PROJECT DELIVERABLE REPORT

NAIADES

Greening the economy in line with the sustainable development goals

D3.12 NAIADES Communication Platform – WMS – Mid-term

A holistic water ecosystem for digitisation of urban water sector

SC5-11-2018

Digital solutions for water: linking the physical and digital world for water solutions

"This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 820985"

Document Information

Grant Agreement Number	820985	Acronym	NAIADES
Full Title	A holistic water ecosystem for digitization of urban water sector		
Topic	SC5-11-2018: Digital solutions for water: linking the physical and digital world for water solutions		
Funding scheme	Innovation action		
Start Date	1st JUNE 2019	**Duration**	36 months
Project URL	www.naiades-project.eu		
EU Project Officer	Alexandre VACHER		
Project Coordinator	CENTER FOR RESEARCH AND TECHNOLOGY HELLAS - CERTH		
Deliverable	D3.12 NAIADES Communication Platform – WMS – Mid-term		
Work Package	WP3 - Data and Sensors Infrastructure		
Date of Delivery	**Contractual** M16	**Actual**	M16
Nature	R - Report	**Dissemination Level**	PU-PUBLIC
Lead Beneficiary	SIMAVI		
Responsible Author	Marius Jianu	Email	marius.jianu@siveco.ro
	Andreea Paunescu	Email	andreea.paunescu@siveco.ro
Reviewer(s):	Efthimios Bothos (ICCS), Babis Magoutas (ICCS), Matej Posinkovic (JSI)		
Keywords	Cloud platform, Data Collection Aggregation, IoT platform, KSI		

Revision History

Version	Date	Responsible	Description/Remarks/Reason for changes
0.1	17/08/2020~ 11/09/2020	Marius Jianu Andreea Paunescu	Report write-up
0.2	15/09/2020 ~ 22/09/2020	ADSYS, UDGA, DISY, ICCS, GT	Inclusion of partners' contributions
0.3	25.09.2020	Marius Jianu Andreea Paunescu	Releasing the document for the Internal Review

1.0	28.09.2020	Marius Jianu Andreea Paunescu	Applying internal comments and final submission
2.0	11.05.2021	Andreea Paunescu	Applying PO comments

Contents

List of Tables

List of Figure

Abbreviations

Amazon EC2	Amazon Elastic Compute Cloud
API	Application Programming Interface
AWS	Amazon Web Service
CDM	Context Data Management
CPU	Central Processing Unit
DCA	Data Collection Aggregation
DFM	Data Fusion Middleware
GUI	Graphical User Interface
HMI	Human Machine Interface
HTTP	Hypertext Transfer Protocol
IAM	Identity and Access Management
IoT	Internet of Things

IP	Internet Protocol
KSI	Keyless Signatures Infrastructure
NGSI	Next Generation Safeguards Initiative
REST	Representational State Transfer
SDK	Software Development Kit
SSD	Solid-State Drive
SSH	Secure Shell network
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UI	User Interface
URL	Uniform Resource Locator
WMS	Water Monitoring System
WP	Work Package

1 Summary

This report describes the NAIADES Communication Platform and it reveals the main aspects regarding its role in providing an environment that supports all NAIADES processing layers which perform monitoring, analysis and reporting of the measured, calculated and simulated parameters for water consumption, production and storage, comfort requirements of end-users, meteorological conditions and cost in real time.
The deliverable also provides an overview of the communication between the NAIADES microservices which will be installed in the platform.

All the significant information about the cloud infrastructure employed for the NAIADES Communication Platform is provided, including the hardware specifications and microservice architecture structure.

Other relevant aspects of the deliverable include the provision of details on the installation of the Communication Platform, the GUI presentation, as well as the security mechanisms employed, all of which are covered in the respective chapters.

2 Introduction

D3.12 NAIADES Communication Platform reports on the outcomes of Task 3.5 NAIDES Communication Platform – WMS and it represents a detailed description of the cloud platform and all the microservices that will be deployed on it, as well as the way those microservices communicate with each other.

2.1 Scope and objectives of the deliverable

The scope of the deliverable is to get an overview of the system's architecture and to reveal how all the microservices will be deployed on the cloud platform.

This report also aims to show the way both the internal users and the external users communicate with the cloud platform.

2.2 Structure of the deliverable

The deliverable was thought to contain all important information about:

- The general aspects of the platform;
- The microservices that will be deployed in the platform;
- How the platform can be installed;

3 Communication Platform -WMS

3.1 Introduction

Cloud computing is the on-demand delivery of Information Technology resources over the Internet and allows to set up the virtual environment, to give users the flexibility of connecting to business services from anywhere, any time. With the growing number of web platforms used in today's business

environment, access to needed data is even easier. Deploying the services to cloud computing may reduce the cost of managing and maintaining a physical IT system. Cloud applications can scale up or scale down the processing resources and storage needs quickly to suit the needs, rather than purchasing and installing expensive upgrades in a physical machine. For a processing continuity whether can happen a natural disaster, power failure or other crisis, having data stored in the cloud ensures it is backed up and protected in a secure and safe location.

The collaboration between developers who integrate cloud services is more efficient in a cloud environment and gives to the entire ecosystem the ability to communicate and share more easily outside the traditional methods. Cloud computing offers to NAIADES solutions many benefits as it allows a robust and flexible communication among different systems not counting the technologies and languages used for the various internal services. Communication Platform WMS represents the Cloud computing environment where the NAIADES services and storage will exist. NAIADES Communication Platform is a cloud-based monitoring and control platform. The platform will be the production environment where NAIADES developers will deploy stable features, databases and will do backed-up regularity. In this way all NAIADES pilots, DCA and external applications can easily integrate with NAIADES cloud services.

In the chapters below, the whole image of the Communication Platform can be viewed starting with the general information about the main flow, continuing with the cloud communication, the installation details and GUI presentation and the Security mechanism.

3.2 Main Flow

For a better understanding of the Communication Platform, this chapter is dedicated to the general aspects and also to the specific aspects regarding authentication and obtaining and sending measured data to the cloud platform. In Figure 1 *NAIADES Architecture*, the layers of NAIADES Communication WMS are presented.

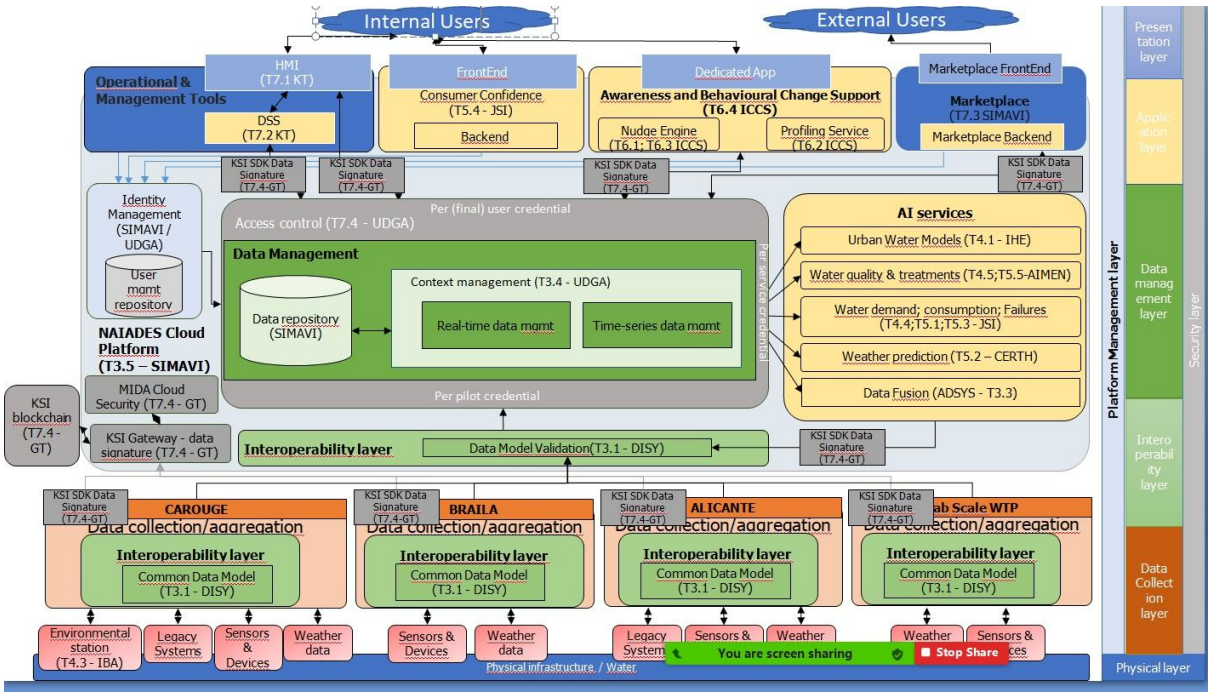

Figure 1 NAIADES Architecture

3.2.1 General

The Data Collection Aggregation modules (detailed in D3.9) for the four pilots: Carouge, Alicante, Braila, and Lab Testbed will connect and send measurements data from sensors to NAIADES cloud platform where the NAIADES service components are running. All data for each pilot will be collected, parsed and aggregated.

The parsed and aggregated data will be processed and saved in Data Repository in order to provide alerts, notification and predictions through an HMI console for each pilot, based on its own use cases.

3.2.2 Authentication

To establish the communication between Data Collection Aggregation and cloud platform, the cloud platform will provide a dedicated microservice for authentication provided by T7.4. NAIADES Security Mechanisms.

The authorization at the cloud platform level will be done using Identity Management using acces control layer (described in T7.4) that will validate users requests received from the DCAs, from UIs (HMI and Marketplace) and from external applications (via Marketplace backend API). KSI can sign and verify the integrity of the requests data.

3.2.3 Obtaining measured data

Data Collection Aggregation will receive raw data from sensors containing different measurements at a given time period.

Data Collection Aggregation will use common data models described in D3.1 Data Harmonization Framework and Tool in order to model the raw measurements data used. These measurements will be sent to the NAIADES cloud platform, as modeled data content, with different time frequency, depending on each pilot and on each measurement type. The lowest frequency of data transmission will be once a day, and the highest frequency will be every hour.

3.2.4 Sending measured data

NAIADES Cloud platform will receive the modeled data from Data Collection Aggregation, in the Interoperability layer, to be first validated using Data Model Validation module described in D3.1 Data Harmonization Framework and Tool.

Inside the cloud platform, if the data Model Validation module validates the data sent, the data will be passed to Context Data Management (described in D3.9).

3.3 Cloud Communication

The Cloud Communication sub-item presents information related to the business aspects of the platform, about the technical performances as well as about the structure of its microservices.

3.3.1 Cloud platform technical information

NAIADES cloud platform is an Amazon Elastic Compute Cloud (Amazon EC2). The Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. Amazon EC2's web service interface allows to obtain and configure capacity with minimal friction. It provides the complete control of computing microservices resources and lets run on Amazon's proven computing environment. Also, it reduces the time required to obtain and boot new server instances to minutes and has the ability to quickly scale the system in both ways, up and down. In such way, the administrator has the power to control the resources in use at any time.

Amazon EC2 provides a truly elastic computing environment, enables to increase or decrease capacity within minutes, control the dockers, adjust the space, etc. NAIADES EC2 instance will be based on operating system Ubuntu, that will permit to install the NAIADES IoT Platform SDK microservices (integrated results of T3.4, WP4, WP5 and WP6), the HMI and the Marketplace (results of WP7).

Microservices deployed on the NAIADES cloud platform, including (1) IoT Platform, (2) Operational and Management tools, (3) Awareness and Behavioral Change Support, (4) Marketplace, are created using open source license tools and programming language. All these SDKs are dockerized using docker and docker-compose configuration. Docker and docker-compose are open source license tools, implicating, that NAIADES platform does not need dedicated services. The SDK can be installed and run using the EC2 Amazon instance.

All major cloud computing providers like Amazon Web Services (AWS), have embraced Docker's availability and added individual support. Docker containers can be run inside Amazon EC2 instance using docker commands via Ubuntu OS. One of Docker's greatest benefits is portability and that it is an open source tool, bringing several advantages to its users: (1) it saves costs; (2) enables standardization for development, build and test. In order to accomplish these, NAIADES SDKs will be deployed in the cloud platform as containerized services using Docker containers. All environment variables and tools used to build a docker image are set internally in docker configurations files and does not affect the rest of images. In NAIADES cloud platform, the communication between docker containers will be done using a docker-compose configuration file, that will put all containers in the same network.

The current amount of available storage space in the cloud, at this moment, is 200 GB.

The cloud platform will also have an elastic IP option. An Elastic IP address is a static, public IPv4 address designed for dynamic cloud computing. It can associate address with any instance or network interface in any Amazon Virtual Private Cloud in the account. With an Elastic IP address can be masked the failure of an instance by rapidly remapping the address to another instance in the Amazon Virtual Private Cloud.

3.3.2 Cloud performance

In the first stage of the project, NAIADES application has concurrent access limited to 4 real user groups, associated to four pilots: Carouge, Alicante, Braila and the Lab Testbed. It has been estimated, that for current stage, the Amazon cloud platform should be limited to 16GRAM and 4 CPU with 200 GB storage

The concurrence access will be increased by the access of the Marketplace by external users, but this will happen in the next steps, and is not part of the pilot's test cases in this midterm report.

3.3.3 Cloud microservices structure

The SDK of each microservice deployed on the cloud will be deployed as docker and docker-compose image. In that way, the microservices could be analyzed, tracked and administered separately/independently.

The applications will be installed as docker images and the communication between the dockers will be done using the configuration within docker-compose files based on image names, inside the cloud.

The platform will be the PRODUCTION environment as prod-platform, an environment where NAIADES developers will deploy stable features, DataBases and backed-up it regularly.

The NAIADES cloud platform will have a suggestible domain name like "NAIADES-application-eu" instead of numeric IP and will be accessible from the outside through two ports only (see Table 1).

Environment	Host	Opened ports	Services running
PRODUCTION	e. g. NAIADES-application-eu	80/tcp 443/tcp	Not public

Table 1 Cloud public ports

IoT Platform services:

The microservices containing Naiades Cloud Platform services can be found in the Table 2 Cloud internal services:

Service	Purpose	open API (REST)
db-crate	database for timeseries (historical data)	None
db-mongo	database for context-manager (currrent data)	None
db-mysql	database for identities and roles	None
fiware-keyrock	manages identities and roles	e.g NAIADES-application-eu/identity-api
fiware-orion	context manager, API server for all entities	e.g. NAIADES-application-eu/context-api
fiware-orion-proxy	enforce access control	WIP
fiware-quantumleap	REST service for storing, querying and retrieving NGSI v2 spatial-temporal data	e.g NAIADES-application-eu /time-series-api
wms-app-example	Dummy example of a WMS component, acts as a consumer(subscriber) and publisher of messages to NAIADES	None
NAIADES-HMI	NAIADES Console Human interface dedicated to internal pilots' users	e.g NAIADES-application-eu or e.g NAIADES-application-eu/login
Marketplace-HMI	NAIADES Marketplace Human Interface dedicated to external users/applications	e.g NAIADES-application-eu/marketplace
Marketplace-backend	NAIADES Marketplace backend dedicated to external users/applications	e.g NAIADES-application-eu/marketplace-api

Table 2 Cloud internal services

Marketplace services

In Table 3 Cloud Marketplace services, are presented Marketplace services that will run in the cloud

Service	Purpose	open API (REST)
Marketplace-HMI	NAIADES Marketplace Human Interface dedicated to external users/applications	e.g NAIADES-application-eu/marketplace
Marketplace-backend	NAIADES Marketplace backend dedicated to external users/applications	e.g NAIADES-application-eu/marketplace-api

Table 3 Cloud Marketplace services

KSI service

In Table 4 Cloud KSI service, is presented KSI service that will run in the cloud.

Service	Purpose	Open API (REST)
KSI - Gateway	KSI gateway tool for establish communication with KSI blockchain	e.g NAIADES-application-eu/authorization-api

Table 4 Cloud KSI service

The cloud platform internal API will be accessed using an NGINX reverse proxy tools described in section 3.5

3.3.3.1 Communication between Data validation and Context Manager.

The data validation module is a module that validates that an HTTP message body complies to the specified data models by the project. It just bypasses messages that do comply with the models or else returns an HTTP error code. No specific communication interface is expected for this integration.

3.3.3.2 Communication between Context Manager and other components

The communications between context manager and internal components such as AI modules, Data Fusion module, etc. all happens over HTTP. These interactions are based on the NGSI-v2 standard. The interfaces are documented in deliverable D3.9. NAIADES IoT Platform – midterm.

Please refer to D3.9 for further information.

3.3.3.3 Communication between Data Collection Aggregation and Cloud Platform

Proxying mechanism is used to distribute the load among several servers, seamlessly show content from different websites, or pass requests for processing to application servers over protocols other than HTTP. In Naiades Cloud Platform we use proxy to distribute requests on HTTP protocols.

Naiades Cloud Platform will use NGINX reverse proxy tool. When NGINX proxies a request, it sends the request to a specified internal server, fetches the response, and sends it back to the client. NGINX can proxy requests to an HTTP server (another NGINX server or any other server) or a non-HTTP server (dedicated microsevice) using a specified protocol. Supported protocols include common Gateway Interface, Web Server Gateway Interface, Simple Common Gateway Interface and Distributed Memory-Caching System.

Passing a request to an HTTP proxied server, the proxy_pass directive is specified inside a location. The measurements data from Data Collection Aggregation will be sent to the cloud using HTTP protocol with REST web service API. The cloud NGINX proxy will redirect the request based on the URL text containing to the Data Validation module.

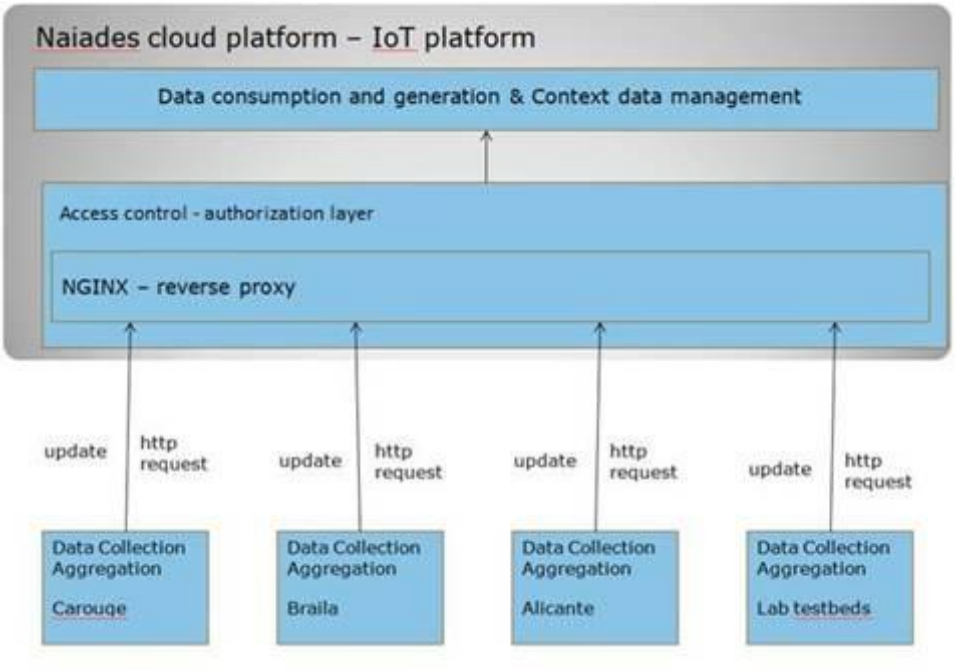

Figure 2 Cloud Proxy

The internal communication between Data Collection Aggregation with Identity Management, DCA with Data Validation, Data Management is described in D.3.9.

3.3.3.4 Communication between KSI Blockchain and KSI SDKs in NAIADES cloud platform

The communication between Guardtime's KSI Blockchain and KSI Gateway will be done over the TCP and UDP protocol. Communication between KSI Gateway and KSI SDKs will be done over HTTP(S) (which is RESTful) and if needed the TCP can be used instead. The request from outside the platform will be handled by NGINX reverse proxy and redirect internally.

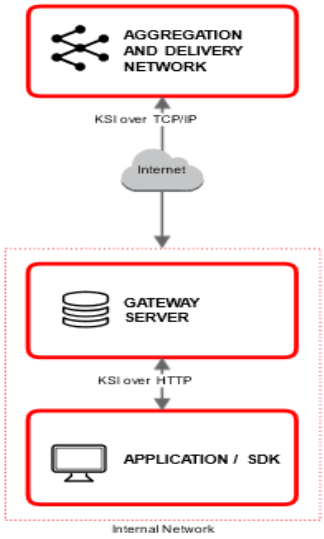

Figure 3 Cloud KSI Communication

3.3.3.5 Communication between internal users HMI and cloud platform IoT

The HMI frontend and backend microservices will be hosted inside the cloud platform. HMI module is a web-based frontend application of the Naiades ecosystem, dedicated to the pilots' users of Carouge, Braila, Alicante and Lab testbed. Pilots users will view their pilot measurements, notifications and statistics in custom windows depending on their usecase. The data that users will see in the interface will come from internal cloud repository and processing services, as a result of processed measurements sent from their Data Collection Aggregation pilot from outside the cloud.

NGINX proxy will be configured to expose outside the platform HMI based on the URL. If the user types the main URL of the platform e. g. "NAIADES-application-eu:" or "e.g. NAIADES-application-eu/login", NGINX will redirect the request inside to the internal application that loads the HMI frontend.

The requests from frontend HMI UI to IoT platform will be authorized by Access Control authorization layer. The credentials for internal users will exist in Identity Management module and will be provided to each pilot. HMI modules cannot directly access the DataBase of Identity Manager, it will be done using Identity Manager API, for login/logout methods. Details will be specified in D7.6.

As a first step, at login time, row credentials will be sent from HMI frontend to the Identity Management, for an authorization token, using REST HTTP requests. The token will be saved locally in browser and will authorize each request that HMI will be sent to the Naiades services. Authorization layer will authorize or reject the requests, based on user roles saved in Identity Manager. If users send requests with bad token the Naiades application will response with 401 Unauthorized HTTP Error.

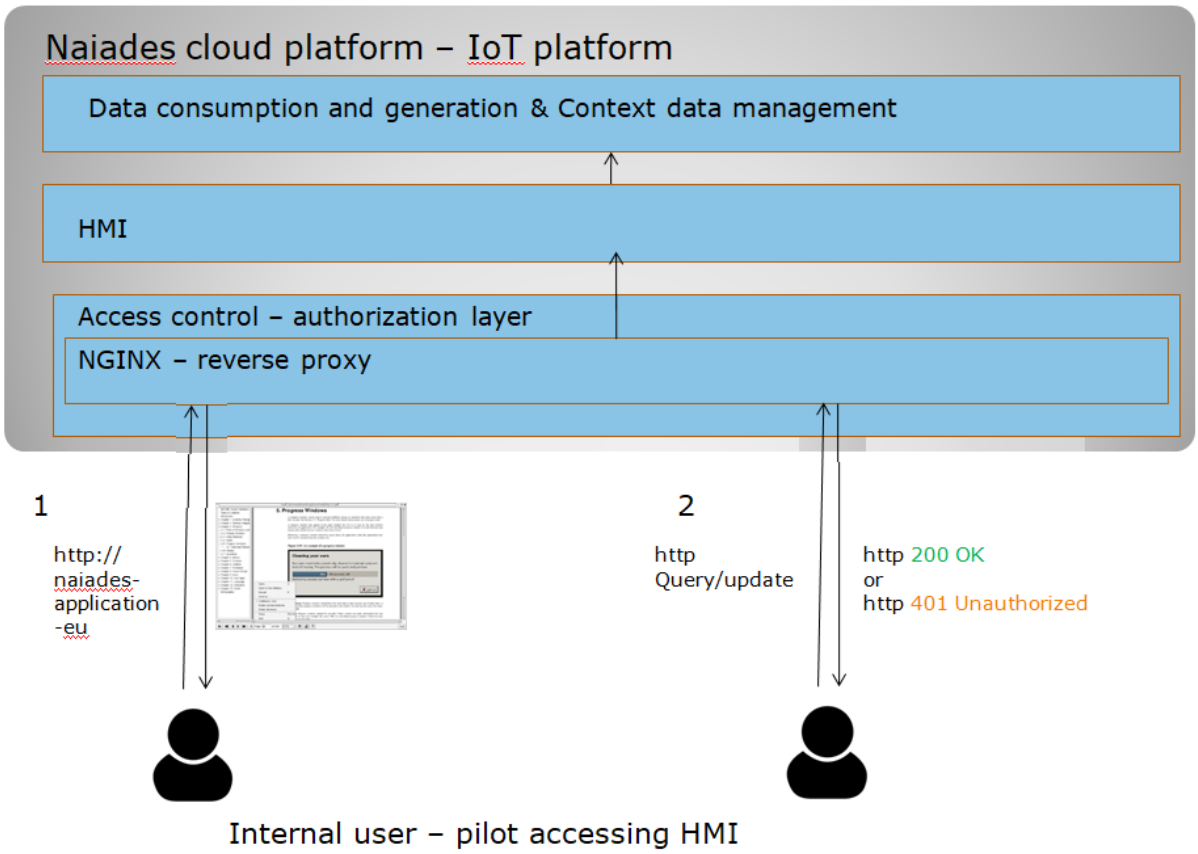

Figure 4 Cloud HMI Communication

3.3.3.6 Communication between external users and cloud platform - Marketplace

The Marketplace frontend and backend microservices will be hosted inside the cloud platform. NGINX proxy will be configured to expose outside the platform HMI based on the URL. If the user types the main URL of the platform e. g. "NAIADES-application-eu/marketplace", NGINX will redirect the request inside to the internal application that load the Marketplace UI.

The requests from frontend Marketplace UI to Marketplace backend inside platform will be authorized by Access Control authorization layer. The request from the external users or external applications that wants to integrate NAIADES application, based on marketplace API described in the interface, will be done using a communication between external-users/applications – marketplace backend using HTTP REST web service protocol. All requests will be authorized by Access Control authorization layer and will be handled using the NGINX proxy. All requests from Marketplace backend to IoT platform process layer will be done internal and they are also authorized.

Marketplace frontend allows external users to create their own account. External users accounts will be stored in Identity Management module. The communication between Marketplace frontend and Identity Management will be done the same as the communication between HMI and Identity Management, using HTTP protocol REST requests. Marketplace Frontend modules cannot access directly the Identity Manager DataBase; it will be done using Identity Manager API, for login, logout and edit user.

External users can access dedicated NAIADES services via Marketplace backend, they do not have direct access to the main processing services, even the user is authorized. Custom web services will be exposed to the Marketplace API in order to let others external application to use it.

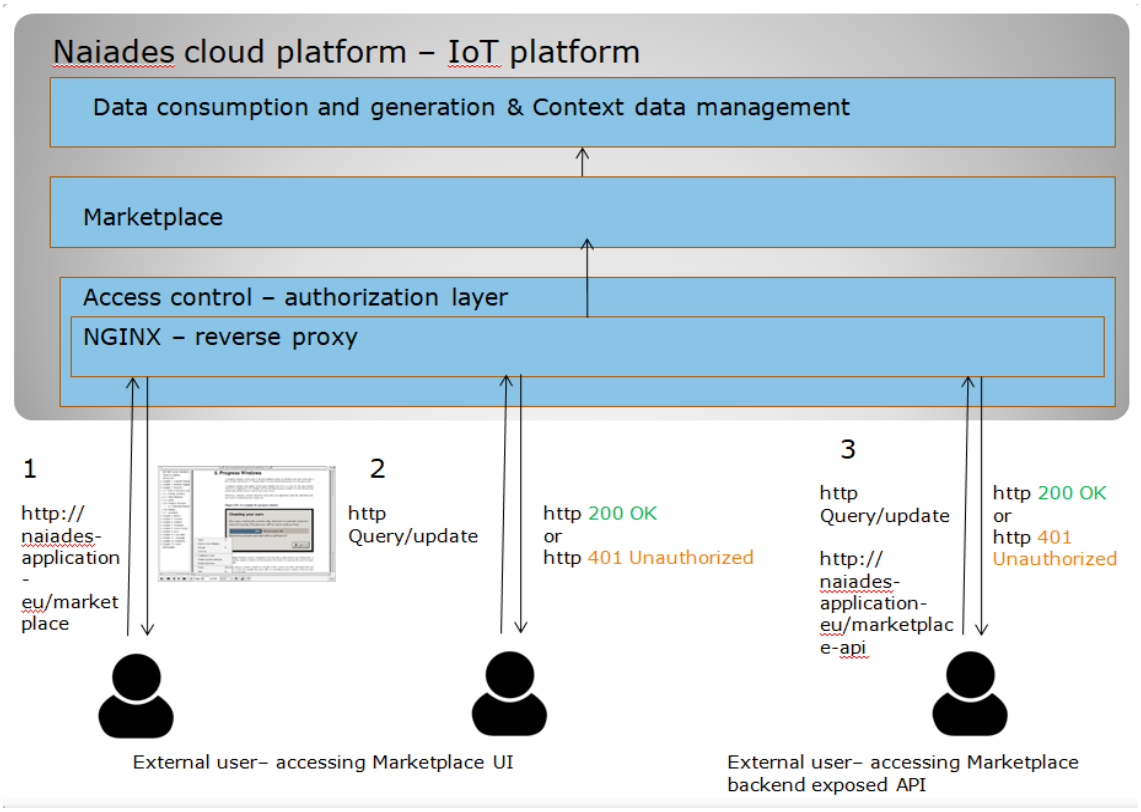

Figure 5 Cloud External users Communication

3.4 Installation details and/or GUI presentation

The installation of the new SDKs on the cloud platform will be done using the admin account for Amazon platform with root permission on the EC2 Amazon instance.

The installation of the new release SDK will require to stop the microservice docker, and redeploy the SDK docker just for the needed microservice. Not all services need to be stopped.

All NAIADES Cloud SDKs are containerized with Docker container, but in the situation where it is needed to install or configure some tool that is not containerized, Ubuntu command line scripts will be used.

3.4.1 For IoT platform

For a detailed procedure of the IoT platform deployment please refer to D3.9 and to the wiki document:

https://gitlab.distantaccess.com/NAIADES/NAIADES-platform-poc/-/wikis/NAIADES-IoT-Platform-installation:

According to the IoT wiki link above, they are presenting the steps of cloning the Iot SDK and run it using Docker container configuration. The wiki provides the clone location of the SDK. Once the binaries are cloned, in side project parent folder, can be execute Linux based commands to build and run the image. Go to the parent folder IoT SDK folder /NAIADES-platform-poc using command:

```
cd NAIADES-platform-poc
```

Build the SDK and start it up using the command:

```
docker-compose up –d
```

For running the services, it will be used docker-compose. The services parameters can be checked in the docker-compose.yml and *.env* configuration files.

After running the command "docker-compose up –d", docker images should be created. In order to have all the services created, it is necessary that all of them are marked with "done" status as it can be seen in Figure 6 bellow.

```
WARNING: The IDM_HTTPS_ENABLED variable is not set. Defaulting to a blank string.
Creating network "NAIADES-platform-poc_default" with driver "bridge"
Creating db-mongo        ... done
Creating db-crate        ... done
Creating wms-app-example ... done
Creating db-mysql        ... done
Creating fiware-orion    ... done
Creating mongo-express   ... done
Creating fiware-keyrock  ... done
Creating fiware-quantumleap ... done
Creating fiware-orion-proxy ... done
```

Figure 6 Cloud IoT build and start Docker info

If services are up, the status of the services can be checked with docker-compose command:

```
docker-compose ps
```

A list with active docker images must appear

```
Name                  Command                State                Ports
-------------------------------------------------------------------------------------------------------------------------------------------------------
db-crate              /docker-entrypoint.sh crat ...   Up                   0.0.0.0:4300->4200/tcp,0.0.0.0:4200->4200/tcp, 4300/tcp, 5432/tcp
db-mongo              docker-entrypoint.sh --bin ...   Up                   0.0.0.0:27017->27017/tcp
db-mysql              docker-entrypoint.sh mysqld      Up                   3306/tcp, 33060/tcp, 0.0.0.0:3307->3307/tcp
fiware-keyrock        /opt/fiware-idm/docker-ent ...   Up (healthy)         3000/tcp, 0.0.0.0:3005->3005/tcp
fiware-orion          orionld -fg -multiservice  ...   Up (health: starting) 0.0.0.0:1026->1026/tcp
fiware-orion-proxy    /opt/fiware-pep-proxy/dock ...   Up (healthy)         0.0.0.0:1027->1027/tcp
fiware-quantumleap    /bin/sh -c python app.py         Up (health: starting) 0.0.0.0:8668->8668/tcp
mongo-express         tini -- /docker-entrypoint ...   Up                   0.0.0.0:8081->8081/tcp
wms-app-example       /bin/sh -c python app.py         Up (healthy)         0.0.0.0:5000->5000/tcp
```

Figure 7 Cloud IoT Docker active status

All services are now up and running. *Docker-compose ps* is a command that allows to check the state and configuration of every running service. It can be done further digging in into the services by checking logs of each of the services with docker logs, by using the command:

```
docker logs wms-app-example
```

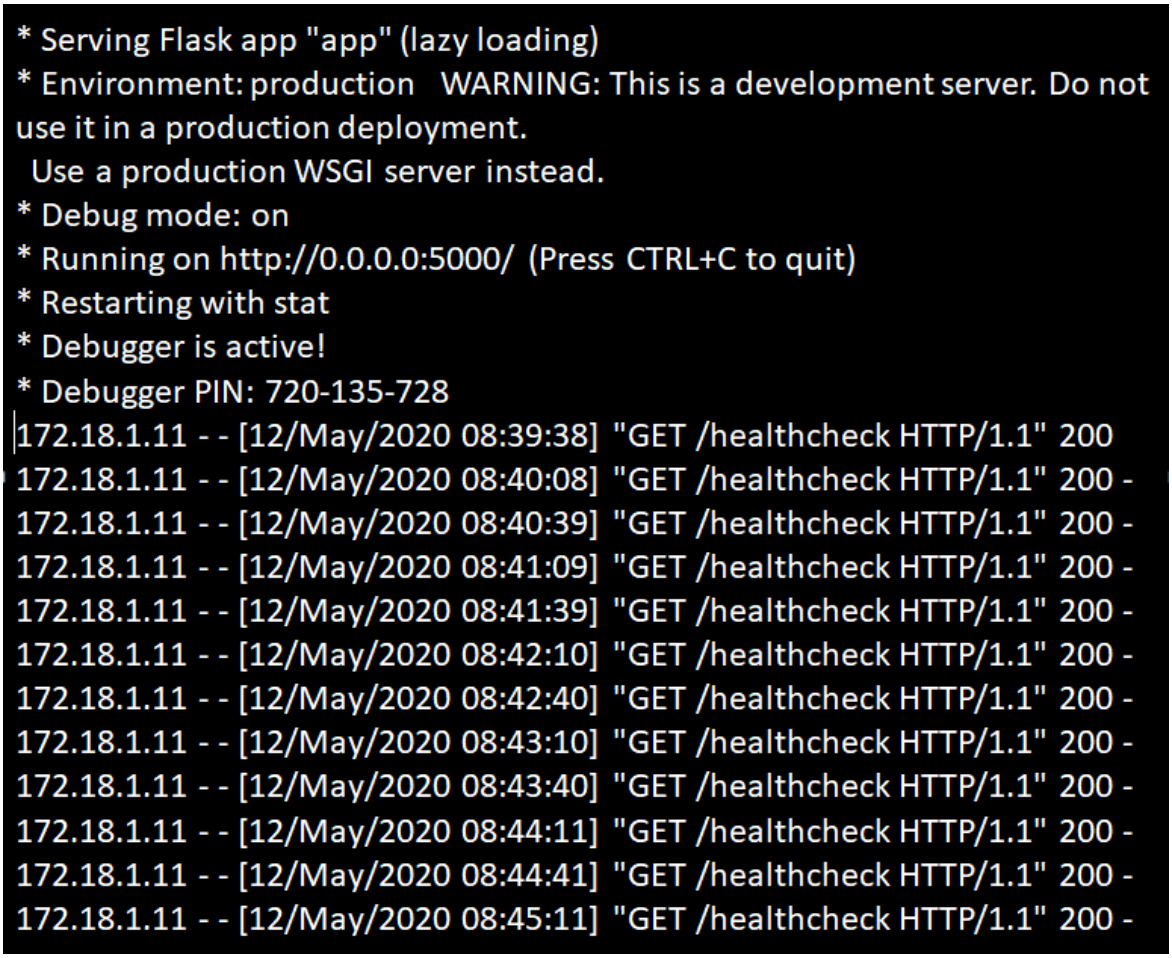

Figure 8 Cloud IoT Docker logs

The default amount of memory allocated by the system to the docker image named "create-db" needed for the creation of one of the data bases is 65530 KB. When the user tries to start the process of the service named "create-db", the system could fail do to an insufficient memory space allocated to this service and informs the user that the allocated space is too low.

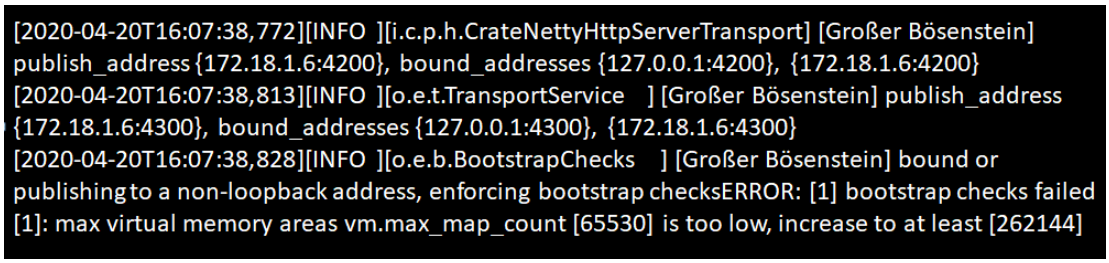

Figure 9 Cloud IoT Docker VM memory size error

To fix the problem, one must adjust the Augment VM memory size with at least 262144 KB using the command:

```
sudo sysctl -w vm.max_map_count=262144
```

3.4.2 For Marketplace

In first version of the NAIADES Cloud Platform SDK's Marketplace will not be available.

Marketplace will be added in the next development stage as a module, containing:

- Marketplace UI
- Marketplace backend

3.4.2.1. Marketplace Frontend UI

Marketplace UI will be a separated application service that will communicate with Marketplace backend service.

From the parent folder of the Marketplace SDK folder /marketplace/UI must be run the docker compose command for build and start the application:

```
docker-compose up –build
```

3.4.2.2. Marketplace backend

From the parent folder of the Marketplace SDK folder /marketplace/backend must be run the docker compose command for build and start the application

```
docker-compose up –build
```

The list of the deploy and build commands for every SDK will be described and shared regarding to the progress of the development.

3.4.3 Awareness and Behavioural Change Support

The services are being implemented and the first version will be available on M18, at this point we do not have the installation details. The SDK should be delivered as a docker image like most of the other services.

The related information will be available in the next version of the deliverable (D3.13 NAIADES Communication Platform – WMS - Final).

3.5 Security mechanism

This chapter provides the guidelines, practices and processes that NAIADES is using for Identity and Access Management to the communication platform, based, internally, on internal roles and externally on KSI signatures.

Privileges

Security best practices shall be followed when granting access by granting least privilege:

- For security-critical resources the simple roles should be avoided, such as (roles/owner, roles/editor, and roles/viewer). Instead, grant predefined roles to allow the least-permissive access necessary.
- Each component of the application is treated as a separate trust boundary. If multiple services require different permissions, a separate service account for each of the services should be created, then the required permissions should be granted to each service account.
- Policy set on a child resource cannot restrict access granted on its parent.
- Grant roles at the smallest scope needed. For example, if a user only needs access to publish Pub/Sub topic, grant the Publisher role to the user for that topic.

- Restrict who can act as service accounts. Users who are granted the Service Account Actor role for a service account can access all the resources for which the service account has access. Therefore, must be cautious when granting the Service Account Actor role to a user.
- Restrict who has access to create and manage service account.
- Granting the Project IAM Admin and Folder IAM Admin predefined roles will allow access to modify Cloud IAM policies without also allowing direct read, write, and administrative access to all resources.
- Grant the Owner (roles/owner) role only when (nearly) universal access is required. Granting the Owner (roles/owner) role to a member will allow them to access and modify almost all resources, including modifying Cloud IAM policies. This amount of privilege is potentially risky.

Service accounts and service account keys

Service account keys, not to be confused with encryption keys, rotation must be automated using the Cloud IAM service account API. Service account keys must not be written into the source code nor left in the Download directory. Specific process must be implemented to manage the user-managed service account keys.

Service accounts shall use its display name to keep track of what they are used for and what permissions they should have. Service accounts that are in used by running instances shall not be deleted before transition to an alternative service account has been made.

Auditing

Cloud Audit logs are regularly used to audit the access to the service account keys and changes in Cloud IAM policies. Additionally, it must be audited that who has the ability to change the Cloud IAM policies. Audit logs shall be exported to Cloud Storage to store them for long periods of time. KSI stamping technology can help to sign the logs and verify its integrity and mutability.

Access to logs is restricted by using the Logging roles. Same access policies must be applied to the Cloud resource that are used to export logs as those that are applied to the logs viewer.

Policy management

In NAIADES the organization-level Cloud IAM polices are set and used to grant access to all parts of the NAIADES projects. Additionally, roles shall be granted to groups instead of individual users when possible as it is easier to add members to and remove members from a group instead of updating a Cloud IAM policy to add or remove users. If there is a need to grant multiple roles to allow a particular task then a specific group is created for that, required roles are granted to that group and finally users are added to that group. This in return should enable the external requests and queries to database be traceable back to the individual.

Security mechanism offers the support for timestamping and hashing accesses and use of the systems by the end users. This would support increased security and avoid unacceptable data queries or data breaches while also supporting the project partners with the functionality of traceable and unified logging

3.6 NAIADES internal ports

The cloud platform will run each microservice on the same host but with different ports, in order to ensure an internal and private communication between the microservices. The internal microservices ports used cannot be accessed outside the platform. Example ports are presented in Figure 10 Cloud internal and private ports.

KEYROCK_PORT	3005 – internal use
KEYROCK_HTTPS_PORT	3443 – internal use
ORION_PROXY_PORT	1027 – internal use
ORION_PORT	1026 – internal use
NAIADES_HMI	8080 – internal use
MARKETPLACE_UI	4200 – internal use
MARKETPLACE_BACKEND	8083 – internal use

Figure 10 Cloud internal and private ports

All the ports will be private. The NGINX proxy mechanism will be configured and will use just two ports: 80/tcp and 443/tcp. The two ports will pe public and accessible from externals applications. All requests that come from applications outside the cloud platform, like Data Collection Aggregation and Marketplace will access the NAIADES cloud platform just using port 80. Based on the URL text, inside the cloud platform, the NGINX proxy will redirect the request internally to the services that run using internal ports. For a secure protocol https, it will be necessary to add security certificates.

4 Deployment and set up

The connection to the cloud platform will be made through a Secure Shell network (SSH) protocol using Amazon Web Service (AWS) credentials. To establish this connection the admin user must use an SSH client together with a command line interfave like PuTTy, using both public and private key.

An example of SSH client is presented in Figure 11 PuTTy, using SSH connection with authorization PuTTy private key.

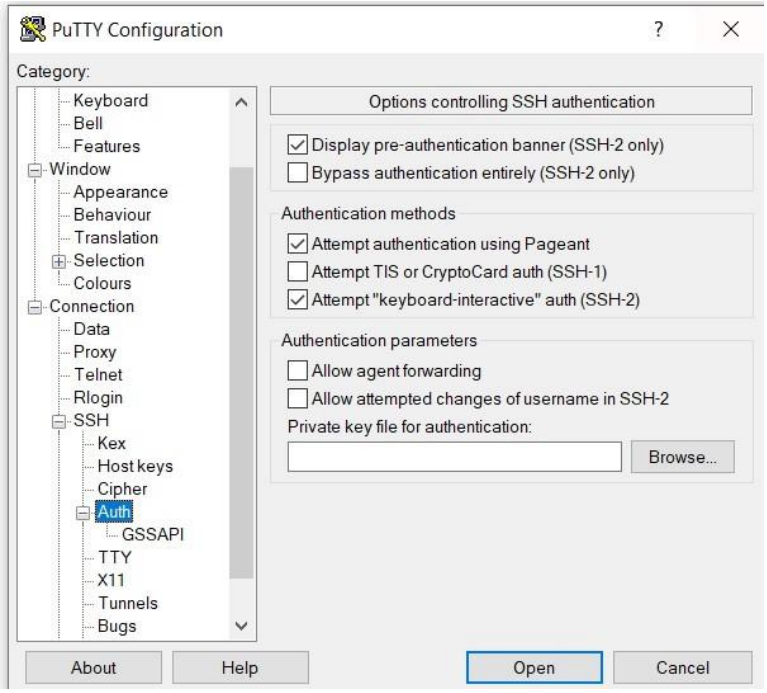

Figure 11 Putty

New SDK binaries and needed resources can be uploaded in the platform using SSH protocol.

Microservices are managed by commands typical for Ubuntu operating system together with the commands necessary for the docker and docker compose cotainerization tool. An installation manual with details about connection and deploy in the cloud platform for all microservices will be available.

5 Conclusions

This report includes the most relevant aspects about the server, the cloud platform, the microservices deployed on the cloud and the installation instructions.

For the first version of NAIADES communication platform, Data Collection Aggregation is prepared for weather use case, to send modelled weather data to the cloud, for each pilot. Other data aggregation services are being developed. Also, as a cloud platform, NAIADES will use an Amazon cloud virtual machine. The first set of services have been deployed and it has been tested the integration and communication of microservices functionalities. The SDKs deployed in the cloud platform are: IoT platform, HMI, KSI.

New features are being developed and related revisions and updates will be included in D.13 "NAIADES Communication Platform – WMS - Final". The first one that should be developed is collecting raw measurements data from pilots' sensorial systems. The following one is improving Data Model for modelling pilots' raw measurements, continuing with integration of Marketplace with NAIADES Data Management and deploying Marketplace and Awareness and Behavioural Change Support inside cloud platform. Last, but not least, is integration and unit tests according to the development progress.