Ref. Ares(2022)2420805 - 31/03/2022

Grant Agreement N° 872592

PLATOON

Digital platform and analytic tools for energy

Deliverable D3.3
Open-source data broker

Contractual delivery date:
M16

Actual delivery date:
30 April 2021

Responsible partner:
P4: IAIS, Germany

Project Title	PLATOON – Digital platform and analytic tools for energy
Deliverable number	D3.3
Deliverable title	Open-source data broker
Author(s):	Hantong Liu
Responsible Partner:	P4 – IAIS
Date:	30.04.2021
Nature	R
Distribution level (CO, PU):	PU
Work package number	WP3 – Data Governance, Security and Privacy
Work package leader	IAIS, Germany
Abstract:	This document provides the introduction of the Open-source Broker, the functions of it, how to interact with it in development and how to deploy it.

Keyword List:	Open-source Broker, guide, interaction, development, deployment

The research leading to these results has received funding from the European Community's Horizon 2020 Work Programme (H2020) under grant agreement no 872592.

This report reflects the views only of the authors and does not represent the opinion of the European Commission, and the European Commission is not responsible or liable for any use that may be made of the information contained therein.

Editor(s):	Hantong Liu (IAIS)
Contributor(s):	ENGIE, TECN, ENG
Reviewer(s):	Erik Maqueda (TECN), Martino Maggio (ENG), Philippe Calvez (ENGIE)
Approved by:	Philippe Calvez (ENGIE) – Platoon Coordinator Erik Maqueda (TECN) – Technical Coordinator
Recommended/mandatory readers:	Mandatory readers WP2-WP7 WP and Task Leaders.

Document Description

Document Revision History

Version	Date	Modifications Introduced	
		Modification Reason	Modified by
0.1	14.04.2021	First creation of the document	Hantong Liu (IAIS)
0.2	18.04.2021	Added more details of interacting with broker	Hantong Liu (IAIS)
0.3	28.04.2021	More functions specified Technical structure added Separated development and deployment Examples and figures added for the functions	Hantong Liu (IAIS)
1.0	30.04.2021	Unit tests explanations added Git repository info added Localhost environment as IDS connector to interact with the broker specified REST endpoints added back More docker-compose components information added	Hantong Liu (IAIS)
1.1	05.05.2021	Minor changes	Hantong Liu (IAIS)
1.2	21.05.2021	Review and Minor changes	Philippe Calvez (ENGIE)
1.3	06.10.2021	Minor changes	Tasneem Tazeen Rashid(IAIS) Tejas Morbagal Harish(IAIS)
1.4	11.03.2022	Resubmission after rejection	Najmeh Mouasvi Nejad (IAIS)

Table of Contents

Terms and abbreviations

CA	Consortium Agreement
CO	Confidential
DM	Dissemination Manager
EC	European Commission
EM	Exploitation Manager
GA	Grant Agreement
GAM	General Assembly Meeting
PM	Project Manager
PU	Public
QA	Quality Assurance
RE	Restricted
SC	Steering Committee
TM	Technical Manager
WP	Work package
WPL	Work package Leader

Executive Summary

This document provides technical information on the PLATOON open-source metadata broker (AKA open-source broker, IDS broker), deliverable D3.3. In PLATOON, the core component of Marketplace is the IDS broker (i.e., open-source broker). The broker in the Marketplace would be capable of handling metadata of connectors, resources, and an extension of this broker will also handle the metadata of data apps. The deliverable D3.3 describes the concept of the open-source broker, which facilitates storing and processing the metadata of connectors and resources. The extension of the broker handling metadata of Data Apps(metadata registry) will be reported in the deliverable D3.5, "Marketplace Demonstrator and Report".

1 What is the Open-source Broker?

The Open-source Broker is a metadata registry for datasets and connectors derived from the International Data Spaces (IDS) Metadata Broker. In contrast to the general IDS Metadata Broker, the Broker Open-source has been tailored to keep the main functionalities of metadata handling for connectors and data resources and querying for the metadata. The open-source broker is one of the main components in the PLATOON Marketplace. As shown in Figure 1, data consumers and data providers with an IDS connector can register their resources in the open-source broker. The broker can be used to register, update, or unregister the connector or resource metadata. Note that the broker does not serve the datasets themselves: querying is performed on metadata only. In PLATOON architecture, as shown in Figure 2, the open-source broker is referring to the functionalities of the Data Source Catalog blob on the Marketplace. It should be clarified here that in Figure 2, the open-source broker and context broker are two completely different components.

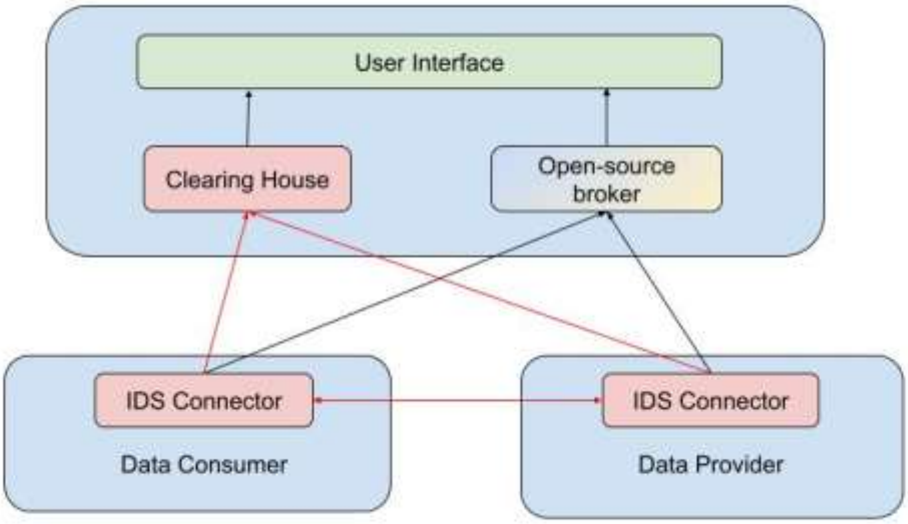

Figure 1 Platoon Marketplace Architecture

This document aims to provide assistance for developers to understand the structure and the functions of the Open-source Broker and for IT administrators to know the deployment of the Open-source Broker.

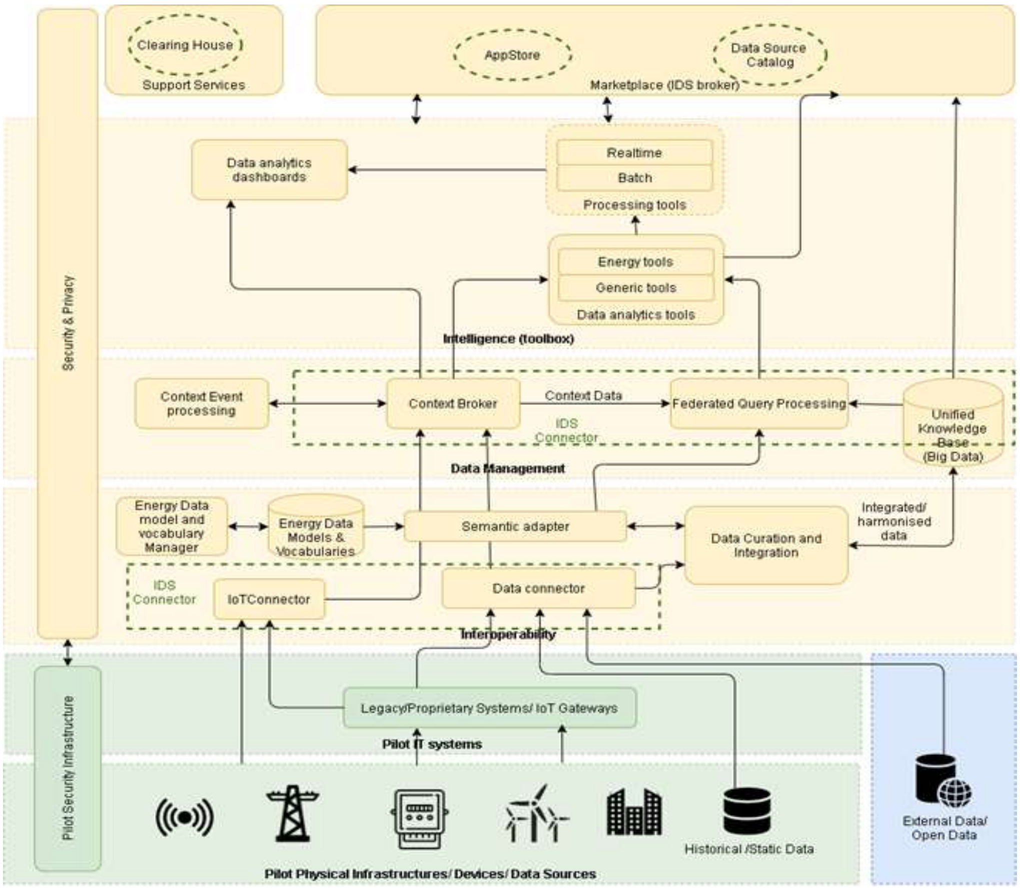

Figure 2 Platoon Architecture

2 Open-source Broker Functions

In Platoon architecture plan, the Open-source Broker will be the main component of Metadata Registry, which is a central component for pilots to register, update or unregister their connector or resource metadata. Pilots can also request the description of the Open-source Broker and also query for any information provided in it.

The following list shows the main functions of the Open-source Broker:

- Description Request:
  - Description Request is to get the self-description of the Open-source Broker.
- Register/Update Connector:
  - Register/Update Connector is to register the connector if it doesn't not exist in the Open-source Broker or update the connector if it exists in the Open-source Broker.
- Unregister Connector:
  - Unregister Connector is to unregister the connector from the Open-source Broker.
- Update Resource:
  - Update Resource is to update the information of a certain resource of a connector.
- Unregister Resource:
  - Unregister Resource is to remove a certain resource from a connector.

- Query:
    - Query is to query the SQARL triples in Fuseki triple store.

Note that all data in the Open-source Broker is metadata only!

The register, update and unregister of connector and resource mentioned above are related to metadata as well.

Each function matches a specific IDS message type. For example, for the description request for the Open-source Broker, the connector can send a DescriptionReqeustMessage.

The following list shows the relations between the functions and IDS messages:

- Description Request for the Open-source Broker:
    - DescriptionRequestMessage
- Register/Update Connector:
    - ConnectorUpdateMessage
- Unregister Connector:
    - ConnectorUnavailableMessage
- Update Resource:
    - ResouceUpdateMessage
- Unregister Resource: ResourceUnavailableMessage
- Query:
    - QueryMessage

Message handlers are created in the code to handle the above messages. In the Open-source Broker, there are four message handlers:

- Description Handler:
    - DescriptionReqeustMessage
- Connector Handler:
    - ConnectorUpdateMessage, ConnectorUnavailableMessage
- Resource Handler:
    - ResouceUpdateMessage, ResouceUnavailableMessage
- Query Handler:
    - QueryMessage

All handlers are added in the AppConfig class to make the Open-source Broker enable such functionalities.

For the further development, the first thing to do is to check which IDS messages should be used to achieve the goal. Then we can create corresponding handlers to handle such messages. At last, we add those handlers in the configuration of Open-source Broker.

The Open-source Broker also includes DAPs service that if the security token of a request is not valid, the Open-source Broker will reject it to ensure unauthorized systems cannot access to the information.

All functions are unit tested in test folders of Maven project. The following command can be used to run all tests:

```
mvn test
```

3 Development

The code of the Open-source Broker is hosted in Platoon Git repository https://github.com/PLATOONProject/open-source-broker

The development environment of Open-source Broker consists of two main components. The first one is a Fuseki triple store which is the database storing all metadata and the other is the Open-source Broker core component.

Prerequisite:

- Docker (19.03.13)
- Docker Compose (1.24.0)
- Java (11)
- Maven (3.6.3)

3.1 Set Up Development Environment

The first step is to run a Fuseki instance. In path *docker/composefiles/DEBUG, run* the command:

```
docker-compose up
```

Now the Fuseki instance is running on port 3030. (Figure 3)

Figure 3 Fuseki instance

The next step is to run Open-source Broker core. In path *broker-core,* run the command:

```
mvn spring-boot:run
```

Now the Open-source Broker core is running on port 8080 and the self-description is also presented. (Figure 4)

Figure 4 Self-Description

3.2 Interacting with the Open-source Broker

The Open-source Broker accepts and sends messages according to the IDS information model. This model uses the Resource Description Framework (RDF) to leverage the power of linked data. As such, all messages are JSON-LD formatted HTTP Multipart messages.

The multipart endpoint of Open-source Broker is “/infrastructure”. If the Open-source Broker is running, an HTTP POST request can be sent to interact with it. The only header should be “Content-Type” and the value should be “multipart/mixed; boundary=msgpart” as shown in Figure 5.

Note that the boundary value can be changed to any value, but it must be the same boundary in the request body.

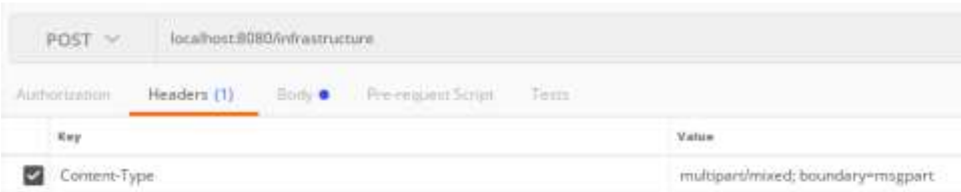

Figure 5 Header of the Multipart Request

The following use cases will show how to interact with the Open-source Broker and it will cover all main functions of the Open-source Broker. Note that the localhost environment here acts as the IDS connector to interact with the Open-source Broker.

3.2.1 Description Request

The multipart message header should be DescriptionRequestMessage and the payload should be empty. (Figure 6)

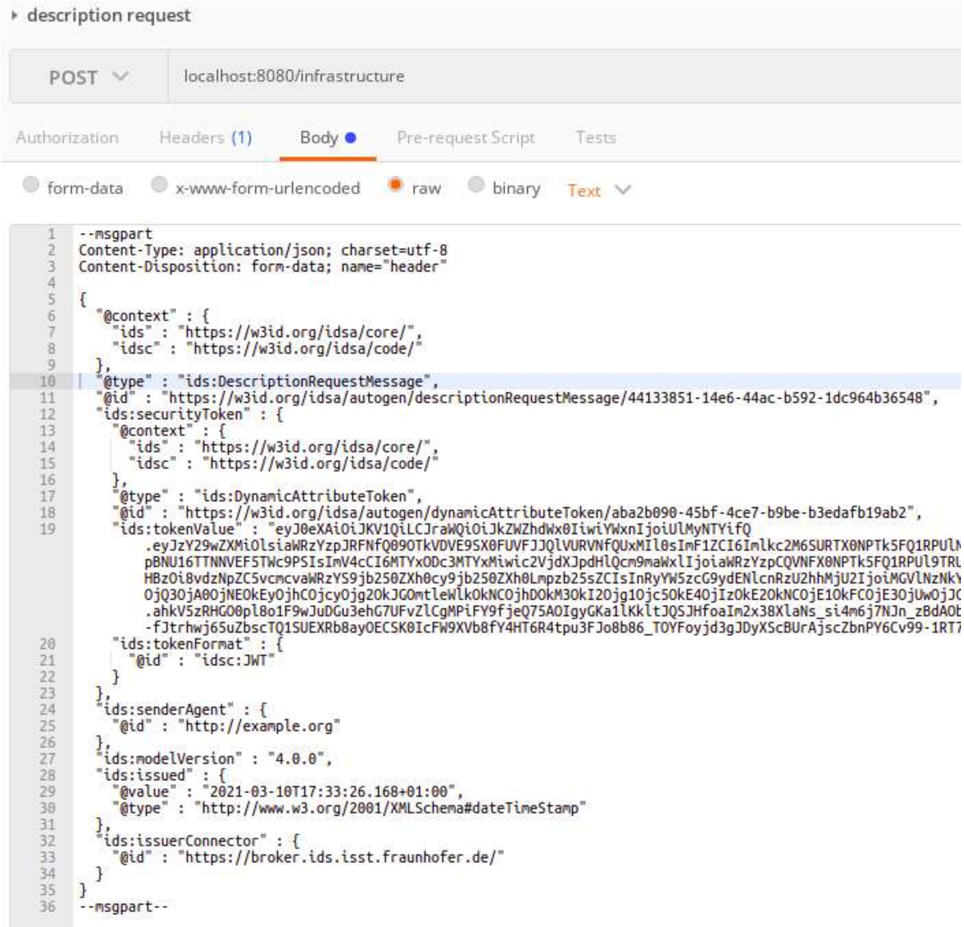

Figure 6 Description Request

The response should be a DescriptionResponseMessage with the self-description of the Open-source Broker in the payload. (Figure 7)


```
--VDPULU3yfnfZ5UM9rv6gL877gENqmVGjqP2s
Content-Disposition: form-data; name="header"
Content-Type: application/ld+json
Content-Length: 2276

{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
    "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:DescriptionResponseMessage",
  "@id" : "https://w3id.org/idsa/autogen/descriptionResponseMessage/611b5dbb-2ec9-4bec-93ca-4857b06ff5bd",
  "ids:correlationMessage" : {
    "@id" : "https://w3id.org/idsa/autogen/descriptionRequestMessage/44133851-14e6-44ac-b592-1dc964b36548"
  },
  "ids:issued" : {
    "@value" : "2021-04-28T12:08:59.651+02:00",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:issuerConnector" : {
    "@id" : "http://localhost:8080/"
  },
  "ids:senderAgent" : {
    "@id" : "https://www.iais.fraunhofer.de"
  },
  "ids:securityToken" : {
    "@type" : "ids:DynamicAttributeToken",
    "@id" : "https://w3id.org/idsa/autogen/dynamicAttributeToken/12f21edc-ed4c-4caf-a1f3-3ad236cad09c",
    "ids:tokenValue" : "eyJ0eXAiOiJKV1QiLCJraWQiOiJkZWZhdWx0IiwiYWxnIjoiUlMyNTYifQ
        .eyJzY29wZXMiOlsiaWRzYzpJRFNfQ09OTkVDVE9SX0FUVFJJQlVURVNfQUxMIl0sImF1ZCI6Imlkc2M6SURTX0NPTk5FQ1RPUlNfQL
        1TXpRMU5ERXhOekF4TmprPSIsImV4cCI6MTYxOTYwNzQ3Mywic2VjdXJpdHlQcm9maWxlIjoiaWRzYzpCQVNFX0NPTk5FQ1RPUl9TRL
        6Imh0dHBzOi8vdzNpZC5vcmcvaWRzYS9jb250ZXh0cy9jb250ZXh0Lmpzb25sZCIsInRyYW5zcG9ydENlcnRzU2hhMjU2IjoiMGVlNz
        zOjI1OjQ5OjQ3OjA0OjNEOkEyOjhCOjcyOjg2OkJGOmtleWlkOkNCOjhDOkM3OkI2Ojg1Ojc5OkE4OjIzOkE2OkNCOjE1OkFCOjE3Oj
        -RJTGmlvafCynAUBjVpcGUd6zk4Lz7eXJmXjmIsqfQOCSpFBuCRqWMEHnH0NrXi34KzHyNJwLLrfERiLSvpCSFq_gkQXEQqbe5Svsw_
        -cRSoHsR_knuvSwAL0bcuEY713FzjtKfBNHaRGaQdmGTOlomak-p8PYUa2tvOTizymeXpK8HUeJD-DmVqcTtcLQ",
    "ids:tokenFormat" : {
      "@id" : "idsc:JWT"
    }
  },
  "ids:modelVersion" : "4.0.3",
  "Serialization" : "Lang:JSON-LD",
  "elementType" : "BrokerImpl"
}
--VDPULU3yfnfZ5UM9rv6gL877gENqmVGjqP2s
Content-Disposition: form-data; name="payload"
Content-Type: application/ld+json
Content-Length: 2724

{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
    "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:Broker",
  "@id" : "http://localhost:8080/",
```


Figure 7 Description Request Response

3.2.2 Register/Update Connector

The multipart message header should be ConnectorUpdateMessage and the payload should be JSON-LD format connector metadata. (Figure 8)

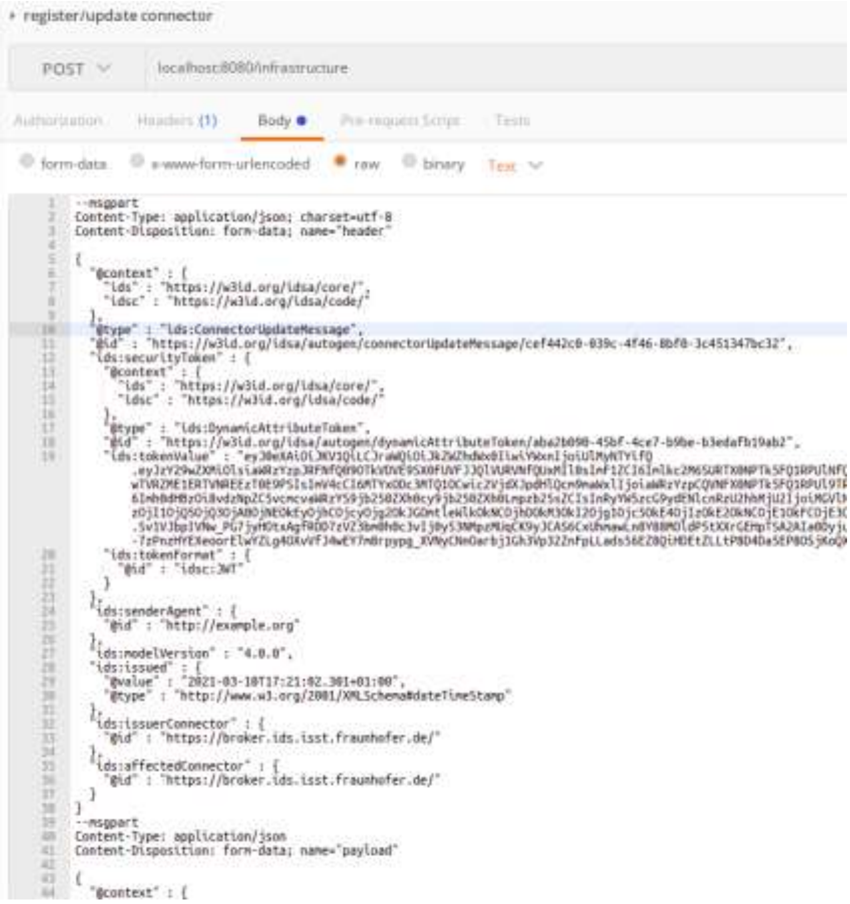

Figure 8 Register/Update Connector Request

The response should be a MessageProcessedNotificationMessage without payload. (Figure 9)


```
--WD3kKgcZoPn-P16XvL0cmSSxjrxflt6V4G45
Content-Disposition: form-data; name="header"
Content-Type: application/ld+json
Content-Length: 2286

{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
    "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:MessageProcessedNotificationMessage",
  "@id" : "https://w3id.org/idsa/autogen/messageProcessedNotificationMessage/e3be96cf-7d4f-4f33-8c12-55b5c811ad22",
  "ids:correlationMessage" : {
    "@id" : "https://w3id.org/idsa/autogen/connectorUpdateMessage/cef442c0-039c-4f46-8bf0-3c451347bc32"
  },
  "ids:issued" : {
    "@value" : "2021-04-28T12:13:13.137+02:00",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:issuerConnector" : {
    "@id" : "http://localhost:8080/"
  },
  "ids:senderAgent" : {
    "@id" : "https://www.iais.fraunhofer.de"
  },
  "ids:securityToken" : {
    "@type" : "ids:DynamicAttributeToken",
    "@id" : "https://w3id.org/idsa/autogen/dynamicAttributeToken/5c617ff4-844f-4ecc-a842-d7499fd70b39",
    "ids:tokenValue" : "eyJ0eXAiOiJKV1QiLCJraWQiOiJkZWZhdWx0IiwiYWxnIjoiUlMyNTYifQ
        .eyJzY29wZXMiOlsiaWRzYzpJRFNfQ09OTkVDVE9SX0FUVFJJQlVURVNfQUxMIl0sImF1ZCI6Imlkc2M6SURTX0NPTk5FQ1RPUlNfQUxMIiwia
        pRMU5ERXhOekF4TmprPSIsImV4cCI6MTYxOTYwNzQ3Mywic2VjdXJpdHlQcm9maWxlIjoiaWRzYzpCQVNFX0NPTk5FQ1RPUl9TRUNVUklUWV9Q
        HBzOi8vdzNpZC5vcmcvaWRzYS9jb250ZXh0cy9jb250ZXh0Lmpzb25sZCIsInRyYW5zcG9ydENlcnRzU2hhMjU2IjoiMGVlNzNkYzBlYmNjMDk
        OjQ3OjA0OjNEOkEyOjhCOjcyOjg2OkJGOmtleWlkOkNCOjhDOkM3OkI2Ojg1Ojc5OkE4OjIzOkE2OkNCOjE1OkFCOjE3OjUwOjJGOkU2OjY1Oj
        -RJTGmlvafCynAUBjVpcGUd6zk4Lz7eXJmXjmIsqfQOCSpFBuCRqWMEHnH0NrXi34KzHyNJwLLrfERiLSvpCSFq_gkQXEQqbe5Svsw_tcUzxOq
        -cRSoHsR_knuvSwAL0bcuEY713FzjtKfBNHaRGaQdmGTOlomak-p8PYUa2tvOTizymeXpK8HUeJD-DmVqcTtcLQ",
    "ids:tokenFormat" : {
      "@id" : "idsc:JWT"
    }
  },
  "ids:modelVersion" : "4.0.3",
  "Location" : "<http://localhost:8080/connectors/541260824>"
}
--WD3kKgcZoPn-P16XvL0cmSSxjrxflt6V4G45--
```


Figure 9 Register/Update Request Connector Response

3.2.3 Unregister Connector

The multipart message header should be ConnectorUnavailableMessage and the payload should be empty. (Figure 10)

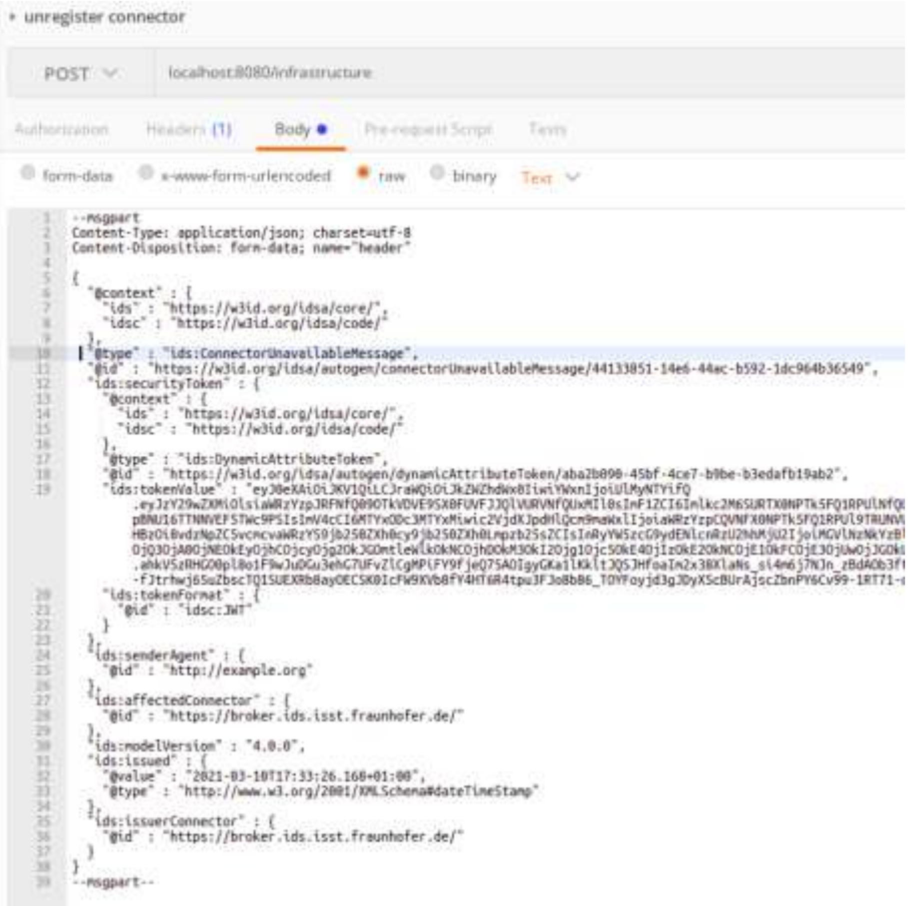

Figure 10 Unregister Connector Request

The response should be a MessageProcessedNotificationMessage without payload. (Figure 11)


```
---CzvI2Z37kimk6miSW5s40B70117CWLxJHROU0q
Content-Disposition: form-data; name="header"
Content-Type: application/ld+json
Content-Length: 2228

{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
    "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:MessageProcessedNotificationMessage",
  "@id" : "https://w3id.org/idsa/autogen/messageProcessedNotificationMessage/97f86be2-a525-4175-98f5-88f5b842e307",
  "ids:correlationMessage" : {
    "@id" : "https://w3id.org/idsa/autogen/connectorUnavailableMessage/44133851-14e6-44ac-b592-1dc964b36549"
  },
  "ids:issued" : {
    "@value" : "2021-04-28T12:22:22.755+02:00",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:issuerConnector" : {
    "@id" : "http://localhost:8080/"
  },
  "ids:senderAgent" : {
    "@id" : "https://www.iais.fraunhofer.de"
  },
  "ids:securityToken" : {
    "@type" : "ids:DynamicAttributeToken",
    "@id" : "https://w3id.org/idsa/autogen/dynamicAttributeToken/8acf83ea-da9a-43a5-a3f2-3d034315ae18",
    "ids:tokenValue" : "eyJ0eXAiOiJKV1QiLCJraWQiOiJkZWZhdWx0IiwiYWxnIjoiUlMyNTYifQ
        .eyJzY29wZXMiOlsiaWRzYzpJRFNfQ090TkVDVE9SX0FUVFJJQlVURVNfQUxMIl0sImF1ZCI6Imlkc2M6SURTX0NPTk5FQ1RPUlNfQUxMIiwiaXN
        pRMU5ERXhOekF4TmprPSIsImV4cCI6MTYxOTYwNzQ3Mywic2VjdXJpdHlQcm9maWxlIjoiaWRzYzpCQVNFX0NPTk5FQ1RPUl9TRUNVUklUWV9QUk
        HBzOi8vdzNpZC5vcmcvaWRzYS9jb250ZXh0cy9jb250ZXh0Lmpzb25sZCIsInRyYW5zcG9ydENlcnRzU2hhMjU2IjoiMGVlNzNkYzBlYmNjMDk3
        OjQ3OjA0OjNEOkEyOjhCOjcyOjg2OkJGOmtleWlkOkNCOjhDOkM3OkI2Ojg1Ojc5OkE4OjIzOkE2OkNCOjE1OkFCOjE3OjUwOjJGOkU2OjY1OjQ
        -RJTGmlvafCynAUBjVpcGUd6zk4Lz7eXJmXjmIsqfQOCSpFBuCRqWMEHnH0NrXi34KzHyNJwLLrfERiLSvpCSFq_gkQXEQqbe5Svsw_tcUzxOqb
        -cRSoHsR_knuvSwAL0bcuEY713FzjtKfBNHaRGaQdmGTOlomak-p8PYUa2tvOTizymeXpK8HUeJD-DmVqcTtcLQ",
    "ids:tokenFormat" : {
      "@id" : "idsc:JWT"
    }
  },
  "ids:modelVersion" : "4.0.3"
}
---CzvI2Z37kimk6miSW5s40B70117CWLxJHROU0q--
```


Figure 11 Unregister Connector Request Response

3.2.4 Update Resource

The multipart message header should be ResourceUpdateMessage and the payload should be JSON-LD format resource metadata. (Figure 12)

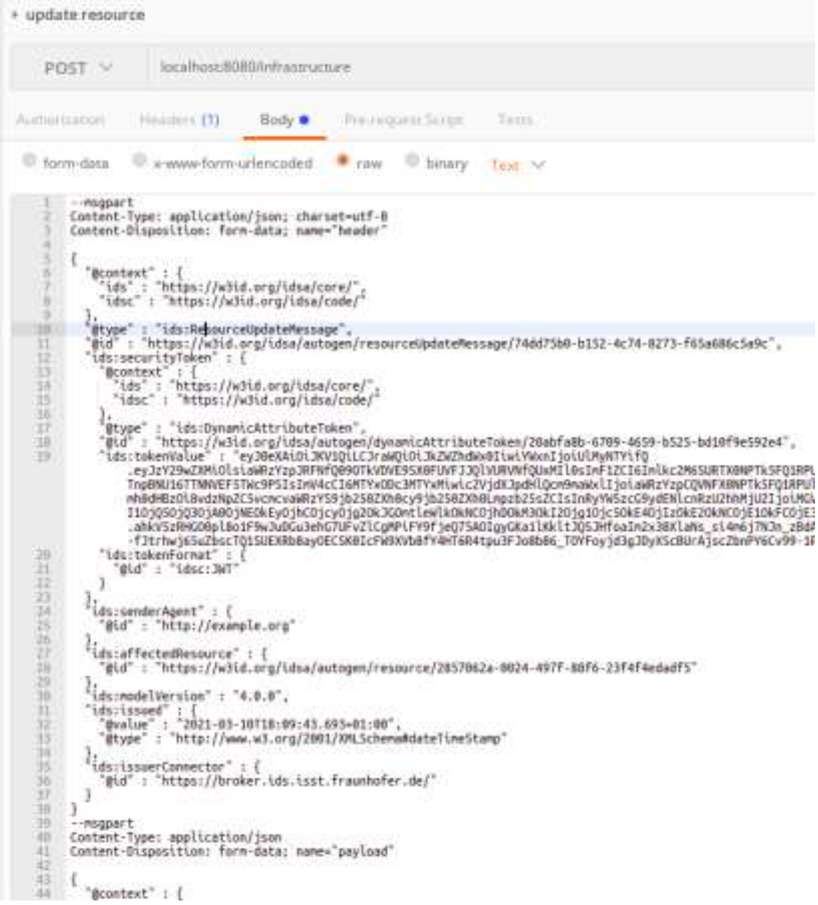

Figure 12 Update Resource Request

The response should be a MessageProcessedNotificationMessage without payload. (Figure13)


```
--cxW4Kz1lFNOTAlXsV3Y_Rk995xyoz5
Content-Disposition: form-data; name="header"
Content-Type: application/ld+json
Content-Length: 2307

{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
    "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:MessageProcessedNotificationMessage",
  "@id" : "https://w3id.org/idsa/autogen/messageProcessedNotificationMessage/7d93ad52-0db8-4b0a-a909-1c45b2384ec0",
  "ids:correlationMessage" : {
    "@id" : "https://w3id.org/idsa/autogen/resourceUpdateMessage/74dd75b0-b152-4c74-8273-f65a686c5a9c"
  },
  "ids:issued" : {
    "@value" : "2021-04-28T12:15:43.039+02:00",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:issuerConnector" : {
    "@id" : "http://localhost:8080/"
  },
  "ids:senderAgent" : {
    "@id" : "https://www.iais.fraunhofer.de"
  },
  "ids:securityToken" : {
    "@type" : "ids:DynamicAttributeToken",
    "@id" : "https://w3id.org/idsa/autogen/dynamicAttributeToken/b5f61010-6b7a-4fdc-8063-74d5a6ca786a",
    "ids:tokenValue" : "eyJ0eXAiOiJKV1QiLCJraWQiOiJkZWZhdWx0IiwiYWxnIjoiUlMyNTYifQ
        .eyJzY29wZXMiOlsiaWRzYzpJRFNfQ09OTkVDVE9SX0FUVFJJQlVURVNfQUxMIl0sImF1ZCI6Imlkc2M6SURTX0NPTk5FQ1RPUlNfQUxMIiwia
        pRMU5ERXhOekF4TmprPSIsImV4cCI6MTYxOTYwNzQ3Mywic2VjdXJpdHlQcm9maWxlIjoiaWRzYzpCQVNFX0NPTk5FQ1RPUl9TRUNVUklUWV9Q
        HBzOi8vdzNpZC5vcmcvaWRzYS9jb250ZXh0cy9jb250ZXh0Lmpzb25sZCIsInRyYW5zcG9ydENlcnRzU2hhMjU2IjoiMGVlNzNkYzBlYmNjMDk
        0jQ3OjA0OjNEOkEyOjhCOjcyOjg2OkJGOmtleWlkOkNCOjhDOkM3OkI2Ojg1Ojc5OkE4OjIzOkE2OkNCOjE1OkFCOjE3OjUwOjJGOkU2OjY1Oj
        -RJTGmlvafCynAUBjVpcGUd6zk4Lz7eXJmXjmIsqfQOCSpFBuCRqWMEHnH0NrXi34KzHyNJwLLrfERiLSvpCSFq_gkQXEQqbe5Svsw_tcUzxOql
        -cRSoHsR_knuvSwAL0bcuEY713FzjtKfBNHaRGaQdmGTOlomak-p8PYUa2tvOTizymeXpK8HUeJD-DmVqcTtcLQ",
    "ids:tokenFormat" : {
      "@id" : "idsc:JWT"
    }
  },
  "ids:modelVersion" : "4.0.3",
  "Location" : "<http://localhost:8080/connectors/541260824/1487174079/-715517746>"
}
--cxW4Kz1lFNOTAlXsV3Y_Rk995xyoz5--
```


Figure 13 Update Resource Request Response

3.2.5 Unregister Resource

The multipart message header should be ResourceUnavailableMessage and the payload should be empty. (Figure 14)

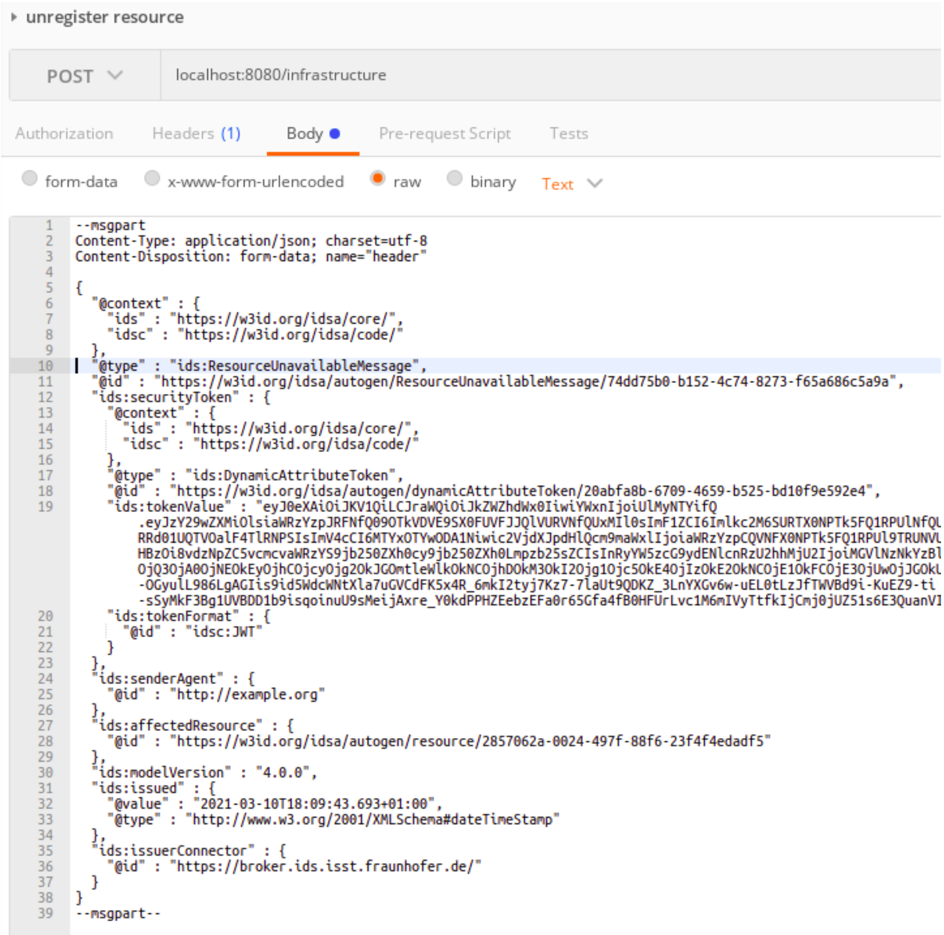

Figure 14 Unregister Resource Request

The response should be a MessageProcessedNotificationMessage without payload. (Figure 15)


```
--uVgWUpgM_X3cQSw7_A2NXZSZyE2zYnFUMv
Content-Disposition: form-data; name="header"
Content-Type: application/ld+json
Content-Length: 2227

{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
    "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:MessageProcessedNotificationMessage",
  "@id" : "https://w3id.org/idsa/autogen/messageProcessedNotificationMessage/b46a9e47-c3c7-410b-adc7-53fd90331796",
  "ids:correlationMessage" : {
    "@id" : "https://w3id.org/idsa/autogen/ResourceUnavailableMessage/74dd75b0-b152-4c74-8273-f65a686c5a9a"
  },
  "ids:issued" : {
    "@value" : "2021-04-28T13:05:49.424+02:00",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:issuerConnector" : {
    "@id" : "http://localhost:8080/"
  },
  "ids:senderAgent" : {
    "@id" : "https://www.iais.fraunhofer.de"
  },
  "ids:securityToken" : {
    "@type" : "ids:DynamicAttributeToken",
    "@id" : "https://w3id.org/idsa/autogen/dynamicAttributeToken/7709f489-daca-4488-ad82-2bd1e2196a34",
    "ids:tokenValue" : "eyJ0eXAiOiJKV1QiLCJraWQiOiJkZWZhdWx0IiwiYWxnIjoiUlMyNTYifQ
        .eyJzY29wZXMiOlsiaWRzYzpJRFNfQ09OTkVDVE9SX0FUVFJJQlVURVNfQUxMIl0sImF1ZCI6Imlkc2M6SURTX0NPTk5FQ1RPUlNfQUxMIiwiaXNzI
        pneE5EUTJNek13TWc9PSIsImV4cCI6MTYxOTYxMTUxNiwic2VjdXJpdHlQcm9maWxlIjoiaWRzYzpCQVNFX0NPTk5FQ1RPUl9TRUNVUklUWV9QUk9
        HBzOi8vdzNpZC5vcmcvaWRzYS9jb250ZXh0cy9jb250ZXh0Lmpzb25sZCIsInRyYW5zcG9ydENlcnRzU2hhMjU2IjoiMGVlNzNkYzBlYmNjMDk3YT
        OjQ3OjA0OjNEOkEyOjhCOjcyOjg2OkJGOmtleWlkOkNCOjhDOkM3OkI2Ojg1Ojc5OkE4OjIzOkE2OkNCOjE1OkFCOjE3OjUwOjJGOkU2OjY1OjQzO
        -4Y4KFqKB8dKrU7lHq5JVcOCPD0mAOUfrZEvuRmnpwNI76d2_jBLevfKf59uCGvnvcrHRulJ-gio8_P0uVm2WSVgOYkWpgMv8a3FrGEsZg1jY7KV-
        -02FvD6sCTG28EAZo8jr3a7P2MYGin8AoRWBlPdllGBZK8JRlLYmz9LDzZIJuGqxLvIcFzy5ITEWhQ0y8xsj0W-0yW4mHNyG2Q6gp88-WjYY9nNuI
    "ids:tokenFormat" : {
      "@id" : "idsc:JWT"
    }
  },
  "ids:modelVersion" : "4.0.3"
}
--uVgWUpgM_X3cQSw7_A2NXZSZyE2zYnFUMv--
```


Figure 15 Unregister Resource Request Response

3.2.6 Query

The multipart message header should be QueryMessage and the payload should be a SPARQL query. (Figure 16)

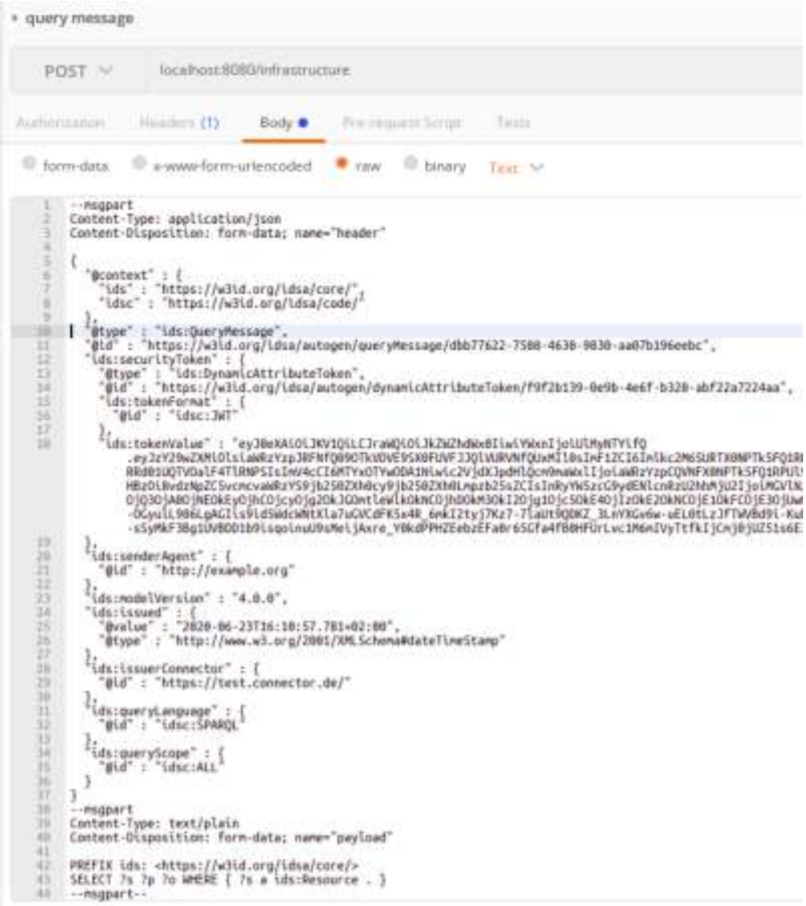

Figure 16 Query Request

The response should be a ResultMessage and the payload should be the result of the SPARQL query. (Figure 17)


```
--NeAedHriCKcot4W-pAhtGdi3eCUF9ep0QjgRj5N
Content-Disposition: form-data; name="header"
Content-Type: application/ld+json
Content-Length: 2169

{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
    "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:ResultMessage",
  "@id" : "https://w3id.org/idsa/autogen/resultMessage/9e588e3c-61f4-4603-b4fe-06555ce863f7",
  "ids:correlationMessage" : {
    "@id" : "https://w3id.org/idsa/autogen/queryMessage/dbb77622-7508-4630-9830-aa07b196eebc"
  },
  "ids:issued" : {
    "@value" : "2021-04-28T12:20:28.191+02:00",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:issuerConnector" : {
    "@id" : "http://localhost:8080/"
  },
  "ids:senderAgent" : {
    "@id" : "https://www.iais.fraunhofer.de"
  },
  "ids:securityToken" : {
    "@type" : "ids:DynamicAttributeToken",
    "@id" : "https://w3id.org/idsa/autogen/dynamicAttributeToken/1355be01-ee20-40dd-8b2d-ddd2460a4f4e",
    "ids:tokenValue" : "eyJ0eXAiOiJKV1QiLCJraWQiOiJkZWZhdWx0IiwiYWxnIjoiUlMyNTYifQ
        .eyJzY29wZXMiOlsiaWRzYzpJRFNfQ09OTkVDVE9SX0FUVFJJQlVURVNfQUxMIl0sImF1ZCI6Imlkc2M6SURTX0NPTk5FQ1RP
        pRMU5ERXhOekF4TmprPSIsImV4cCI6MTYxOTYwNzQ3Mywic2VjdXJpdHlQcm9maWxlIjoiaWRzYzpCQVNFX0NPTk5FQ1RPUl9
        HBzOi8vdzNpZC5vcmcvaWRzYS9jb250ZXh0cy9jb250ZXh0Lmpzb25sZCIsInRyYW5zcG9ydENlcnRzU2hhMjU2IjoiMGVlNz
        OjQ3OjA0OjNEOkEyOjhCOjcyOjg2OkJGOmtleWlkOkNCOjhDOkM3OkI2Ojg1Ojc5OkE4OjIzOkE2OkNCOjE1OkFCOjE3OjUwO
        -RJTGmlvafCynAUBjVpcGUd6zk4Lz7eXJmXjmIsqfQOCSpFBuCRqWMEHnH0NrXi34KzHyNJwLLrfERiLSvpCSFq_gkQXEQqbe
        -cRSoHsR_knuvSwAL0bcuEY713FzjtKfBNHaRGaQdmGTOlomak-p8PYUa2tvOTizymeXpK8HUeJD-DmVqcTtcLQ",
    "ids:tokenFormat" : {
      "@id" : "idsc:JWT"
    }
  },
  "ids:modelVersion" : "4.0.3"
}
--NeAedHriCKcot4W-pAhtGdi3eCUF9ep0QjgRj5N
Content-Disposition: form-data; name="payload"
Content-Type: text/plain
Content-Length: 78

?s  ?p  ?o
<http://localhost:8080/connectors/541260824/1487174079/1538291463>

--NeAedHriCKcot4W-pAhtGdi3eCUF9ep0QjgRj5N--
```


Figure 17 Query Request Response

3.2.7 Rejection

The above responses are all successful responses. If anything in the request is not correct, a rejection message will be sent in the response from the Open-source Broker. For example, if the security token is not valid, there will be a rejection message with "Error processing token" in the response. (Figure 18)


```
---Vj6M6km5eSsLkijU67A2Tj5XvGgpXLuq
Content-Disposition: form-data; name="header"
Content-Type: application/ld+json
Content-Length: 2260

{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
    "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:RejectionMessage",
  "@id" : "https://w3id.org/idsa/autogen/rejectionMessage/ed0633a9-df8f-4cca-a8e0-9f53c4fa0091",
  "ids:correlationMessage" : {
    "@id" : "https://w3id.org/idsa/autogen/connectorUnavailableMessage/44133851-14e6-44ac-b592-1dc964b36549"
  },
  "ids:issued" : {
    "@value" : "2021-04-28T12:21:49.255+02:00",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:issuerConnector" : {
    "@id" : "http://localhost:8080/"
  },
  "ids:senderAgent" : {
    "@id" : "https://www.iais.fraunhofer.de"
  },
  "ids:rejectionReason" : {
    "@id" : "idsc:NOT_AUTHENTICATED"
  },
  "ids:securityToken" : {
    "@type" : "ids:DynamicAttributeToken",
    "@id" : "https://w3id.org/idsa/autogen/dynamicAttributeToken/41d08598-508c-44ac-8539-49598e391f19",
    "ids:tokenValue" : "eyJ0eXAiOiJKV1QiLCJraWQiOiJkZWZhdWx0IiwiYWxnIjoiUlMyNTYifQ
        .eyJzY29wZXMiOlsiaWRzYzpJRFNfQ09OTkVDVE9SX0FUVFJJQlVURVNfQUxMIl0sImF1ZCI6Imlkc2M6SURTX0NPTk5FQ1RPUlNfQUx
        pRMU5ERXhOekF4TmprPSIsImV4cCI6MTYxOTYwNzQ3Mywic2VjdXJpdHlQcm9maWxlIjoiaWRzYzpCQVNFX0NPTk5FQ1RPUl9TRUNVUk
        HBzOi8vdzNpZC5vcmcvaWRzYS9jb250ZXh0cy9jb250ZXh0Lmpzb25sZCIsInRyYW5zcG9ydENlcnRzU2hhMjU2IjoiMGVlNzNkYzBlY
        OjQ3OjA0OjNEOkEyOjhCOjcyOjg2OkJGOmtleWlkOkNCOjhDOkM3OkI2Ojg1Ojc5OkE4OjIzOkE2OkNCOjE1OkFCOjE3OjUwOjJGOkU2
        -RJTGmlvafCynAUBjVpcGUd6zk4Lz7eXJmXjmIsqfQOCSpFBuCRqWMEHnH0NrXi34KzHyNJwLLrfERiLSvpCSFq_gkQXEQbe5Svsw_t
        -cRSoHsR_knuvSwAL0bcuEY713FzjtKfBNHaRGaQdmGTOlomak-p8PYUa2tvOTizymeXpK8HUeJD-DmVqcTtcLQ",
    "ids:tokenFormat" : {
      "@id" : "idsc:JWT"
    }
  },
  "ids:modelVersion" : "4.0.3"
}
---Vj6M6km5eSsLkijU67A2Tj5XvGgpXLuq
Content-Disposition: form-data; name="payload"
Content-Type: text/plain
Content-Length: 23

Error processing token.
---Vj6M6km5eSsLkijU67A2Tj5XvGgpXLuq--
```


Figure 18 Rejection Response

3.3 REST Endpoints

The Open-source Broker also provides optional REST endpoints. The theory behind it is to put the required fields in multipart header to the header of the REST request, such as ids-securityToken, ids-senderAgent, ids-modelVersion, ids-issued, ids-issuerConnector, etc... and keep the multipart body in the body of the REST request.

4 Deployment

In this chapter, we will provide guidance how to deploy the software on a server. We'll create docker images for each Open-source Broker components and run the corresponding docker containers in the same docker-compose environment so that each component can communicate with each other internally.

The Open-source Broker consists of three components:

- broker-core
- broker-fuseki
- broker-reverseproxy

The broker-core component is the main component of the Open-source Broker, which is a Java environment running the Maven package of our code. The broker-core component will take requests from broker-reverseproxy, handle requests using the same handlers we mentioned in Chapter 3, and use APIs that broker-fuseki provides to either read or write in the Fuseki triple store.

The broker-fuseki component is an instance of Fuseki triple store, which hosts all metadata of the Open-source Broker.

The broker-reverseproxy is a NGINX reverse proxy instance, which hosts the certificate to provide security connection to our Open-source Broker. It also acts as the gateway to redirect requests to the broker-core component.

Prerequisite:

- Docker (19.03.13)
- Docker Compose (1.24.0)

4.1 Recommended System Specifications

In this section, we will provide some guidance as to recommendations for the amount of resources that should be available to smoothly operate the Open-source Broker. The actual amount of resources required heavily depends on the load. In case of very little traffic, fewer resources than listed below might be required.

4.1.1 Hardware

2GB of disk space is required for operating the Open-source Broker, though we recommend providing at least 20GB of free disk storage to avoid running out of disk space with increasing number of registered items. We also recommend reserving at least 2GB of RAM.

To provide enough processing power for all Docker containers, we recommend using a 64bit quad core processor or higher.

4.1.2 Software

We recommend using a Linux based operating system. However, any operating system with a Docker installation can be used (tested on Ubuntu 20.04 and Windows 10). More strict hardware requirements than listed above might apply if a non-Linux operating system is used.

4.1.3 Other

A valid X.509 certificate, signed by a trusted certification authority, is strongly recommended to avoid warnings about insecure HTTPS connections. The certificate needs to be of .crt format and must have the name server.crt. In case your certificate is of .pem format, it can be converted with the following commands, which require OpenSSL to be installed:

```
openssl x509 -in mycert.pem -out server.crt
openssl rsa -in mycert.pem -out server.key
mkdir cert
mv server.crt cert/
mv server.key cert/
```

4.2 Configuring the docker-compose File

The docker-compose file is located in path *docker/composefiles/broker-localhost*.

The most crucial part of adapting the configuration is to provide the correct location of the X.509 certificate in the broker-reverseproxy service. Assuming the location of the certificate is "/home/ids/cert", the corresponding configuration is:

```
services:
  broker-reverseproxy:
    image: registry.gitlab.cc-asp.fraunhofer.de:4567/eis-ids/broker-open/reverseproxy
    volumes:
      - /home/ids/cert:/etc/cert/
    […]
```

4.3 Downloading the Docker Images

All of the Open-source Broker docker images are currently hosted at the GitLab of Fraunhofer IAIS. No credentials needed to download the images. The following command is for pulling all docker images (in path *docker/composefiles/broker-localhost*):

```
docker-compose pull
```

Note that the docker images will hosted in a Platoon image registry in the future and how to download the image may change afterwards.

4.4 Starting up the Open-source Broker

To start up the Open-source Broker, run the following command inside the directory of the docker-compose.yml file (in path *docker/composefiles/broker-localhost*):

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

This process can take several minutes to complete. You can test whether the broker has successfully started by opening https://localhost. The result should be a JSON document, providing some general metadata about the Open-source Broker, which should be the same as Figure 2.

4.5 Updating the Open-source Broker

To update an existing installation of the Open-Source Broker, first repeat the steps explained in section "Downloading the Docker Containers". Containers can be either hot updated or restarted to apply the changes. To hot update a container, run the following command:

```
docker-compose up -d --no-deps --build <container name>
```

Alternatively, one can restart the entire service by running:

```
docker-compose down
docker-compose up –d
```

5 Conclusion

This deliverable describes how the open-source broker handles the messages sent from the Connectors and stores the metadata of the Connectors in the RDF triple store. Once the Connector (data provider) is registered with the open-source broker, the other Connectors (data consumers) can query the metadata. The deliverable presents the functions that the Open-source Broker supports and provides detailed description for setting up, deploying and interacting with the Open-source Broker in the developing environment. This open-source broker will be extended to handle metadata from Data Apps in M24 and will be reported in deliverable D3.5, "Marketplace Demonstrator and Report".