Grant Agreement N° 872592

PLATOON

Digital platform and analytic tools for energy

Deliverable D3.4
Open Source Data Connector

Contractual delivery date:
M20

Actual delivery date:
10 September 2021

Responsible partner:
IAIS, Germany

Project Title	PLATOON – Digital platform and analytic tools for energy
Deliverable number	D3.4
Deliverable title	Open Source Data Connector
Author(s):	Martino Maggio (ENG) Vincenzo Savarino (ENG) Tasneem Tazeen Rashid (IAIS) Tejas Morbagal Harish (IAIS) Idoia Murua (TECN)

	Urtza Iturraspe (TECN)
Responsible Partner:	IAIS, Germany
Date:	10.09.2021
Nature	O
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 open-source Data Connector, the functions of it, and how to deploy and interact with it.
Keyword List:	IDS Connector, data security, privacy, personal data

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

Editor(s):	Martino Maggio (ENG) Vincenzo Savarino (ENG)
Contributor(s):	Tasneem Tazeen Rashid (IAIS) Tejas Morbagal Harish (IAIS) Idoia Murua (TECN) Urtza Iturraspe (TECN)
Reviewer(s):	Erik Maqueda (TECN) Valentin Sanchez (TECN) Philippe Calvez (ENGIE) Martino Maggio (ENG)
Approved by:	Philippe Calvez (ENGIE) Erik Maqueda (TECN)
Recommended/mandatory readers:	WP2-WP6, WP8, WP9, and Task Leaders

Document Revision History

Document Revision History

Version	Date	Modifications Introduced	
		Modification Reason	Modified by
0.1	21/07/2021	Table of Contents	Martino Maggio (ENG)
0.2	26/01/2021	Added content about CaPe integration	Vincenzo Savarino (ENG)
0.3	30/07/2021	Added content related to Data Usage Controll app	Idoia Murua (TECN) Urtza Iturraspe (TECN)
0.4	02/08/2021	Added content about IDS connector	Tasneem Tazeen Rashid (IAIS) Tejas Morbagal Harish (IAIS)
0.5	23/08/2021	Added content about TRUE connector and overall finalisation	Martino Maggio (ENG)
0.9	29/08/2021	Final revision for internal review	Martino Maggio (ENG)
1.0	03/09/2021	Revision after internal review	Martino Maggio (ENG), Tasneem Tazeen Rashid (IAIS) Tejas Morbagal Harish (IAIS)
1.1	10/09/2021	Editing executive summary	Martino Maggio (ENG), Tasneem Tazeen Rashid (IAIS) Tejas Morbagal Harish (IAIS)

Table of Contents

List of Figures

Terms and Abbreviations

API	Application Program Interface
CR	Consent Record
CSR	Consent Status Record
DMZ	Demilitarized Zone
ECC	Execution Core Container
GDPR	General Data Protection Regulation
IdM	Identity Management
IDS	International Data Spaces
IP	Internet Protocol
JSON	JavaScript Object Notation
ODRL	Open Digital Rights Language
PoP	Proof of Possession
REST	Representational state transfer
SDK	Software Development Kit
SLR	Service Link Record
SLSR	Service Link Status Records
SSR	Service Link Status Records
TRUE	TRUsted Engineering
UC	Usage-Control
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WS	Web service

1 Introduction

This document is a technical report related to the "Open-Source Data Connector" that has been provided in PLATOON. In particular the aim of this activity was to provide in the PLATOON project an open-source instance of an IDS Connector that fits project requirements and can be easily integrated in the PLATOON technical platform. To achieve this objective it was decided, inside the project, to reuse an existing asset, the "TRUE (TRUsted Engineering) connector" to be extended and customised for the specific project needs. The TRUE connector is an open-source instance of an IDS Connector developed by Engineering inside other European initiatives, in particular is one of the main outcomes of the project Market 4.0[1]. The decision to reuse this software was based on different considerations: (1) TRUE connector was one of the few open-source IDS connector instances available (2) it is a stable component and a project that is currently evolved (3) it is developed by Engineering, the leader of the architectural task in PLATOON, (4) it is easily extensible to fit PLATOON requirement, and can be integrated in the overall project technical architecture.
The TRUE Connector has been extended to provide new capabilities required in PLATOON. In particular:

- Integration with the CaPe suite: CaPe in an ICT suite for a consent-based user-centric personal data management that acts as an intermediary and as a tool of communication between data subjects and controllers/processors, supporting the generation and management of dynamic consents. In this way the IDS connector will also support, not only the traditional exchange of industrial data, but also personal data, using in this specific case the capability of CaPe to manage user data consents in compliance with GDPR.
- Development of a new, open-source, Data Usage Control module as an evolution of the open-source IDS Dataspace Connector[2] that supports usage policies written in the IDS Usage Control Language[3] based on ODRL[4]. In particular, the Data Usage Control module is able to specifically filter personal data interacting with CaPe suite.
- Integration with the other PLATOON components and pilot infrastructure customising the trivial Data App that allows to exchange data with the connector following IDS message specification.

This document describes the functionalities, architecture and interfaces of the above-mentioned component together with basic installation and deployment instructions. More detailed technical information, together with the source code of the components, can be found in the PLATOON GitHub repository[5].

[1] http://market40.eu/
[2] https://github.com/International-Data-Spaces-Association/DataspaceConnector
[3] https://internationaldataspaces.org/wp-content/uploads/IDSA-Position-Paper-Usage-Control-in-the-IDS-V3.0.pdf
[4] https://www.w3.org/TR/odrl-model/
[5] https://github.com/PLATOONProject

2 Executive Summary

This document provides technical information about the PLATOON Open-Source Data Connector (Deliverable 3.4) and its integration with external components. The scope of PLATOON Open-Source Data Connector is to provide a secure communication component, based on IDS specification and compliant with PLATOON architecture, that can assure privacy and security in data transmission between different parties. The PLATOON Open-Source Data Connector is based on the "TRUE" connector[6] an open-source reference implementation of an IDS connector that have been customised in order to fit project requirements, in particular supporting the management of personal information, in compliance with GDPR, through the integration of CaPe suite[7] and developing a new Data Usage Control module that supports usage policies written in the IDS Usage Control Language based on ODRL. The PLATOON Open-Source Data Connector can be easily integrated with the specific pilot infrastructure implementing dedicated Data application enabling the secure data transmission between different pilots in the project. The deliverable provides architectural and functional information about the developed components, but also technical documentation related to installation and API usage.

[6] https://github.com/Engineering-Research-and-Development/true-connector
[7] https://github.com/OPSILab/Cape

3 PLATOON Open-Source Connector

3.1 IDS Connector Overview

In International Data Spaces (IDS)[8], the Connector is the central technological building block. It is a dedicated software component allowing Participants to exchange, share and process digital content. At the same time, the Connector ensures that the data sovereignty of the Data Owner is always guaranteed. Depending on the type of configuration, the Connector's tamper-proof runtime hosts a variety of system services ensuring, for example, secure bidirectional communication, enforcement of content usage policies, system monitoring, and logging of content transactions for clearing purposes. The functional range of a generic Connector may be extended by custom software (Data Apps), allowing data processing, visualization, persistence, etc.

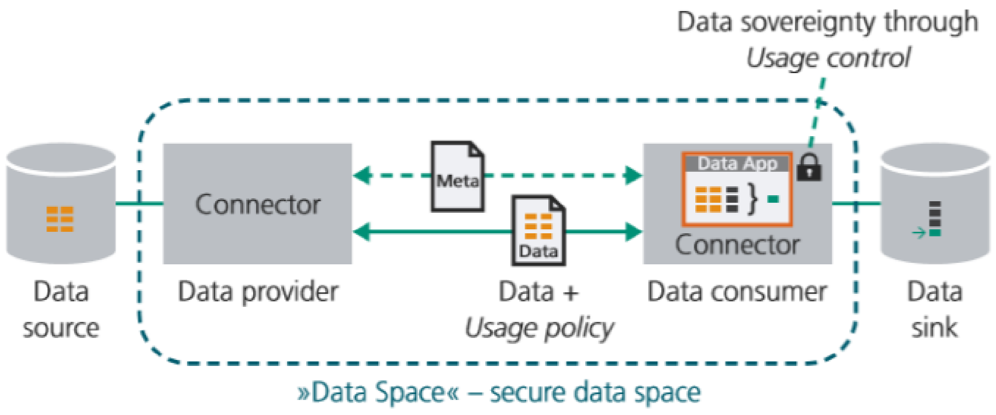


Figure 1 - Data Exchange in the IDS between Connectors

As shown in Figure 1[9] , the Connector provides metadata to the Data Consumer Connector as specified in the Connector's self-description. For example, technical interface description, authentication mechanism, exposed data sources, and associated data usage. Following the peer-to-peer network concept, the data is transferred between the Connectors of the Data Provider and the Data Consumer.

Based on different technologies and functionalities, there may be different types of Connectors. The fundamental variants of the Connector are:

1. **Base Connector**: The Base Connector contains all the basic components described in the IDS Architecture.
2. **Trusted Connector**: Trusted Connector is the extension of the Base Connector to all the security specifications.

Furthermore, the Connectors can be distinguished between:

1. **External Connector**: An External Connector executes the data exchange between participants of the International Data Spaces. An External Connector is typically operated behind a firewall in a specially secured network segment of a participant (so-

[8] P. D.-I. B. Otto, S. Steinbuß, A. Teuscher and D.-I. S. Lohmann, REFERENCE ARCHITECTURE MODEL, International Data Spaces Association, 2019.
[9] D.-I. S. Pretzsch, H. Drees and D. L. Rittershaus, Mobility Data Space, Fraunhofer-Institut für Verkehrs- und Infrastruktursysteme IVI, 2020.

called "Demilitarized Zone", DMZ). External Connector can be reached using the standard Internet Protocol (IP), and can be operated in any appropriate environment.
2. **Internal Connector**: An Internal Connector is typically operated in an internal company network which is a network which is not accessible from outside.

Implementations of Internal Connectors and External Connectors may be identical, as only the purpose and configuration differ.

3.1.1 Connector Architecture

The Connector Architecture uses Application Container Management technology to ensure an isolated and secure environment for individual Data Services. A Data Service matches a system which offers an API to store, access or process data. To ensure privacy of sensitive data, data processing should take place as close to the data source as possible. Any data pre-processing (e.g., filtering, anonymization, or analysis) should be performed by Internal Connectors. Only data intended for being made available to other participants should be made visible through External Connectors.

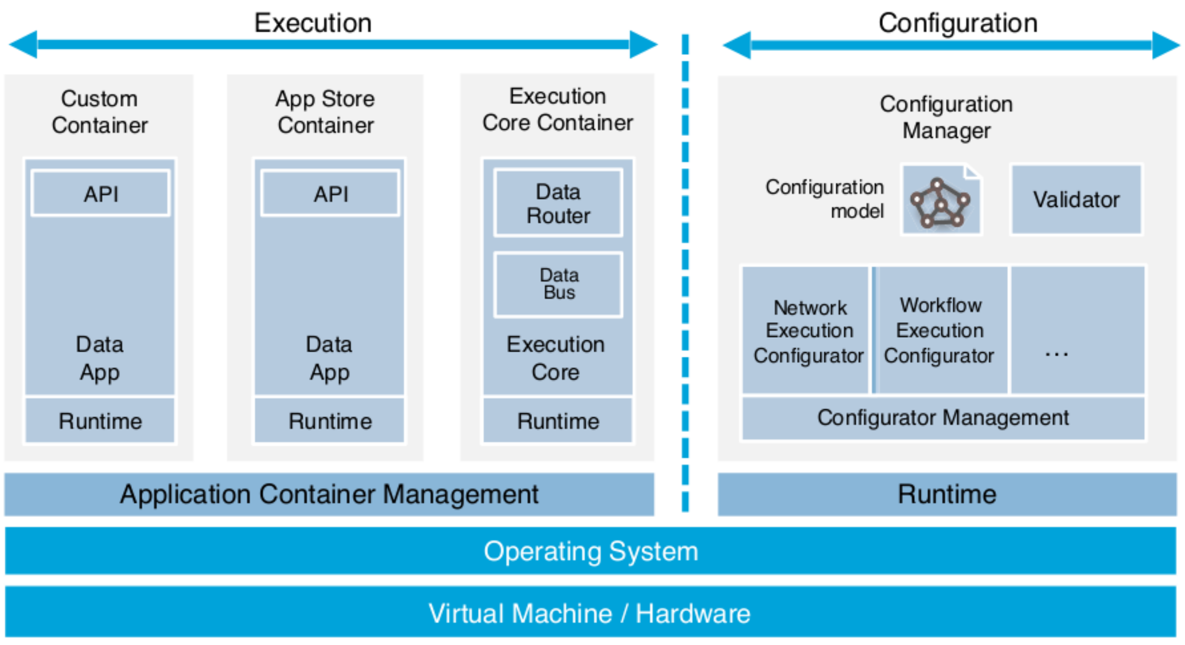


Figure 2 - Connector Architecture

Figure 2 illustrates the internal structure of the Connector. A concrete installation of a Connector may differ from this structure, as existing components can be modified and optional components added. The components shown in Figure 2 can be assigned to two phases: Execution and Configuration.

The Execution phase of a Connector involves the following components:

1. **Application Container Management**: Application Containers allows the deployment of an Execution Core Container and selected Data Services. Extended control of Data Services and Containers can be enforced using Application Container Management,
2. **Execution Core Container**: An Execution Core Container provides components for interfacing with Data Services and supporting communication (e.g., Data Router or Data Bus to a Connector).
3. **App Store Container**: An App Store Container is a certified Container downloaded from the App Store, providing a specific Data Service to the Connector.
4. **Custom Container:** A Custom Container provides a self-developed Data Service. Custom Containers usually require no certification.
5. **Data Service**: Data Service defines a public API, which is invoked from a Data Router. This API is formally specified in a meta-description that is imported into the configuration model.
6. **The Runtime**: The Runtime of a Data Service depends on the selected technology and programming language. Different Containers may use different runtimes. The availability of runtimes depends mainly on the host computer's operating system. The service architect selects the suitable runtimes from the available ones.

The Configuration phase of a Connector involves the following components:

1. **Configuration Manager**: Configuration Manager is the administrative part of a Connector that manages and validates the Configuration Model followed by deployment of the Connector.
2. **Configuration Model**: Configuration Model is the extendable domain model that describes the configuration of a Connector.
3. **Configurator Management**: Configurator Management loads and manages an exchangeable set of Execution Configurators and assigns each task to a special Execution Configurator during the deployment of a Connector.
4. **Execution Configurators**: Execution Configurators are exchangeable plug-ins that execute or translate single aspects of the Configuration Model to a specific technology. Depending on the technology the execution of the Configuration is maintained. For example, the generation of configuration files or the usage of a configuration API.
5. **Validator**: Validator checks if the Configuration Model complies with both self-defined rules and the general rules specified by the International Data Spaces and violation of rules can be treated as warnings or errors and the deployment may fail or be rejected.

3.1.2 Connector Configuration Model

The Connector Configuration Model, as shown in Figure 3 , is technology-independent describing the configuration of a Connector, which is set up during deployment. A Connector can be configured for different statuses (development, test, or live).

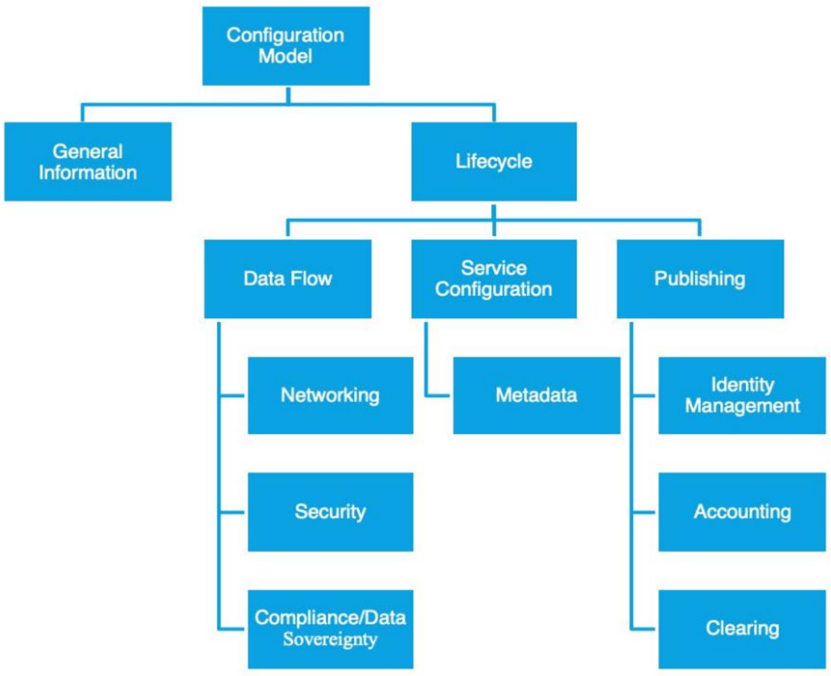


Figure 3 - Connector Configuration Model

The components of the Connector Configuration Model are implemented with the following Configurators:

1. **General Information**: General Information” includes the configuration type, the connector type (Base, Trusted, Mobile, Embedded, Developer), the version of the connector, a timestamp of the last change made to the configuration, the status of configuration (development, test, live), and a name of a contact person.
2. **Lifecycle:**
   a. **Data Flow:** “Data Flow” defines the tasks and connections configurations established by the Data Router between the Data Services and the Data Bus (for multiple data pipelines).
      i. **Networking**: “Networking” relates to the definition of network parameters (ports, IPs, etc.) for being used inside the Connector as well as for connections to External Connectors.
      ii. **Security**: “Security” contains information about the SSL certificates that is used for connections, or which public key infrastructure should be used.
      iii. **Compliance / Data Sovereignty**: “Compliance / Data Sovereignty” specifies rules to be checked by the Validator before Connector deployment. If warnings or errors occur, deployment may be cancelled. This feature is used to prevent incorrect configuration of the Connector.

   b. **Service Configuration:** Service Configuration explains how the configuration parameters for Data Services or other Connector components need to be set.
      i. **Metadata:** “Metadata” describes the data types for input and output used by different Connector components. Data Services can provide metadata descriptions, which can be imported to the Configuration Model. This information is used to configure the Data Flow.

c. **Publishing:** Publishing describes the Data Flows or the Data Services that are provided to the external participants and submitted to a Broker.

i. **Identity Management:** Identity Manager describes the Identity Provider closely integrated with the Connector. Data Service may need additional libraries to connect Identity Provider.

ii. **Accounting:** It is important to record additional information for example contract specifications, pricing models, or billing details for accounting of a data exchange transaction between participants.

iii. **Clearing:** Clearing describes which Clearing House should be informed regarding a certain data exchange transaction.

3.2 TRUE Connector

The TRUE (**TRU**sted **E**ngineering) Connector is an open-source implementation, developed by Engineering, of the IDS connector. The TRUE connector has been developed in other research projects, in particular the Market 4.0 project[10]. In PLATOON the TRUE Connector has been reused and extended in order to fit the specific project needs.

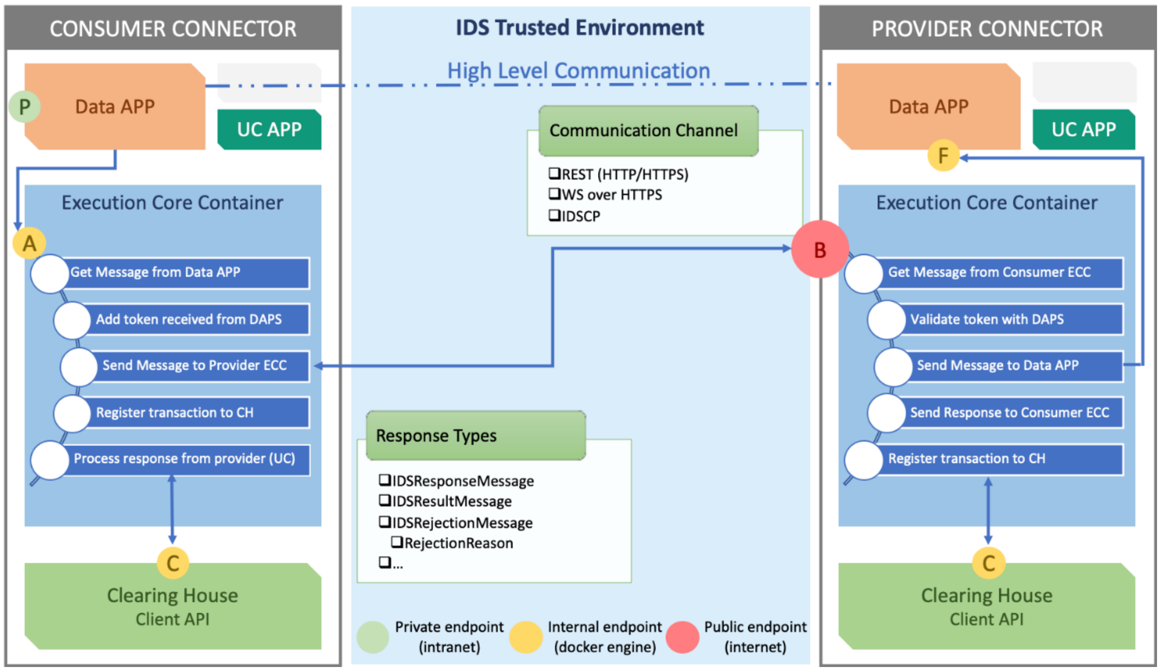

CONSUMER CONNECTOR
IDS Trusted Environment
PROVIDER CONNECTOR
High Level Communication
P
Data APP
UC APP
Data APP
UC APP
F
Communication Channel
REST (HTTP/HTTPS)
WS over HTTPS
IDSCP
Execution Core Container
Execution Core Container
A
B
Get Message from Data APP
Add token received from DAPS
Send Message to Provider ECC
Register transaction to CH
Process response from provider (UC)
Get Message from Consumer ECC
Validate token with DAPS
Send Message to Data APP
Send Response to Consumer ECC
Register transaction to CH
Response Types
IDSResponseMessage
IDSResultMessage
IDSRejectionMessage
RejectionReason
...
C
Clearing House Client API
C
Clearing House Client API
Private endpoint (intranet)
Internal endpoint (docker engine)
Public endpoint (internet)

Figure 4: TRUE Connector Architecture and Interactions

Figure 4 depicts, on one hand, the architecture of the TRUE Connector and, on the other, the exchange of information between a Consumer connector and a Provider connector. Each connector is composed of the following components:

- **Execution Core Container (ECC):** it is the core component of the TRUE Connector. It is in charge of:

[10] http://market40.eu/

    - The data exchange using HTTP/HTTPS, WS over HTTPS or IDSCP2[11], taking advantage of the IDS Information Model to represent the data.
    - The interactions with an external Identity Provider to require and validate a token.
    - The communication with an IDS Broker for managing the information and with the Clearing House for registering the transactions.
- **Data APP:** this component represents a trivial data application build to generate and/or consume data on top of the Execution Core Container.
- **Usage-Control Application (UC APP):** a component in charge of applying usage control policies to the data, please refer to section 3.3.2 for additional details.

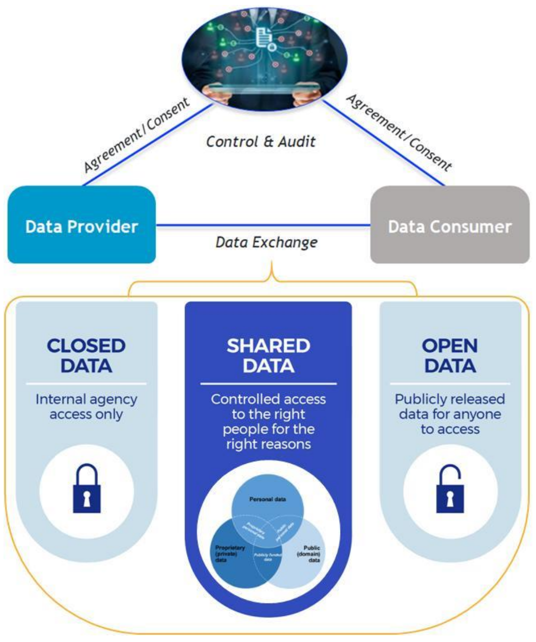
Considering the flows of interactions, the scenario depicted in Figure 4 shows how the Consumer connector accesses data from the Provider connector. Specifically, the Consumer receive a request to its P endpoint, it forwards the request to its internal Execution Core Container (A endpoint) which is in charge of establishing a secure communication with the Provider Execution Core Container to access the data. The Consumer's ECC receives the message from its Data APP, interacts with an external Identity Provider to retrieve the token of the Consumer and send the appropriate IDS message to the Provider's ECC using one of the provided communication channels (B endpoint). The Provider's ECC receives the message and validates the token against the Identity provider, then retrieves the actual data from its Data APP (F endpoint), and returns it to the Consumer's ECC who, finally, processes the response, applies the usage control policies and forward the data to the original requester. It is important to underline that every transaction is logged into the Clearing House (C endpoints).

3.3 Evolution and Improvements In PLATOON

Data sharing, supported by security and mutual trusted mechanisms provides enormous benefits, but in some contexts, we have to deal with different type of exchanged data.
Taking into account the traditional definitions of data spectrum we talk about in this context on a mix of proprietary data, open data and personal data.

[11] https://industrial-data-space.github.io/trusted-connector-documentation/docs/idscp2/

In the scenario of IDS connector, but in general scenario of data sharing, we have a *consumer party* and a *provider party* exchanging data according to an *agreement* (for example a contract). This agreement authorizes provider party to data provision to consumer party and authorizes the latter to process that data. Also, that agreement can refer to a *data usage policy* describing processing restriction obligations and duties.



In the case of personal data, the scenario is more complex, we have not only data provider and consumer, but also the *data subject*, and usage rules are user centric and consent based, so they can vary dynamically according to the preference modification performed by the data subject.
Therefore, the presence of personal data suggests to follow a user-centric approach of data sharing, which also involves aspects of compliancy, from a legislative point of view, and other aspects about technological and semantic interoperability and the use of standards.

The next sections will describe the adopted open-source solutions that have been extended and integrated in the PLATOON Data Connector to support, on one hand a data usage control according to shared agreement among the parties and on the other a user-centric approach to data management on providing data subjects with transparency and consent management tools, as well as data controllers and processors, with tools for processing data on the legal basis provided by the GDPR and on the basis of the preferences collected from the data subjects.

3.3.1 CaPe

CaPe provides an ICT suite for a consent-based, user-centric personal data management during the interaction among data subjects and public and private services as Data Controller and processors. It provides tools for lawful data sharing processes, with the ability to grant and withdraw consent to third parties for accessing own personal data.
It follows the MyData principles[12] to exploit the potential of personal data, and it facilitates its control and new business opportunities in compliance with the GDPR.
In this frame CaPe acts (Figure 5) as an intermediary and as a tool of communication between data subjects and controllers/processors, supporting the generation and management of dynamic consents.

[12] https://mydata.org/

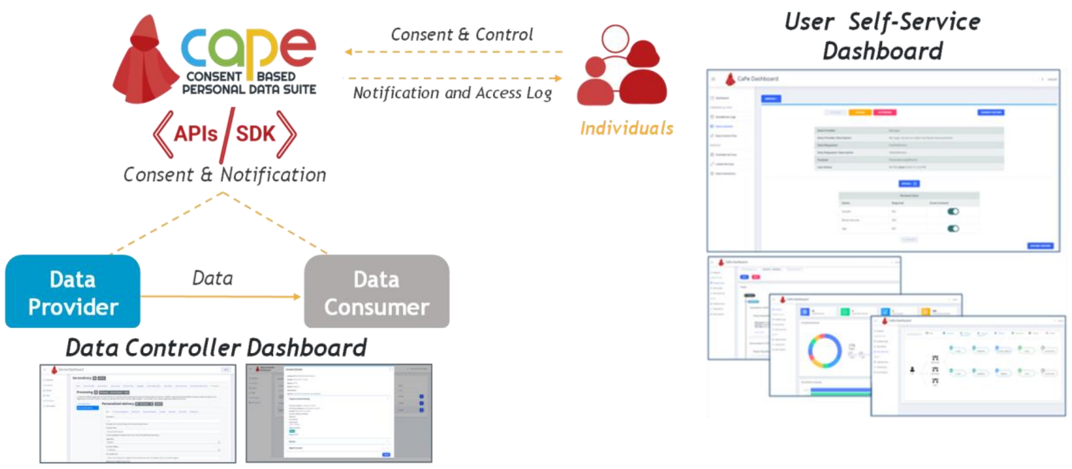

cape
CONSENT BASED PERSONAL DATA SUITE
APIs / SDK
Consent & Notification
Consent & Control
Notification and Access Log
Individuals
User Self-Service Dashboard
Data Provider
Data
Data Consumer
Data Controller Dashboard

Figure 5 - CaPe solution as intermediary between Data Controller/Processor and Data Subject

CaPe supports fine granularity in consent management and provides open and machine readable consent format to be processed in usage enforcement. It provides self service transparency tools for individuals by means they have the possibility at any time to manage their consent, receive notification and exercise data subject rights: objection, right to be forgotten, rectification and copy of data.
CaPe provides an API ecosystem to have consents status so that any changes are automatically available to be used with other systems or with partner organization.

CaPe suite is a web platform based on the microservices paradigm, in which several modules expose a set of APIs through an API Gateway, to be consumed by front-end applications and other external services. The following picture (Figure 6) illustrates the architecture of the CaPe Suite.

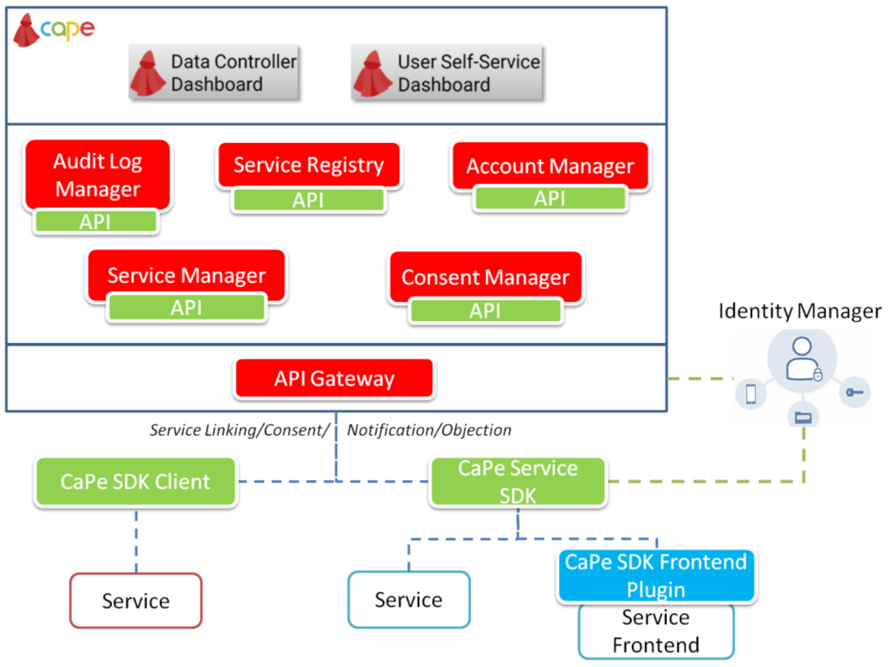


Figure 6- CaPe architecture

Each component of that modular architecture is involved to support the end to end process of consent management, from the formal definition of privacy rules disclaimer to the collection and management of data subject consents and related privacy enforcement.
In order to use all the functionalities of CaPe components, a workflow has been defined (Figure 7) and it consists of the following steps:

1. Service description and registration
2. Service Linking
3. Consent Request (for processing within a service or sharing among services)
4. Data Request, Notification and Activity Logs
5. Consent Management & User Data Usage Control

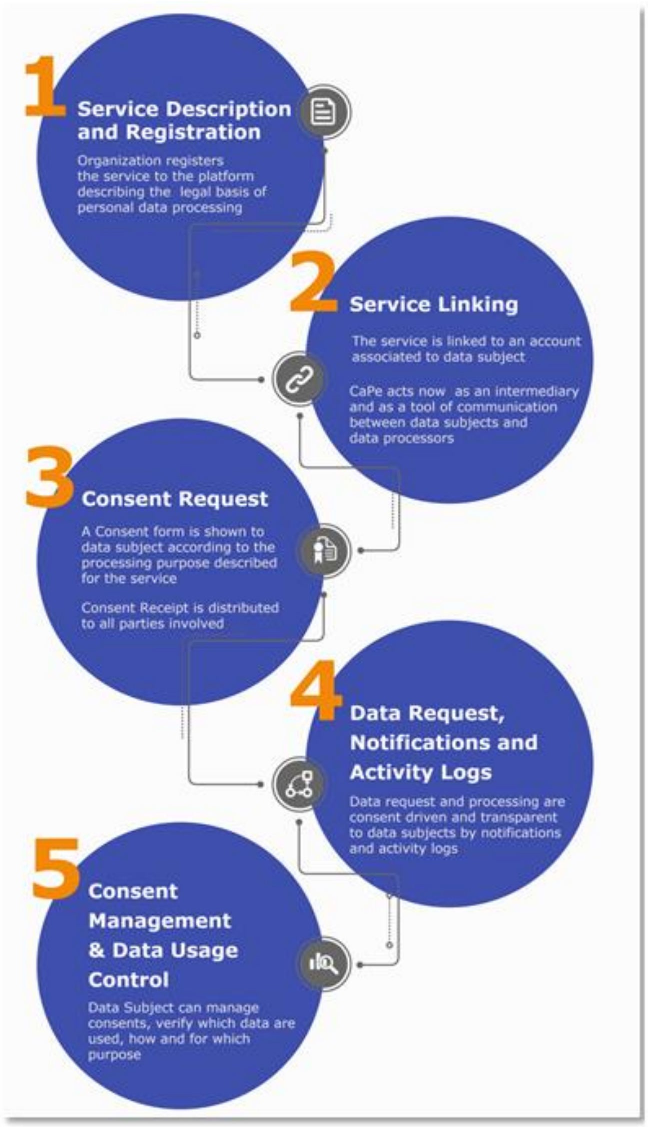

1
Service Description and Registration
Organization registers the service to the platform describing the legal basis of personal data processing
2
Service Linking
The service is linked to an account associated to data subject
CaPe acts now as an intermediary and as a tool of communication between data subjects and data processors
3
Consent Request
A Consent form is shown to data subject according to the processing purpose described for the service
Consent Receipt is distributed to all parties involved
4
Data Request, Notifications and Activity Logs
Data request and processing are consent driven and transparent to data subjects by notifications and activity logs
5
Consent Management & Data Usage Control
Data Subject can manage consents, verify which data are used, how and for which purpose

Figure 7- CaPe workflow for a user centric end-to-end consent management

With CaPe (Figure 8), through the Data Controller Dashboard an organization can model the legal basis for the processing of personal data describing in a standard format (taxonomies, service models…) the relevant information (i.e., purpose, processing, type of data and so on) in line with the related privacy policy. According to the derived model, CaPe automatically generates the consent form that can be shown to the data subject.

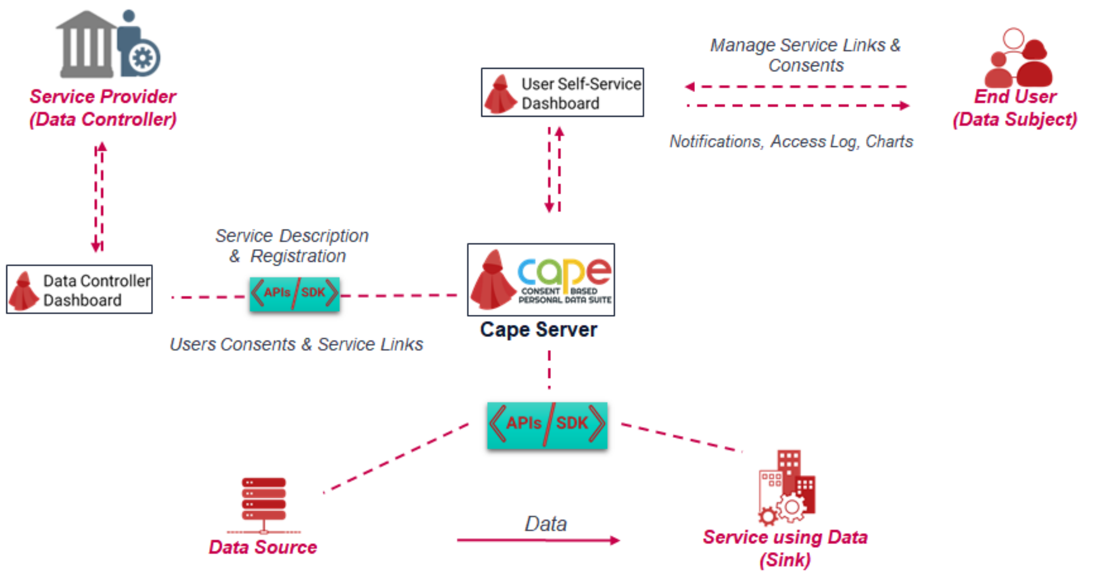

Service Provider (Data Controller)
User Self-Service Dashboard
Manage Service Links & Consents
Notifications, Access Log, Charts
End User (Data Subject)
Service Description & Registration
Data Controller Dashboard
APIs SDK
cape CONSENT BASED PERSONAL DATA SUITE
Cape Server
Users Consents & Service Links
APIs SDK
Data Source
Data
Service using Data (Sink)

Figure 8- CaPe core components in action

The two separated dashboards can let on one side, the Data Controller to view and manage all the consents collected, on the other, the Data Subject, through the User Self-Service Dashboard, to check which data is used, how and for what purpose and to manage the related consents.

3.3.2 Open-Source Data Usage Control

The Data Usage Control module has been developed by modifying the open-source IDS Dataspace Connector[13]. Specifically, version v5.0.1 of the Dataspace Connector has been used as the starting point for the development, extracting the Java packages required for this module. This code has been improved by adding the following new functionalities:

- REST services to get, upload and remove the Contract Agreements from the Contract Agreements storage. The format of these Contract Agreements is the one specified by the IDS Information Model[14]. These contracts will be used to apply the Data Usage Control enforcement.
- A REST service to apply the Data Usage Control enforcement on the input data according to the Contract Agreements related to the pair consumer-producer indicated as input parameters.
- A new policy is supported which indicates if the data contains Personal Data, in which case the module invokes CaPe for each data set belonging to a specific data subject. CaPe will filter out the content that is consented by each data subject. This new policy rule is described in section 5.2.8.

The following figure shows the architecture of this module.

[13] https://github.com/International-Data-Spaces-Association/DataspaceConnector
[14] https://international-data-spaces-association.github.io/InformationModel/docs/index.html#

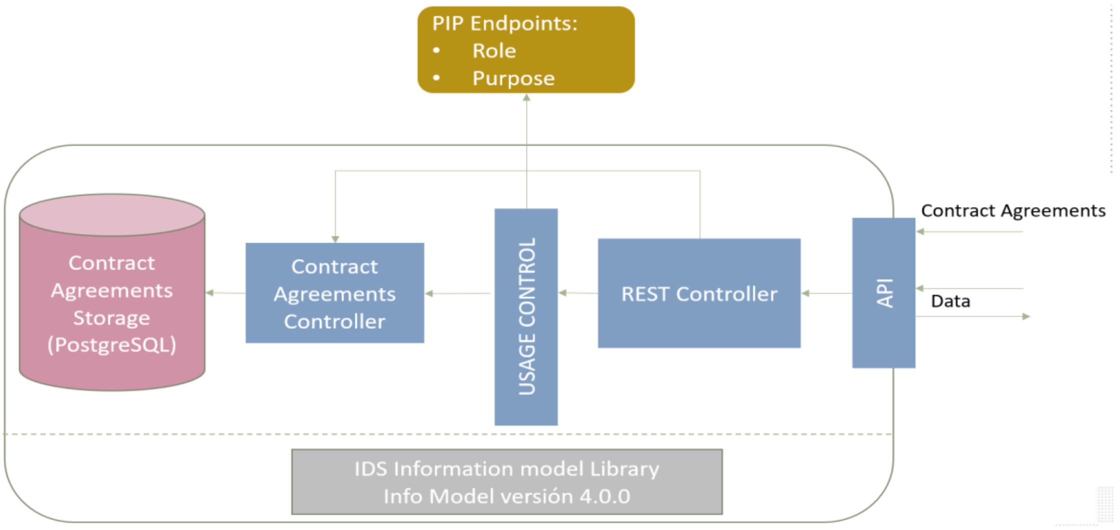


Figure 9 - Data Usage Control architecture

The Data Usage Control is composed of the following components:

- A PostgreSQL database, where the Constract Agreements are stored. This database contains the following tables:
    - **Contract_store:** the whole contract agreement content in JSON-LD format is stored in the "contract_as_string" column.
    - **Rule_store:** a contract may contain several rules. Each of these rules is stored in this table and the rule content in JSON-LD format is stored in the "rule_content" column.
    - **Access_store:** this table stores the number of times a consumer has accessed a specific target/data, in the "num_access" column. This table is used to apply the policy pattern "N Times Usage" detailed in section 5.2.7.

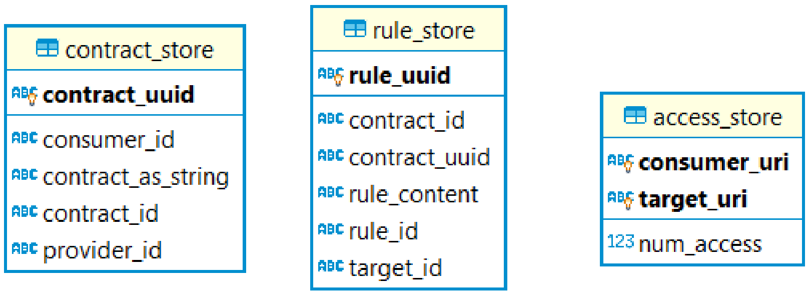


Figure 10 - Data Usage Control database schema

- The Contract Agreement Controller, which implements the REST services to get, update and remove the Contract Agreements in the database.
- The Usage Control module, which applies the usage control enforcement over the input data according to the rules specified in the corresponding Contract Agreements.
- The REST Controller.

On the other hand, PIP endpoints have been also developed to get the Role and Purpose of the data consumer. These PIP endpoints are only for testing purposes. The URL-s of these endpoints are specified in the Contract Agreements when the rule is referred to a Purpose/Role based permission/prohibition/obligation. Each consumer should implement its PIP endpoint, to inform about its Role or Purpose when consuming the data.

The steps to be taken to do enforcement are the following:

1. Once the consumer and provider connectors have negotiated and established a Contract Agreement, this Contract Agreement is stored in the Data Usage Control by invoking the corresponding REST service.
2. The usage control enforcement REST service is invoked before transferring the data from the Provider Connector to the Consumer Connector (parameter consuming=false), and before transferring the data from the Consumer Connector to the Data App (parameter consuming=true). This service will return the data according to the policies defined in the Contract Agreement.

The data Usage Control module supports usage policies written in the IDS Usage Control Language[15] based on ODRL[16]. The policy patterns supported by the Data Usage Control module are the following ones:

- Allow the Usage of the Data: provides data usage without any restrictions.
- Prohibit the Usage of the Data: prohibits data usage.
- Interval-restricted Data Usage: provides data usage within a specified time interval.
- Duration-restricted Data Usage: allows data usage for a specified time period.
- Role-restricted Data Usage.
- Purpose-restricted Data Usage Policy.
- Restricted Number of Usages: allows data usage for n times.
- Personal Data: filter out the contents of the data according to the data subject´s consents. To apply this rule, the Usage Control module interacts with CaPe.

Depending on the input parameter “consuming” passed to the REST service in charge of doing the usage control enforcement, the policy patterns to be verified are:

- Consuming=False. Providing data to another connector:
  - Allow the Usage of the Data
  - Prohibit the Usage of Data
  - Role-restricted Data Usage
  - Purpose-restricted Data Usage Contracts
  - Interval-restricted Data Usage
- Consuming=True. Consuming data:
  - Interval-restricted Data Usage
  - Duration-restricted Data Usage

[15] https://internationaldataspaces.org/wp-content/uploads/IDSA-Position-Paper-Usage-Control-in-the-IDS-V3.0.pdf
[16] https://www.w3.org/TR/odrl-model/

- Restricted Number of Usages

The model of these policies is detailed in section 5.2 via examples. All of them, except the Personal Data related policy, detailed in section 5.2.8, are policy patterns included in the IDS Information Model. When a policy of this type is present in a Contract Agreement, the input data must be in JSON format, to make it possible to parse the data belonging to each user and the field containing the user identification, which is indicated in the policy rule. E.g.:

```
[{"firstName":"MyName","lastName":"Mylastname", "address":"Myaddress",
"dateOfBirth":"130174", "mth_avg_cons_":"22kWh", "email": "userId1@domine1.com"},
{"firstName":"MyName2","lastName":"Mylastname2", "address":"Myaddress2",
"dateOfBirth":"130175", "mth_avg_cons_":"32kWh", "email": "userId2@domine1.com"}]
```

That is, it will be a JSON array where each element in the array refers to data of a particular data subject.

The following steps are carried out to enforce a policy rule of Personal Data type:

- The input data is parsed, and the following processing is carried out for each element in the JSON array:
  - The value of the property specified in the “idsc:JsonPath” property of the Personal Data rule (e.g.: "idsc:JsonPath":"$.email") is extracted from the data subset of a user (E.g.: “`userId1@domine1.com`”). This value is the user identification required to invoke Cape SDK usage enforcement API.
  - CaPe will return the data of each user filtered out according to the Consents given by the data subject. That is, if the user did not allow to use his address. CaPe will return all the data except the address. E.g.:
    ```
    [{"firstName":"MyName","lastName":"Mylastname",
    "dateOfBirth":"130174", "mth_avg_cons_":"22kWh", "email":
    "userId1@domine1.com"}
    ```
- The elements in the JSON array previously processed via CaPe are put together again in a JSON array and returned. E.g.:
  ```
  [{"firstName":"MyName","lastName":"Mylastname", "dateOfBirth":"130174",
  "mth_avg_cons_":"22kWh", "email": "userId1@domine1.com"},
  {"firstName":"MyName2","lastName":"Mylastname2", "dateOfBirth":"130175",
  "mth_avg_cons_":"32kWh", "email": "userId2@domine1.com"}]
  ```

3.3.3 CaPe and Data Usage Control Interaction

As the Usage Control concept is an extension of Access Control, the target of Data Usage Control (UC) component is to enforce restrictions on data usage and data processing, after access to data has been granted.

CaPe extends (Figure 11) the feature of UC component in the specific case of a request involving Personal Data, the component also enforces the compliancy with data usage/data processing Consents given by Data Owner, according to the GDPR regulation, by interacting with the Consent Manager component.

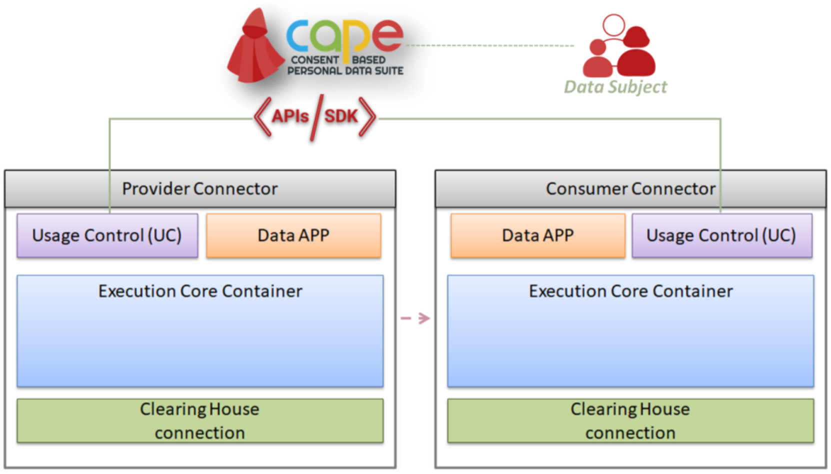
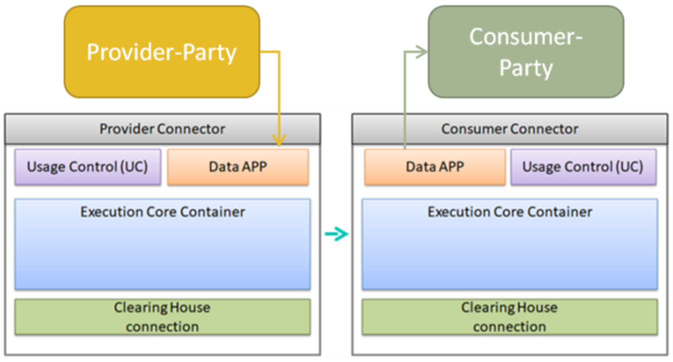


Figure 11 - CaPe and Usage control interaction in Data Connector

By interacting with CaPe, UC component enforces the compliancy, not only with the contract agreement between the parties (consumer and provider), but also with the privacy preferences included in the consent given by data subject.
In the following a scenario describes the interaction of a consumer and a provider party on data sharing where the usage control and CaPe interacts for data usage enforcement including personal data.



Two companies agreed on sharing data, about month average of domestic energy consumption, according to a declared contract agreement. In particular "***consumer-party***" requests data, by means a digital service of "***provider-party***".

In order to interact, the two services use IDS based ***PLATOON Data Connector*** for a secure and privacy exchange of structured data:

```
{"firstName":”","lastName":”","address":”“,"dateOfBirth":”“, “mth_avg_cons_”:””}
```

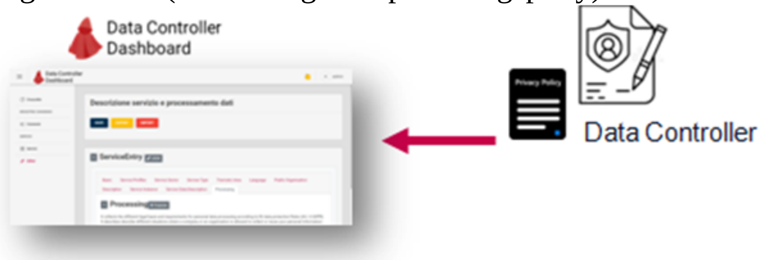
That dataset is identified in the contract agreement as:

```
‟http://w3id.org/engrd/connector/artifact/1”
```

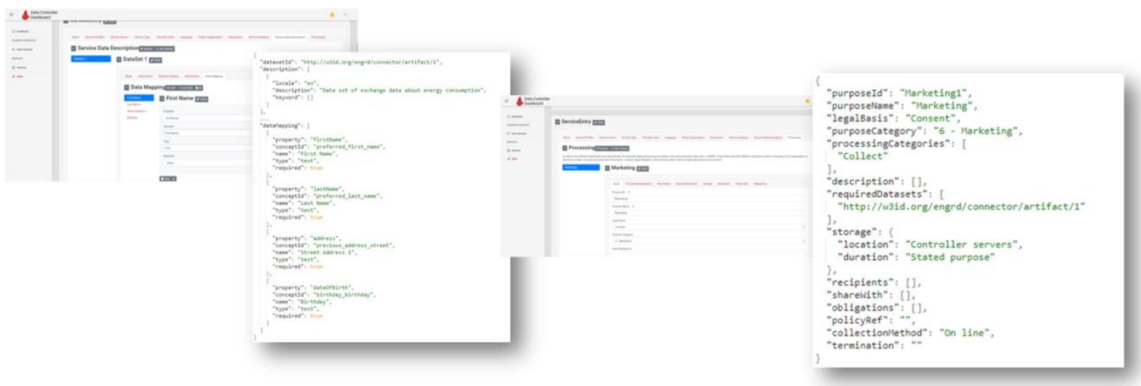
The contract agreement is formalized and stored by each party by means of the Usage Control application.
The agreed dataset will contain personal data, and its processing has to be performed in compliance with GDPR, in a legal basis and in particular with an informed consent given by

the data subject. In order to manage this aspect, the usage control component will interact with CaPe components (by means of APIs/SDK) responsible to collect and manage all consents. This is a preliminary and mandatory phase before the actual exchange of personal data between the two data connectors.

In order to interact with CaPe, to collect and manage all the consents for the specific purpose of data exchange the two interacting services (consuming and providing party) have to be described and registered in CaPe, describing the legal basis of personal data processing of the above identified dataset.



In particular the users acting, respectively, as data controller of the two services, describe, by means of CaPe Data Controller dashboard, the legal basis, the type of data processed and related purposes (Figure 12).



```
{
  "datasetId": "http://w3id.org/engrd/connector/artifact/1",
  "description": [
    {
      "locale": "en",
      "description": "Data set of exchange data about energy consumption",
      "keyword": []
    }
  ],
  ...
  "dataMapping": [
    {
      "property": "firstName",
      "conceptId": "preferred_first_name",
      "name": "First Name",
      "type": "text",
      "required": true
    },
    {
      "property": "lastName",
      "conceptId": "preferred_last_name",
      "name": "Last Name",
      "type": "text",
      "required": true
    },
    {
      "property": "address",
      "conceptId": "previous_address_street",
      "name": "Street Address 1",
      "type": "text",
      "required": true
    },
    {
      "property": "dateOfBirth",
      "conceptId": "birthday_birthday",
      "name": "Birthday",
      "type": "text",
      "required": true
    }
  ]
}
```

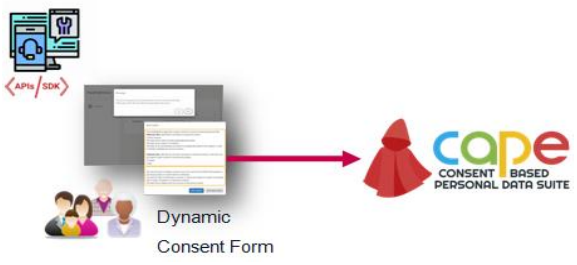
```
{
  "purposeId": "Marketing1",
  "purposeName": "Marketing",
  "legalBasis": "Consent",
  "purposeCategory": "6 - Marketing",
  "processingCategories": [
    "Collect"
  ],
  "description": [],
  "requiredDatasets": [
    "http://w3id.org/engrd/connector/artifact/1"
  ],
  "storage": {
    "location": "Controller servers",
    "duration": "Stated purpose"
  },
  "recipients": [],
  "shareWith": [],
  "obligations": [],
  "policyRef": "",
  "collectionMethod": "On line",
  "termination": ""
}
```

Figure 12 - Data processing description by means of CaPe Data Controller dashboard

Now two scenarios can arise:

1) The provider service has not yet collected the required data and before collecting it consent collection is mandatory.
2) Collection of data have already been performed but an updated consent request is needed for the "new" data processing agreement with the consumer-party service.

In both cases CaPe, starting from the above processing description, can generate the consent form to be used by services to collect data subject privacy preferences.



The consent is stored by CaPe server and forwarded to the involved parties (Consumer-party and Provider-Party).

Once consents related to the specified dataset and between the two parties for the specific purpose have been collected, the UC component can interact with CaPe APIs to retrieve the consents to be enforced during the data exchange between the two data connector

The following figure (Figure 13) summarizes the overall interactions of the involved parties in data exchange by using data connector and applying data usage.

The consumer-party service interacts (steps 1, 2) with PLATOON data connector (see details in section 4.4) to request the specific dataset. The consumer execution core container (ECC) interacts (step 3) with the dual provider ECC in order to enforce the request (steps 4, 5) to the provider party. The provider ECC then forwards the response (step7) to the consumer ECC, by verifying previously (if configured) the contract agreement constraints, and finally the requested dataset (steps 9, 10) is provided to the Consumer-party service after the data usage policy enforcement performed by the consumer data connector (step 8).

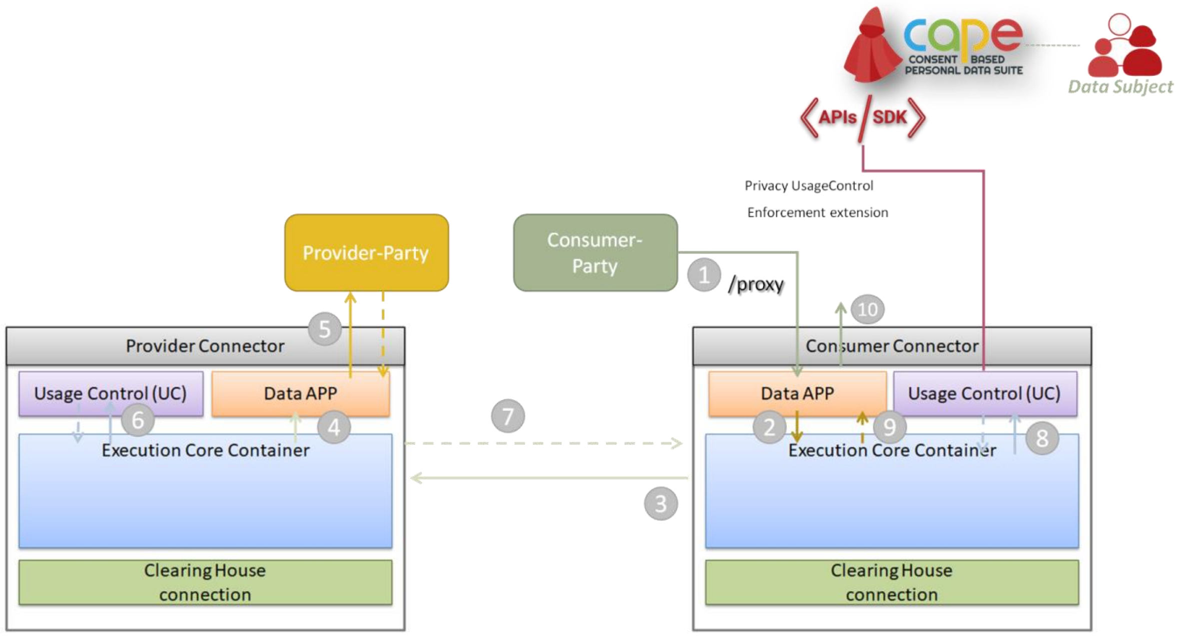
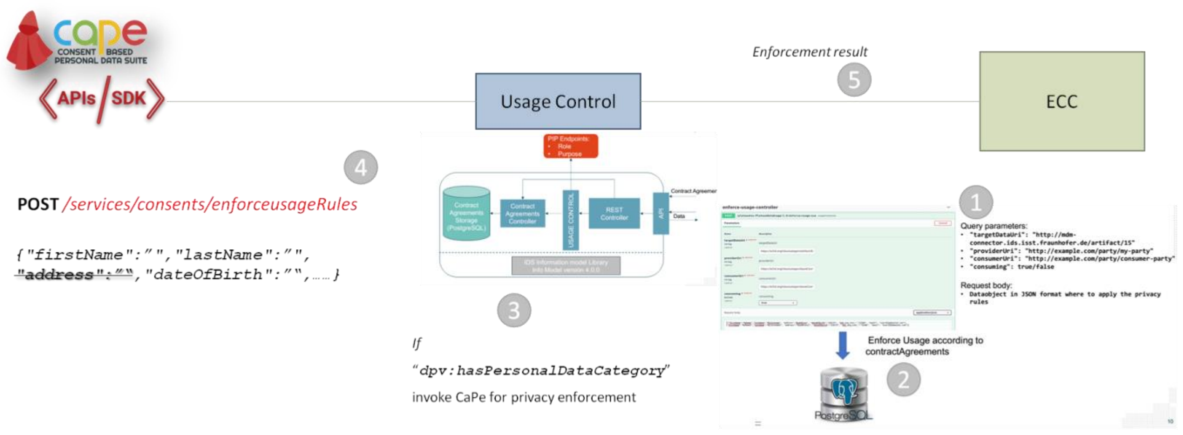


Figure 13 - Data connector supporting data exchange and usage enforcement based on contract agreement and privacy rules.

The data usage policy enforcement is performed by the consumer ECC by interacting with the UC component as depicted in the Figure 14.


cape
CONSENT BASED PERSONAL DATA SUITE
APIs
SDK
Usage Control
Enforcement result
5
ECC
4
3
1
2
Enforce Usage according to contractAgreements
PostgreSQL
If
"dpv:hasPersonalDataCategory"
invoke CaPe for privacy enforcement

POST */services/consents/enforceusageRules*

```
{"firstName":"","lastName":"",
~~"address":""~~,"dateOfBirth":"",……}
```

Query parameters:
- "targetDataUri": "http://mdm-connector.ids.isst.fraunhofer.de/artifact/15"
- "providerUri": "http://example.com/party/my-party"
- "consumerUri": "http://example.com/party/consumer-party"
- "consuming": true/false

Request body:
- Dataobject in JSON format where to apply the privacy rules

Figure 14- Interaction between UC component and CaPe APIs for policy enforcement

The ECC invokes Usage Control APIs to enforce the received dataset (step 1) to any data usage rules. The UC verifies the availability of any policy to enforce and associated to the specified dataset (step 2). If the retrieved policy provides references to personal data processing the UC invokes (step 3) CaPe APIs for privacy enforcement. CaPe performs the privacy usage rules according to the collected consents (step 4), for example removing any personal data not included in the consent. The enforced dataset is finally returned to ECC component to be forwarded to consumer service.

4 Technical Documentation

Following sections will describe the steps needed to correctly install and configure CaPe Suite, Data Usage Control and TRUE Connector.

4.1 CaPe

CaPe Suite is publicly available on CaPe GitHub repository[17]. Open a command prompt and execute the following command to clone the source code from the CaPe GitHub repository (Git software must be installed):

```
git clone https://github.com/OPSILab/Cape.git
```

CaPe installation will involve the deployment of different architectural components (Figure 15), each of which can be deployed either in an "on premise", "as a service" or mixed approach:

- **CaPe Server**
- **CaPe Dashboards**
- **CaPe SDK**

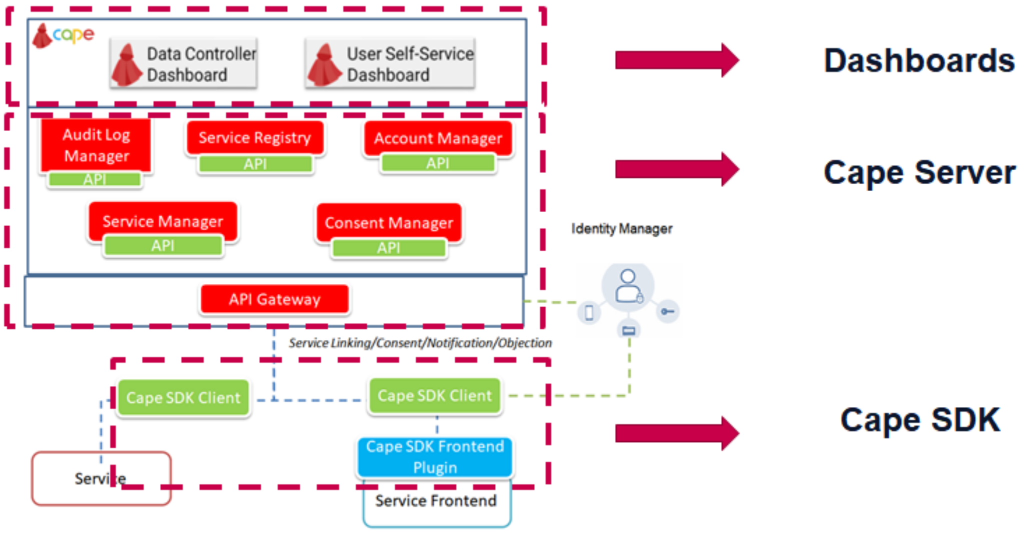


Figure 15 - CaPe main layers and components

For simplicity installation using Docker will be described. Additionally, CaPe must interact with any Identity Manager supporting OpenId Connect authorization framework (e.g. . Keycloak). For further installation mode and details see the CaPe ReadTheDocs[18] installation manual.

[17] https://github.com/OPSILab/Cape
[18] https://cape-suite.readthedocs.io/en/latest/install/

Although deployment with Docker can be accomplished by building and running each container individually from relative Docker files, it is recommended to use directly the provided ***docker-compose.yml*** files, which allow to start easily the whole stack by pulling already built images published on CaPe Suite Docker Hub repository[19].

Prerequisites

- **Docker (with docker-compose)**: version >= 20.10

4.1.1 CaPe Server

CaPe Server is the core backend of the CaPe platform. It implements and exposes all the main functionalities provided by CaPe, regarding the lifecycle management and storage of Service Descriptions, Service Linking, Consent Records and Auditing.
Its main components are:

- **Account Manager**: it manages the lifecycle of the CaPe Account, Account signing keys for Service Linking.
- **Service Manager**: it manages the Service Linking internal processes and Service Link Record storage.
- **Consent Manager**: it manages the Consent Records lifecycle, the generation of Consent Forms, etc.
- **Auditlog Manager**: it collects aggregated auditing statistics, triggered by incoming Event Logs (ServiceLink, Consent or Data Processing) regarding a specific Account.
- **Service Registry**: it collects the service descriptions and registrations (Signing keys and certificates).

Each of them is implemented as Spring Boot Java microservices, and will be deployed with a tightly coupled storage service (MongoDB 4.2+). They will be deployed as Docker containers (based on Tomcat Alpine image and each paired with a MongoDB container) running in the same Docker network (in order to resolve automatically their hostnames).
Docker Compose allows to run the whole stack and to link each component to the other under the same Docker network.

Deployment

Before launching docker-compose.yml, modify it to configure properly environment variables for OAuth2/OpendIdConnect authentication (see installation manual for further details).
Move into Cape/cape-server folder and run the docker-compose file with:

```
docker-compose up
```

[19] https://hub.docker.com/search?q=capesuite&type=image

The containers will be automatically started and attached to the created cape-network network.

After installing CaPe Server

Once CaPe Server has been deployed, in order to start using its APIs directly, through Dashboards or through CaPe SDK, a Data Operator Description must be created, by calling the relative API exposed by Service Manager (see its OpenApi specification in the relative service-manager/doc subfolder for further details).
Data Operator Description will describe the installed instance of CaPe Server. This Data Operator (intended according to the My Data specification) is not to be confused with the Service Provider's operator that will interact with CaPe by the means of Data Controller Dashboard and CaPe SDK/APIs.
In order to create the Operator Description on CaPe Server, issue a POST with a JSON body like the following one to the `/api/v2/dataOperatorDescriptions`

```
{
  "operatorId": "cape",
  "serviceProvider": {
    "businessId": "ENG",
    "name": "string",
    "address1": "string",
    "address2": "string",
    "postalcode": "string",
    "city": "string",
    "state": "string",
    "country": "string",
    "email": "string",
    "phone": "string",
    "jurisdiction": "string"
  },
  "operatorServiceDescriptionVersion": "string",
  "supportedProfiles": [
    "Consenting"
  ],
  "operatorUrls": {
    "domain": "string",
    "linkingUri": "http://localhost/cape-dashboard/serviceLinking",
    "linkingRedirectUri": "http://localhost/cape-dashboard/pages/services/linkedServices"
  },
  "createdOnDate": "2021-04-10T10:06:55.333Z",
  "createdByUserId": "string"
}
```

4.1.2 CaPe Service SDK

CaPe SDK is the software package of CaPe suite that will be provided to a Service Provider (Data Controller). He can manage several own services with CaPe, whose Service descriptions (Service Instance section) will have in common his Business Id (assigned by the running CaPe-Server instance).

It will expose a set of APIs (CaPe SDK APIs) providing integration of all services with the main functionalities provided by CaPe, regarding the lifecycle management and storage of:

- Service Signing keys (used during Service Linking phase).
- Service Link and Service Link Status Records (SLR, SSR) copies issued by CaPe after service linking phase and next status updates.
- Consent and Consent Status Records (CR, CSR) copies issued by CaPe after Consenting phase and next status updates.
- Proof of Possession (PoP) keys issued by CaPe during linking phase for Sink services.
- Authorisation Tokens issued by CaPe during consenting phase for Sink services.
- Management of Service Linking phases (start, SLR/SSR payloads sign and verification, status updates, etc).
- Management of Consenting Phase (Consent form generation, CR/CSR payloads verification, Giving consent and its status updates, etc).
- Management of Data Transfer request (Generation and signing of Data request for Sink; Request, Authorisation token and check of associated Consent status for Source, etc).

These functionalities will be used both by Data Controller dashboard and by the specific integrated Service Provider's Services.

CaPe SDK is composed by:

- **CaPe SDK Client**: Backend part exposing CaPe SDK APIs and implemented as Spring Boot Java service. It will be deployed with a tightly coupled storage service (MongoDB). It acts as Client by mirroring through its APIs the calls made directly to CaPe Server's API.
- **CaPe Angular Frontend Plugin**: Implemented as Angular library to be imported in a Angular project as module. Used in the integration of existing Angular Frontend Services with CaPe functionalities, in particular for generating the dynamic Consent Form and to perform Consenting workflow. In general, for the PLATOON use case, which involves interaction between True Connector and Usage Control, CaPe functionalities will be exploited via APIs.

CaPe SDK Client component will be deployed as Docker containers, based on Tomcat Alpine image (service-sdk) and paired with a MongoDB container (service-sdk-mongo). Default configuration will let run the container in the same Docker network of CaPe Server (in order to resolve automatically their hostnames).

Deployment

Before launching *docker-compose.yml*, modify it to configure properly environment variables, in particular for Oauth2/Oidc authentication (see installation manual for further details).
Move into **Cape/cape-sdk/service-sdk** folder and run the docker-compose file with:

```
docker-compose up
```

The containers will be automatically started and attached to the created *cape-network* network.

4.1.3 CaPe User Self-Service Dashboard

This section covers the steps needed to properly install CaPe User Self-Service Dashboard. It is an Angular portal based on Nebular[20] framework that can be installed in the following ways:

- Build as Angular distribution and deploy natively on a Web Server.
- Run as Docker containerized environment (recommended).

As done in previous sections, Docker installation will be described below.

Deployment

Docker Compose allows to run the Docker container by pulling the already built image from CaPe Docker Hub repository. In order to accomplish this:

- Move into **Cape/cape-dashboard** folder.
- Ensure you modified config.json file properly, as described in the section below.
- Run the docker-compose file with:

```
docker-compose up
```

The containers will be automatically started and attached to the created cape-network network.

Configuration

The provided docker-compose.yml file has also directives to mount the provided nginx.conf file, needed to correctly handle deep-linking on deployed Dashboard Angular application. It contains also the mount to the src/assets/config.json file, which allows configuring IdM endpoints for authentication and CaPe endpoints, accordingly to where CaPe Server and CaPe SDK have been deployed. See Deployment and Configuration section of User Dashboard installation manual[21] for further details.

4.1.4 CaPe Data Controller Dashboard

This section covers the steps needed to properly install CaPe Data Controller Dashboard. It is an Angular Web portal based on Nebular[22] framework. As done in previous sections, Docker installation will be described below.

Deployment

[20] https://akveo.github.io/nebular/
[21] https://cape-suite.readthedocs.io/en/latest/install/install-cape-user-dashboard/#deployment-and-configuration
[22] https://github.com/akveo/nebular

Docker Compose allows to run the Docker container by pulling the already built image from CaPe Docker Hub repository. In order to accomplish this:

- Move into **Cape/cape-service-editor** folder.
- Ensure you modified config.json file properly, as described in the section below.
- Run the docker-compose file with:

```
docker-compose up
```

The containers will be automatically started and attached to the created *cape-network* network.

4.2 Data Usage Control

Data Usage Control will be publicly available in GitHub repository.
The first step to install it is to download the code from GitHub issuing the `git clone` command (Git software must be installed):

Data Usage Control installation will involve the deployment of its different architectural components:

- The data Usage Control component.
- The PostgreSQL database where the Contract Agreements are stored.
- The PIP endpoints for testing purposes.

The installation will be carried out using Docker. The ***docker-compose.yml*** file is provided in the GitHub repository. Additionally, it must interact with an Identity Manager supporting OpenId Connect authorization framework (e.g.:. Keycloak).

Prerequisites

- **Docker (with docker-compose)**: version >= 20.10

Deployment

Before launching ***docker-compose.yml***, the following files should be modified:

- Modify ***docker-compose.yml***, file to configure properly the following properties:
    - Set the available ports and then modify the PostgreSQL port accordingly in the ***datausage.env*** file.
    - Set the environment variable UC_IDM_ISSUER_URI with the JWT Issuer Uri of installed IdM (e.g. https://IDM_HOST/auth/realms/platoon) This endpoint will be used to verify token issued for the Oauth2/OpendIdConnect client application `usage_control` registered in the `platoon` realm. Change IDM_HOST with the real hostname where IdM (e.g. Keycloak) has been deployed.

- Modify the ***postgres.env*** file to the set a more secure password for accessing the PostgreSQL database, and then modify the password accordingly in the ***datausage.env*** file.
- Modify the file ***datausage/etc/platoon_datausage_cape.properties*** setting the proper value for the ***cape.auth.client.secret*** property. This value will be provided by the CaPe system manager.

Move into the folder where the ***docker-compose.yml*** file is located and run the docker-compose file with the following command:

```
docker-compose up
```

The containers will be automatically started.

Open the https://HOST_NAME/platoontec/PlatoonDataUsage/1.0/swagger-ui.html page in a web browser and the available REST services of the Usage Control module will be shown.
Currently, the following web page installed in Tecnalia is publicly available: https://platoon.tecnalia.com/platoontec/PlatoonDataUsage/1.0/swagger-ui.html .

4.3 TRUE Connector

This section covers the step needed to build, configure, and start the TRUE Connector. As already reported in section 3.2, the tool is mainly composed of two different components: the Execution Core Container and the trivial Data APP.
The source code of each component is publicly available in PLATOON's GitHub organization repository[23]. Specifically, the following repositories are provided:

- Execution Core Container: the source code of the ECC[24]
- Data APP: source code of the trivial Data APP[25]
- IDS Multipart Message Processor: additional package needed to build the ECC[26]
- WebSocket Message Streamer: additional package needed to build the ECC[27]
- TRUE Connector: the integrated repository used to start the connector[28]

The following installation and deployment guide takes advantage of the dockerized version of the TRUE Connector. Please, refer to the official guide[29] in order to build and start the tool using a different approach. It is important to underline that the repository provides a working

[23] https://github.com/PLATOONProject
[24] https://github.com/PLATOONProject/market4.0-execution_core_container_business_logic/tree/platoon_uc
[25] https://github.com/PLATOONProject/market4.0-data_app_test_BE/tree/platoon_uc
[26] https://github.com/PLATOONProject/market4.0-ids_multipart_message_processor
[27] https://github.com/PLATOONProject/market4.0-websocket_message_streamer
[28] https://github.com/PLATOONProject/true-connector/tree/platoon_uc
[29] https://github.com/Engineering-Research-and-Development/true-connector-execution_core_container#developer-guide-section

example of two connectors following the architecture depicted into Figure 4, as a matter of fact the provided docker-compose.yml[30] file contains two ECCs and two trivial Data APPs containers representing, respectively, the Consumer and the Provider connectors.

To exploit the TRUE Connector functionalities, open a command prompt and execute the following command to clone its repository:

```
git clone https://github.com/PLATOONProject/true-connector.git
```

Enter the newly created *true-connector* directory and checkout the PLATOON's branch:

```
git checkout -b platoon_uc origin/platoon_uc
```

Prerequisites

- **Docker (with docker-compose)**: version >= 20.10

Configuration

The configuration of the tool can be managed by editing the *.env[31]* file located in the top repository directory. This file contains all the needed environmental variables used by the Consumer and Provider connectors to work properly, for instance, the data app endpoints (PROVIDER_DATA_APP_ENDPOINT and CONSUMER_DATA_APP_ENDPOINT) or the content type that should be used for the communications among each Data APP and the specific ECC (PROVIDER_MULTIPART_EDGE or CONSUMER_MULTIPART_EDGE) or between the two ECCs (MULTIPART_ECC). The possible content type values alternatives are *mixed* for Multipart/mixed, *form* for Multipart/form-data and *http-header* for Multipart/http-header.
Moreover, through the *.env* file it is also possible to enable or disable the different data exchange methods used by the ECC, that, as already reported in section 3.2, are REST over HTTP/HTTPS, Web Socket over HTTPS (WSS), and IDSCP2. In particular:

- **REST endpoints** are enabled if WS_EDGE=false and WS_ECC=false
- **IDSCP2** is enabled if IDSCP2=true and WS_ECC = false. For WS_EDGE=true (use websocket on the edge, false for REST on the edge)
- **WSS** is enabled if WS_EDGE=true and WS_ECC=true and IDSCP2=false for configuration which uses web socket on the edge and between connectors.

Furthermore, advanced configurations can be managed by the *application-docker.properties* files located in the different folders of the repository. For instance, the *application-*

[30] https://github.com/PLATOONProject/true-connector/blob/platoon_uc/docker-compose.yml
[31] https://github.com/PLATOONProject/true-connector/blob/platoon_uc/.env

docker.properties[32] that refers to the Consumer's ECC is located within the *ecc_resources_consumer* folder. Through such files is possible, for instance, to enable/disable the Usage Control or the Clearing House modifying, respectively, the *application.isEnabledUsageControl* and *application.isEnabledClearingHouse* properties. Please, refer to the official documentation for further details.

Deployment

At the time this document is written, to successfully start the containers the images need to be built manually following the guidelines provided in the repository[33]. Once each docker image is successfully built, move to the TRUE Connector top folder, and execute the following command:

```
docker-compose up
```

The containers will be automatically started.

4.4 Integration with PLATOON Data Connector

PLATOON Data Connector represent the integrated component that takes advantage of the functionalities provided by the TRUE Connector, Data Usage Control and CaPe, if the data exchange involves personal data.

A possible approach for the integration with PLATOON Data Connector envisages the implementation of a specific Data APP that needs to be placed on top of the Execution Core Container provided by the TRUE Connector. As already explained in section 3.2 and in section 4.3, the provided implementation of the Data APP represents a trivial example to demonstrate the functionalities offered by the connector itself and the data exchange different methods. On the other hand, the other components described in this deliverable can be used as is without further intervention.

To implement such Data APP, a developer could start from the trivial Data APP source code and implement the business logic needed to access the legacy data source.

A different approach consists in the implementation, from scratch, of a solution that directly uses the ECC endpoints defined in section 5.1.3. It is important to underline that, by using this approach, the developer needs to follow IDS Messages specifications to interact with the ECC.

[32] https://github.com/PLATOONProject/true-connector/blob/platoon_uc/ecc_resources_consumer/application-docker.properties

[33] https://github.com/PLATOONProject/true-connector/blob/platoon_uc/doc/docker-installation.md#alternative-build-docker-images-from-sources

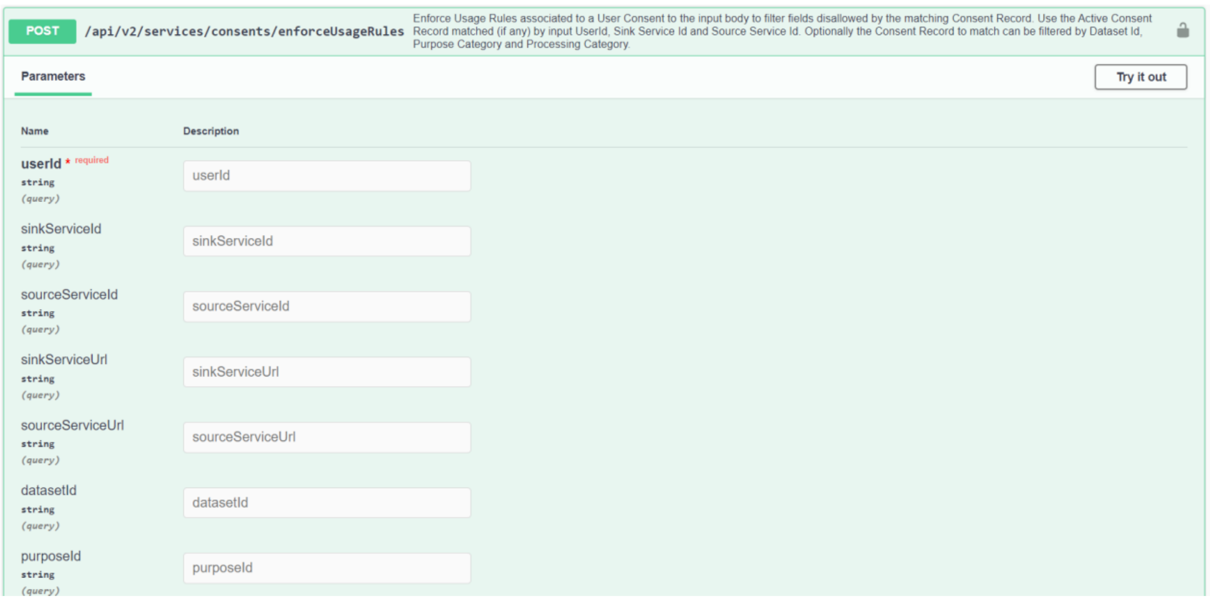
5 Appendix 1: API and Policy Models

5.1 API Documentation

5.1.1 CaPe

CaPe Suite with its main components, namely CaPe Server and CaPe Service SDK, will expose a set of APIs implementing CaPe functionalities.

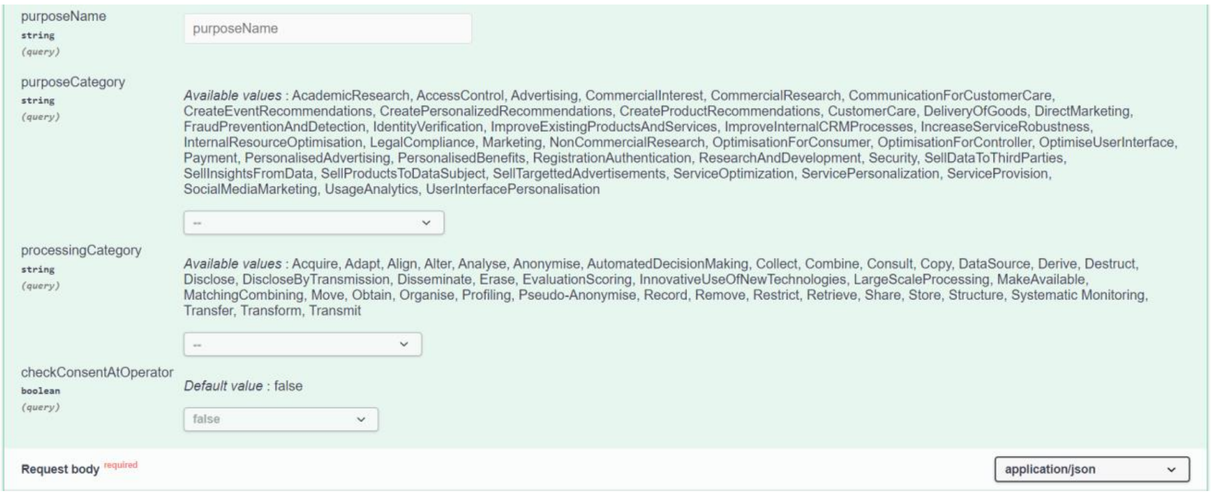
In particular CaPe Service SDK will expose the set of APIs to be used both by the Data Controller Dashboard and directly by Data Controller/Service Provider services that will be integrated in the CaPe workflow. These APIs will interact with the ones exposed by CaPe Server to perform functionalities requested at SDK endpoint.
Further details with OpenAPI specifications of CaPe APIs can be found in relative */doc* folders of each component.
This section will focus on the SDK usage enforcement API used by Data Usage Control during the data flows described in the section 3.3.3. A Swagger-UI screenshot is reported below (Figure 16), in order to give an overview of the API parameters.



POST /api/v2/services/consents/enforceUsageRules Enforce Usage Rules associated to a User Consent to the input body to filter fields disallowed by the matching Consent Record. Use the Active Consent Record matched (if any) by input UserId, Sink Service Id and Source Service Id. Optionally the Consent Record to match can be filtered by Dataset Id, Purpose Category and Processing Category.

Parameters

Name	Description
userId * required string (query)	userId
sinkServiceId string (query)	sinkServiceId
sourceServiceId string (query)	sourceServiceId
sinkServiceUrl string (query)	sinkServiceUrl
sourceServiceUrl string (query)	sourceServiceUrl
datasetId string (query)	datasetId
purposeId string (query)	purposeId



purposeName
string
(query)

purposeName

purposeCategory
string
(query)

Available values : AcademicResearch, AccessControl, Advertising, CommercialInterest, CommercialResearch, CommunicationForCustomerCare, CreateEventRecommendations, CreatePersonalizedRecommendations, CreateProductRecommendations, CustomerCare, DeliveryOfGoods, DirectMarketing, FraudPreventionAndDetection, IdentityVerification, ImproveExistingProductsAndServices, ImproveInternalCRMProcesses, IncreaseServiceRobustness, InternalResourceOptimisation, LegalCompliance, Marketing, NonCommercialResearch, OptimisationForConsumer, OptimisationForController, OptimiseUserInterface, Payment, PersonalisedAdvertising, PersonalisedBenefits, RegistrationAuthentication, ResearchAndDevelopment, Security, SellDataToThirdParties, SellInsightsFromData, SellProductsToDataSubject, SellTargettedAdvertisements, ServiceOptimization, ServicePersonalization, ServiceProvision, SocialMediaMarketing, UsageAnalytics, UserInterfacePersonalisation

--

processingCategory
string
(query)

Available values : Acquire, Adapt, Align, Alter, Analyse, Anonymise, AutomatedDecisionMaking, Collect, Combine, Consult, Copy, DataSource, Derive, Destruct, Disclose, DiscloseByTransmission, Disseminate, Erase, EvaluationScoring, InnovativeUseOfNewTechnologies, LargeScaleProcessing, MakeAvailable, MatchingCombining, Move, Obtain, Organise, Profiling, Pseudo-Anonymise, Record, Remove, Restrict, Retrieve, Share, Store, Structure, Systematic Monitoring, Transfer, Transform, Transmit

--

checkConsentAtOperator
boolean
(query)

Default value : false

false

Request body required

application/json

Figure 16 - CaPe SDK: Usage Rules Enforcement API

Mandatory query parameters to match a Consent Record:

- ***userId*:** Id of the Data Subject. This value is extracted from initial Artifact Request originated at True Connector end (Consumer), forwarded to Usage Control and then finally to CaPe in this parameter.
- ***sinkServiceId / sinkServiceUrl:*** Id defined in CaPe Service Description or URL (matching with the Consumer URI in the True Connector) of service requesting data from Provider.
- ***sourceServiceId / sourceService:*** Id defined in CaPe Service Description or URL (matching with the Consumer URI in the True Connector) of service requesting data from Provider.

Optionally query parameters to filter the Consent Records:

- ***datasetId:*** Id of the dataset contained in the Resource Set of Consent Record.
- ***purposeId / purposeName:*** Id and name of the processing basis purpose in the Usage Rules of Consent Record.
- **purposeCategory:** Purpose Category of the processing basis in the Usage Rules of Consent Record, selected among available values.
- **processingCategory:** Processing category indicating type of operations made on exchanged personal data.
- ***checkConsentAtOperator*:** Whether Consents must be searched among local copies delivered by CaPe Server to the SDK storage in the consenting phase, or directly at CaPe Server end.

The Enforcement API indeed enforces Usage Rules against a Json payload in the input body, by fetching a specific consent usage rule, if any, or in case it returns not found, meaning that a consent does not exist. In detail, it will check if there is any active Consent for the input UserId and pair of Services Ids or URLs (Consumer and Provider parties), indicating that the user given a consent for that specific dataset. If the Consent is present, the input payload is filtered against the data concept fields present in the Dataset of the Consent Resource Set.

Taking as input payload the example given in the previous section:

```
{
      "firstName": "John",
      "lastName": "Doe",
      "address": "somewhere",
      "mth_avg_cons_": "10KWH"
}
```

Considering a Consent for which the User deselected from proposed dataset in the shown consent form the optional field “address”. The resulting payload will be the following:

```
{
      "firstName": "John",
      "lastName": "Doe",
      “mth_avg_cons_": "10KWH"
}
```

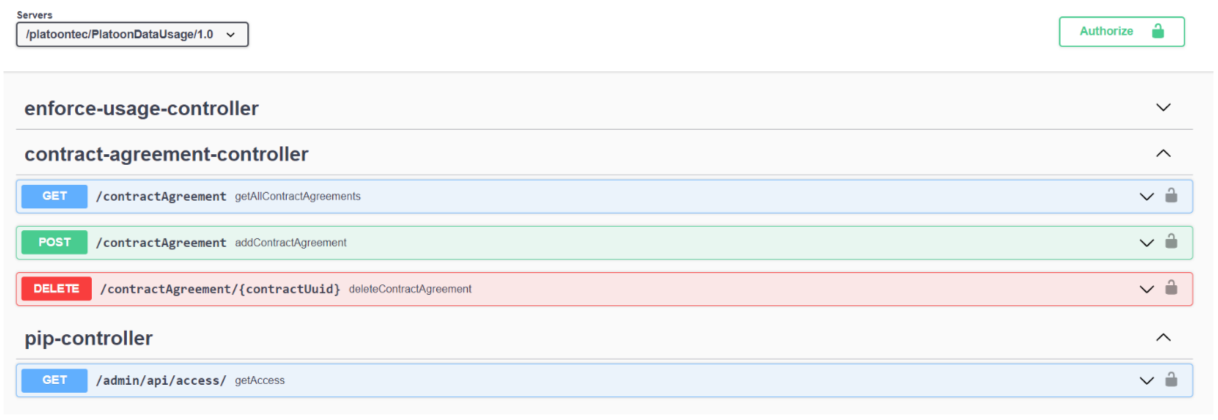
Since the user in this way denied the consent to transfer and use that specific personal data from Provider to Consumer party.

5.1.2 Data Usage Control

Data Usage Control exposes a set of APIs which implement the functionalities of the Data UC.

Swagger-UI screenshots are reported below, in order to give an overview of each of the API parameters.

The following figure shows a summary of the API-s offered by the UC module.



Servers
/platoontec/PlatoonDataUsage/1.0

Authorize

enforce-usage-controller

contract-agreement-controller

GET /contractAgreement getAllContractAgreements

POST /contractAgreement addContractAgreement

DELETE /contractAgreement/{contractUuid} deleteContractAgreement

pip-controller

GET /admin/api/access/ getAccess

Figure 17 – Data Usage Control: summary of offered API-s

Enforce usage rules

This service applies the usage control enforcement on the input data according to the rules included in the Contract Agreement. The Contract Agreement applied will be the one that corresponds according to the input parameters provided to the service.

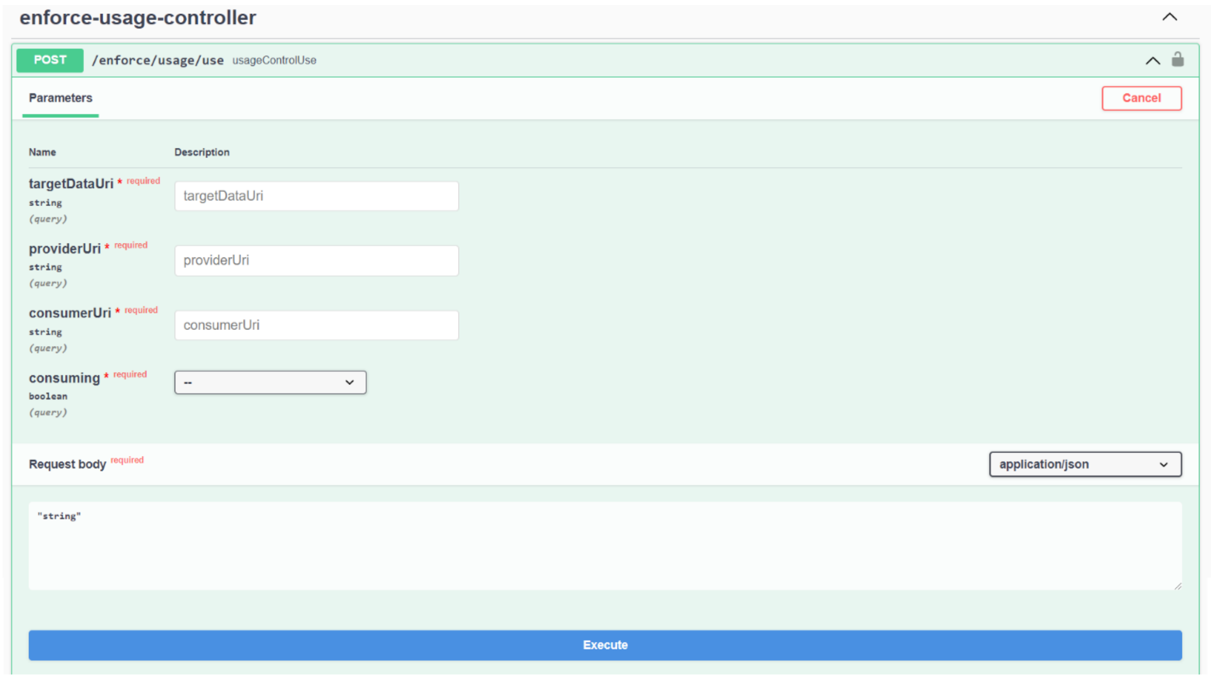

enforce-usage-controller
POST /enforce/usage/use usageControlUse
Parameters
Cancel
Name
Description
targetDataUri * required string (query)
providerUri * required string (query)
consumerUri * required string (query)
consuming * required boolean (query)
Request body required
application/json
"string"
Execute

Figure 18 – Data Usage Control: Usage Rules Enforcement API

The service URL is `POST /platoontec/PlatoonDataUsage/1.0/enforce/usage/use`

Mandatory parameters to do this service are:

- ***targetDataUri***: Id of the dataset. E.g.: https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-37aa08b2aedd .
- ***providerUri:*** Id of the Provider Connector. E.g.: https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-669219dde4ea .
- ***consumeruri:*** Id of the Consumer Connector. E.g.: https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-669219dde4ea .
- ***consuming:*** true/false. Boolean value which informs if the data is being provided by a Provider Connector (false) or if the data is being consumed at the Consumer Connector (true).

Request Body is also mandatory, and it will contain the data on which the usage rules defined in the Contract Agreement must be applied. When the Contract Agreement includes a Personal Data policy, then the data must be in JSON format, to make it possible to parse the data belonging to each user and the field containing the user identification, which is indicated in the policy rule. E.g.:

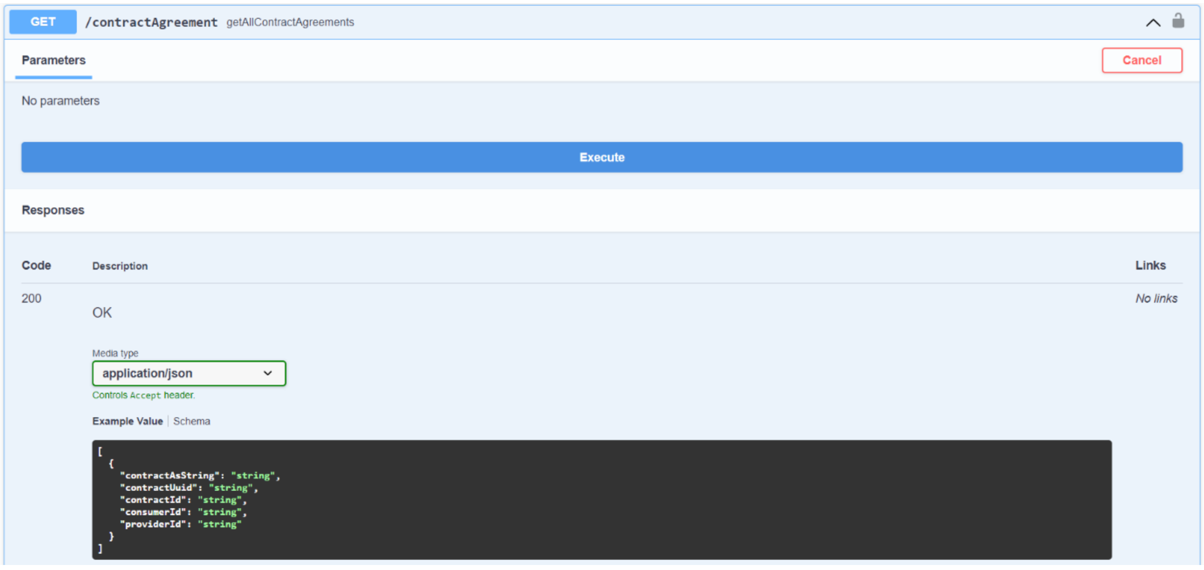
```
[{"firstName":"MyName","lastName":"Mylastname", "address":"Myaddress",
"dateOfBirth":"130174", "mth_avg_cons_":"22kWh", "email": "userId1@domine1.com"},
{"firstName":"MyName2","lastName":"Mylastname2", "address":"Myaddress2",
"dateOfBirth":"130175", "mth_avg_cons_":"32kWh", "email": "userId2@domine1.com"}]
```

The following possible responses can be returned by this service:

- HTTP 200 OK: the response body will contain the data filtered according to the rules defined in the Contract Agreement and applied according to the input parameters provided.
- HTTP 403 Forbidden: this code will be returned when after applying the rules defined in the Contract Agreement, it concludes that data usage is not allowed.
- HTTP 400 Bad Request: this code will be returned when no valid Contract Agreements are found for the consumer/provider/target values provided as input parameters to the service.

Get All Contract Agreements

This service returns the Contract Agreements stored in the Data Usage Control database.



No parameters

Code	Description	Links
200	OK	*No links*

Media type: application/json

```
[
  {
    "contractAsString": "string",
    "contractUuid": "string",
    "contractId": "string",
    "consumerId": "string",
    "providerId": "string"
  }
]
```

Figure 19 – Data Usage Control: Get Contract Agreements API

The service URL is `GET /platoontec/PlatoonDataUsage/1.0/contractAgreement`

No input parameters are required.

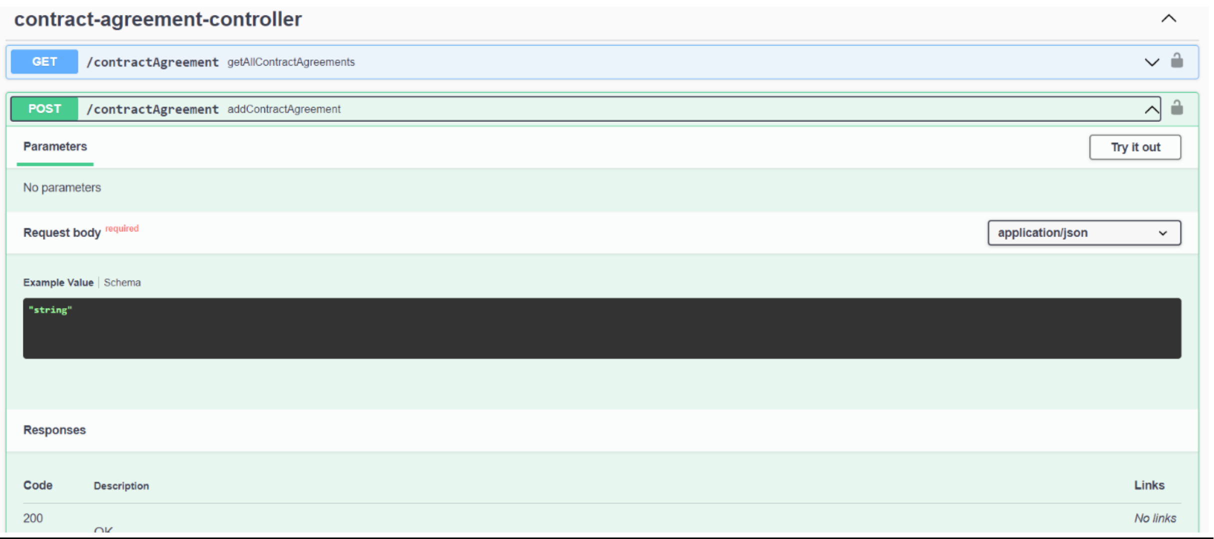
The following possible responses can be returned by this service:

- HTTP 200 OK: the response body will contain a JSON array with the Contract Agreements stored in the database. E.g.:

```
[
  {
    "contractAsString":
"{\"@context\":{\"ids\":\"https://w3id.org/idsa/core/\"},\"@type\":\"ids:Contra
ctAgreement\",\"@id\":\"https://w3id.org/idsa/autogen/contractAgreement/contrac
tAgree4\",\"ids:permission\":[{\"@type\":\"ids:Permission\",\"@id\":\"https://w
3id.org/idsa/autogen/permission/perm4\",\"ids:target\":{\"@id\":\"https://w3id.
org/idsa/autogen/artifact/4\"},\"ids:title\":[{\"@value\":\"Example Usage
Policy N Times
Usage\",\"@type\":\"http://www.w3.org/2001/XMLSchema#string\"}],\"ids:descripti
on\":[{\"@value\":\"n-times-
usage\",\"@type\":\"http://www.w3.org/2001/XMLSchema#string\"}],\"ids:action\":
[{\"@id\":\"idsc:USE\"}],\"ids:constraint\":[{\"@type\":\"ids:Constraint\",\"@i
d\":\"https://w3id.org/idsa/autogen/constraint/2030a8f2-f03d-4af9-bce5-
b9222e129dce\",\"ids:rightOperand\":{\"@value\":\"5\",\"@type\":\"xsd:double\"}
,\"ids:operator\":{\"@id\":\"idsc:LTEQ\"},\"ids:leftOperand\":{\"@id\":\"idsc:C
OUNT\"},\"ids:pipEndpoint\":{\"@id\":\"http://localhost:8080/platoontec/Platoon
DataUsage/1.0/admin/api/access/\"}}]}],\"ids:provider\":{\"@id\":\"https://w3id
.org/idsa/autogen/baseConnector/provider1\"},\"ids:consumer\":{\"@id\":\"https:
//w3id.org/idsa/autogen/baseConnector/consumer1\"},\"ids:contractDate\":{\"@val
ue\":\"2021-02-
18T10:15:21.137Z\",\"@type\":\"http://www.w3.org/2001/XMLSchema#dateTimeStamp\"
},\"ids:contractStart\":{\"@value\":\"2021-02-
18T10:15:21.137Z\",\"@type\":\"http://www.w3.org/2001/XMLSchema#dateTimeStamp\"
}}",
    "contractUuid": "b7b88b8f-0f36-4b60-9479-c530204176cf",
    "contractId":
"https://w3id.org/idsa/autogen/contractAgreement/contractAgree4",
    "consumerId": "https://w3id.org/idsa/autogen/baseConnector/consumer1",
    "providerId": "https://w3id.org/idsa/autogen/baseConnector/provider1"
  },
…
]
```

Insert/Update a Contract Agreement

This service is used to insert or update a Contract Agreement in the Data Usage Control database.



contract-agreement-controller

GET /contractAgreement getAllContractAgreements

POST /contractAgreement addContractAgreement

Parameters

Try it out

No parameters

Request body required

application/json

Example Value | Schema

```
"string"
```

Responses

Code	Description	Links
200	OK	No links

Figure 20 – Data Usage Control: Insert/Update Contract Agreement API

The service URL is `POST /platoontec/PlatoonDataUsage/1.0/ContractAgreement`

It does not require any input parameters.

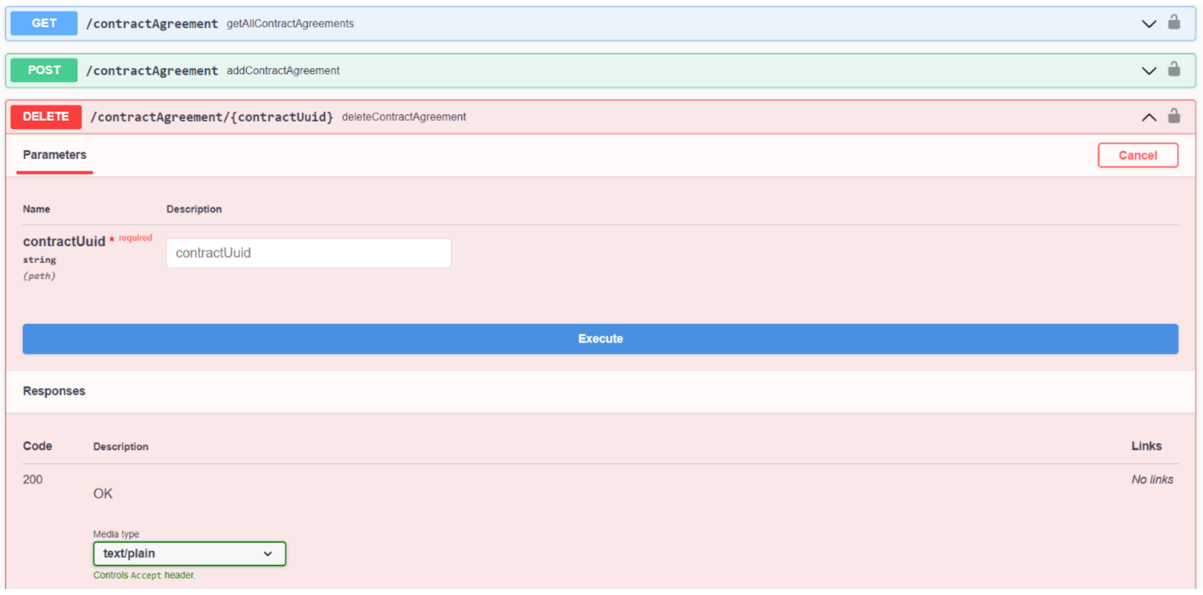
Request Body is mandatory, and it will contain the Contract Agreement to be inserted or updated in JSON-LD format. E.g.:

```
{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
      "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:ContractAgreement",
  "@id" : "https://w3id.org/idsa/autogen/contractAgreement/52272512-dcbd-4b15-
8f1f-f409327a4a9a",
  "ids:permission" : [ {
    "@type" : "ids:Permission",
    "@id" : "https://w3id.org/idsa/autogen/permission/59b0a20a-11bd-4276-8341-
af40c8960e98",
    "ids:target" : {
      "@id" : "https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-
37aa08b2aedd"
    },
    "ids:title" : [ {
      "@value" : "Example Usage Policy",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:description" : [ {
      "@value" : "provide-access",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:action" : [ {
      "@id" : "idsc:USE"
    } ]
  } ],
  "ids:provider" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:consumer" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:contractDate" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractStart" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractEnd" : {
    "@value" : "2022-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  }
}
```

If everything works properly, it will create or update the Contract Agreement and return a HTTP 200 OK.

Delete a Contract Agreement

This service removes the specified Contract Agreement in the Data Usage Control database.


GET /contractAgreement getAllContractAgreements

POST /contractAgreement addContractAgreement

DELETE /contractAgreement/{contractUuid} deleteContractAgreement

Parameters

Cancel

Name	Description
contractUuid * required string (path)	contractUuid

Execute

Responses

Code	Description	Links
200	OK Media type text/plain Controls Accept header.	No links

Figure 21 – Data Usage Control: Delete Contract Agreement API

The service URL is `DELETE /platoontec/PlatoonDataUsage/1.0/contractAgreement/{contractUuid}`

Mandatory path parameter is the contract unique identifier `{contractUuid}` of the Contract Agreement to remove from the database, e.g.: `DELETE /platoontec/PlatoonDataUsage/1.0/contractAgreement/b7b88b8f-0f36-4b60-9479-c530204176cf`

If everything works properly, it will remove the specified Contract Agreement and return a HTTP 200 OK.

PIP endpoint to get number of times a data has been accessed at Consumer side

This service returns the number of times the specified data has been accessed by the specified Consumer Connector. This PIP endpoint is used to apply the "N Times Usage" rule described in section 5.2.7, and its URL will appear in the rule definition.

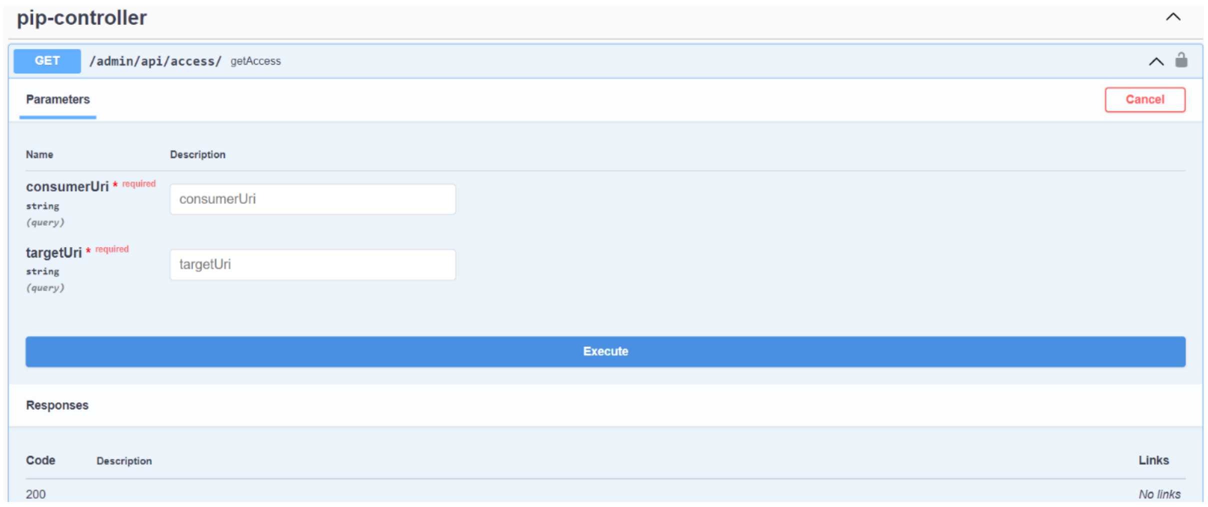


Figure 22 – Data Usage Control: PIP N Times access API

The service URL is `GET /platoontec/PlatoonDataUsage/1.0/admin/api/access/`

Mandatory parameters are:

- ***targetUri***: Id of the dataset. E.g.: https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-37aa08b2aedd .
- ***consumeruri:*** Id of the Consumer Connector. E.g.: https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-669219dde4ea .

If everything works properly, it will return a number that specifies the number of times the specified data has been accessed by the specified Consumer Connector.

5.1.3 TRUE Connector

This section describes the set of APIs and endpoints provided by the TRUE Connector. As stated in section 3.2, the TRUE Connector is composed of mainly two components, the Execution Core Container and the trivial Data APP, each of which can be configured to act as a Consumer or a Provider.
Both components expose the following utility endpoint that returns the business logic version of the tool, and it is used to verify if the components itself are running successfully:

- `/about/version`

Acting as a Consumer, the ECC exposes the following set of camel[34] endpoints:

[34] https://camel.apache.org/

- `/incoming-data-app/multipartMessageBodyBinary:` to receive data with binary body form the Data APP (mixed content type)
- `/incoming-data-app/multipartMessageBodyFormData:` to receive data with form-data body from the Data APP (form content type)
- `/incoming-data-app/multipartMessageHttpHeader:` represented with IDS-Message using http headers and payload in body from Data APP (http-headers content type)

And the Data APP exposes the following endpoint:

- `/proxy:` to receive data incoming request, and based on received request, forward request to Execution Core Connector

Acting as a Provider, the ECC exposes the following camel endpoint:

- `/data:` to receive data (IDS Message) from a sender connector

And the Data APP exposes the following:

- `/data:` to provide the real data to the ECC

The following snippets provides an example of exchanges using the `/proxy` endpoint of the Consumer's Data APP using the *mixed* content type configuration:

```
curl --location --request POST
'https://CONSUMER_DATA_APP_HOST:CONSUMER_DATA_APP_PORT/proxy' \
--header 'Content-Type: application/json' \
--data-raw '{
    "multipart": "mixed",
    "Forward-To": "https://ECC_PROVIDER_HOST:ECC_PROVIDER_PORT/data",
    "message": {
        "@context": {
            "ids": "https://w3id.org/idsa/core/"
        },
        "@type": "ids:ArtifactRequestMessage",
        "@id":
"https://w3id.org/idsa/autogen/artifactRequestMessage/76481a41-8117-4c79-
bdf4-9903ef8f825a",
        "ids:issued": {
            "@value": "2021-04-14T16:43:27.051+01:00",
            "@type": "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
        },
        "ids:modelVersion": "4.0.0",
        "ids:issuerConnector": {
            "@id": "https://w3id.org/idsa/autogen/baseConnector/7b934432-
a85e-41c5-9f65-669219dde4eb"
        },
        "ids:requestedArtifact": {
            "@id": "https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-
42e1-a1c3-37aa08b2aedd"
        }
    },
    "payload": {}
}'
```

It is important to underline that the *Forward-To* attribute in the request needs to be set to the `/data` camel endpoint of the Provider's ECC. The `ids:requestedArtifact` represents the resource requested by the Consumer.

Internally, the Consumer's Data APP forward the request to its ECC who is in charge of contacting the Provider's ECC to access the requested data, validating and controlling each step, as described in section 3.2. Depending on the request's `multipart` attribute, the Consumer's Data APP will forward the request to one of the already described endpoints provided by the ECC acting as a Consumer. The Provider's ECC, through its `/data` endpoint, will finally use the Provider's Data APP `/data` endpoint to retrieve the data. An example of the payload received by the Consumer's Data APP is provided:

```
--qOtSEYP5N-bpFaSHwFj4cQi1YRzbDH
Content-Disposition: form-data; name="header"
Content-Length: 643

{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/"
  },
  "@type" : "ids:ArtifactResponseMessage",
  "@id" : "https://w3id.org/idsa/autogen/artifactResponseMessage/9385ca94-
3eda-4ef6-9467-79cc0a04a3ee",
  "ids:modelVersion" : "4.0.0",
  "ids:issued" : {
    "@value" : "2021-08-27T09:15:42.289Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:issuerConnector" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-
41c5-9f65-669219dde4ea"
  },
  "ids:correlationMessage" : {
    "@id" : "https://w3id.org/idsa/autogen/artifactRequestMessage/76481a41-
8117-4c79-bdf4-9903ef8f825a"
  }
}
--qOtSEYP5N-bpFaSHwFj4cQi1YRzbDH
Content-Disposition: form-data; name="payload"
Content-Length: 588


[{"firstName":"Bill","lastName":"Doe","mth_avg_cons":"33
kWh","address":"592 My Street, My Country",
"checksum":"ABC123 2021/08/27 11:15:42","dateOfBirth":"2021/08/27
11:15:42","email":"user2@mail.it"},
{"firstName":"John","lastName":"Doe","mth_avg_cons":"64 kWh","address":"591
My Street, My Country",
"checksum":"ABC123 2021/08/27 11:15:42","dateOfBirth":"2021/08/27
11:15:42","email":"user1@mail.it"},
{"firstName":"Mario","lastName":"Rossi","mth_avg_cons":"72
kWh","address":"via Roma",
"checksum":"ABC123 2021/08/27 11:15:42","dateOfBirth":"2021/08/27
11:15:42","email":"user3@mail.it"}]
--qOtSEYP5N-bpFaSHwFj4cQi1YRzbDH--
```

5.2 Policy Model

This section contains the policy patterns supported by the Data Usage Control module. These policy patterns are presented via examples.
All of them, except the Personal Data related policy, are policy patterns included in the IDS Information Model.

5.2.1 Provide Access

```
{
   "@context":{
      "ids":"https://w3id.org/idsa/core/",
      "idsc":"https://w3id.org/idsa/code/"
   },
   "@type":"ids:ContractAgreement",
   "@id":"https://w3id.org/idsa/autogen/contractAgreement/52272512-dcbd-4b15-8f1f-
f409327a4a9a",
   "ids:permission":[
      {
         "@type":"ids:Permission",
         "@id":"https://w3id.org/idsa/autogen/permission/59b0a20a-11bd-4276-8341-
af40c8960e98",
         "ids:target":{
            "@id":"https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-
37aa08b2aedd"
         },
         "ids:title":[
            {
               "@value":"Example Usage Policy",
               "@type":"http://www.w3.org/2001/XMLSchema#string"
            }
         ],
         "ids:description":[
            {
               "@value":"provide-access",
               "@type":"http://www.w3.org/2001/XMLSchema#string"
            }
         ],
         "ids:action":[
            {
               "@id":"idsc:USE"
            }
         ]
      }
   ],
   "ids:provider":{
      "@id":"https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
   },
   "ids:consumer":{
      "@id":"https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
   },
   "ids:contractDate":{
      "@value":"2021-02-18T10:15:21.137Z",
      "@type":"http://www.w3.org/2001/XMLSchema#dateTimeStamp"
   },
   "ids:contractStart":{
      "@value":"2021-02-18T10:15:21.137Z",
      "@type":"http://www.w3.org/2001/XMLSchema#dateTimeStamp"
   },
   "ids:contractEnd":{
      "@value":"2022-02-18T10:15:21.137Z",
      "@type":"http://www.w3.org/2001/XMLSchema#dateTimeStamp"
   }
}
```

5.2.2 Prohibit Access

```
{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
      "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:ContractAgreement",
  "@id" : "https://w3id.org/idsa/autogen/contractAgreement/52272512-dcbd-4b15-
8f1f-f409327a4a9a",
  "ids:prohibition" : [ {
    "@type" : "ids:Prohibition",
    "@id" : "https://w3id.org/idsa/autogen/permission/59b0a20a-11bd-4276-8341-
af40c8960e99",
    "ids:target" : {
      "@id" : "https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-
37aa08b2aedd"
    },
    "ids:title" : [ {
      "@value" : "Example Usage Policy",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:description" : [ {
      "@value" : "prohibit-access",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:action" : [ {
      "@id" : "idsc:USE"
    } ]
  } ],
  "ids:provider" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:consumer" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:contractDate" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractStart" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  }
}
```

5.2.3 Usage During Interval

```
{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
       "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:ContractAgreement",
  "@id" : "https://w3id.org/idsa/autogen/contractAgreement/52272512-dcbd-4b15-
8f1f-f409327a4a9a",
  "ids:permission" : [ {
    "@type" : "ids:Permission",
    "@id" : "https://w3id.org/idsa/autogen/permission/59b0a20a-11bd-4276-8341-
af40c8960e98",
    "ids:target" : {
      "@id" : "https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-
37aa08b2aedd"
    },
    "ids:title" : [ {
      "@value" : "Example Usage Policy",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:description" : [ {
      "@value" : "provide-access",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:action" : [ {
      "@id" : "idsc:USE"
    } ],
    "ids:constraint" : [ {
      "@type" : "ids:Constraint",
      "@id" : "https://w3id.org/idsa/autogen/constraint/0b7c4ca7-1f9e-4e30-8fa1-
7551700c1980",
      "ids:rightOperand" : {
        "@value" : "2020-07-11T00:00:00Z",
        "@type" : "xsd:dateTimeStamp"
      },
      "ids:operator" : {
        "@id" : "idsc:AFTER"
      },
      "ids:leftOperand" : {
        "@id" : "idsc:POLICY_EVALUATION_TIME"
      }
    }, {
      "@type" : "ids:Constraint",
      "@id" : "https://w3id.org/idsa/autogen/constraint/9f2e0197-2ad9-442b-806b-
5bb4951a2943",
      "ids:rightOperand" : {
        "@value" : "2021-07-11T00:00:00Z",
        "@type" : "xsd:dateTimeStamp"
      },
      "ids:operator" : {
        "@id" : "idsc:BEFORE"
      },
      "ids:leftOperand" : {
        "@id" : "idsc:POLICY_EVALUATION_TIME"
      }
    } ]
  } ],
  "ids:provider" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:consumer" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:contractDate" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractStart" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractEnd" : {
    "@value" : "2022-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  }
}
```

5.2.4 Duration Usage

```
{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
      "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:ContractAgreement",
  "@id" : "https://w3id.org/idsa/autogen/contractAgreement/52272512-dcbd-4b15-
8f1f-f409327a4a9a",
  "ids:permission" : [ {
    "@type" : "ids:Permission",
    "@id" : "https://w3id.org/idsa/autogen/permission/59b0a20a-11bd-4276-8341-
af40c8960e98",
    "ids:target" : {
      "@id" : "https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-
37aa08b2aedd"
    },
    "ids:title" : [ {
      "@value" : "Example Usage Policy",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:description" : [ {
      "@value" : "provide-access",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:action" : [ {
      "@id" : "idsc:USE"
    } ],
    "ids:constraint" : [ {
      "@type" : "ids:Constraint",
      "@id" : "https://w3id.org/idsa/autogen/constraint/a5aa4243-432f-4360-aff4-
c95da99eb266",
      "ids:rightOperand" : {
        "@value" : "PT4H",
        "@type" : "xsd:duration"
      },
      "ids:operator" : {
        "@id" : "idsc:SHORTER_EQ"
      },
      "ids:leftOperand" : {
        "@id" : "idsc:ELAPSED_TIME"
      }
    } ]
  } ],
  "ids:provider" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:consumer" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
```

```
  },
  "ids:contractDate" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractStart" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractEnd" : {
    "@value" : "2022-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  }
}
```

5.2.5 Role Based Usage

```
{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
      "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:ContractAgreement",
  "@id" : "https://w3id.org/idsa/autogen/contractAgreement/52272512-dcbd-4b15-
8f1f-f409327a4a9a",
  "ids:permission" : [ {
    "@type" : "ids:Permission",
    "@id" : "https://w3id.org/idsa/autogen/permission/59b0a20a-11bd-4276-8341-
af40c8960e98",
    "ids:target" : {
      "@id" : "https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-
37aa08b2aedd"
    },
    "ids:title" : [ {
      "@value" : "Example Usage Policy",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:description" : [ {
      "@value" : "provide-access",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:action" : [ {
      "@id" : "idsc:USE"
    } ],
     "ids:constraint" : [ {
      "@type" : "ids:Constraint",
      "@id" : "https://w3id.org/idsa/autogen/constraint/constraint6",
      "ids:rightOperandReference":{
            "@id": "http://example.com/ids-role:riskManager"
          },
       "ids:operator" : {
         "@id" : "idsc:HAS_MEMBERSHIP"
```

```
        },
        "ids:leftOperand" : {
          "@id" : "idsc:USER"
        },
        "ids:pipEndpoint" : {
          "@id": "http://pip:8085/DataUsage/Pip/1.0/admin/api/role/"
        }
      } ]
    } ],
  "ids:provider" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:consumer" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:contractDate" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractStart" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractEnd" : {
    "@value" : "2022-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  }
}
```

5.2.6 Purpose Based Usage

```
{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
      "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:ContractAgreement",
  "@id" : "https://w3id.org/idsa/autogen/contractAgreement/52272512-dcbd-4b15-
8f1f-f409327a4a9a",
  "ids:permission" : [ {
    "@type" : "ids:Permission",
    "@id" : "https://w3id.org/idsa/autogen/permission/59b0a20a-11bd-4276-8341-
af40c8960e98",
    "ids:target" : {
      "@id" : "https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-
37aa08b2aedd"
    },
    "ids:title" : [ {
      "@value" : "Example Usage Policy",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:description" : [ {
      "@value" : "provide-access",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:action" : [ {
      "@id" : "idsc:USE"
    } ],
    "ids:constraint" : [ {
      "@type" : "ids:Constraint",
      "@id" : "https://w3id.org/idsa/autogen/constraint/constraint7",
      "ids:rightOperandReference":{
        "@id": "http://example.com/ids-purpose:Marketing"
      },
      "ids:operator" : {
        "@id" :  "idsc:SAME_AS"
      },
      "ids:leftOperand" : {
        "@id" : "idsc:PURPOSE"
      },
      "ids:pipEndpoint" : {
        "@id": "http://pip:8085/DataUsage/Pip/1.0/admin/api/purpose/"
      }
    } ]
  } ],
  "ids:provider" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:consumer" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:contractDate" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractStart" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractEnd" : {
    "@value" : "2022-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  }
}
```

5.2.7 N Times Usage

It will be required to access the PIP endpoint specified in the rule to get the number of times the Consumer has accessed the data. See section 5.1.2 for a description of this endpoint.

```
{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
      "idsc" : "https://w3id.org/idsa/code/"
  },
  "@type" : "ids:ContractAgreement",
  "@id" : "https://w3id.org/idsa/autogen/contractAgreement/52272512-dcbd-4b15-
8f1f-f409327a4a9a",
  "ids:permission" : [ {
    "@type" : "ids:Permission",
    "@id" : "https://w3id.org/idsa/autogen/permission/59b0a20a-11bd-4276-8341-
af40c8960e98",
    "ids:target" : {
      "@id" : "https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-
37aa08b2aedd"
    },
    "ids:title" : [ {
      "@value" : "Example Usage Policy",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:description" : [ {
      "@value" : "provide-access",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:action" : [ {
      "@id" : "idsc:USE"
    } ],
    "ids:constraint" : [ {
      "@type" : "ids:Constraint",
      "@id" : "https://w3id.org/idsa/autogen/constraint/2030a8f2-f03d-4af9-bce5-
b9222e129dce",
      "ids:rightOperand" : {
        "@value" : "5",
        "@type" : "xsd:double"
      },
      "ids:operator" : {
        "@id" : "idsc:LTEQ"
      },
      "ids:leftOperand" : {
        "@id" : "idsc:COUNT"
      },
      "ids:pipEndpoint" : {
        "@id" :
"http://localhost:8080/platoontec/PlatoonDataUsage/1.0/admin/api/access/"
      }
    } ]
  } ],
  "ids:provider" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:consumer" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:contractDate" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractStart" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractEnd" : {
    "@value" : "2022-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  }
}
```

5.2.8 Personal Data

This new policy pattern has been introduced in PLATOON project to make it possible to announce that the data set contains personal data, so that data will be processed on the legal basis provided by the GDPR and on the basis of the preferences collected from the data subjects when they gave their Consents via CaPe.
A new property called "dpv:hasPersonalDataCategory" has been introduced to indicate that there is personal data in the data set being transferred. This property belongs to the Data Privacy Vocabulary (DPV)[35].
Moreover, the following properties are used:

- **"ids:preDuty":** this property informs that before providing the data, this data has to undergo a modification considering the following sub-properties:
  - **"idsc:JsonPath":** the value of this property indicates the field in the dataset that contains the user identification. This value will be used when invoking the CaPe service to filter out the data of a specific user according to the consents given by him.
  - **"ids:pipEndpoint":** the value of this property contains the URL of the PIP endpoint to which it has to be invoked to obtain the purpose of the Consumer Connector. The purpose returned by the PIP endpoint will be used when invoking the CaPe service to filter out the data of a specific user according to the purpose of the Consumer Connector.

```
{
   "@context":{
      "ids":"https://w3id.org/idsa/core/",
      "idsc":"https://w3id.org/idsa/code/",
      "dpv":"http://www.w3.org/ns/dpv#"
   },
```

[35] https://w3c.github.io/dpv/dpv/#

```
   "@type":"ids:ContractAgreement",
   "@id":"https://w3id.org/idsa/autogen/contractAgreement/52272512-dcbd-4b15-8f1f-
f409327a4a9a",
   "ids:permission":[
      {
         "@type":"ids:Permission",
         "@id":"https://w3id.org/idsa/autogen/permission/59b0a20a-11bd-4276-8341-
af40c8960e98",
         "ids:target":{
            "@id":"https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-
37aa08b2aedd"
         },
         "dpv:hasPersonalDataCategory":"dpv:DerivedPersonalData",
         "ids:title":[
            {
               "@value":"Example Usage Policy",
               "@type":"http://www.w3.org/2001/XMLSchema#string"
            }
         ],
         "ids:description":[
            {
               "@value":"provide-access",
               "@type":"http://www.w3.org/2001/XMLSchema#string"
            }
         ],
         "ids:action":[
            {
               "@id":"idsc:USE"
            }
         ],
         "ids:preDuty":[
            {
               "@type":"ids:Duty",
               "@id":"https://w3id.org/idsa/autogen/duty/preduty1",
               "ids:action":[
                  {
                     "@id":"idsc:MODIFY"
                  }
               ],
               "idsc:JsonPath":"$.email",
               "ids:constraint":[
                  {
                     "@type":"ids:Constraint",
                     "@id":"https://w3id.org/idsa/autogen/constraint/constraint8",
                     "ids:operator":{
                        "@id":"idsc:SAME_AS"
                     },
                     "ids:leftOperand":{
                        "@id":"idsc:PURPOSE"
                     },
                     "ids:pipEndpoint":{
```

```
"@id":"http://pip:8085/DataUsage/Pip/1.0/admin/api/purposeName/"
                     }
                  }
               ]
            }
         ]
      }
   ],
   "ids:provider":{
      "@id":"https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
   },
   "ids:consumer":{
      "@id":"https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
   },
   "ids:contractDate":{
      "@value":"2021-02-18T10:15:21.137Z",
      "@type":"http://www.w3.org/2001/XMLSchema#dateTimeStamp"
   },
   "ids:contractStart":{
      "@value":"2021-02-18T10:15:21.137Z",
      "@type":"http://www.w3.org/2001/XMLSchema#dateTimeStamp"
   },
   "ids:contractEnd":{
      "@value":"2022-02-18T10:15:21.137Z",
      "@type":"http://www.w3.org/2001/XMLSchema#dateTimeStamp"
   }
}
```

5.2.9 Combination of Patterns

The previous patterns can be combined in one Contract Agreement. For example, the following Contract Agreement includes the **Usage During Interval**, **N Times Usage** and **Personal Data** rules. That is, data will be returned if the following conditions are met:

- The current time is inside the time interval specified in the **Usage During Interval** rule.
- The Consumer has accessed less or equal times than the number specified in the **N Times Usage** rule.
- The input data will be filtered so that only the fields to which the data subject has given permission to use in the specified Consents via CaPe will be returned.

```
{
  "@context" : {
    "ids" : "https://w3id.org/idsa/core/",
       "idsc" : "https://w3id.org/idsa/code/",
    "dpv": "http://www.w3.org/ns/dpv#"
  },
  "@type" : "ids:ContractAgreement",
  "@id" : "https://w3id.org/idsa/autogen/contractAgreement/52272512-dcbd-4b15-
8f1f-f409327a4a9a",
  "ids:permission" : [ {
    "@type" : "ids:Permission",
    "@id" : "https://w3id.org/idsa/autogen/permission/59b0a20a-11bd-4276-8341-
af40c8960e98",
    "ids:target" : {
      "@id" : "https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-
37aa08b2aedd"
    },
    "ids:title" : [ {
      "@value" : "Example Usage Policy",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:description" : [ {
      "@value" : "provide-access",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:action" : [ {
      "@id" : "idsc:USE"
    } ],
    "ids:constraint" : [ {
      "@type" : "ids:Constraint",
      "@id" : "https://w3id.org/idsa/autogen/constraint/0b7c4ca7-1f9e-4e30-8fa1-
7551700c1980",
      "ids:rightOperand" : {
        "@value" : "2020-07-11T00:00:00Z",
        "@type" : "xsd:dateTimeStamp"
      },
      "ids:operator" : {
        "@id" : "idsc:AFTER"
      },
      "ids:leftOperand" : {
        "@id" : "idsc:POLICY_EVALUATION_TIME"
      }
    }, {
      "@type" : "ids:Constraint",
      "@id" : "https://w3id.org/idsa/autogen/constraint/9f2e0197-2ad9-442b-806b-
5bb4951a2943",
      "ids:rightOperand" : {
        "@value" : "2021-07-11T00:00:00Z",
        "@type" : "xsd:dateTimeStamp"
      },
      "ids:operator" : {
        "@id" : "idsc:BEFORE"
      },
      "ids:leftOperand" : {
        "@id" : "idsc:POLICY_EVALUATION_TIME"
      }
    } ]
  },
{
    "@type" : "ids:Permission",
    "@id" : "https://w3id.org/idsa/autogen/permission/59b0a20a-11bd-4276-8341-
af40c8960e91",
    "ids:target" : {
      "@id" : "https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-
37aa08b2aedd"
    },
      "dpv:hasPersonalDataCategory": "dpv:DerivedPersonalData",
    "ids:title" : [ {
      "@value" : "Example Usage Policy",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:description" : [ {
      "@value" : "provide-access",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    }],
    "ids:action" : [ {
      "@id" : "idsc:USE"
    } ],
      "ids:preDuty": [{
        "@type": "ids:Duty",
        "@id" : "https://w3id.org/idsa/autogen/duty/preduty1",
        "ids:action": [{
              "@id" : "idsc:MODIFY"
        }],
      "idsc:JsonPath": "$.email",
      "ids:constraint" : [ {
          "@type" : "ids:Constraint",
          "@id" : "https://w3id.org/idsa/autogen/constraint/constraint8",
          "ids:operator" : {
              "@id" :  "idsc:SAME_AS"
          },
          "ids:leftOperand" : {
              "@id" : "idsc:PURPOSE"
          },
          "ids:pipEndpoint" : {
              "@id": "http://pip:8085/DataUsage/Pip/1.0/admin/api/purposeName/"
          }
      }]
      }]
  },
  {
    "@type" : "ids:Permission",
    "@id" : "https://w3id.org/idsa/autogen/permission/59b0a20a-11bd-4276-8341-
af40c8960e99",
    "ids:target" : {
      "@id" : "https://w3id.org/idsa/autogen/artifact/8e3a5056-1e46-42e1-a1c3-
37aa08b2aedd"
    },
    "ids:title" : [ {
      "@value" : "Example Usage Policy",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:description" : [ {
      "@value" : "provide-access",
      "@type" : "http://www.w3.org/2001/XMLSchema#string"
    } ],
    "ids:action" : [ {
      "@id" : "idsc:USE"
    } ],
    "ids:constraint" : [ {
      "@type" : "ids:Constraint",
      "@id" : "https://w3id.org/idsa/autogen/constraint/2030a8f2-f03d-4af9-bce5-
b9222e129dcf",
      "ids:rightOperand" : {
        "@value" : "5",
        "@type" : "xsd:double"
      },
      "ids:operator" : {
        "@id" : "idsc:LTEQ"
      },
      "ids:leftOperand" : {
        "@id" : "idsc:COUNT"
      },
      "ids:pipEndpoint" : {
        "@id" :
"http://pip:8080/platoontec/PlatoonDataUsage/1.0/admin/api/access/"
      }
    } ]
  }],
  "ids:provider" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:consumer" : {
    "@id" : "https://w3id.org/idsa/autogen/baseConnector/7b934432-a85e-41c5-9f65-
669219dde4ea"
  },
  "ids:contractDate" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractStart" : {
    "@value" : "2021-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  },
  "ids:contractEnd" : {
    "@value" : "2022-02-18T10:15:21.137Z",
    "@type" : "http://www.w3.org/2001/XMLSchema#dateTimeStamp"
  }
}
```

6 Internal Review

Mark with X the corresponding column:

Y= yes	N= no	NA = not applicable

Name of reviewer: Valentín Sánchez
Organisation: TECNALIA
Date: 09/09/2021

ELEMENT TO REVIEW	Y	N	NA	Comments	Author
FORMAT: Does the document …?					
…include editors, deliverable name, version number, dissemination level, date, and status?	X				
… contain a license (in case of public deliverables)?		X			
… include the names of contributors and reviewers?	X				
… contain a version table?	X				
… contain an updated table of contents?	X				
… contain a list of figures?	X				
… contain a list of tables?			X		
… contain a list of terms and abbreviations?	X				
… contain an Executive Summary?	X				
… contain a Conclusions section?		X			
… contain a List of References (Bibliography) in the appropriate format?			X		
… use the fonts and sections defined in the official template?	X				
… use correct spelling and grammar?	X				
… conform to length guidelines (50 pages maximum (plus Executive Summary and annexes)	X				
… conform to guidelines regarding Annexes (inclusion of complementary information)	X				

ELEMENT TO REVIEW	Y	N	NA	Comments	Author
… present consistency along the whole document in terms of English quality/style? (to avoid accidental usage of copy & paste text)	X				
About the content…					
Is the deliverable content correctly written?	X				
Is the overall style of the deliverable correctly organized and presented in a logical order?	X				
Is the Executive Summary self-contained, following the guidelines and does it include the main conclusions of the document?	X				
Is the body of the deliverable (technique, methodology results, discussion) well enough explained?	X				
Are the contents of the document treated with the required depth?		N			
Does the document need additional sections to be considered complete?	X				
Are there any sections in the document that should be removed?		N			
Are all references in the document included in the references section?			N		
Have you noticed any text in the document not well referenced? (copy and paste of text/picture without including the reference in the reference list)		N			
TECHNICAL RESEARCH WPs (WP2-WP5)					
Is the deliverable sufficiently innovative?	X				
Does the document present technical soundness and its methods are correctly explained?	X				
What do you think is the strongest aspect of the deliverable?				Data usage control and personal data management	

ELEMENT TO REVIEW	Y	N	NA	Comments	Author
What do you think is the weakest aspect of the deliverable?			X		
Please perform a brief evaluation and/or validation of the results, if applicable.			X		
VALIDATION WP (WP6)					
Does the document present technical soundness and the validation methods are correctly explained?					
What do you think is the strongest aspect of the deliverable?					
What do you think is the weakest aspect of the deliverable?					
Please perform a brief evaluation and/or validation of the results, if applicable.					
DISSEMINATION AND EXPLOITATION WPs (WP8 & WP9)					
Does the document present a consistent outreach and exploitation strategy?					
Are the methods and means correctly explained?					
What do you think is the strongest aspect of the deliverable?					
What do you think is the weakest aspect of the deliverable?					
Please perform a brief evaluation and/or validation of the results, if applicable.					

SUGGESTED IMPROVEMENTS

PAGE	SECTION	SUGGESTED IMPROVEMENT

CONCLUSION

Mark with X the corresponding line.

X	Document accepted; no changes required.
	Document accepted; changes required.
	Document not accepted; it must be reviewed after changes are implemented.

Please rank this document globally on a scale of 1-5.
(1-Poor; 2–Fair; 3–Average; 4–Good; 5–Excellent)
Using a half point scale.

Mark with X the corresponding grade.

Document grade	1	1.5	2	2.5	3	3.5	4	4.5	5
								X	