Ref. Ares(2025)4589364 - 08/06/2025

openZDM

OPEN PLATFORM FOR REALIZING ZERO DEFECTS IN CYBER PHYSICAL MANUFACTURING

D3.6 – NDIs SW apps integration to openZDM platform

openZDM

Version	1.0
WP	3
Delivery Date	06 June 2025
Dissemination level	PU
Deliverable lead	INTRA
Authors	INTRA
Reviewers	UNIVPM, LMS
Abstract	This document is a demonstration of the openZDM NDI integration approach. The document presents the basic concepts related to NDIs and the integration methods used in openZDM. In the last part of the document, a step-by-step integration of a sample NDI is presented.
Keywords	NDI, AAS, Software Integration, API, MQTT
License	This work is licensed under a Creative Commons Attribution-No Derivatives 4.0 International License (CC BY-ND 4.0). See: https://creativecommons.org/licenses/by-nd/4.0/

Dissemination Level:	
PU	**Public, fully open**
SEN	Sensitive, limited under the conditions of the Grant Agreement
Classified R-UE/EU-R	EU RESTRICTED under the Commission Decision No2015/444
Classified C-UE/EU-C	EU CONFIDENTIAL under the Commission Decision No2015/444
Classified S-UE/EU-S	EU SECRET under the Commission Decision No2015/444
Type	
R	Document, report (excluding the periodic and final reports)
DEM	**Demonstrator, pilot, prototype, plan designs**
DEC	Websites, patents filing, press & media actions, videos, etc.
DATA	Data sets, microdata, etc.
DMP	Data management plan
ETHICS	Deliverables related to ethics issues.
SECURITY	Deliverables related to security issues
OTHER	Software, technical diagram, algorithms, models, etc.

Funded by the European Union

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or EUROPEAN HEALTH AND DIGITAL EXECUTIVE AGENCY (HADEA). Neither the European Union nor the granting authority can be held responsible for them.

Version History

Version	Date	Owner	Author(s)	Changes to previous version
0.1	02-04-2025	INTRA	INTRA	Outline
0.5	25-04-2025	INTRA	INTRA	Full draft
0.6	20-05-2025	INTRA	LMS, UNIVPM	Commented version
0.7	23-05-2025	INTRA	INTRA	Revised version
1.0	06-06-2025	INTRA	LMS	Final draft

Table of Contents

List of Abbreviations & Acronyms

AAS	:	Asset Administration Shell
API	:	Application Programming Interface
HTTP	:	Hypertext Transfer Protocol
ICT	:	Information Communication Technology
JSON	:	JavaScript Object Notation
MQTT	:	Message Queue Telemetry Transport
NDI	:	Non-Destructive Inspection system
OPC-UA	:	Open Platform Communications Unified Architecture

List of Figures

List of Tables

Executive Summary

This document represents a demonstration of how openZDM NDI systems are integrated with the openZDM platform. This work is based on the following concepts:

1. NDIs: These are complex systems designed to measure characteristic features of a product and perform quality control, i.e. conformity assessment, diagnosis and classification of defects.
2. AAS is used to provide a digital representation of an asset and consists of several sub models in which all the information and functionalities of a given asset are modelled. In section 2.2.1 details are given about the structure of the AAS model.
3. AAS Middleware: software that facilitates the interaction with the AAS model with third-party components. It provides an HTTP API to access and manage the AAS model.
4. openZDM platform: a group of software components working together to support production networks' zero-defect processes. To achieve this goal, various data sources are needed, including NDIs. The platform provides those components (including the AAS middleware) that are required for integrating the NDIs, and for realising the goal of a zero-defect production process. More details about the openZDM platform are presented in the section 2.3.

The NDI integration approach to the openZDM platform is common for all NDIs developed in openZDM and provides two options:

- MQTT-based approach
- HTTP AAS API based approach

Both approaches rely on AAS and assume that an appropriate AAS model has been developed for the NDI. This model is then fed with data based on the MQTT or HTTP approach. For the MQTT approach, the NDI publishes data in the MQTT Broker and the openZDM AAS Middleware using an appropriate mapping between the AAS model and the MQTT topic, and the data is responsible for populating the AAS model with data. On the other hand, using the HTTP AAS API approach, the AAS Middleware API is used by the NDI software to directly populate the AAS model with data. The integration process of NDIs in openZDM is as follows:

1. Designing the NDI's AAS using an appropriate tool such as AASX Package Explorer.
2. Export the design as a JSON file.
3. Upload the JSON file to the openZDM platform. As soon as this step is complete, the AAS of the NDI appears in the list of AAS of the platform and is available to users with the role of 'Administrator' to manage (e.g. configure, change values, etc.) and users with the role 'Operator' to view its data.
4. The last step is the configuration of the integration. In case of HTTP API is just a matter of creating appropriate credentials for the NDI software to access the AAS API. MQTT is more complex in configuration and requires more steps to complete, such as the mapping between the AAS Model and MQTT data. In section 4.3 an example of how the MQTT approach works in detail is presented, including an example where an NDI is sending data to the platform using the MQTT approach.
5. Lastly, the conclusions are summarised as follows:
6. An integration approach for NDIs based on the concept of AAS has been defined for openZDM. The approach applies to all NDIs developed in openZDM. Moreover, it can be used to integrate NDIs outside the project development.
7. Two different approaches have been provided:
8. The HTTP AAS API based approach requires more complexity on the NDI software as the developer must be aware of the AAS model and the openZDM AAS Middleware API. However, it provides more control over the integration of the NDI for the developer.
9. The MQTT-based approach is far simpler for the NDI developer and lets them use their format of choice of the output of the NDI (e.g. JSON) while insulating them from the complexities of the AAS model.
10. The AAS approach provides a significant advantage by insulating the component developers of other parts of the platform from NDI software complexities.

1. Introduction

This document represents a demonstration of how openZDM Non-Destructive Inspection (NDI) systems are integrated with the openZDM platform. It presents an overview of the integration and a step-by-step guide on how to integrate a new NDI into the openZDM platform. The main section of the document covers:

1. Essential concepts, where the reader is introduced to the technologies facilitating the NDI integration and provided details on the openZDM platform. These concepts are NDIs, AAS, and the openZDM platform.
2. The integration approach used in the openZDM platform, where explanations are given on how integration between NDIs and the platform is achieved.
3. A step-by-step demonstration of the integration of sample NDI in the platform, including a link to a video demonstration.
4. Conclusions on the work carried out.

2. Essential concepts

This section introduces the various concepts that are associated with the aspect of NDI integration into the openZDM platform. It provides explanations on:

1. NDIs
2. AASs
3. openZDM platform

2.1 Non-Destructive Inspection (NDI) systems

NDIs are complex systems designed to measure characteristic features of a product and perform quality control, in particular, assessment of conformity to specifications, diagnosis and classification of possible defects. More specifically, the NDIs measure quantitative values of the product and determine if the measurement lies within the design specifications of the product. Quality control ranges from a simple measurement of one characteristic (such as the temperature of product after it has been processed by one machine) and its comparison to thresholds set by the designers to more complex operations where the execution of a test procedure is required, meaning that for each product several measurements may be carried for one or more characteristics under specific test conditions. NDIs must perform a minimally invasive and non-destructive test, meaning that measurement techniques applied must not alter the product's characteristics. In openZDM, the NDIs typically follow, from an ICT point of view, the structure presented in Figure 1.

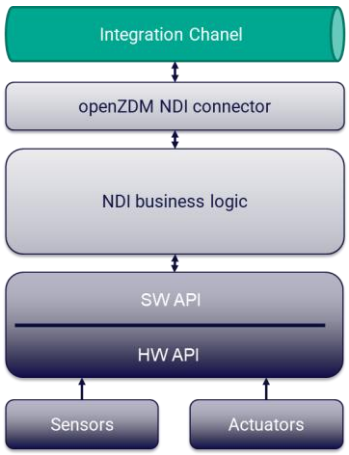

Figure 1: openZDM NDI overview.

The NDIs in openZDM typically consist of one or more sensors that take measurements of the product; for example, NDIs in openZDM use various types of cameras as sensors. The actuators are used to execute the test and allow the measurement process, for example, by placing the product in front of the sensor. Both sensors and actuators typically are packaged with Hardware and Software APIs to facilitate access by third-party software, in this case, the NDI business logic. The NDI business logic processes the measured data to determine if the characteristic features are within the product specification. Lastly, the output of the NDIs is communicated to the platform using the openZDM NDI connector that connects to the platform's infrastructure either directly through software APIs or through an integration channel.

2.2 Asset Administration Shell (AAS)

openZDM uses the concept of AAS to facilitate the integration of assets, such as NDIs, to the platform. AAS is the digital representation of an asset and consists of several sub models in which all the information and functionalities of a given asset, including its features, characteristics, properties, statuses, parameters, measurement data and capabilities, can be described [1].

2.2.1 Modelling an NDI using AAS in openZDM

It becomes evident from the Platform Industrie 4.0 [1] definition of AAS that in order to create an AAS, the first step is to create its model in terms of sub models, and according to the Platform Industrie 4.0 specifications [2]. This process is facilitated using a visual editor such as AASX Package Explorer 2023-02-03.alpha. In openZDM, the NDIs have been modelled as AAS using a series of predefined sub models as illustrated in Figure 2. The following sub models are used in all NDIs:

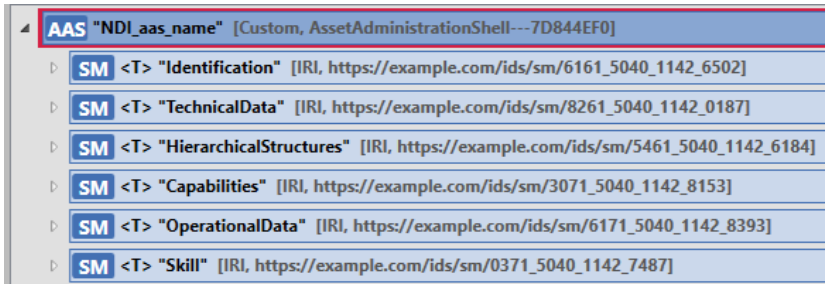

Figure 2: AAS template of openZDM NDIs from the AASX Package Explorer.

- Identification sub model: it aims to collect information that allows the NDI to be distinguished uniquely from other NDIs or even other AAS.
- TechnicalData sub model: It allows describing the assets, which are provided to the market, employing technical data. For example, this sub model includes information about the model of the camera, its manufacturer, its resolution and so on.
- Hierarchical structure sub model: It represents the composition of the NDI. This sub model typically contains all the software and hardware components that make up each NDI. For example, if an NDI consists of 2 cameras and the NDI business logic, all this information is included in this sub model.
- Capabilities sub model: It represents the tasks that the NDI can perform, for example, a temperature sensor would have the capability to measure the temperature of a product.
- Skill sub model: It is the implementation of a Capability, meaning that the skill provides an interface to invoke a given capability. For the temperature sensor, invoking the skill that realises the temperature measurement capability would cause the sensor to take a measurement.
- OperationalData sub model: It is a representation of the properties measured by the NDI. In this sub model the value of the temperature measured by a temperature sensor is recorded.

2.2.2 AAS Middleware

The openZDM AAS Middleware is a piece of software that facilitates interaction with the AAS model (i.e. accessing its structure and data) through a standardised API. A part of the AAS Middleware API is illustrated in Figure 3 and Figure 4.

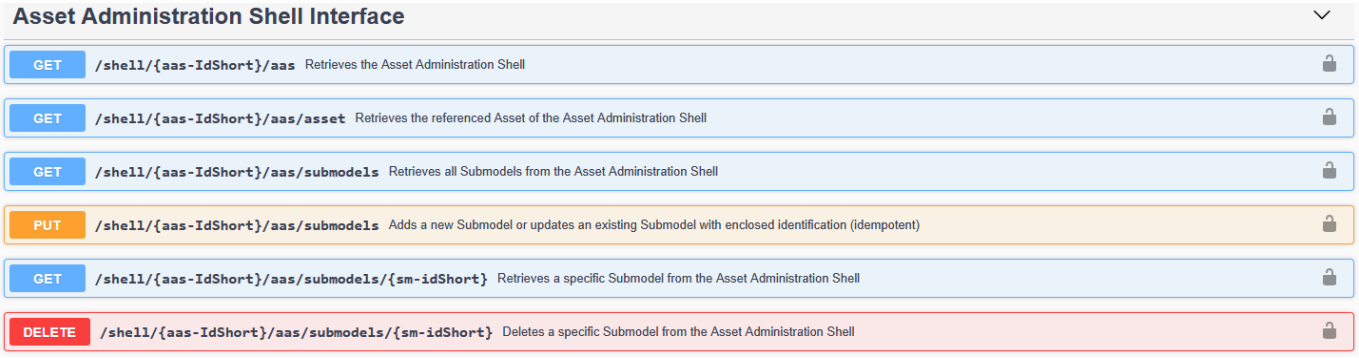

Figure 3: AAS Middleware API - Asset Administration Shell Interface.

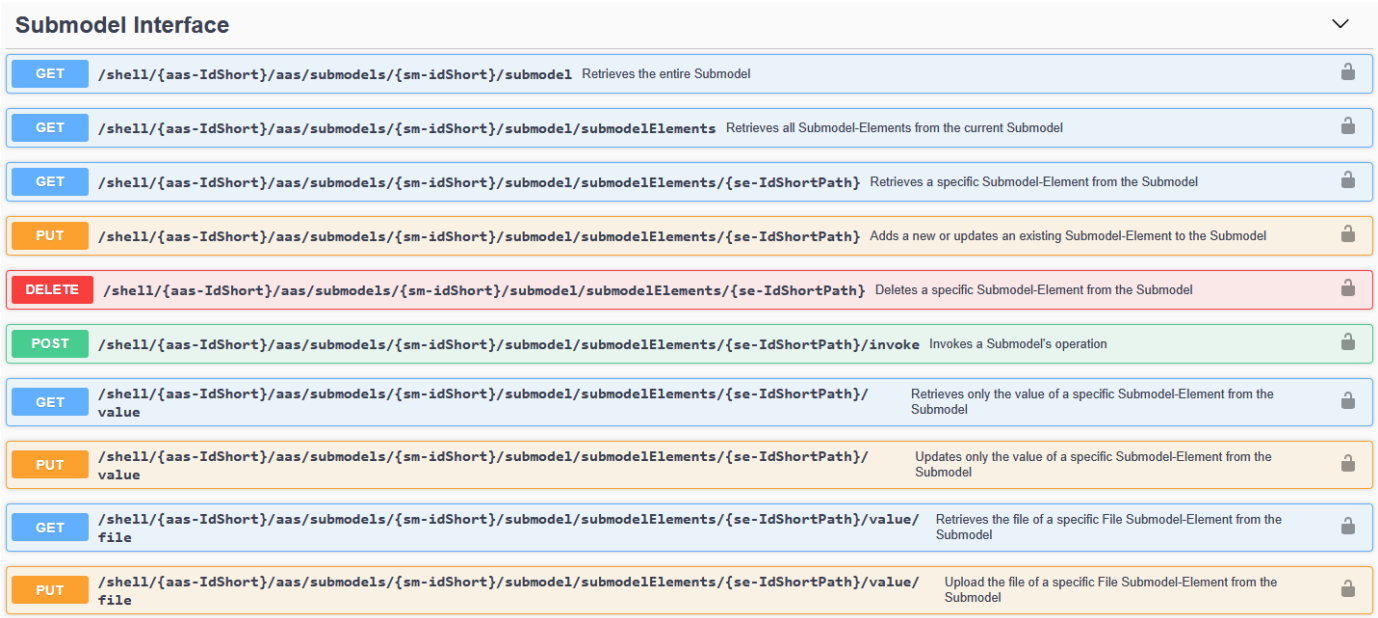

Figure 4: AAS Middleware API – Submodel Interface.

This API provides a uniform way for developers to access the data of the AAS, including values of sub model elements, sub model names and IDs and so on. In addition, the openZDM AAS Middleware provides the appropriate integration mechanisms so that the data from a physical asset (i.e. an NDI) can be copied into the AAS of the NDI. The integration mechanisms may vary in terms of technology used and may include communication via HTTP API calls, MQTT, OPC-UA, etc.

2.3 openZDM platform

The openZDM platform is a group of software components working together to support production networks' zero-defect processes. The platform accesses data from a variety of data sources, including NDIs, equipment deployed at a manufacturing company's shop floor, and other legacy systems (such as databases). The data are stored and processed in the platform by a variety of tools having as the following key objectives:

1. Monitoring of manufacturing processes to identify deviations that cause defects, through the application of Digital Twins.
2. Data analysis for quality assessment and prediction of defects.
3. Decision support through the evaluation of alternative zero-defect manufacturing strategies and the suggestion of adaptation strategies.

The openZDM architecture that facilitates the goals of the project is illustrated in Figure 5. The definition of this architecture is based on the Reference Architectural Model Industrie 4.0 (RAMI 4.0) [3], and the work carried out in context of Sense&React [4], and Boost 4.0 [5] architectures.

Figure 5: The openZDM platform architecture [6].

The architecture of the openZDM platform is comprised of three tiers. Each tier can be either deployed on the cloud, locally deployed or hybrid. Each layer within a tier of the architecture is responsible for grouping together specific components. The platform layers include the following horizontal layers:

- **External data source and infrastructural layer:** This layer represents all the assets of the end-users that need to be connected to the openZDM platform, such as NDIs, existing equipment, and legacy systems.
- **Integration layer:** The integration layer facilitates the communication between the platform and all the assets. In this layer, the openZDM AAS Middleware is applied to represent assets as AASs and manage the integration of NDIs.
- **Data layer:** The data layer's role is to intercept messages from all the assets.
- **Service layer:** In the service layer, components that implement some of the platform's functionalities are included, and the layer facilitates access to the data layer from applications in the Business layer (Service API).
- **Business layer:** This layer groups together the functionality that is offered to the users and supports their business objectives. At a minimum, this layer can be conceived as a User Interface that provides access to the data of the platform. In more complex cases, the User Interface is the entry point for openZDM applications offering production monitoring, data analysis, and decision support.

The architecture also includes the three vertical layers, which provide functionalities to all the horizontal layers:

- **Authentication & authorisation:** Provides functionalities for user and services identification, mechanisms to secure communications and authorisation mechanisms that permit access to resources of the platform. This layer is realised by the Identity Server of the platform that relies on the OAuth2.0 protocol [7] for authorisation.
- **Logging & Monitoring:** This layer logs user activity and records system errors.
- **Data sharing:** The data sharing layer enables the integration of the platform into a data sharing ecosystem.

In the rest of the document, the various services in the openZDM platform will be access over the user interface to provision a new NDI through an 'Administrator' user and then view its data using an 'Operator' user that has mainly viewing privileges in the platform and access to specific applications designed for viewing data. In this example, the

'Operator' user will have a Grafana Dashboard to view the historical data of the NDI. It is also assumed that the NDI has already published data to a local MQTT Broker.

3. Integration approach

The NDI integration approach to the openZDM platform is common for all NDIs developed in openZDM and provides two options (see Figure 6):

- MQTT-based approach
- HTTP AAS API based approach

Both approaches rely on AAS and assume that an appropriate AAS model has been developed for the NDI. This model is then fed with data based on the MQTT or HTTP approach. This approach covers all NDIs developed in the project and deployed in the pilot lines. It may also be used to add new NDIs to the platform. The proposed approach integrates the NDIs on the software level, thus making the integration approach hardware agnostic. The integration steps are presented in section 4, and in the demonstration video at the end of the deliverable.

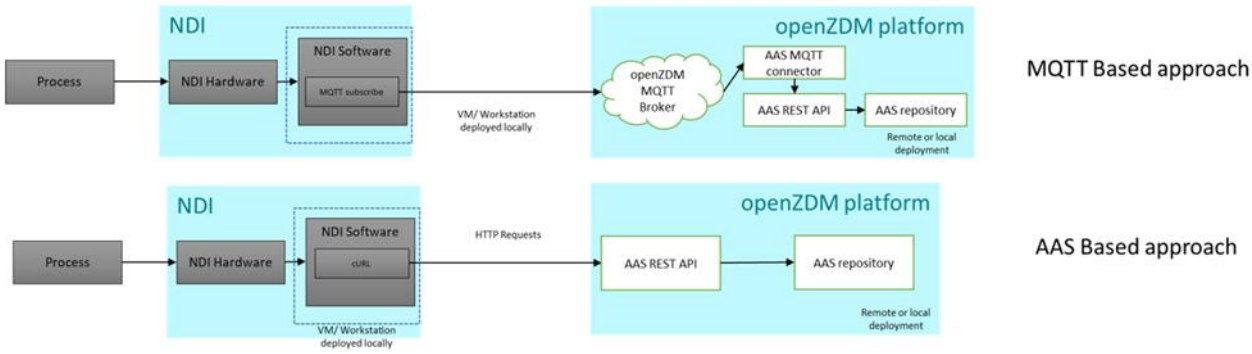

Figure 6: NDI integration approaches to the openZDM platform.

While a variety of protocols are available for communicating NDI data, the considerations behind the selection of MQTT and HTTP are presented in Table 1.

Table 1: NDI integration technology selection

Technology	Pros	Cons
MQTT [8], [9]	• Lightweight and efficient • Low power consumption • Reliable over unstable networks • Wide support • NDI developer controls the data format. • Integration is simpler on the NDI developer side • No direct connection needed between NDIs and openZDM applications. • Immediate delivery of data to subscribers as soon as it's published.	• Not ideal for large binary data. • Integration relies heavily on the platform and would require support from a company acting as the integrator or owner of the platform to configure the integration.
HTTP [10], [11]	• Wide support • Works well with firewalls and proxies • Readable and compatible with web apps. • Handles large binary data better. • Provides more control over the integration and better understanding of whether the data has reached the platform. • The integration is handled predominantly on the NDI side. Some information is required on the platform side.	• High overhead. • Not suitable for real-time or battery-sensitive devices. • Some complexity in the implementation due to the AAS API.

In the MQTT-based approach, the NDI software publishes data in predefined topics in an MQTT broker. The topics are configured in the AAS MQTT connector, which is part of the openZDM AAS Middleware. The AAS MQTT connector is responsible for monitoring all topics and copying any values into the appropriate AAS sub model elements. The AAS MQTT connector uses a mapping between the topics and the AAS model to copy the values.

In the HTTP AAS API-based scenario, the NDI software directly communicates with the AAS API to populate the AAS model with data. Figure 7 presents the relevant API methods that enable this communication. These methods enable the creation of new values (PUT) or accessing the current value (GET). Figure 7 is actually a subset of the methods presented in Figure 4.

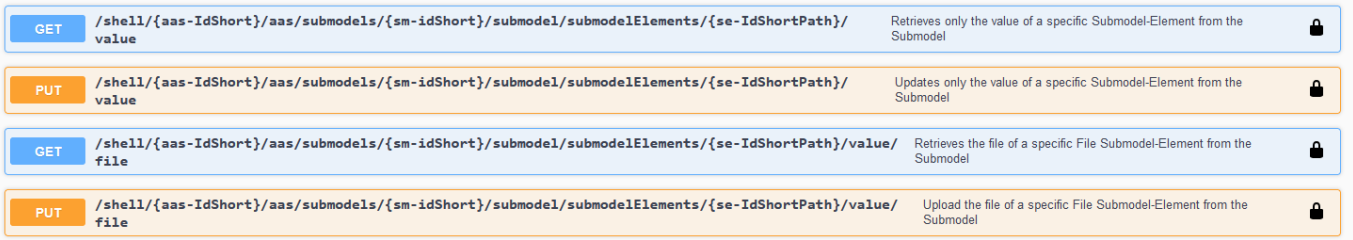

Figure 7: Extract of the AAS API online specification for accessing data.

The input of each method is the AAS ID, the sub model ID, and the sub model element path (dot-separated if required). Both approaches are used in the openZDM project to integrate NDIs. Based on the approach selected, different security mechanisms are applied to secure data:

1. MQTT: Authentication in combination with TLS/SSL Encryption is used. Each pilot has its own MQTT Broker. It is feasible to handle separate MQTT Brokers for each NDI to further isolate their data.
2. HTTP: Authentication and authorisation are applied on the AAS API using the OAuth 2.0 protocol.

An overview of the NDIs integrated in the context of the openZDM project is presented in Table 2. As most NDIs output is relatively small in terms of data size produced, all of them rely on the MQTT protocol. The exception, NDI#6, which exports several images, it uses HTTP. Lastly, NDI#8 is portable using a battery for its power and uses MQTT due to its low power consumption.

Table 2: NDI integrated in openZDM

NDI Number	NDI Type	Description	Integration Approach
NDI#0	X-ray inspection instrument	X-ray inspection technique used in the VDLWEW use case for residual stress detection	-
NDI#1	Laser line triangulation instrument	Laser line triangulation instrument used in the VDLWEW use case for straightness measurements at low temperature	MQTT
NDI#2	Vision based instrument	Vision based instrument used in the VDLWEW use case for measuring the product's temperature profile after the induction furnace process	MQTT
NDI#3	Vision based instrument	Vision based instrument used in the VDLWEW use case for measuring the product's temperature profile after the descaling unit process	MQTT
NDI#4	Laser line triangulation instrument	Laser line triangulation instrument used in the VDLWEW use case for straightness measurements at high temperature	MQTT
NDI#5	Laser line triangulation instrument	Laser line triangulation instrument that is used in the VDLWEW use case for 3D dimension measurements of the product	MQTT
NDI#6	Vision based instrument	Vision based instrument used in the VDLWEW use case for surface defects detection	HTTP

NDI#7	X-ray inspection technique	X-ray inspection technique used in the VDLWEW use case for residual stress detection	-
NDI#8	Laser line triangulation instrument	Portable IoT sensor used in the VWAE use case for gap & flush measurements	MQTT
NDI#9a	Vision based instrument	Vision based instrument used in the VIDRALA use case for glass bottle thickness measurement	MQTT
NDI#9b	Vision based instrument	Vision based instrument used in the VIDRALA use case for glass gob quality assessment	MQTT
NDI#10	Vision based instrument	Vision based instrument used in the APTIV use case for the detection of welding process defects	MQTT
NDI#11	Vision based instrument	Vision based instrument used in the APTIV use case for welding process monitoring and aesthetical defects detection	MQTT

4. NDI provisioning in the openZDM platform

Connecting new NDIs to the openZDM platform follows a simple succession of steps, as illustrated in Figure 8. It should be noted that the provisioning process is only available to users with administration privileges to the openZDM platform.

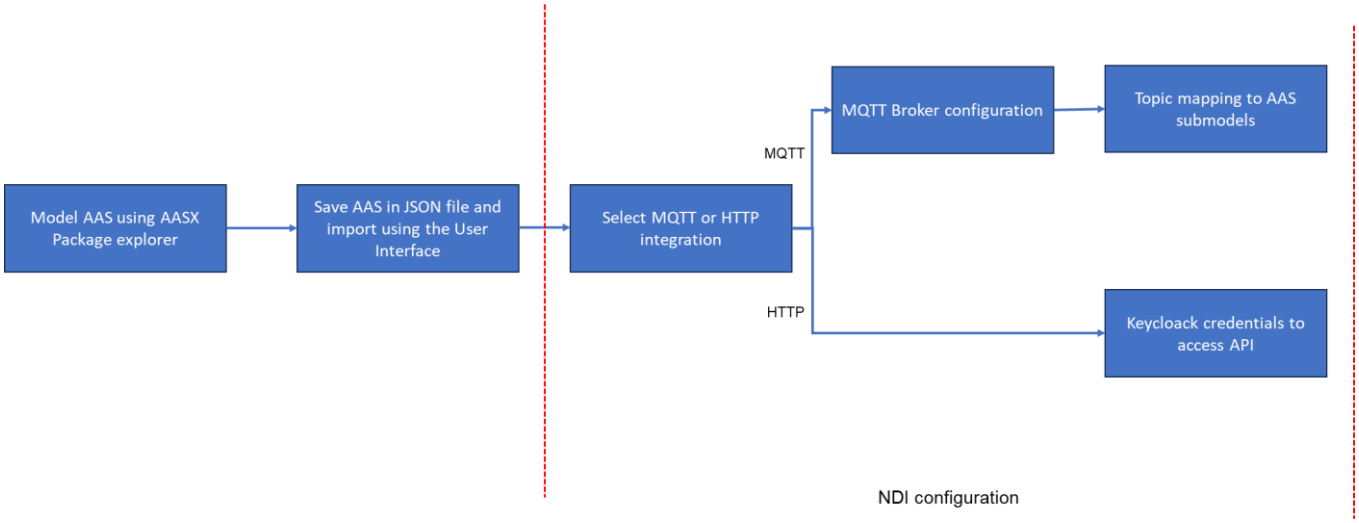

Figure 8: NDI provisioning steps in openZDM platform.

The first step for adding an NDI into the openZDM platform is to create its AAS model according to the structure in the section 2.2.1, using AASX Package Explorer. The next step is to save the model as a JSON file. After the model has been completed, the administrator can directly upload it to the platform using the User Interface. As soon as this step is complete, the AAS becomes available in the platform. Next, the configuration of the AAS begins. The first step is to select the integration type between MQTT and HTTP. If HTTP is selected, a set of credentials for accessing the API of the AAS Middleware is generated. These credentials can then be added in the NDI software to complete the integration. For the MQTT integration, first, the MQTT Broker credentials used by the NDI are configured. Apart from a valid configuration, the Broker needs to be accessible from the platform infrastructure. After the Broker details are configured, the process of mapping MQTT topics to AAS sub models elements begins. For each sub model that should receive data from the physical asset, one configuration is needed. When all sub model elements are covered, the configuration is completed.

4.1 Overview of a sample NDI as AAS

For the NDI, we consider a simple Pyrometer that reads a temperature every five seconds. The AAS of this NDI is defined in Figure 9.

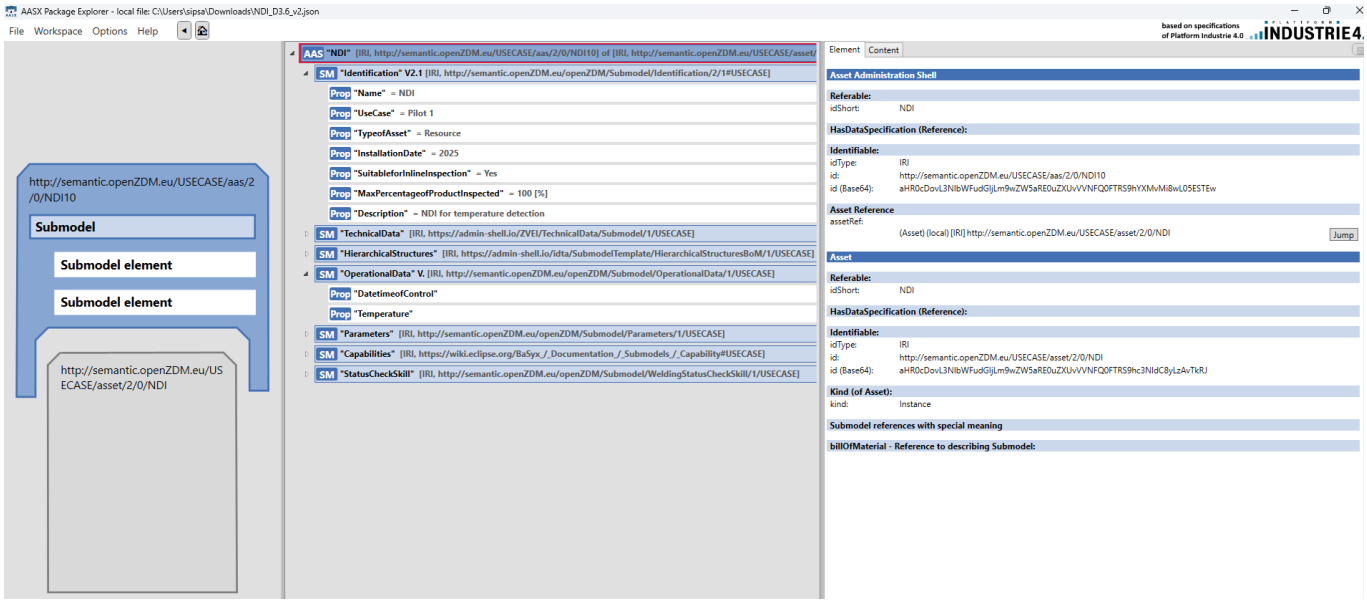

Figure 9: Definition of a sample NDI using AASX Package Explorer.

As illustrated in Figure 9 the NDI follows the structure presented in section 2.2.1. The Identification sub model contains information about the NDI AAS, such as its name, description, the pilot it belongs to, and so on. The most important sub model, however, is the OperationalData subodel, where two properties have been added:

1. Temperature
2. DateTimeOfControl

These properties represent the data that this NDI produces. For this demonstration, the assumption has been made that the NDI publishes data to a local Broker using the JSON format in Table 3. The JSON field named 'value' corresponds to the Temperature AAS property, and the JSON field named 'timestamp' corresponds to the DateTimeOfControl AAS property. In the next steps, the NDI will be connected to the openZDM platform by uploading the AAS and configuring its integration.

Table 3: NDI MQTT data format.

```
{
  "value": 86.74,
  "timestamp": "2025-04-16T11:53:06.211720Z"
}
```

4.2 Import the NDI into the openZDM platform

For importing the NDI AAS, the JSON file exported by AASX Package Explorer is directly uploaded into the openZDM platform using an 'Administrator' user and the user interface. In Figure 10 the 'Asset Administration Shells' view is presented with the appropriate button for uploading AAS files in the top right corner (inside the red box). Also note that in this demonstrator, the AAS repository of the platform is empty; thus, when searching for AAS, no results are presented.

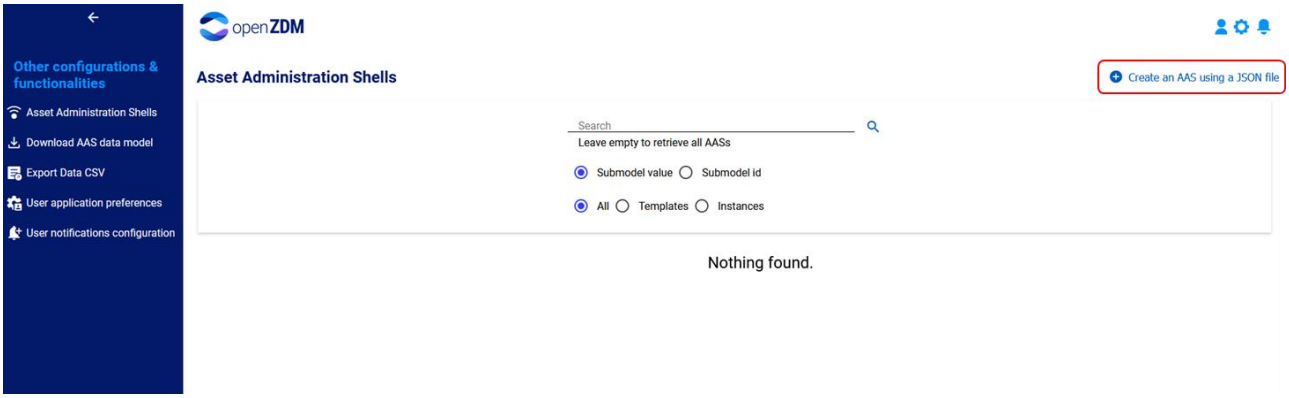

Figure 10: Adding an NDI AAS through the openZDM platform interface.

The next step is to select the appropriate file using the upload dialog show in Figure 11.

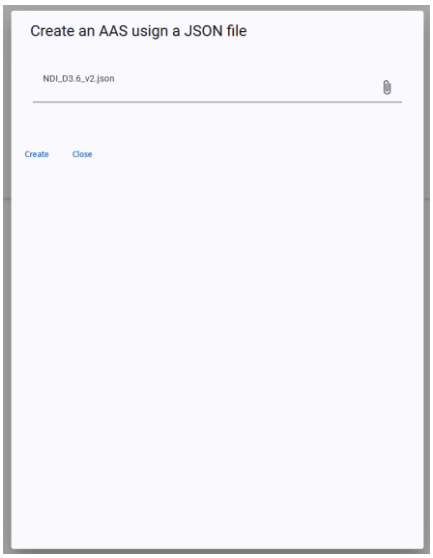

Figure 11: AAS file upload dialogue.

Upon completion, the NDI AAS has been registered in the openZDM platform, and the AAS view is updated with the new AAS ID, as shown in Figure 12.

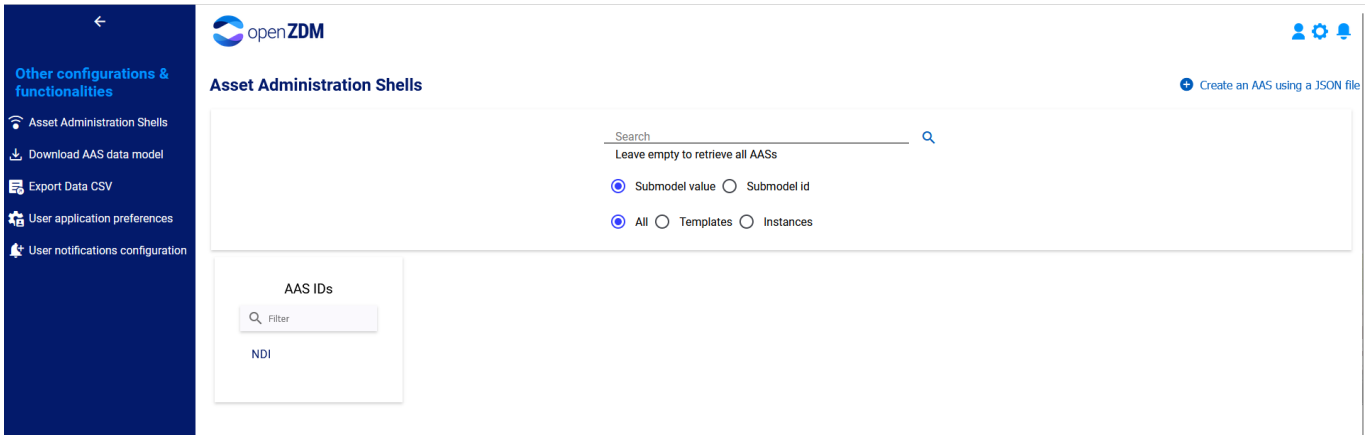

Figure 12: The NDI has been registered in the openZDM platform.

After this step, the user can configure the integration of the AAS, which is achieved by selecting the appropriate AAS ID from the side menu in Figure 12. This will reveal the AAS submodels and the relevant button for managing the NDI integration (see Figure 13).

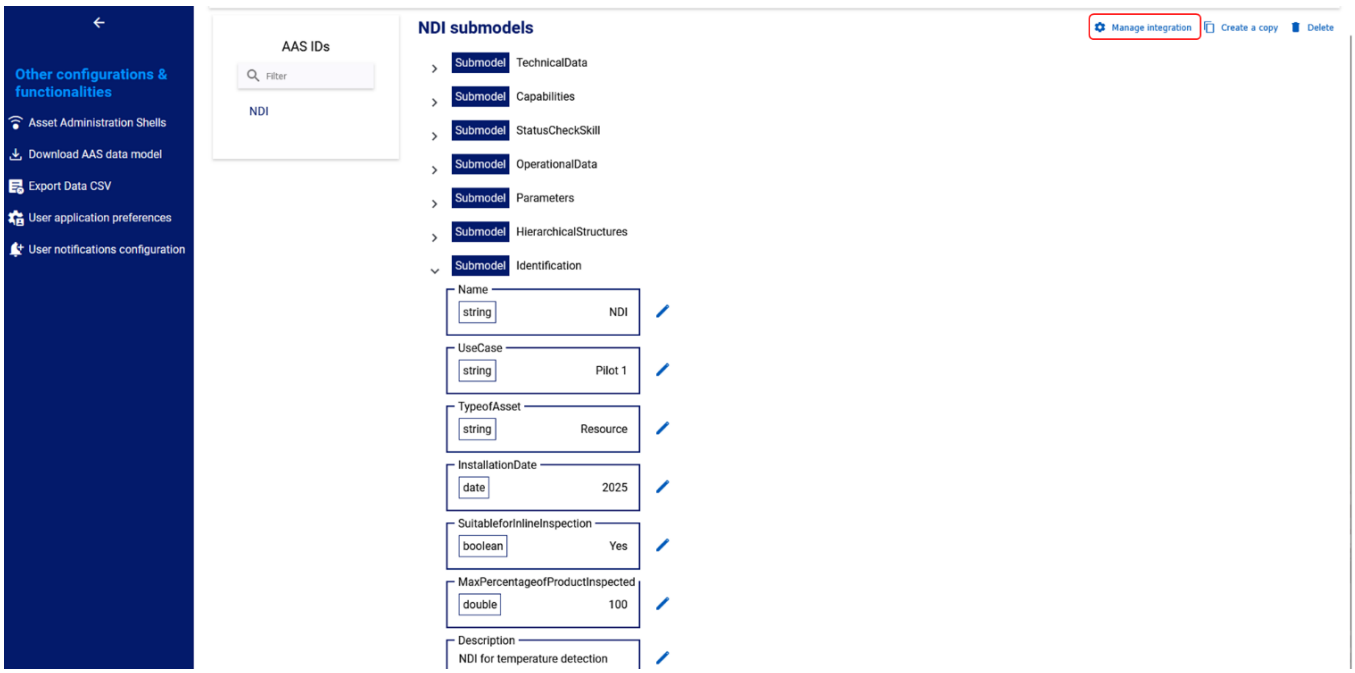

Figure 13: Details of the AAS registered in the openZDM platform along with the 'Manage Integration' button.

4.3 Configuring and testing the integration

The configuration view (see Figure 14) starts with the simple step of selecting the integration approach (MQTT or HTTP AAS API).

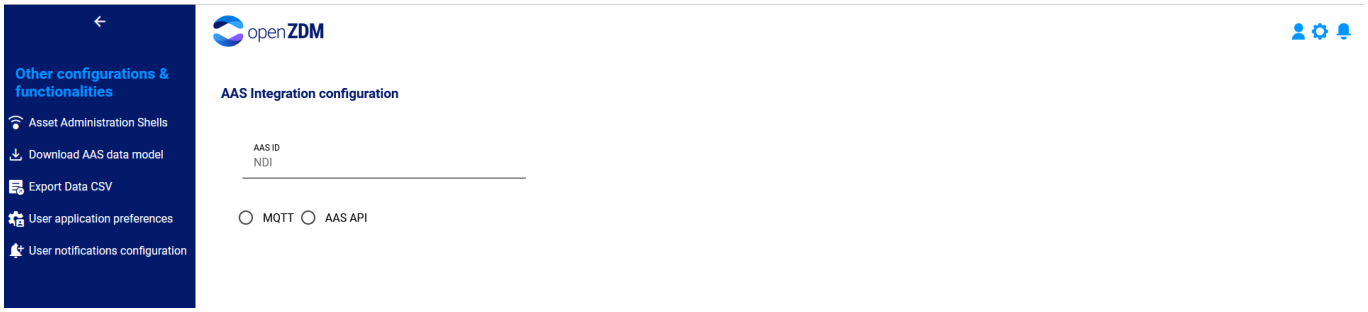

Figure 14: NDI integration configuration view.

If the AAS API is selected, a Client ID and Client secret are generated for this NDI, enabling the NDI software to retrieve an access token from the Identity Server of the platform, which allows access to the AAS API presented in Figure 7.

Figure 15: Results of HTTP AAS API based integration.

If the MQTT option is selected, a different view is presented, prompting the user to fill in the MQTT Broker details as shown in Figure 16. In addition, in this view, a table with existing mappings between AAS properties and topics is also presented. In this case (see Figure 16) the table is empty.

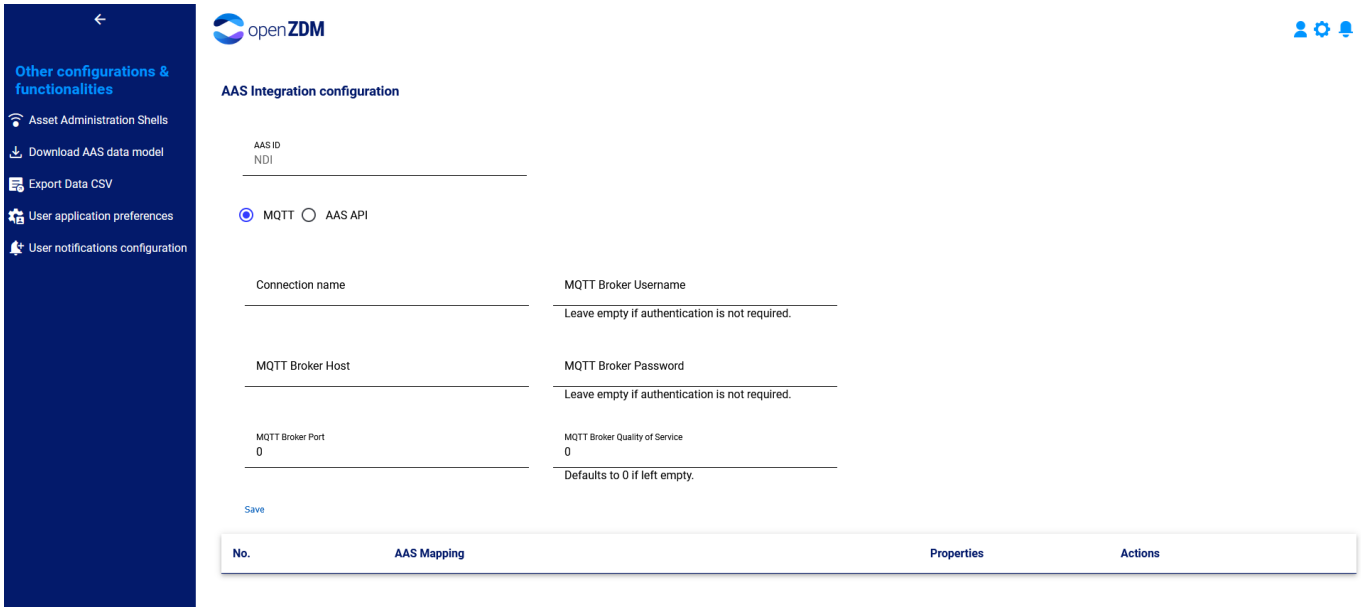

Figure 16: Results of MQTT based integration.

As soon as the MQTT Broker details are defined, the mapping process can start using the dialogue in Figure 17.

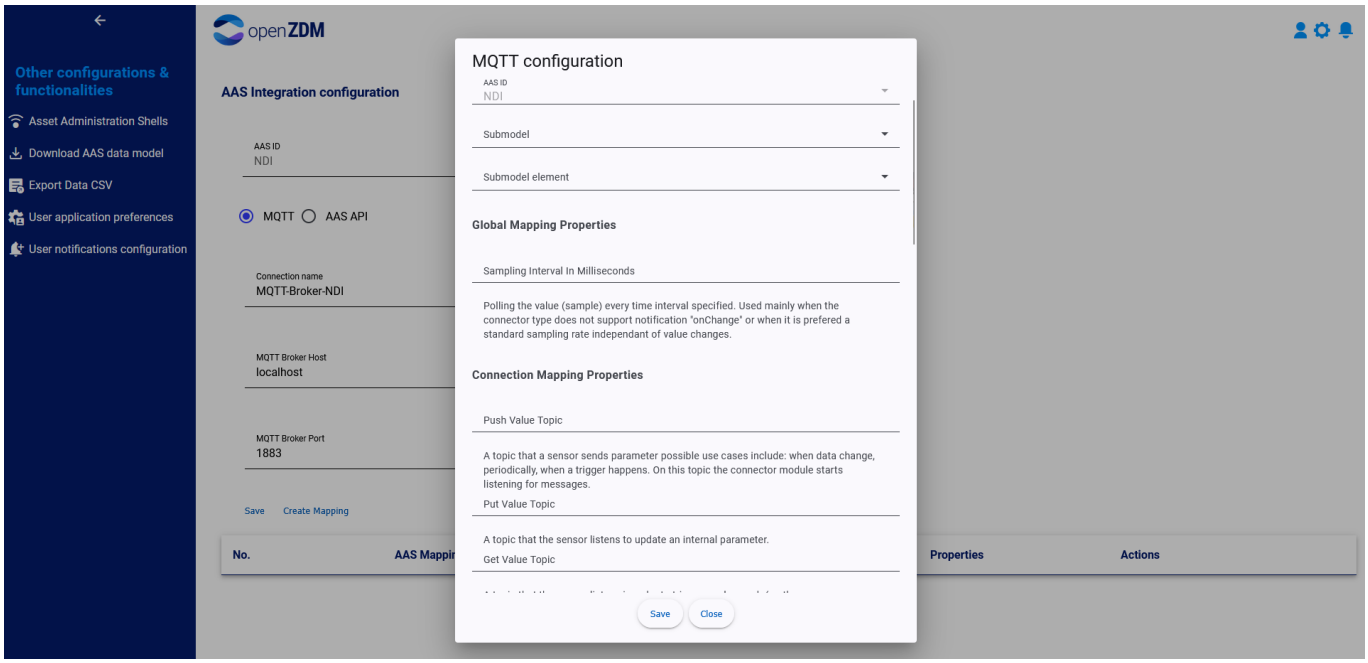

Figure 17: Mapping AAS properties and MQTT topics.

The dialogue starts with the selection of the AAS property to be mapped to an MQTT topic, and then the MQTT configuration options are presented. For this demonstration, the property ‘Temperature’ of the submodel OperationalData is mapped to the topic ‘ndi/data’. Additionally, the option ValueJsonPath is set to ‘$.value’. This configuration will make the system parse the JSON published by the NDI and copy the value of the property named ‘value’ into the AAS Submodel named ‘Temperature’. The configuration is shown in Figure 18.

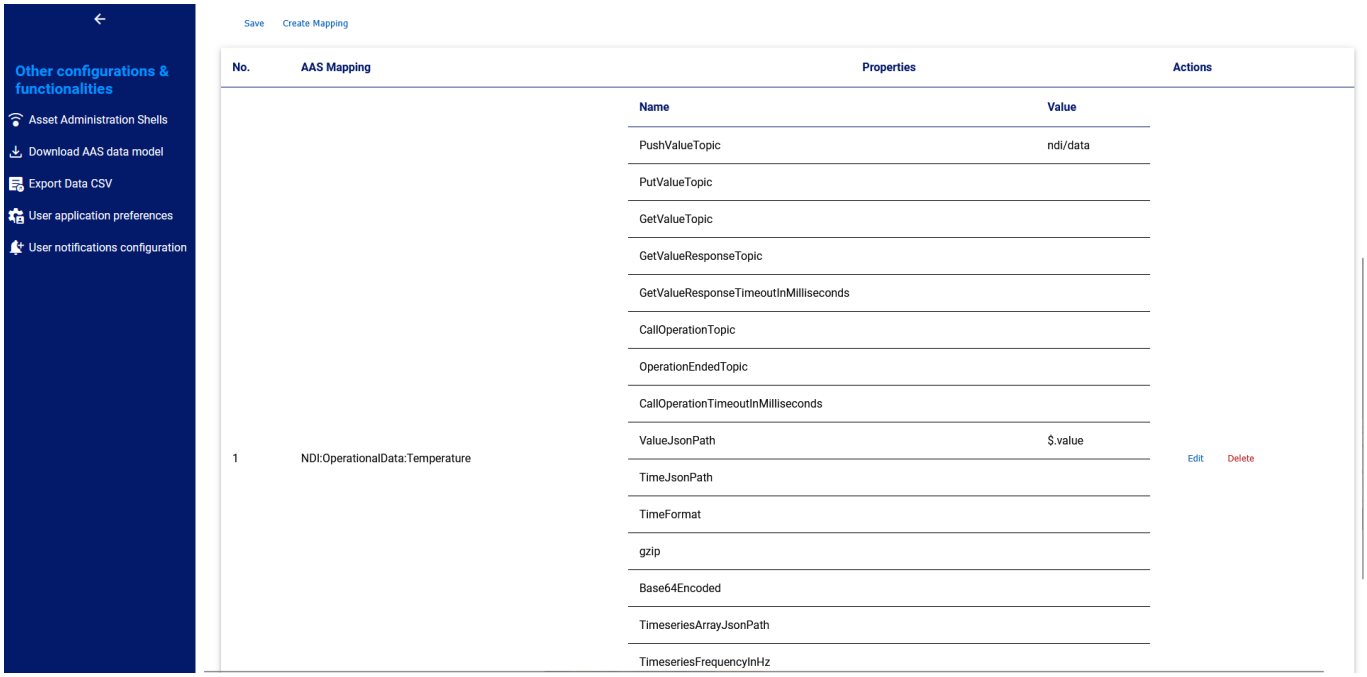

Figure 18: MQTT configurations created in the openZDM platform.

A similar approach is followed for the property 'DateTimeOfControl'; in this case, the ValueJsonPath is set to '$.timestamp'. Now the integration configuration is complete. Testing the integration will take place through the view of an 'Operator' user, where data from the NDI is presented. The first view is the NDI with AAS data in a simplified form (see Figure 19). This view shows parts of the 'Identification' and 'OperationalData' sub models.

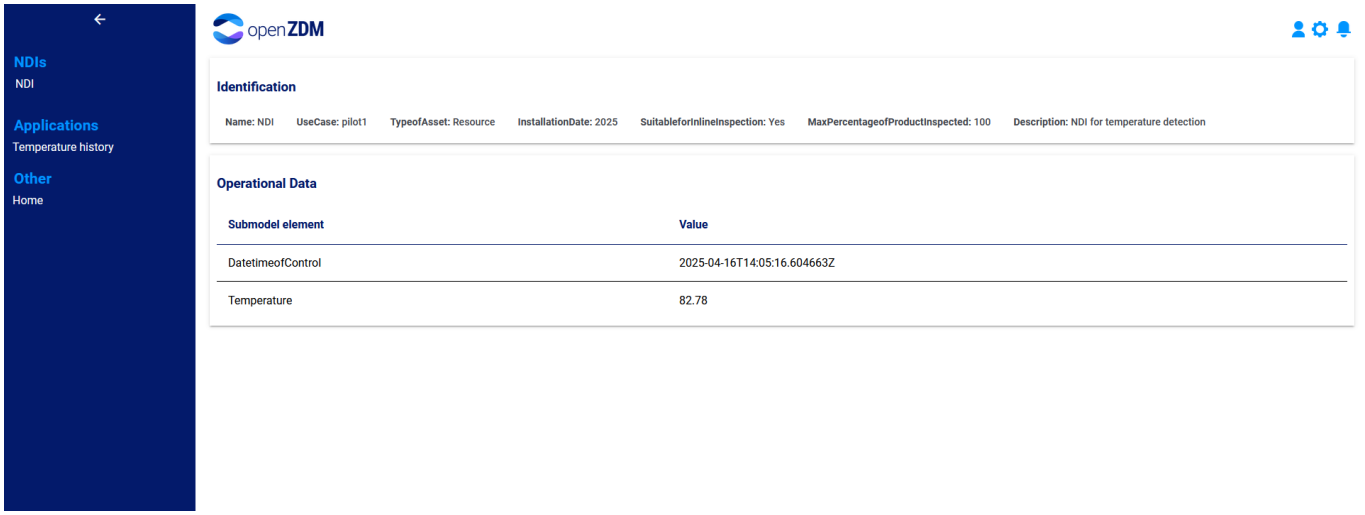

Figure 19: NDI data shown in the view of a user with the 'Operator' role.

In addition to the current NDI values of the AAS model, the platform also tracks historical data. By default, no applications are configured to visualise NDI historical data. However, with the support of the historical data API of the platform, access to the data is available for visualisation, analysis, and so on. For the demonstration, Grafana is used to access the historical data API and visualise it in a graph. The Grafana dashboard can then be added to the user's application through configuration and will be available in their main screen. This dashboard of the NDI's historical data is presented in Figure 20.

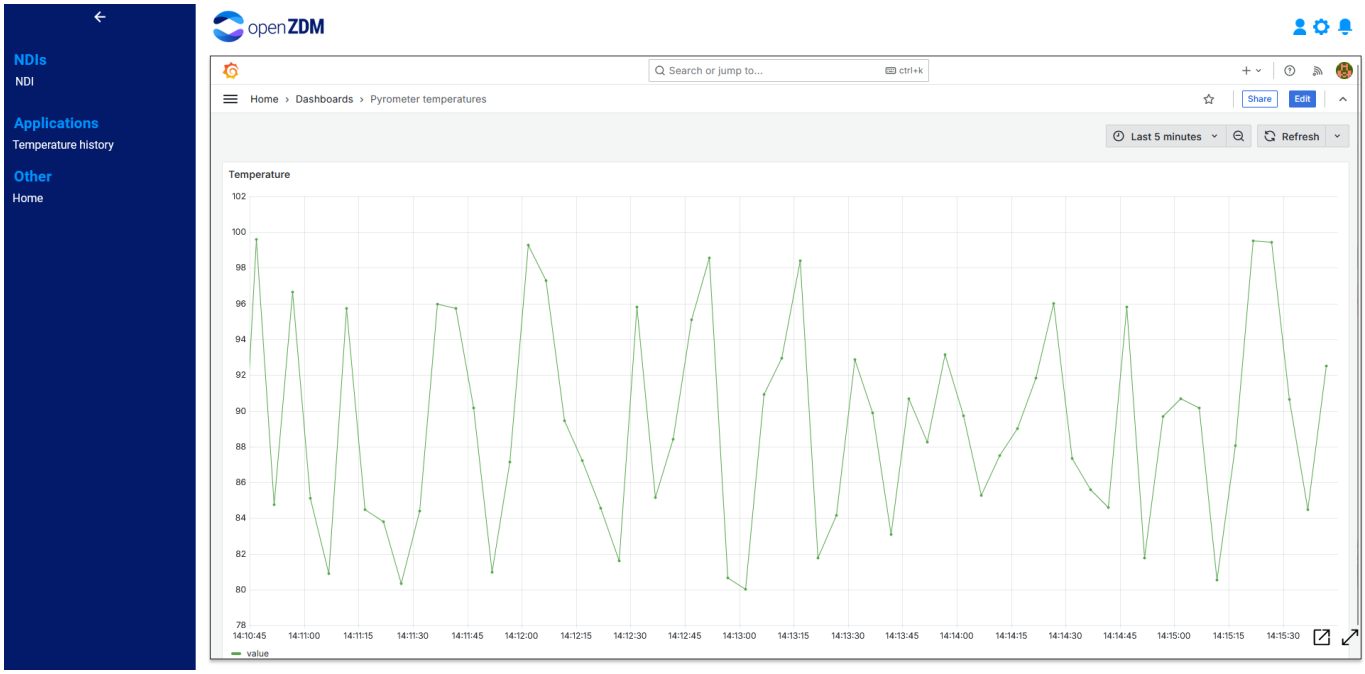

Figure 20: NDI historical data.

4.4 Video

A walkthrough of this demonstration is provided hereafter:

- First, the NDI modelling as an AAS is explained using a predefined AAS for a temperature sensor.
- The AAS file is exported as JSON and imported into the platform.
- Then we move on to the AAS view of a regular user, the AAS is available to all relevant users with the data that is available in the file.
- The next step is the AAS integration configuration. The first part is the MQTT Broker definition. The second part is the mapping between AAS submodel elements and MQTT topics.
- Lastly, in the view of the regular user, both current and historical data are visualised.

A video showing the corresponding demonstrator is available through the following link:

LINK

5. Conclusions

An integration approach for NDIs based on the concept of AAS has been defined for openZDM. The approach has been developed and applied to all NDIs developed in openZDM, whose variety is rather large and representative of many industrial installations. Moreover, thanks to the generality of the approach, it can be used to integrate NDIs outside the project development. Two different approaches have been provided:

1. The HTTP AAS API based approach requires more complexity on the NDI software, as the developer must be aware of the AAS model and the openZDM AAS Middleware API. However, it provides more control over the integration of the NDI to the developer, including a better understanding of the developer if the data has successfully arrived in the platform. This approach can be considered more suited for larger companies with resources to devote to understanding AAS and its complexities, and those NDI developers closely linked to the digital environment (in this case, the openZM platform) or with past knowledge of AAS.
2. The MQTT-based approach is far simpler for the NDI developer and lets them use their format of choice for the output of the NDI (e.g. JSON) while insulating them from the complexities of the AAS model. In fact, the NDI developer may be completely unaware of the AAS approach used in the platform. The NDI developer has a limited understanding of whether the NDI data has reached the platform successfully. They can only understand if the data has reached the MQTT Broker successfully. However, using an appropriate MQTT Broker configuration with data persistence and Quality of Service 2, data loss can be avoided. It should be noted that this approach facilitates the integration of NDIs from independent developers by isolating them from the

complexities of AAS and enabling them to focus only on the NDI development, and in parallel making easier access to the market for them.

The AAS approach provides a significant advantage by insulating the component developers of other parts of the platform from NDI software complexities. These developers have to interact only with the uniform API of the openZDM AAS Middleware and understand the AAS models.

References

[1] Platform Industrie 4.0, “Asset Administration Shell Reading Guide”, 2022. [Online]. Available: https://www.plattform-i40.de/IP/Redaktion/DE/Downloads/Publikation/AAS-ReadingGuide_202201.pdf?__blob=publicationFile&v=1

[2] Platform Industrie 4.0, “What is the Asset Administration Shell from a technical perspective?”, 2021. [Online]. Available: https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/2021_What-is-the-AAS.pdf?__blob=publicationFile&v=1

[3] ZVEI, “Reference Architecture Model Industrie 4.0 (RAMI4.0)”, Jul. 2015. [Online]. Available: https://www.zvei.org/fileadmin/user_upload/Presse_und_Medien/Publikationen/2016/januar/GMA_Status_Report__Reference_Archtitecture_Model_Industrie_4.0__RAMI_4.0_/GMA-Status-Report-RAMI-40-July-2015.pdf. [Accessed Sept. 26, 2022].

[4] K. Alexopoulos, K. Sipsas, E. Xanthakis, S. Makris & D. Mourtzis, “An industrial Internet of things based platform for context-aware information services in manufacturing”, International Journal of Computer Integrated Manufacturing, 31:11, 1111-1123, DOI: 10.1080/0951192X.2018.1500716, 2018

[5] O. Lázaro, et al., “Big Data-Driven Industry 4.0 Service Engineering Large-Scale Trials: The Boost 4.0 Experience”, In: Curry, E., Auer, S., Berre, A.J., Metzger, A., Perez, M.S., Zillner, S. (eds) Technologies and Applications for Big Data Value . Springer, Cham. https://doi.org/10.1007/978-3-030-78307-5_17, 2022

[6] The openZDM Platform [Online]. Available: https://www.openzdm.eu/wp-content/uploads/2024/06/Technical-Articles_The-openZDM-Platform.pdf

[7] OAuth2.0 Protocol. 2025. [Online]. Available: https://oauth.net/2/

[8] H. J. Jara Ochoa, R. Peña, Y. Ledo Mezquita, E. Gonzalez, and S. Camacho-Leon, “Comparative Analysis of Power Consumption between MQTT and HTTP Protocols in an IoT Platform Designed and Implemented for Remote Real-Time Monitoring of Long-Term Cold Chain Transport Operations,” Sensors, vol. 23, no. 10, p. 4896, May 2023, doi: 10.3390/s23104896.

[9] K. T. M. Tran, A. X. Pham, N. P. Nguyen, and P. T. Dang, “Analysis and Performance Comparison of IoT Message Transfer Protocols Applying in Real Photovoltaic System,” Int J Netw Distrib Comput, vol. 12, no. 1, pp. 131–143, Jun. 2024, doi: 10.1007/s44227-024-00021-4.

[10] C. Bayılmış, M. A. Ebleme, Ü. Çavuşoğlu, K. Küçük, and A. Sevin, “A survey on communication protocols and performance evaluations for Internet of Things,” Digital Communications and Networks, vol. 8, no. 6, pp. 1094–1104, Dec. 2022, doi: 10.1016/j.dcan.2022.03.013.

[11] D. Silva, L. I. Carvalho, J. Soares, and R. C. Sofia, “A Performance Analysis of Internet of Things Networking Protocols: Evaluating MQTT, CoAP, OPC UA,” Applied Sciences, vol. 11, no. 11, p. 4879, May 2021, doi: 10.3390/app11114879.


openZDM