
  - COASTSERV Model

    - Oil Spill Service
    - Pump station observation
- Services
    - Timeseries Service
    - NETCDF Downloader
    - NETCDF API
    - Grid Data Service
    - Database

3.2 Deployment

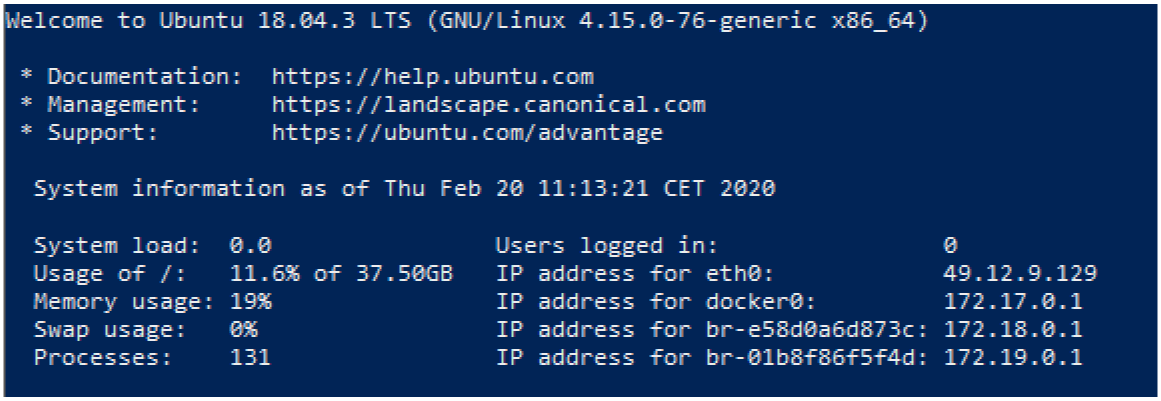
The original deployment plan was to set up Docker on a DIAS server, however, the platform is not yet deployed in DIAS server due to the current stage of DIAS infrastructure. A few more detailed information on the causes of such are described in the section below.

3.2.1 DIAS

Through a comparison of the individual DIAS providers, it was found that the most promising providers are WEKEO and CREODIAS. Therefore, test accesses were set up by these providers to test the environment and the features. Unfortunately, despite support from the Wekeo team, it was not possible to establish a SSL connection to the virtual machine. The Consortium is still busy working on establish such a connection and positive that the latest will be achieved soon. The second provider CREODIAS could be tested successfully, however, real time data were not provided, so that only very old data set could be used. For this reason, the use of the CREODIAS virtual machine was not pursued further.

3.2.2 Server

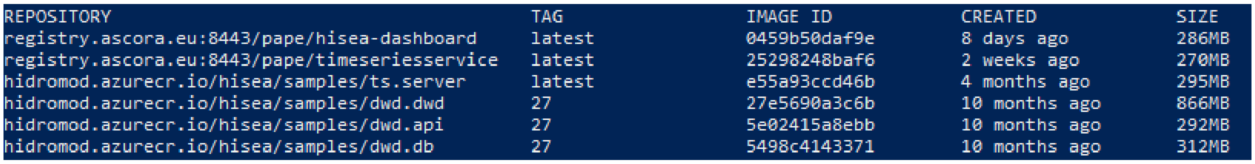
Due to the problems to get a DIAS server running, it was decided to use a backup solution to start the deployment of components. Therefore, an independent external server was rented to setup a running HiSea platform. Docker was installed on the machine, and docker images were pushed to it. The server is secured via an SSL/TLS connection and the use of public and private keys. Ascora personnel takes care of this server and have access to it. The different components are then downloaded from private Docker registries from Hidromod and Ascora. To finally run those images as containers docker compose and dockerfiles are used. To avoid address complications and collisions, care has always been taken to use different ports. At this stage the server is accessible via an IP (e.g. http://49.12.9.129:338/), another domain can also be obtained very quickly. This temporary solution on a rented server currently has no https certificate, necessary to secure the individual web components. However, if it is foreseeable that this solution will continue in the long term, the necessary certificates will be obtained quickly and easily from Let's Encrypt[2]. Further details of the server can be viewed in Figure 1.



```
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-76-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Thu Feb 20 11:13:21 CET 2020

  System load:  0.0                Users logged in:                0
  Usage of /:   11.6% of 37.50GB   IP address for eth0:            49.12.9.129
  Memory usage: 19%                IP address for docker0:         172.17.0.1
  Swap usage:   0%                 IP address for br-e58d0a6d873c: 172.18.0.1
  Processes:    131                IP address for br-01b8f86f5f4d: 172.19.0.1
```

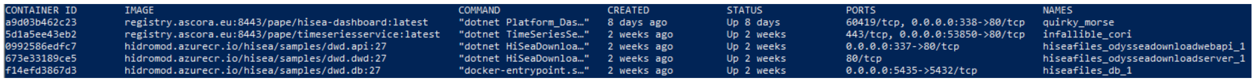
Figure 1: Server Information

On this server, only the backend services on which the demonstrators run, and the Web Dashboard are running at the moment. The other applications run containerized on different servers of the respective partners. The currently running components are integrated as images in Docker (see Figure 2) and are also running as containers (see Figure 3).



```
REPOSITORY                                       TAG         IMAGE ID        CREATED          SIZE
registry.ascora.eu:8443/pape/hisea-dashboard     latest      0459b50daf9e    8 days ago       286MB
registry.ascora.eu:8443/pape/timeseriesservice   latest      25298248baf6    2 weeks ago      270MB
hidromod.azurecr.io/hisea/samples/ts.server      latest      e55a93ccd46b    4 months ago     295MB
hidromod.azurecr.io/hisea/samples/dwd.dwd        27          27e5690a3c6b    10 months ago    866MB
hidromod.azurecr.io/hisea/samples/dwd.api        27          5e02415a8ebb    10 months ago    292MB
hidromod.azurecr.io/hisea/samples/dwd.db         27          5498c4143371    10 months ago    312MB
```

Figure 2: Docker Images



```
CONTAINER ID   IMAGE                                                   COMMAND                    CREATED         STATUS        PORTS                                 NAMES
a9d03b462c23   registry.ascora.eu:8443/pape/hisea-dashboard:latest      "dotnet Platform_Das…"    8 days ago      Up 8 days     60419/tcp, 0.0.0.0:338->80/tcp        quirky_morse
5d1a5ee43eb2   registry.ascora.eu:8443/pape/timeseriesservice:latest    "dotnet TimeSeriesSe…"    2 weeks ago     Up 2 weeks    443/tcp, 0.0.0.0:53850->80/tcp        infallible_cori
0992586edfc7   hidromod.azurecr.io/hisea/samples/dwd.api:27             "dotnet HiSeaDownloa…"    2 weeks ago     Up 2 weeks    0.0.0.0:337->80/tcp                   hiseafiles_odysseadownloadwebapi_1
673e33189ce5   hidromod.azurecr.io/hisea/samples/dwd.dwd:27             "dotnet HiSeaDownloa…"    2 weeks ago     Up 2 weeks    80/tcp                                hiseafiles_odysseadownloadserver_1
f14efd3867d3   hidromod.azurecr.io/hisea/samples/dwd.db:27              "docker-entrypoint.s…"    2 weeks ago     Up 2 weeks    0.0.0.0:5435->5432/tcp                hiseafiles_db_1
```

Figure 3: Docker Containerization

4 Next Activities

The next goal is to unite and orchestrate all running Docker applications on a virtual server in the DIAS environment. Thanks to the flexibility of the platform built so far, the move of a Docker environment will require only little effort in contrast to an orchestration. Kubernetes was chosen for orchestration, as Kubernetes will be developed further in contrast to Docker Swarm. As soon as a movement to a DIAS

[2] https://letsencrypt.org

provider is possible, the images and containers will be pushed to the virtual machine of a DIAS server, and the work continues there. Additional services will be integrated to facilitate the routing of the individual components, including SSL encryption, such as Traefik[3]. To detect issues early, monitoring will be established that monitors the running containers and server resources. This is necessary to enhance visibility into the performance of the entire containerized environment, including container orchestration.

In summary, the following actions will continue:

- Setup on DIAS infrastructure
- Reverse Proxy and Routing with Traefik
- Container Monitoring with Prometheus
- Establishment of Continuous Integration

[3] https://docs.traefik.io/