D5.4 Final technical design of interfaces and common specifications

Document Identification			
Status	Final	**Due Date**	28/02/2023
Version	1.0	**Submission Date**	01/03/2023

Related WP	WP5	**Document Reference**	D5.4
Related Deliverable(s)	D2.5, D3.6, D4.1, D4.5, D4.9, D5.2, D5.3	**Dissemination Level (*)**	PU
Lead Participant	MAETD-SGAD	**Lead Author**	Javier Ferrero Merchán (SGAD, ES)
Contributors	Hugo Berdón Hinojosa (SGAD) Javier Ferrero Merchán (SGAD) Philip Helger (BRZ) Eduardo Moreno Alcázar (SGAD)	**Reviewers**	Miguel Correia (INESC-ID) Francisco Aragó (UJI)

Keywords:
Intermediation, user-supported intermediation, subscription, notification, lookup, feature, patterns, common, components, interfaces, specifications

Document Information

List of Contributors	
Name	Partner
Hugo Berdón Hinojosa (SGAD)	MAETD-SGAD (Spain)
Javier Ferrero Merchán (SGAD)	MAETD-SGAD (Spain)
Philip Helger (BRZ)	BRZ (Austria)
Eduardo Moreno Alcázar (SGAD)	MAETD-SGAD (Spain)

Document History			
Version	Date	Change editors	Changes
0.1	28/11/2022	Javier Ferrero (SGAD)	Initial version of document and preliminary comments.
0.2	16/01/2023	Hugo Berdón (SGAD) Eduardo Moreno (SGAD) Javier Ferrero (SGAD)	Addition of chapters 2, 4, 6 and 7.
0.3	28/01/2023	Philip Helger (BRZ)	Addition of chapters concerning third-party specifications and components, along with content related to the Central IAL.
0.3.1	01/02/2023	Javier Ferrero (SGAD)	Review
0.3.2	02/02/2023	Javier Ferrero (SGAD)	Review and addition of chapters 1.2, 3.1.3.2 and 3.1.4.6.
0.3.3	03/02/2023	Philip Helger (BRZ)	Moved Directory to a top-level chapter. Added content to the IAL Service chapter.
0.3.4	06/02/2023	Javier Ferrero (SGAD)	Review and addition of the executive summary, document structure and conclusions.
0.4.0	10/02/2023	Javier Ferrero (SGAD) Eduardo Moreno (SGAD)	Final internal review. Document ready for the formal review.
0.5.0	21/02/2023	Javier Ferrero (SGAD)	Modifications resulting from the formal review of the deliverable.
0.5.1	22/02/2023	Javier Ferrero (SGAD)	Minor corrections.
1.0	28/02/2023	Ana Piñuela (ATOS)	Final quality review

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable leader	Javier Ferrero Merchán (SGAD, ES)	22/02/2023
Quality manager	Julia Wells (ATOS)	28/02/2023
Project Coordinator	Ana Piñuela Marcos (ATOS)	28/02/2023

Table of Contents

List of Tables

List of Figures

List of Acronyms

Abbreviation / acronym	Description
API	Application Programming Interface
AS4	Applicability Statement 4
ATU	Administrative Territorial Unit
BST	Binary Security Token
CEF	Connecting Europe Facility
CRL	Certificate Revocation List
DC	Data Consumer
DE	Data Evaluator
DE4A	Digital Europe for All
DNS	Domain Name System
DO	Data Owner
DP	Data Provider
DR	Data Requestor
DT	Data Transferor
Dx.y	Deliverable number y, belonging to WP number x
EC	European Commission
ESL	Exchange Service Locator
IAL	Issuing Authority Locator
IEM	Information Exchange Model
IDK	Information Desk
IM	Intermediation (pattern)
ITx	Iteration (1 or 2)
KPI	Key Performance Indicator
MOR	Multilingual Ontology Repository
OOP	Once-Only Principle
OOTS	Once-Only Technical System
PDF	Portable Document Format
PM	Person-month
REST	Representational State Transfer
SML	Service Metadata Locator
SMP	Service Metadata Publisher
SOAP	Simple Object Access Protocol
URI	Uniform Resource Identifier
USI	User-Supported Intermediation (pattern)
UUID	Universally Unique Identifier
WP	Work Package

Abbreviation / acronym	Description
XML	Extensible Markup Language

Executive Summary

This deliverable provides the technical design of the interfaces between the DE4A components that facilitate the integration among different technologies used by Member States. These interfaces include the ones required to communicate with the Data Consumers, with the Data Providers and with the common intermediation components, such as the DE4A Connector and the components of the Information Desk (IDK) defined by "WP3 Semantic Interoperability Solutions". D5.4 also defines the common specifications that must be known and complied with by all DE4A network participants to exchange messages, including DE4A-specific protocols and third-party protocols and components.

Within the DE4A network, the Data Consumer entity is composed of the Data Evaluator (online public service provider who requires some evidence) and a Data Requestor (DR), while the Data Provider (DP) entity is composed of a Data Owner (DO) (sender of the requested evidence) and a Data Transferor (DT). The Data Requestor and Data Transferor are each a DE4A Connector, which is the component that establishes the cross-border communication. For the communication between DE4A Connector, the European Commission eDelivery building block [9] is used: the DE4A Connector integrates an eDelivery Access Point, and the dynamic discovery of this technology is used, so that the SML of European Commission and an SMP per Country are also involved. Finally, there is a single instance of the Central Issuing Authority Locator (IAL) component, which provides routing functionalities for the Connectors.

WP2 defined five interaction patterns [2] (i.e., five different ways of orchestrating the exchange of the information) in the DE4A project. Four of them use the DE4A Connector as the way to carry out the exchange:

- **Intermediation pattern**: exchange of evidence where the preview takes place at the electronic procedure portal of the Data Evaluator.
- **User-Supported Intermediation pattern**: exchange of evidence where the preview takes place at the Data Owner portal.
- **Subscription and Notification pattern**: pattern aimed at configuring alerts (notifications) triggered by some events that relate to businesses.
- **Lookup pattern**: exchange of evidence, closely related to the previous pattern, where there is no direct involvement of a user (and thus, there is no preview).

The **Verifiable Credentials pattern** is elaborated in its own deliverable as a result of task T5.4 DE4A Self-sovereign Identity Framework Support [4].

These interaction patterns use a data model for exchanging the evidence called Information Exchange Model (IEM), based on the set of XSD files defined by WP3 [3]. In addition, WP3 defined the Information Desk conceptual component [3], consisting of three functionalities: Issuing Authority Locator (IAL), Exchange Service Locator (ESL) and Multilingual Ontology Repository (MOR). The Central IAL fully implements the IAL functionality. It also stores and allows the retrieval of the content files of the MOR functionality (otherwise implemented by WP3). The Exchange Service Locator (ESL) functionality has been implemented with the dynamic discovery mode of the eDelivery infrastructure (i.e., by the use of SMPs and the SML).

Finally, for everything to work properly, a set logs and error messages has been defined by WP5, which must be complied with by all common and pilot components to inform about different types of situations that may occur throughout the evidence exchange process.

1 Introduction

1.1 Purpose of the document

The present document is written in the context of DE4A work package **"WP5 Common Component Design & Development"** and under the task **"T5.2 Technical design of interfaces and common specifications of products"**.

The scope of WP5 is the design and development of the DE4A common software components, required for the pilot implementations, following the outputs of "WP2 Architecture vision and framework" and "WP3 Semantic Interoperability Solutions" on architecture and semantic components.

Task T5.2 specifies the common products/components to build from the inventory of features created in the previous task "T5.1 Consolidation of Features and Patterns", providing technical inputs to "T5.3 Construction of Common Products and Components & Change Management".

Therefore, deliverable D5.4 provides the technical design of the interfaces between the DE4A components to facilitate the integration among different technologies used by Member States. These interfaces include the ones required to communicate with the Data Consumers, with the Data Providers and with the common intermediation components, such as the DE4A Connector or the components of the IDK defined by WP3. D5.4 also defines the common specifications that must be known and complied with by all DE4A network participants to exchange messages, including DE4A-specific protocols and third-party protocols and components.

This deliverable D5.4 is related to the following DE4A deliverables:

- D2.5 Project Start Architectures (PSA), second iteration
- D3.6 Semantic Toolkit, Final version
- D4.1 Studying abroad - Use cases definition & requirements
- D4.5 Doing Business Abroad - Use cases definition & requirements
- D4.9 Moving Abroad - Use cases definition & requirements
- D5.2 Final inventory of features for products/components
- D5.6 Final release of DE4A Common Components

1.2 Structure of the document

This document is divided into seven main sections plus an annex:

- Chapter 2 DE4A infrastructure
- Chapter 3 Third-party specifications and components
- Chapter 4 Common behaviour and interactions between components
- Chapter 5 Central IAL interface
- Chapter 6 Summary of software interfaces
- Chapter 7 DE4A Logs and error messages
- Annex I Lists of codes

2 DE4A infrastructure

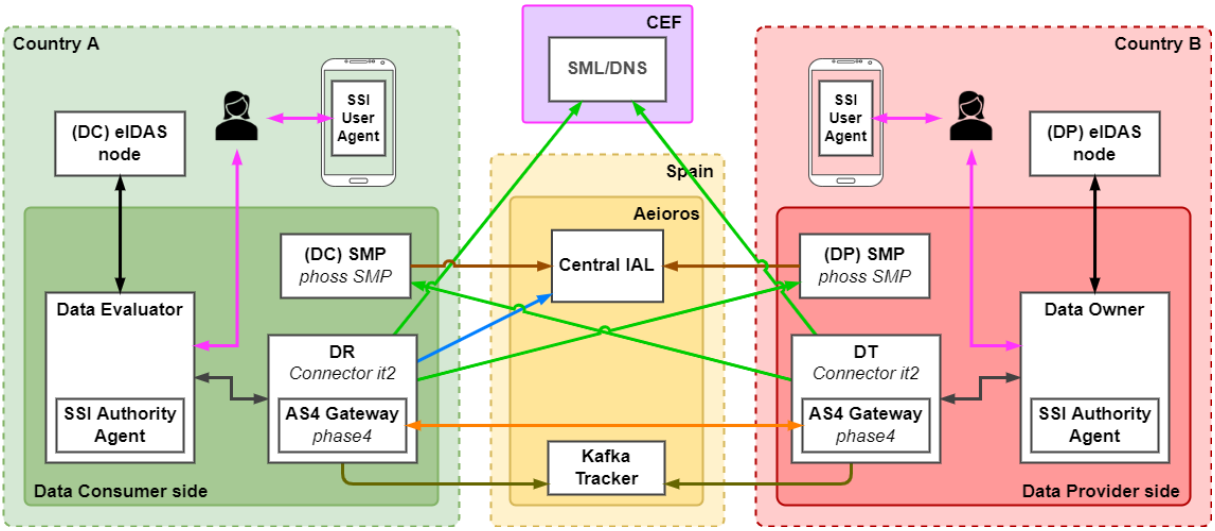

Figure 1: General infrastructure of the DE4A network

Figure 1 shows the general infrastructure of the DE4A network. It requires at least two participants establishing cross-border communication. Each participant must have their own software components needed to communicate with the other.

Regardless of whether the participant infrastructure is acting as a Data Consumer (DC) or as a Data Provider (DP), the following software components are required:

- **Connector**: The component that will establish the cross-border communication.
- **SMP:** The component that will store the routing and certificate information to inform other components when requested.
- When acting as a DC:
  - **Data Evaluator (DE):** The component requesting the information.
- When acting as a DP:
  - **Data Owner (DO):** The component that has access to the local information and will send it to the Data Consumer side.

Additionally, each pilot DE and DO will validate the user authentication with the **eIDAS** node of the user's nationality.

The user can also participate using a mobile application. Using this feature will require a deployment of the **SSI User Agent**, a mobile digital wallet app that enables users to manage their digital diplomas and interact with the Issuer's/Verifier's portals.

There are other common software components that will be used by each DE4A Connector:

- **SML**: The Service Metadata Locator (SML) is a unique instance in the DE4A network. It is operated by the EC and provided to the DE4A project at no cost. Every SMP that wants to join the DE4A network needs to register once at the SML using the specific DNS zone. The SML creates DNS entries and is required to find foreign SMPs within the DE4A network.
- **Central IAL**: Central component that allows to retrieve the information of the IAL functionality and the content of the MOR component.

- **Kafka Tracker**: A centralized application to monitor the communication between components, useful to trace the full process flow.

The following two diagrams outline the general infrastructure expected from the Member States and participants involved in the DE4A pilots. Each Connector is in turn connected and exchanges data with the corresponding Connector (Data Requestor or Data Transferor) of the Playground or other participants.

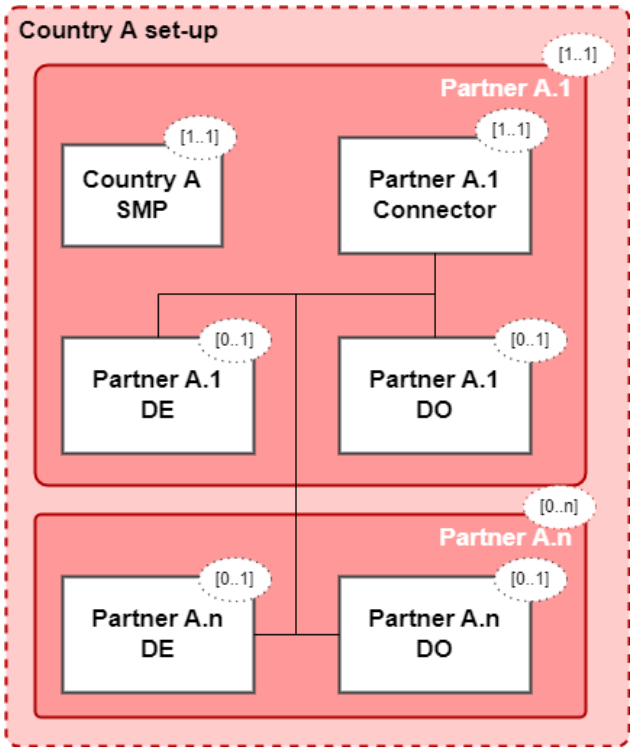

Figure 2: Country A set-up

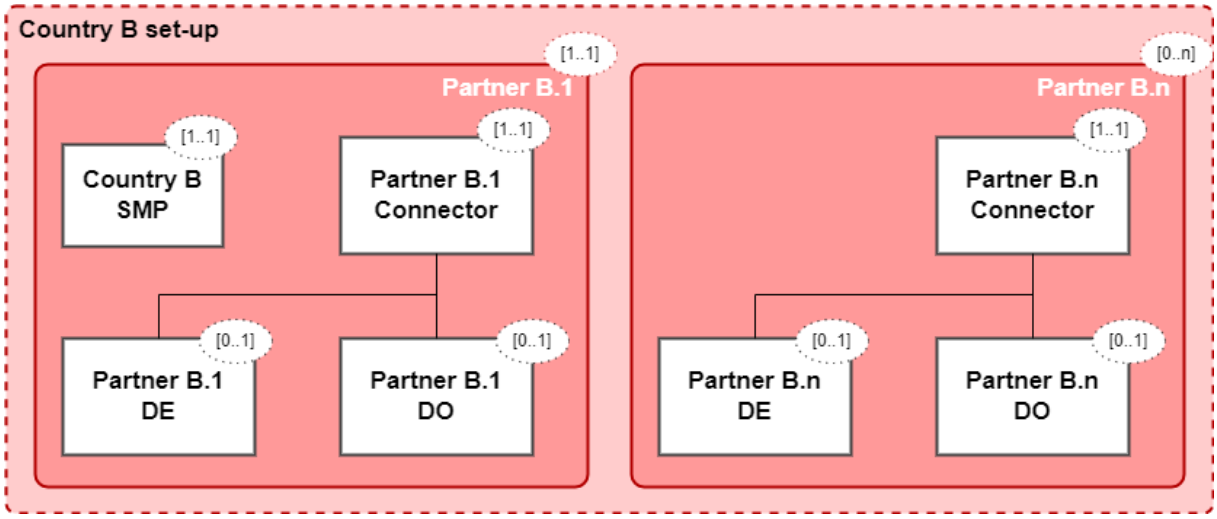

Figure 3: Country B set-up

Each participant may have a different infrastructure depending on their requirements. There are two main set-ups expected in DE4A:

- One participant deploys and maintains a single DE4A Connector instance that is used by the other participants of the same country (see Figure 2).
- Each participant deploys and maintains their own DE4A Connector instance (see Figure 3).

In any case, both set-up types use a single SMP application for all participants of the same country (even though there is no technical or business constraint to enforce this logic).

3 Third-party specifications and components

3.1 eDelivery

3.1.1 Introduction

The exchange of a single document between a DE and a DO always requires two eDelivery exchanges: the first one initiated by DE and targeted for DO, and the second one is initiated by DO and targeted for the DE. Technically speaking both transmissions are “requests” even though their semantics are “request” and “response”.

The foundation of the document exchange is the so called “4-corner model”, which differentiates between the business sender of a document (Corner 1 aka C1), the technical sender of a document (Corner 2 aka C2), the technical receiver of a document (Corner 3 aka C3) and the business receiver of a document (Corner 4 aka C4). Depending on the order of a message exchange, the assignment of the corner varies. In DE4A the “DE4A Connector” (sometimes just “Connector”) can play the role of both DR and DT and therefore acts as C2 or C3, depending on whether a message is sent or received.

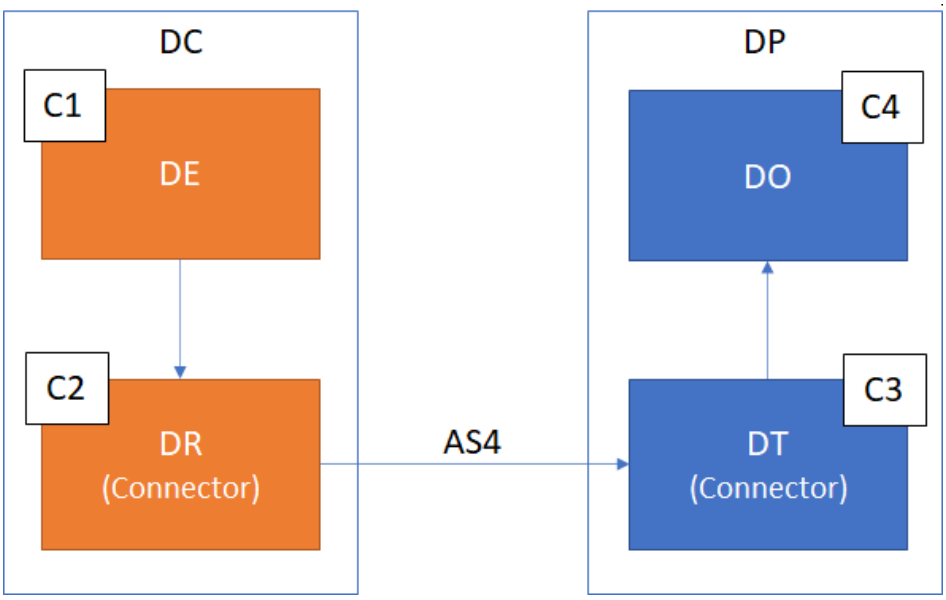

Figure 4: eDelivery business request from DE to DO

Figure 4 depicts the structural message exchange initiated by DE (C1), sent by DR (C2), received by DT (C3) and forwarded to DO (C4). The message exchange between C1 and C2 as well as the message exchange between C3 and C4 are not specified by eDelivery, even though AS4 may be used for this, but they must be defined by the DE4A Connector.

If DO sends a message back to DE, the order of the messages change as well as the corner assignment, as shown in Figure 5: the DO becomes C1, forwarding the response to DT which is now C2. The AS4 transmission targets DR as C3 who in turn forwards the payload to DE which is the C4 in this scenario.

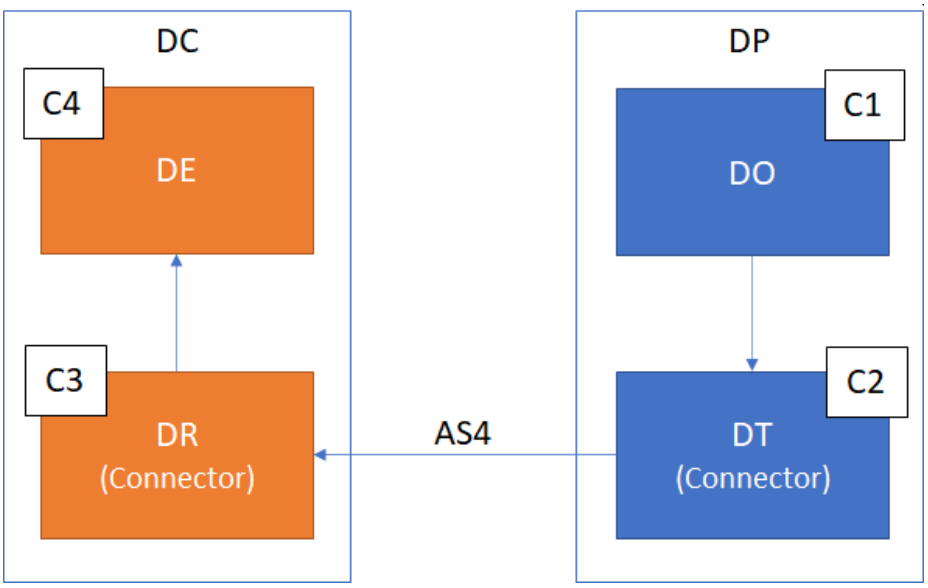

Figure 5: eDelivery business response from DO to DE

This duality of the message exchange means, that each of the named nodes (DE, DR, DT and DO) requires both sending and receiving capabilities.

For the sake of clarity, the rest of the document only shows images with messages flowing from DE to DO because it seems easier to understand, even though the image would be perfectly valid for the return direction from DO to DE (except when stated differently).

The eDelivery message exchange in DE4A uses the so called “Dynamic Discovery” which is an extension of the basic eDelivery in the sense that it adds the usage of SML and SMP. Both components as well as the lookup process are described below.

3.1.2 Identification of Components

Each C1 and C4 of a message exchange is called a “Participant” and is uniquely identified by a “Participant Identifier”. The nodes C2 and C3 are not participants and have no respective identifier, they are only accessed by URLs.

Different types of documents exchanged via eDelivery are classified via “Document Type Identifiers”. The orchestrations in which document types are exchanged are classified via “Process Identifiers”.

There is a separate policy document on the usage of identifiers within its network. This document, called “DE4A Policy for the use of identifiers” [1], contains the details about the following identifier types:

- Participant Identification
- Document Type Identification
- Process Identification
- Transport Profile Identification

Each Participant ID, Document Type ID and Process ID consists of two separate parts – one “scheme” part and one “value” part. A scheme defines the layout and constraints of the value. This allows to add new types of identifiers in different scenarios, without interfering with existing used identifiers.

This document was heavily inspired by the "Peppol Policy for use of Identifiers", the identifier reference document of the Peppol[1] network.

Additional to the guiding document, the project also created a set of Code Lists, that contains the allowed values[2]. The Code Lists are available in Annex I. Lists of codes.

3.1.3 SML/DNS

The SML and DNS are often used synonymously but they serve different purposes. The SML is the registry of all known SMPs and Participants from any project or service that use the dynamic discovery mode of eDelivery and is responsible for creating DNS entries in a specific DNS zone. For every Participant a unique DNS entry is created. Only SMPs interacts with the SML. Other components will never explicitly interact with the SML.

During the main message exchange between C2 and C3 the DNS is queried for the SMP URL of C3. The SML is neither queried in the message exchange nor before or after. Due to the distributed nature of DNS, this message exchange is performed without a single point of failure, which is one of the key benefits of eDelivery.

The SML is operated centrally by the European Commission. Its service is gratefully offered free of charge to the DE4A project.

The SML creates DNS records for each SMP itself (see chapter 3.1.4.1) and for each Participant (aka Service Group; see chapter 3.1.4.2) pointing to the owning SMP. These DNS NAPTR records need to be read by the sending party (the DE4A Connector), to determine the URL of the SMP that needs to be queried.

The SML and the DNS are centralized third-party components for DE4A. For the SMPs to be able to communicate with the SML, a Client Certificate is needed (see chapter 3.4), so only legitimate requestors can create DE4A participants in the SML.

3.1.3.1 SML query process in DE4A

DE4A only uses the production SML and the dedicated domain `de4a.edelivery.tech.ec.europa.eu.` for its purposes.

The lookup process from the Connector solely requires the Participant Identifier (see chapter 3.1.2) of the receiver. Using a SHA-256 hash value of the Participant Identifier, a unique domain name is created, which will be looked up from the DNS using the "NAPTR" record query type. This NAPTR record then contains the base path of the SMP to be queried. The details of the SMP query are described in chapter 3.1.4.5.

The structure of the Participant IDs used in DE4A is described in chapter 3.1.3.2.

A real-life example for looking up the Participant Identifier `iso6523-actorid-upis::9915:de4atest` looks like this:

- Apply the following algorithm to the Participant Identifier:
  - `strip-trailing (base32 (sha256 (lowercase (ID-VALUE))), "=") + "." + ID-SCHEME + "." + SML-ZONE-NAME`
  - In the above example identifier, the "ID-SCHEME" is "iso6523-actorid-upis" and the "ID-VALUE" is "9915:de4atest".
- The created domain name is:

[1] The Peppol network is the outcome of the former EU project "PEPPOL".
[2] The DE4A Code Lists also reside on GitHub at https://github.com/de4a-wp5/de4a-codelists/

`54VMPCQA26DNZS74VHQOKJ7U6IRBBI5KPMQ6AO3KVCQC3F6YR2YA.iso6523-actorid-upis.de4a.edelivery.tech.ec.europa.eu`

- The look up can be done with the “dig” tool on the command line:

```
C:\tools\dig-9.16.12>dig -t naptr 54VMPCQA26DNZS74VHQOKJ7U6IRBBI5KPMQ6AO3KVCQC3F6YR2YA.iso6523-actorid-upis.de4a.edelivery.tech.ec.europa.eu

; <<>> DiG 9.16.12 <<>> -t naptr 54VMPCQA26DNZS74VHQOKJ7U6IRBBI5KPMQ6AO3KVCQC3F6YR2YA.iso6523-actorid-upis.de4a.edelivery.tech.ec.europa.eu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 40227
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;54VMPCQA26DNZS74VHQOKJ7U6IRBBI5KPMQ6AO3KVCQC3F6YR2YA.iso6523-actorid-upis.de4a.edelivery.tech.ec.europa.eu. IN NAPTR

;; ANSWER SECTION:
54VMPCQA26DNZS74VHQOKJ7U6IRBBI5KPMQ6AO3KVCQC3F6YR2YA.iso6523-actorid-upis.de4a.edelivery.tech.ec.europa.eu. 60 IN NAPTR 100 10 "U" "Meta:SMP" "!.*!https://de4a-smp.usp.gv.at!" .

;; Query time: 362 msec
;; SERVER: 212.186.211.21#53(212.186.211.21)
;; WHEN: Thu Jan 26 23:14:43 Mitteleuropäische Zeit 2023
;; MSG SIZE  rcvd: 195
```

Figure 6: DNS NAPTR record query result

- The relevant part of the response is `"!.*!https://de4a-smp.usp.gv.at!"` that contains the link to the SMP of this particular Participant Identifier that can be queried for the details, embedded into a regular expression as defied by the NAPTR record specification.
- Using the extracted URL `https://de4a-smp.usp.gv.at` the regular SMP query process, as described in chapter 3.1.4.5, can be performed.

As can be seen from this description, the SML system itself is not invoked in the lookup – only the DNS system is involved. Because the DNS system is inherently replicated, a fast and fail-safe operation is guaranteed.

3.1.3.2 Domains and types of Participant IDs used in DE4A

In DE4A, there are two types of participants:

- Imaginary participants: participants for whom example evidence and fake data have been made up to be used for testing purposes.
- Real participants: the actual DE4A partners participating in the pilots of the project.

At the same time, for the real participants, there are also two “domains”:

- Test domain: only fake data is used for test purposes. It is the domain related to the DE4A Playground, where participants return datasets from test sources.
- Pilot domain: where real data from real citizens and companies are expected to be used. It is the environment used for the running phase of the pilots, where participants access to their real registries and return real evidence from them.

Finally, the project is divided into two pilots iterations where each of them needs a copy of those two domains, and where some overlap between them occurs.

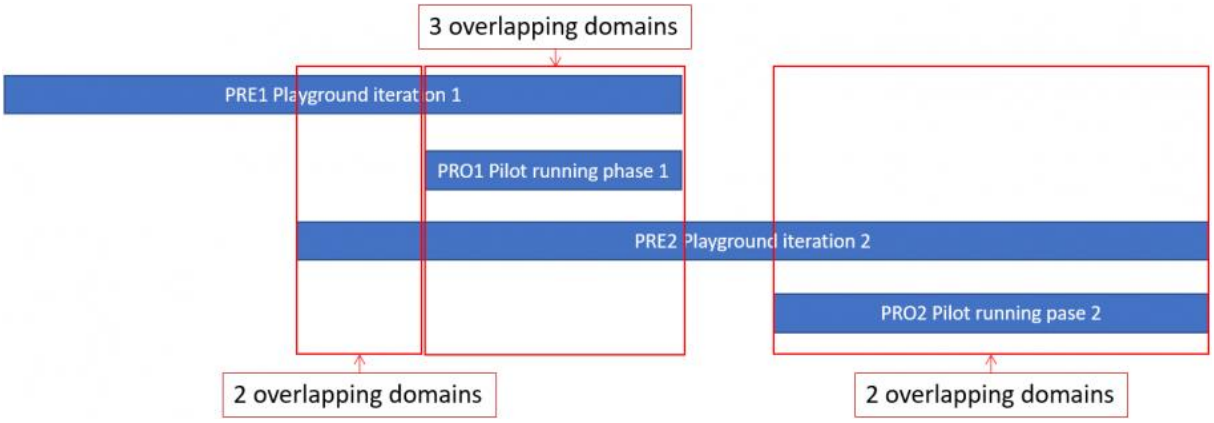

Figure 7: Overlapping between domains over the course of the two iterations of the project

As the DE4A project uses the dynamic discovery of eDelivery, to query the DNS there cannot be duplicate participant IDs targeting to different SMPs (depending on the domain concerned). Thus, a different participant ID schema per each domain has been defined:

- **PRE1a**: Imaginary participants used for testing during iteration 1 of the project. The related evidence is stored in the Mocked DO of the Playground iteration 1.
  - Scheme identifier used: "**9999**".
  - Suffix used: "**-it1**".
  - E.g. iso6523-actorid-upis::99**99**:ess2833002e-**it1**
- **PRE1b:** Real participants queried about testing data through the Playground iteration 1. Simulated data will be returned when querying for the participants. The related evidence is located in each partner's infrastructure, within their test domains.
  - Scheme identifier used: "**99XX**", where "XX" depends on the identifier each participant is using.
  - Suffix used: "**-it1**".
  - E.g. iso6523-actorid-upis::99**20**:ess2833002e-**it1**
- **PRO1:** Real participants queried about actual data during the execution of phase 1 of the pilot. The related evidence is located in each partner's infrastructure, within their pilot domains.
  - Scheme identifier used: "**99XX**", where "XX" depends on the identifier each participant is using. It is the "real" participant ID.
  - Suffix used: *none*, since there is no overlap between phases 1 and 2 of the pilots.
  - E.g. iso6523-actorid-upis::99**20**:ess2833002e
- **PRE2a:** Imaginary participants used for testing during iteration 2 of the project. The related evidence is stored in the Mocked DO of the Playground iteration 2.
  - Scheme identifier used: "**9999**".
  - Suffix used: "**-mock-it2**".
  - E.g. iso6523-actorid-upis::99**99**:ess2833002e-**mock-it2**
- **PRE2b:** Real participants queried about testing data through the Playground iteration 2 Simulated data will be returned when querying for the participants. The related evidence is located in each partner's infrastructure, within their test domains.
  - Scheme identifier used: "**99XX**", where "XX" depends on the identifier each participant is using.
  - Suffix used: "**-test-it2**".
  - E.g. iso6523-actorid-upis::99**20**:ess2833002e-**test-it2**
- **PRO2:** Real participants queried about actual data during the execution of phase 2 of the pilot. The related evidence is located in each partner's infrastructure, within their pilot domains.

- Scheme identifier used: "**99XX**", where "XX" depends on the identifier each participant is using. It is the "real" participant ID.
- Suffix used: *none*, since there is no overlap between phases 1 and 2 of the pilots.
- E.g. iso6523-actorid-upis::99**20**:ess2833002e

These different participant IDs must be stored in their corresponding IDK components and SMPs and configured to return the proper information:

- **PRE1a:** Imaginary participants during the first iteration
  - IDK: none (for these participants, the DemoUI of the Playground knows which the imaginary participants are and which DO to query, namely, the Mocked DO of the first iteration).
  - SMP: shared SMP of the Playground IT1. It returns the routing information of the targeted Connector (DR/DT) of the Playground IT1.
- **PRE1b**: Simulated data from real participants during the first iteration
  - IDK: Mocked IDK of the Playground IT1.
  - SMP: shared SMP of the Playground IT1. It returns the routing information of the targeted Connector (DR/DT) of the requested participant.
- **PRO1:** Real data from real participants during the first iteration
  - IDK: Mocked IDK of the pilot running environment.
  - SMP: national SMP of the requested participant. It returns the routing information of the targeted Connector (DR/DT) of the requested participant.
    - For the first pilot running phase, partners who had not deployed their own SMPs were able to use a shared SMP provided by one of the partners with an available SMP. In such cases, the routing information of those partners was stored in that shared SMP.
- **PRE2a:** Imaginary participants during the second iteration
  - IDK: Central IAL. Its information is automatically updated and fed from the SMPs connected to it.
  - SMP: shared SMP of the Playground IT2. It returns the routing information of the targeted Connector (DR/DT) of the Playground IT2.
- **PRE2b:** Simulated data from real participants during the second iteration
  - IDK: Central IAL. Its information is automatically updated and fed from the SMPs connected to it.
  - SMP: national SMP of the requested participant. It returns the routing information of the targeted Connector (DR/DT) of the requested participant.
- **PRO2:** Real data from real participants during the second iteration
  - IDK: Central IAL. Its information is automatically updated and fed from the SMPs connected to it.
  - SMP: national SMP of the requested participant. It returns the routing information of the targeted Connector (DR/DT) of the requested participant.

3.1.4 SMP

The Service Metadata Publisher (SMP) is a decentralized registry with routing information. For DE4A a solution that is compatible with the OASIS BDXR SMP v1 specification must be used, as indicated by the eDelivery specification.

The SMP is responsible for maintaining the relationship between a Participant Identifier and its technical addressing details, such as the AS4 endpoint URL and the X.509 certificate. Every SMP must implement a standardized REST API for querying.

All SMPs MUST provide the two REST APIs mandated by the specification, identified as `/{participantID}` and `/{participantID}/services/{docTypeID}`. The first API returns a list of all document types the participant is capable to receive (which may be an empty list) and the second API returns the details on the receiving "Endpoints" including the endpoint URL and

the X.509 certificate of the receiver. Both APIs can only return XML content, and only the second API response is digitally signed with the SMP certificate of the SMP maintainer. Each of the response data structures contain an optional Extension element that could be used for additional content.

Each SMP MUST have one Certificate from the SMP PKI configured (see chapter 3.4), independent of the number of Participants it manages. This certificate is used as a client certificate for the communication with the SML (see chapter 3.1.3), as a client certificate for the communication with the DE4A Directory and as an XML signing certificate for its REST responses.

An SMP MUST be registered once in the SML before it can be used in the network.

3.1.4.1 Big Picture for SMP Registration

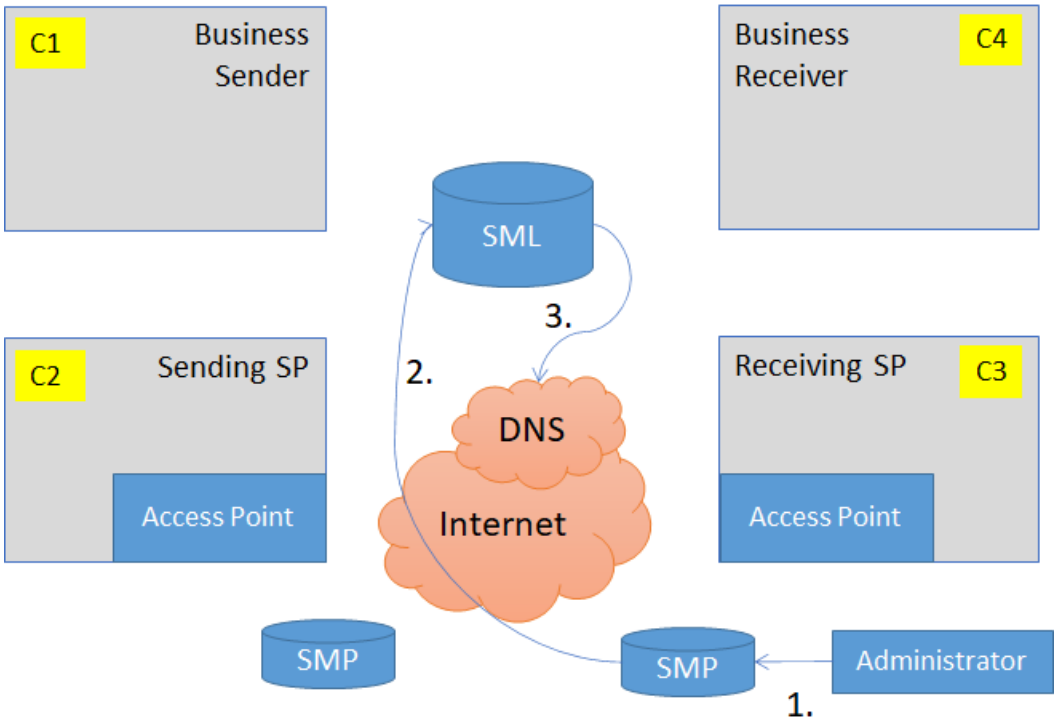

Figure 8: Big Picture of SMP registration

The initial registration of an SMP to the SML is depicted in the above figure. This process needs to be triggered manually by the SMP Administrator. It requires a trusted SMP certificate which is used as a client certificate when invoking the SML's API.

Input parameters to the SML registration are:

- The SMP ID
- The public IP address of the SMP server (which is a legacy parameter)
- The public hostname of the SMP server

The results of this registration process are:

- The SMP ID is linked inside the SML with the SMP certificate.
- The creation of the "Publisher DNS entry", which is a generated DNS CNAME entry based on the SMP's "SMP ID" and the base DNS zone the SML is operating in. The target is the public hostname of the SMP server.

Note: every time an SMP certificate is updated it MUST be updated in the SMP and the SML.

3.1.4.2 Big Picture for Participant Registration

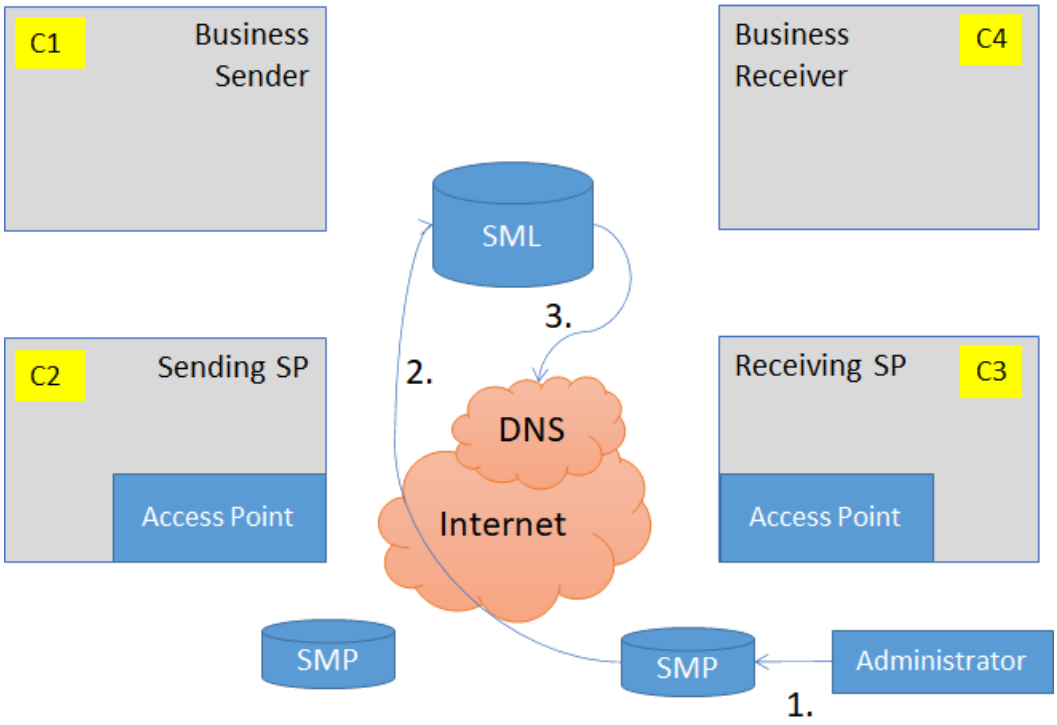

Figure 9: Big Picture of SMP participant registration

The above picture shows the Participant registration process – and it looks exactly the same as Figure 8 because the involved components are the same. It requires a trusted SMP certificate which is used as a client certificate when invoking the SML's API.

Each Participant Registration in the SML performs the following actions:

- Link the Participant ID with the owning SMP. This implicitly checks that the Participant ID is unique and not already registered.
- Create a new DNS record of type "NAPTR" [10] that links the Participant ID with the owning SMP. NAPTR is a special DNS record type, different from the usual "A" or "CNAME" record types, and is able to store absolute URLs. The domain name is created using a hash algorithm and the target of the DNS entry is the public domain name of the SMP.

Note: Of course, an SMP should be able to handle multiple participants.

3.1.4.3 Business Card Extension

As an addition to the routing information, an SMP must also support the "Business Card" API as specified by the Peppol Directory specification. It adds non-routing information to a "Participant Identifier". The data model of the Business Cards is depicted in the following Figure 10.

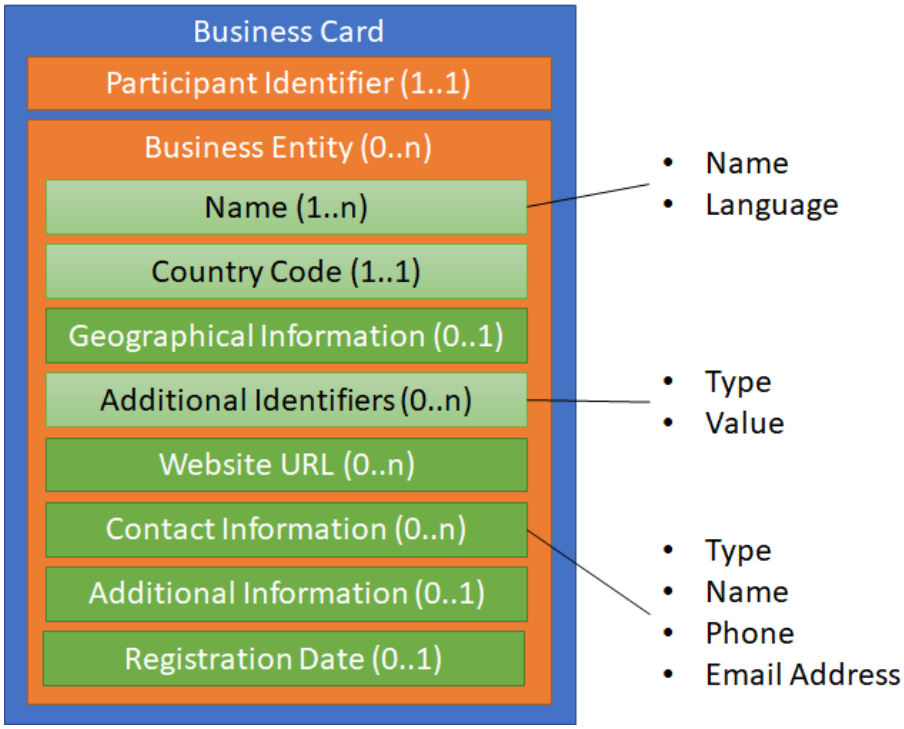

Figure 10: Data model of a Business Card

All the "Business Cards" will be collected in the DE4A Directory and made available centrally for querying. The SMP is responsible for keeping the data in the Directory up to date.

The API to query the business cards from an SMP is `/businesscard/{participantID}` and returns XML only. The supported XML Schemas are available on GitHub [11] - any of these versions may be returned from an SMP.

The SMP itself triggers the DE4A Directory via a REST API to indicate that the data of a Participant needs to be re-indexed. This API only takes the participant ID and the Directory will perform a DNS lookup with the participant ID (see chapter 3.1.3) to determine the owning SMP, and query the Business Card via the previously mentioned REST API from that SMP. The reason why the SMP is not sending the full Business Card is to avoid that somebody else than the data owner can publish data into the Directory.

The phoss SMP, provided in GitHub [12], is the only open source SMP known to the author that supports the Business Card extension out of the box.

3.1.4.4 Big Picture for Message Transmission

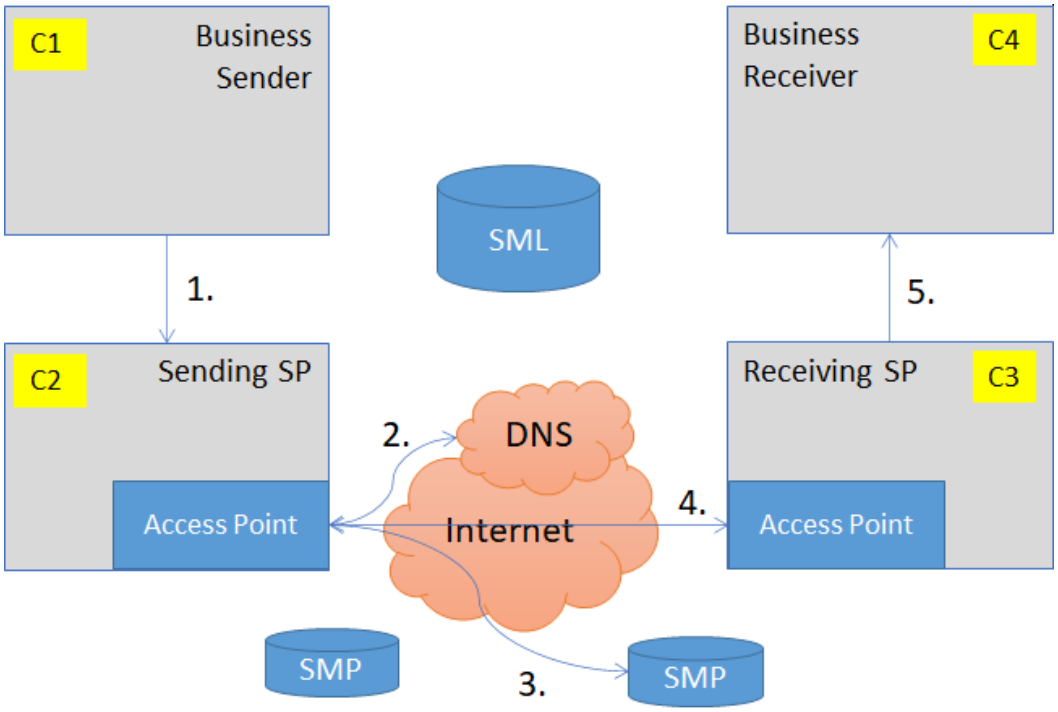

Figure 11: Big Picture of an eDelivery Message Exchange

Figure 11 shows the big picture of a message exchange. It includes the following steps. Error handling is purposely left out.

- Step 1: C1 submits the document anyhow to C2. "Anyhow" from an eDelivery perspective means, that it is not specified by the eDelivery components – any protocol, payload and addressing mechanisms can be used here. In DE4A this communication is defined by a DE4A Connector interface.
- Step 2: C2 requires the Participant ID of C4, calculate the DNS name (as outlined in chapter 3.1.4.2) and perform a DNS lookup for the "NAPTR" record. The outcome is the public URL of the SMP. The DNS is a distributed system itself and is one of the corner stones of the Internet as we know it.
- Step 3: Perform the SMP client lookup with the public URL from step 2. The response XML format is described in the OASIS SMP specification. Select the best matching SMP Endpoint, based on the business requirements, which results in an "Endpoint URL" and the X.509 Certificate to be used in the AS4 transmission. The specific SMP Endpoint selection rules may vary from pilot to pilot. Access to the SMP SHOULD be transport layer secured[3].
- Step4: Create the AS4 message, encrypt it with the certificate from Step 3 and sign it with your AS4 certificate. Transmit the document via AS4 to the URL retrieved from Step 3. The transmission MUST be transport-layer secured.
- Step 5: C3 submits the document anyhow to C4. In DE4A this communication is defined by a DE4A Connector interface.

The list of these steps is complete, and no central, single-instance components are involved in the transmission.

[3] Since the response of an SMP is digitally signed, authenticity can be verified anyway.

3.1.4.5 SMP Query Process

After the base URL of the SMP was determined using the DNS lookup (see chapter 3.1.3.1), the regular SMP lookup can take place. This action is performed by the DE4A Connector using the specified query API `/{participantID}/services/{docTypeID}`. The parameters to this query API are the Participant ID (that is already required for the DNS lookup) and Document Type ID. The Document Type ID defines what kind of document should be received and needs to be taken from the DE4A Code List (see chapter 3.1.2).

For the Participant ID `iso6523-actorid-upis::9915:de4atest` and the Document Type ID `urn:de4a-eu:CanonicalEvidenceType::CompanyRegistration:1.0` (used to identify company registration data from the DBA pilot) the following query URL is built:

`/iso6523-actorid-upis::9915:de4atest/services/urn:de4a-eu:CanonicalEvidenceType::CompanyRegistration:1.0`

The result of this SMP HTTP GET query[4] is an XML document that contains a list of all Processes and for each Process the list of Endpoints.

The structure and values for the Participant IDs, Document Type IDs and Process IDs used during the SMP query process is described in chapter 3.1.4.6.

Inside each Process returned by the SMP query is a list of so called "Endpoints". Each endpoint represents the connection details for one particular transport protocol. In DE4A we are only supporting the AS4 transport protocol with the identifier `bdxr-transport-ebms3-as4-v1p0` – so each returned Process may only contain a single Endpoint.

Figure 12: SMP Endpoint example XML representation

The above figure Figure 12 shows the technical content of an SMP Endpoint. Inside an SMP Endpoint the two main elements that are of interest are the contents of:

- element `EndpointURI` containing the URL of the Connector where this Participant can retrieve the queried Document Type and
- element `Certificate` containing the public X.509 certificate of the receiving Connector, so that the message can be digitally encrypted for that specific receiver.

[4] The absolute SMP query URL is https://de4a-smp.usp.gv.at/iso6523-actorid-upis%3A%3A9915%3Ade4atest/services/urn%3Ade4a-eu%3ACanonicalEvidenceType%3A%3ACompanyRegistration%3A1.0

With these information elements at hand, the message can be encrypted for the specific receiver and send to the correct URL. The main message exchange via AS4 can start now.

3.1.4.6 Values of the parameters for the SMP query process

This section specifies the general values of the Participant ID, Document Type ID and Process ID parameters when querying the SMPs for the three main types of DE4A messages. Remember that, in DE4A, the Transport Protocol ID value is always the same (see chapter 3.1.4.5). The complete set of values for the following *evidenceID* and *catalogueID* variables are available at Annex I. Lists of codes.

3.1.4.6.1 Evidence request

When a Data Consumer wants to send an evidence request message based on the IM, USI or LU patterns, or when it wants to send subscription request based on the S&N pattern, its DE4A Connector will query the SMP of the recipient Data Provider about this data:

- ParticipantId:
  - Identifier of the **Data Owner to request** (see chapter 3.1.3.2).
- DocumentTypeId:
  - Identifier of the required item, depending on whether it is a **canonical evidence type** or a **canonical event catalogue**, and whether it is a multi-item request or not.
    - For a single piece of evidence:
      - urn:de4a-eu:CanonicalEvidenceType::*evidenceID*
    - For multiple pieces of evidence:
      - urn:de4a-eu:CanonicalEvidenceType::MultiItem
    - For a single subscription:
      - urn:de4a-eu:CanonicalEventCatalogueType::*catalogueID*
    - For multiple subscriptions:
      - urn:de4a-eu:CanonicalEventCatalogueType::MultiItem
- ProcessId:
  - urn:dea4-eu:MessageType::**request**

Data Provider's SMP will reply with the endpoint and the certificate of the Data Transferor's AS4 Gateway.

3.1.4.6.2 Evidence response

When a Data Provider wants to send an evidence response message based on the IM, USI or LU patterns, or a subscription confirmation message based on the S&N pattern, or a redirection user message based on the USI pattern, its DE4A Connector will query the SMP of the recipient Data Consumer about this data:

- ParticipantId:
  - Identifier of the **Data Evaluator to respond** (see chapter 3.1.3.2).
- DocumentTypeId:
  - Identifier of the required item, depending on whether it is a **canonical evidence type** or a **canonical event catalogue**, and whether it is a multi-item request or not.
    - For a single piece of evidence:
      - urn:de4a-eu:CanonicalEvidenceType::*evidenceID*
    - For multiple pieces of evidence:
      - urn:de4a-eu:CanonicalEvidenceType::MultiItem
    - For a single subscription:
      - urn:de4a-eu:CanonicalEventCatalogueType::*catalogueID*

- For multiple subscriptions:
  - urn:de4a-eu:CanonicalEventCatalogueType::MultiItem

- ProcessId:
  - urn:dea4-eu:MessageType::**response**

Data Consumer's SMP will reply with the endpoint and the certificate of the Data Requestor's AS4 Gateway.

3.1.4.6.3 Notification

When a Data Provider wants to send an event notification message based on the S&N pattern, its DE4A Connector will query the SMP of the recipient Data Consumer about this data:

- ParticipantId:
  - Identifier of the **Data Evaluator to notify** (see chapter 3.1.3.2).
- DocumentTypeId:
  - Identifier of the involved **canonical event catalogue**, depending on whether it is a multi-item request or not.
    - For a single event notification:
      - urn:de4a-eu:CanonicalEventCatalogueType::*catalogueID*
    - For multiple event notifications:
      - urn:de4a-eu:CanonicalEventCatalogueType::MultiItem
- ProcessId:
  - urn:dea4-eu:MessageType::**notification**

Data Consumer's SMP will reply with the endpoint and the certificate of the Data Requestor's AS4 Gateway.

3.1.5 AS4

AS4 is the corner stone of the eDelivery document exchange. It ensures that messages are transmitted in a secure, reliable, standardised and interchangeable manner. AS4 1.0 is an OASIS Standard and defines a subset of OASIS ebMS 3.0 [13]. The technology builds on top of SOAP messages and the usage of SOAP with attachments. The security specifications used are WS Security and WS Reliable Messaging.

The usage of AS4 is mandatory when using eDelivery: every message that is exchanged between a DE and a DO, independent of its direction, must be sent via AS4. The sending DE4A Connector acting as the DR and receiving Connector acting as the DT are the only components that directly deal with AS4. Neither DE nor DO need to know the details of the protocol.

AS4 messages are encrypted and signed on the protocol level (leveraging the WS-Security 1.1.1 specification) and by governance the usage of TLS 1.2 or later on the transport layer (with strong cipher suites only) is required – see the CEF eDelivery specification for details.

Messages sent from C2 to C3 are encrypted with the public key of C3 and signed with the private key of C2. The public key of C2 is transmitted as a BST as part of the message to C3. Each AS4 installation needs exactly one X.509 certificate (see chapter 3.4) independent of the number of Participants for which it exchanges messages.

Each AS4 message exchange matches one HTTP exchange – it always consists of one request and one response. The message exchange pattern used by DE4A is the "One Way Push" pattern and that means that the requestor always sends a so called "User Message" with a payload to the receiver who has to respond with a so called "Signal Message" that contains either an Error or a Receipt with the non-

repudiation of receipt information. In case of a successful message transmission and a positive response, no further payload besides the pure acceptance information is allowed.

In DE4A each request from a DR to a DT contains a RegRep document (see chapter 3.2) that contains the DE4A Core Data Model (e.g. an Evidence Request). The response from DT back to DR contains also a RegRep document containing the DE4A Code Data Model (e.g. an Evidence Response) but also the main evidences as so called "attachments". Depending on the number of pieces of evidence requested and the number of formats provided, the number of AS4 attachments in the response may vary.

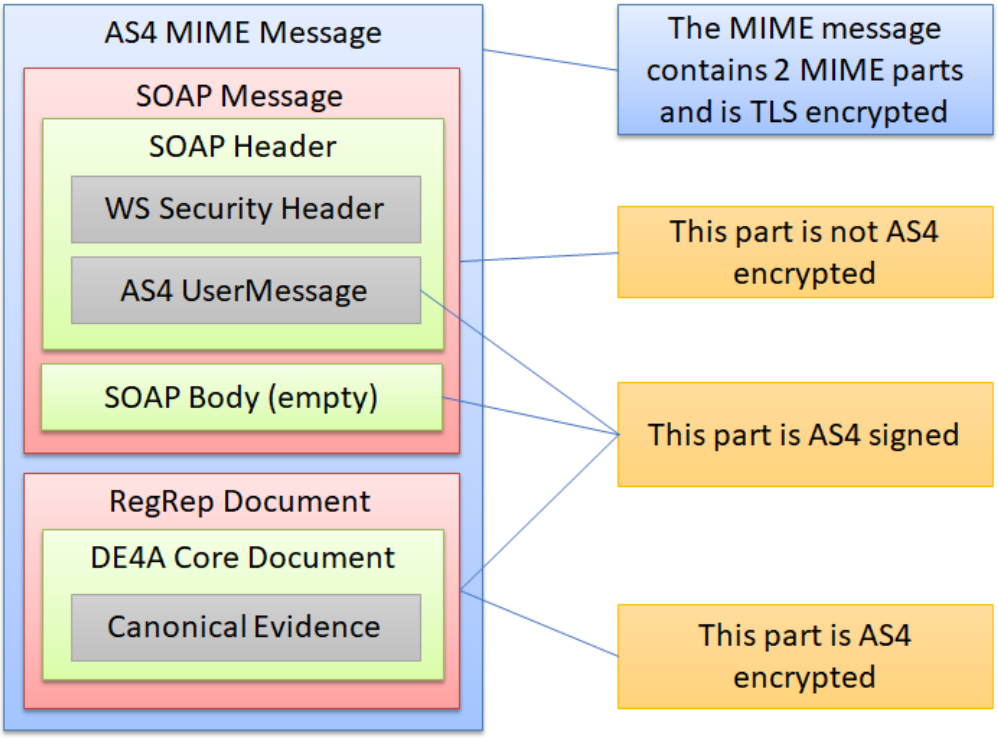

Figure 13: AS4 Message Layout

The above figure shows the structural parts of an AS4 message. The figure depicts a DE4A evidence response that only contains the Canonical Evidence but no other evidence formats. A DE4A evidence request looks very similar – just without the Canonical Evidence.

3.2 RegRep

RegRep is a data format standard by OASIS [17]. It adds another layer of abstraction onto XML schema and splits the data into so called "slots".

The usage of RegRep is very limited to inside the DE4A Connector. Before the main DEA message is send it is wrapped into a thin RegRep wrapper.

3.2.1 RegRep Evidence Request

The RegRep Evidence Request consists of the following slots:

- **SpecificationIdentifier** (of type String) – Defines the underlying specification that is implemented. Must contain the value `de4a-iem-v2`.
- **IssueDateTime** (of type DateTime) – Defines the date and time when request was created.

- **DE4AQuery** (of type Any) – Contains the main DE4A data model

The following example code shows a complete DE4A RegRep query.

```
<query:QueryRequest xmlns:lcm="urn:oasis:names:tc:ebxml-
regrep:xsd:lcm:4.0" xmlns:query="urn:oasis:names:tc:ebxml-
regrep:xsd:query:4.0" xmlns:spi="urn:oasis:names:tc:ebxml-
regrep:xsd:spi:4.0" xmlns:rs="urn:oasis:names:tc:ebxml-
regrep:xsd:rs:4.0" xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:4.0"
id="fa62b109-b564-4985-a322-afb92e82d906">
  <rim:Slot name="SpecificationIdentifier">
    <rim:SlotValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:type="rim:StringValueType">
      <rim:Value>de4a-iem-v2</rim:Value>
    </rim:SlotValue>
  </rim:Slot>
  <rim:Slot name="IssueDateTime">
    <rim:SlotValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:type="rim:DateTimeValueType">
      <rim:Value>2023-01-28T20:10:49.827+01:00</rim:Value>
    </rim:SlotValue>
  </rim:Slot>
  <query:ResponseOption returnType="LeafClassWithRepositoryItem" />
  <query:Query queryDefinition="DE4AQueryIt2">
    <rim:Slot name="DE4AQuery">
      <rim:SlotValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:type="rim:AnyValueType">
        <!-- main DE4A request here -->
      </rim:SlotValue>
    </rim:Slot>
  </query:Query>
</query:QueryRequest>
```

3.2.2 RegRep Evidence Response

The RegRep Evidence Response consists of the following slots:

- **SpecificationIdentifier** (of type String) – Defines the underlying specification that is implemented. Must contain the value `de4a-iem-v2`.
- **IssueDateTime** (of type DateTime) – Defines the date and time when request was created.
- **DE4AResponse** (of type Any) – Contains the main DE4A data model

The following example code shows a complete DE4A RegRep query response.

```
<query:QueryResponse xmlns:lcm="urn:oasis:names:tc:ebxml-
regrep:xsd:lcm:4.0" xmlns:query="urn:oasis:names:tc:ebxml-
```

```
regrep:xsd:query:4.0" xmlns:spi="urn:oasis:names:tc:ebxml-
regrep:xsd:spi:4.0" xmlns:rs="urn:oasis:names:tc:ebxml-
regrep:xsd:rs:4.0" xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:4.0"
status="urn:oasis:names:tc:ebxml-regrep:ResponseStatusType:Success"
requestId="reqid">
  <rim:Slot name="SpecificationIdentifier">
    <rim:SlotValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:type="rim:StringValueType">
      <rim:Value>de4a-iem-v2</rim:Value>
    </rim:SlotValue>
  </rim:Slot>
  <rim:Slot name="IssueDateTime">
    <rim:SlotValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:type="rim:DateTimeValueType">
      <rim:Value>2023-01-28T20:18:46.598+01:00</rim:Value>
    </rim:SlotValue>
  </rim:Slot>
  <rim:RegistryObjectList>
    <rim:RegistryObject id="910a4e63-a08d-4d8b-b736-e94e9098b1fd">
      <rim:Slot name="DE4AResponse">
        <rim:SlotValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:type="rim:AnyValueType">
          <!-- main DE4A response here -->
        </rim:SlotValue>
      </rim:Slot>
    </rim:RegistryObject>
  </rim:RegistryObjectList>
</query:QueryResponse>
```

3.3 Kafka

The Apache Kafka instance used in DE4A is an Open-Source distributed event streaming platform. It is used to receive messages from the different services used in the project. The received messages can be viewed online as they are received in the Package Tracker made available for everyone. The main purpose is to be able to quickly see if there are any problems between services and what type of problem (e.g. connection problems) to help pilot partners and developers find those issues. Therefore, the main use case is distributed event logging and not digital traffic reporting where full logs are sent and stored off-site. The reasoning for this is that the Kafka instance on the Aeioros servers is fully open, meaning that anyone can read and write messages on that service. To reduce security risks, the Kafka instance does not store messages sent for extended periods of time, i.e. messages are essentially ephemeral.

The particular configuration used for the DE4A project uses ZooKeeper as the coordinator, Kafka as the message broker and the Package Tracker as the web service configured to consume and display any messages received by the Kafka broker. The Package Tracker can only display "live" messages, i.e. any messages received in the past will never be displayed.

3.4 Certificates

The existing eDelivery components are designed to work with a single PKI. That means, that all SMP certificates MUST be based on a single SMP root certificate, and all AS4 certificates MUST be based on a single AS4 root certificate. This rule only applies to the SMP and AS4 certificates, but NOT to TLS certificates used for transport security.

The usage of a single root certificate provides an easy way to check if a certificate is valid or not. It requires a functioning OCSP or CRL revocation check to work properly. For a production PKI to function, it needs a strong governance and appropriate controls and measures.

All certificates int the DE4A pilots are based on the “CommisSign2 PKI” provided by the European Commission to the project free of charge. This PKI was used for to create certificates for SMPs and for the AS4 message exchange.

4 Common behaviour and interactions between components

4.1 Communication diagrams of interaction patterns

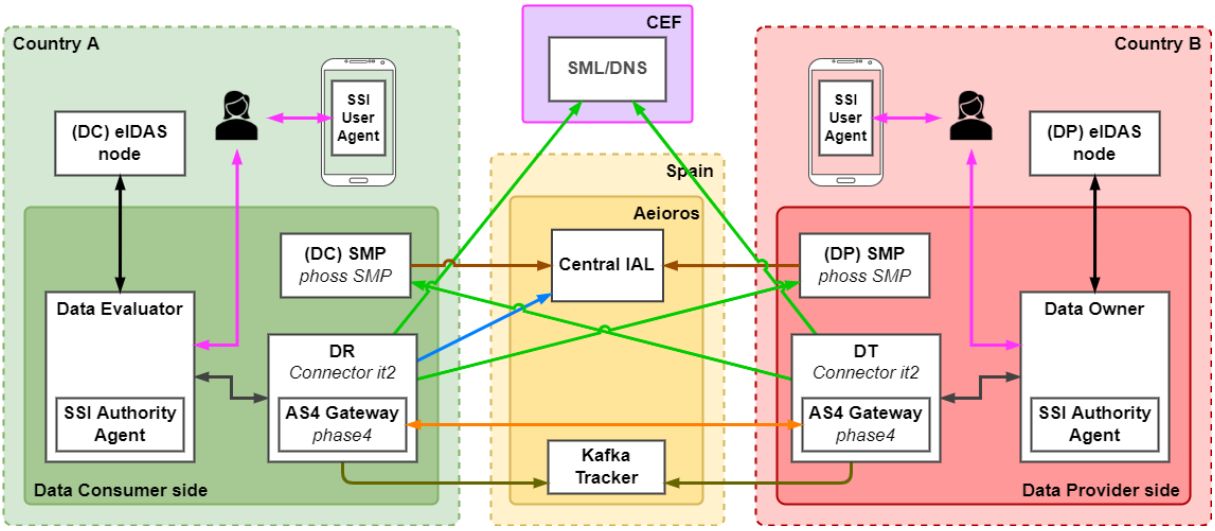

Figure 14: Communication between DE4A components

Figure 14 shows a general view of the communication between components of the DE4A network.

In the following sections, each interaction pattern is described by means of a communication diagram explaining the entire flow, including references to error messages that an exceptional situation may throw. These referenced error messages are available in chapter 7.

4.1.1 Intermediation Pattern (IM)

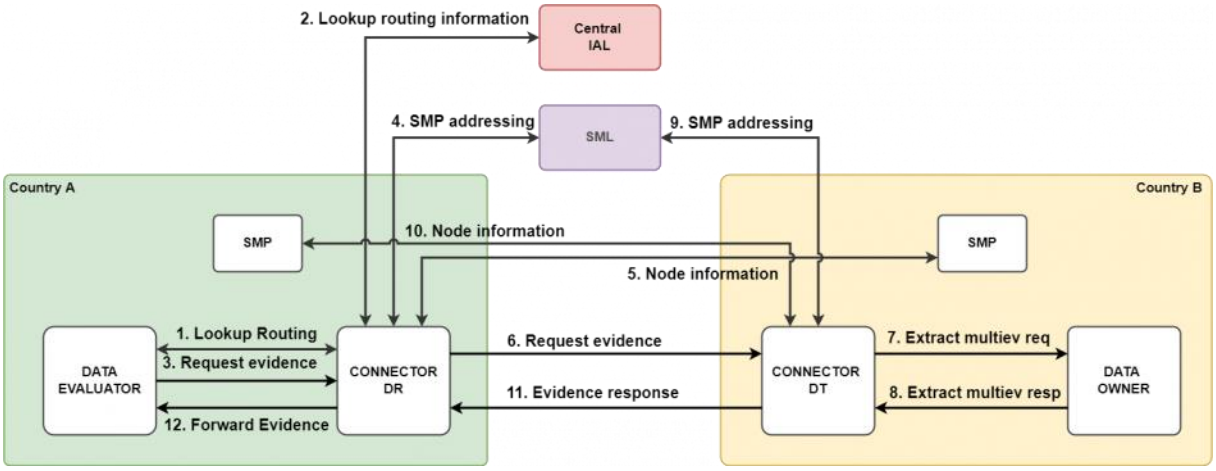

Figure 15: Intermediation pattern communication diagram

In the IM pattern the sequence is very straightforward: the Data Transferor receives the request for

evidence, which is forwarded to the Data Owner; this one recover and returns the evidence, which is sent back to the Data Requestor by the Data Transferor, all in a synchronous sequence.

Figure 15 shows the interaction between components for this intermediation pattern.

- IM pattern sequence:

  - **1. Lookup routing information**

- Request
  - *Description*: The Data Evaluator sends a request to the Connector DR to obtain the Data Owner identifier. A canonical evidence type must be provided, also an optional country code can be specified to obtain the response.
  - *Message type:* Detailed in Central IAL interface section.
- Response
  - *Description:* The Connector DR sends the response to the Data Evaluator.
  - *Message type:* **ResponseLookupRoutingInformationType**. Detailed in Central IAL interface section.
- Error handling
  - If Data Evaluator cannot be reached there will be a connection error. [DRE01]
  - Mandatory fields must be present in the request. [DRE05]

  - **2. Lookup routing information**

- Request
  - *Description:* The Connector DR sends a request to the Central IAL to get the routing information. The request received from the Data Evaluator is sent to the IAL.
  - *Message type:* Detailed in Central IAL interface section.
- Response
  - *Description:* The Central IAL sends a response to the Connector DR providing the routing information.
  - *Message type:* **ResponseLookupRoutingInformationType**. Detailed in Central IAL interface section.
- Error handling
  - If IAL cannot be reached there will be a communication error. [DRE06]
  - If request information does not obtain results, there will be a no data found error. [DRE02]

  - **3. Request evidence**

- Request
  - *Description:* The Data Evaluator sends a request to the Connector DR to retrieve evidence. In the message payload, Data Evaluator and Data Owner identify both components.
  - *Message type:* **RequestExtractMultiEvidenceIMType.**
- Response
  - Description: The asynchronous message will have a response with the result of this communication.

- Message type: **ResponseErrorType**

- Error handling
  - If Connector DR cannot be reached there will be a connection error. [DRE01]

- **4. SMP addressing**

- *Description:* The Connector DR sends a request to the SML to get the SMP address.
- *Message type:* Detailed in eDelivery section.

- **5. Node information**

- Request
  - *Description:* The Connector DR sends a request to the SMP to get gateway information.
  - *Message type:* Detailed in eDelivery section.
- Response
  - *Description:* The SMP sends a response to the Connector DR providing the gateway information. The destination endpoint and the certificate will be provided in the response.
  - *Message type:* Detailed in eDelivery section.
- Error handling
  - If the SMP cannot be reached there will be a connection error. [DRE01]
  - Mandatory fields must be present in the request. [DRE05]
  - If request information does not obtain results, there will be a no data found error. [DRE02].

- **6. Request evidence**

- Request
  - *Description:* The Connector DR sends a request to the Connector DT to retrieve the evidence.
  - *Message type:* **RequestExtractMultiEvidenceIMType**
- Response
  - Description: The asynchronous message will have a response with the result of this communication.
  - Message type: **ResponseErrorType**

- Error handling
  - If Connector DT cannot be reached there will be a connection error. [DRE01]

- **7. Extract multi-evidence request**

- Request
  - *Description*: The Connector DT sends a request to the Data Owner to retrieve the evidence.
  - *Message type:* **RequestExtractMultiEvidenceIMType**
- Response
  - Description: The asynchronous message will have a response with the result of this communication.
  - Message type: **ResponseErrorType**

- Error handling
  - If the Data Owner cannot be reached there will be a connection error. [DTE01]
  - If request information does not obtain results, there will be a no data found error. [DTE02]

- **8. Extract multi-evidence response**

- Request
  - *Description:* The Data Owner sends a response to the Connector DT with the evidence.

▸ *Message type:* **ResponseExtractMultiEvidenceType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
  - Message type: **ResponseErrorType**

- Error handling
  - If the evidence cannot be created there will be an error extracting the evidence. [DOE02]
  - If the evidence is not available when requested there will be an error. [DOE03]
  - If the identity cannot be matched there will be an identity matching error. [DOE04]
  - If the preview cannot be completed there will be an error. [DOE05]
  - If the user identity cannot be re-established there will be an error. [DOE06]
  - If the preview is rejected by the user there will be an error. [DOE07]
  - If the canonical evidence cannot be generated there will be an error. [DOE08]

- **9. SMP addressing**

Description: Connector DT sends a request to the SML to get the SMP address. Data Evaluator identifier will be sent as the participant identifier.

- **10. Node information**

- Request
  - *Description:* The Connector DT sends a request to the SMP to get gateway information.
  - *Message type:* Detailed in eDelivery section.
- Response
  - *Description:* The SMP sends a response to the Connector DT providing the gateway information. The destination endpoint and the certificate will be provided in the response.
  - *Message type:* Detailed in eDelivery section.
- Error handling
  - If the SMP cannot be reached there will be a connection error. [DRE01]
  - Mandatory fields must be present in the request. [DRE05]
  - If request information does not obtain results, there will be a no data found error. [DRE02].

Detailed information is available in 'eDelivery BDXL' documentation.

- **11. Evidence response**

- Request
  - *Description:* The Connector DT sends a response to the Connector DR with the evidence information.

▸ *Message Type:* **ResponseExtractMultiEvidenceType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If the Connector DR cannot be reached there will be a connection error. [DTE01]

- **12. Forward evidence**

- Request
  - *Description:* The Connector DR sends the response to the Data Evaluator with the evidence information.

▸ *Message Type:* **ResponseExtractMultiEvidenceType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If Data Evaluator cannot be reached there will be a connection error. [DRE01]

For more detail about error codes go to section 7 DE4A Logs and error messages.

4.1.2 Backwards compatibility layer for the Intermediation Pattern IT1

The DE4A Connector offers a **backwards compatibility feature** for the former synchronous IM pattern. It can handle the old messaging structure, so the synchronous implementation is fully compatible with the current asynchronous version.

When a Data Evaluator sends an old message request (it1 structure), the connector DR transforms it into a new message request (it2 structure) to be exchanged between connectors. After that, before sending the request to the Data Owner, the message is transformed to its original form (it1 structure). After retrieving the evidence from the Data Owner, the connector DT will transform again the response message to the new structure and will send back the evidence to the connector DT and this one will transform back the response to an old message response (it1 structure) to send the response to the Data Evaluator.

These transformations allow the pilots remaining in the it1 messaging structure to work with the new common components without adopting the it2 message exchange.

4.1.3 User-Supported Intermediation Pattern (USI)

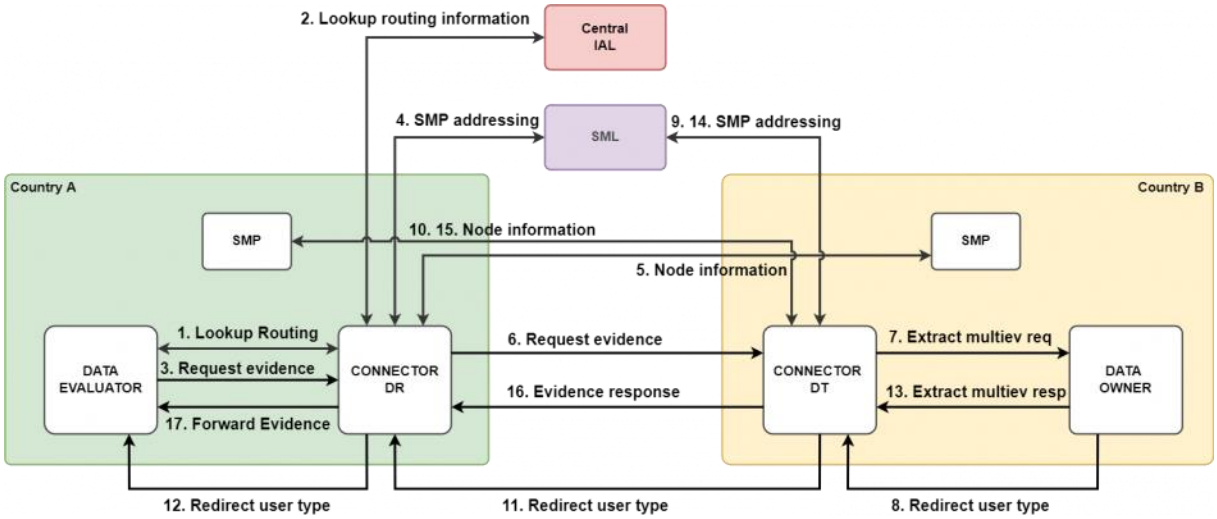

Figure 16: User-supported intermediation pattern communication diagram

In the USI pattern again the request for evidence is sent to the Data Provider. However, in the case of the USI pattern, the evidence must be previewed by the User on the Data Provider side, rather than on the Data Consumer side as in the Intermediation pattern. Therefore, the evidence is stored by the Data Owner pending the arrival of the User at its portal. When the User arrives and has been identified by the Data Owner, the preview is presented to them. Their response, acceptance or rejection, is then sent to the Data Consumer, along with the evidence (if approved). The User is also redirected back to the Data Evaluator's portal.

The Figure 16 shows the interaction between common components for this user-supported intermediation pattern.

- USI pattern sequence:

  - **1. Lookup routing information**

- Request
  - *Description*: The Data Evaluator sends a request to the Connector DR to obtain the Data Owner identifier. A canonical evidence type must be provided, also an optional country code can be specified to obtain the response.
  - *Message type:* Detailed in Central IAL interface section.
- Response
  - *Description:* The Connector DR sends the response to the Data Evaluator.
  - *Message type:* **ResponseLookupRoutingInformationType.** Detailed in Central IAL interface section.
- Error handling
  - If Data Evaluator cannot be reached there will be a connection error. [DRE01]
  - Mandatory fields must be present in the request. [DRE05]

- **2. Lookup routing information**

- Request
  - *Description:* The Connector DR sends a request to the Central IAL to get the routing information. The request received from the Data Evaluator is sent to the IAL.
  - *Message type:* Detailed in Central IAL interface section.
- Response
  - *Description:* The Central IAL sends a response to the Connector DR providing the routing information.
  - *Message type:* **ResponseLookupRoutingInformationType.** Detailed in Central IAL interface section.
- Error handling
  - If IAL cannot be reached there will be a communication error. [DRE06]
  - If request information does not obtain results, there will be a no data found error. [DRE02]

- **3. Request evidence**

- Request
  - *Description:* The Data Evaluator sends a request to the Connector DR to retrieve evidence. In the message payload, Data Evaluator and Data Owner identify both components.

Message type:

▸ **RequestExtractMultiEvidenceUSIType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
  - Message type: **ResponseErrorType**

- Error handling
  - If Connector DR cannot be reached there will be a connection error. [DRE01]

- **4. SMP addressing**

- *Description:* The Connector DR sends a request to the SML to get the SMP address.
- *Message type:* Detailed in eDelivery section.

- **5. Node information**

- Request
  - *Description:* The Connector DR sends a request to the SMP to get gateway information.
  - *Message type:* Detailed in eDelivery section.
- Response
  - *Description:* The SMP sends a response to the Connector DR providing the gateway information. The destination endpoint and the certificate will be provided in the response.
  - *Message type:* Detailed in eDelivery section.

- Error handling
  - If the SMP cannot be reached there will be a connection error. [DRE01]
  - Mandatory fields must be present in the request. [DRE05]
  - If request information does not obtain results, there will be a no data found error. [DRE02].

- **6. Request evidence**

- Request
  - *Description:* The Connector DR sends a request to the Connector DT to retrieve the evidence.

Message type:

▸ **RequestExtractMultiEvidenceUSIType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If Connector DT cannot be reached there will be a connection error. [DRE01]

- **7. Extract multi-evidence request**

- Request
  - *Description*: The Connector DT sends a request to the Data Owner to retrieve the evidence.

Message type:

▸ **RequestExtractMultiEvidenceUSIType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If the Data Owner cannot be reached there will be a connection error. [DTE01]
  - If request information does not obtain results, there will be a no data found error. [DTE02]

- **8. Redirect user type (DO to DT)**

- Request
  - *Description:* The Data Owner sends redirection URL to the Connector DT.
- *Message type:* **RedirectUserType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.

- Message type: **ResponseErrorType**

- Error handling
  - If the Connector DT cannot be reached there will be a connection error. [DOE01]

- **9. SMP addressing**
- *Description:* The Connector DR sends a request to the SML to get the SMP address. Detailed information is available in eDelivery section.
- *Message type:* Detailed in eDelivery section.

- **10. Node information**
- Request
  - *Description:* The Connector DR sends a request to the SMP to get gateway information.
  - *Message type:* Detailed in eDelivery section.
- Response
  - *Description:* The SMP sends a response to the Connector DR providing the gateway information. The destination endpoint and the certificate will be provided in the response.
  - *Message type:* Detailed in eDelivery section.
- Error handling
  - If the SMP cannot be reached there will be a connection error. [DRE01]
  - Mandatory fields must be present in the request. [DRE05]
  - If request information does not obtain results, there will be a no data found error. [DRE02].

- **11. Redirect user type (DT to DR)**
- Request
  - *Description:* The Connector DT sends redirection URL to the Connector DR.
- *Message type:* **RedirectUserType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If the Data Evaluator cannot be reached there will be a communication error. [DTE01]
  - Mandatory fields must be present in the request. [DTE05]

- **12. Redirect user type (DR to DE)**
- Request
  - *Description:* The Connector DR sends redirection URL to the Data Evaluator.
- *Message Type:* **RedirectUserType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
  - Message type: **ResponseErrorType**

- Error handling
  - If the Data Evaluator cannot be reached there will be a communication error. [DRE01]
  - Mandatory fields must be present in the request. [DRE05]

- **13. Extract multi-evidence response**

- Request
  - *Description:* The Data Owner sends a response to the Connector DT with the evidence.

▸ *Message type:* **ResponseExtractMultiEvidenceType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
  - Message type: **ResponseErrorType**

- Error handling
  - If the evidence cannot be created there will be an error extracting the evidence. [DOE02]
  - If the evidence is not available when requested there will be an error. [DOE03]
  - If the identity cannot be matched there will be an identity matching error. [DOE04]
  - If the preview cannot be completed there will be an error. [DOE05]
  - If the user identity cannot be re-established there will be an error. [DOE06]
  - If the preview is rejected by the user there will be an error. [DOE07]
  - If the canonical evidence cannot be generated there will be an error. [DOE08]

- **14. SMP addressing**

- *Description:* The Connector DT sends a request to the SML to get the SMP address.
- *Message type:* Detailed in eDelivery section.

- **15. Node information**

- Request
  - *Description:* The Connector DT sends a request to the SMP to get gateway information.
  - *Message type:* Detailed in eDelivery section.
- Response
  - *Description:* The SMP sends a response to the Connector DT providing the gateway information. The destination endpoint and the certificate will be provided in the response.
  - *Message type:* Detailed in eDelivery section.
- Error handling
  - If the SMP cannot be reached there will be a connection error. [DTE01]

- Mandatory fields must be present in the request. [DTE05]
- If request information does not obtain results, there will be a no data found error. [DTE02].

- **16. Evidence response**

- Request
  - *Description:* The Connector DT sends a response to the Connector DR with the evidence information.

▸ *Message Type:* **ResponseExtractMultiEvidenceType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If the Connector DR cannot be reached there will be a connection error. [DTE01]

- **17. Forward evidence**

- Request
  - *Description:* The Connector DR sends the response to the Data Evaluator with the evidence information.

▸ *Message Type:* **ResponseExtractMultiEvidenceType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If Data Evaluator cannot be reached there will be a connection error. [DRE01]

For more detail about error codes go to section 7 DE4A Logs and error messages.

4.1.4 Subscription & Notification Pattern (S&N)

After receiving evidence from a Data Owner (DO), it can be essential for a Data Evaluator (DE) to be informed on changes regarding the subject of this evidence to be able to take appropriate action. The goal of this interaction pattern is to allow the DE to subscribe to a service of the DO that provides automatic and regular notifications. The cross-border message exchange for the subscriptions and notifications are put in the responsibility of the Data Requester (DR) and the Data Transferor (DT) respectively to allow an easier distribution of responsibility on national level, i.e. to intermediary platforms and national gateway providers.

There are two distinct purposes, or business requirements for Subscription and Notification, both of which are relevant for the DE4A Doing Business Abroad Pilot: Evidence update notification and Event notification.

- Evidence Update Notification

The goal is to keep previously shared evidence data that is stored at the DE up to date.

Description: Data may change in the base register. In case the DE wants an exact copy of the evidence data on record, they need to be notified in case the data has changed in the base registry.

- Business or Life Event Notification

The goal is to assess the impact of changes to the subject (e.g., company) on the public services provided by the data evaluator.

Description: Some public services oblige the subject (i.e., company or citizen) to continue in a specific situation or state to remain entitled to the benefits of the public service provided. An agricultural company may, for example, receive a subsidy for its permanent pasture. As a prerequisite, the company must preserve the pasture for five consecutive years. The data evaluator needs to be notified of the company ending its operations and hence not meeting the five-year requirement. "Ending its operation" is an example of a business event. Other examples are: going bankrupt, a merger, etc.

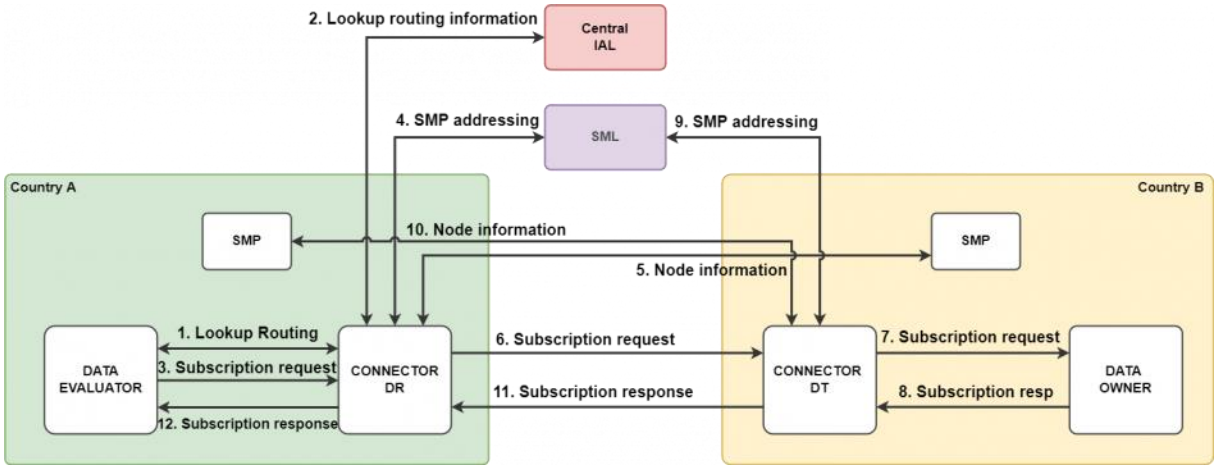

Figure 17: Subscription pattern communication diagram

- Subscription pattern sequence:

- **1. Lookup routing information**

- Request
  - *Description*: The Data Evaluator sends a request to the Connector DR to obtain the Data Owner identifier. A canonical evidence type must be provided, also an optional country code can be specified to obtain the response.
  - *Message type:* Detailed in Central IAL interface section.
- Response
  - *Description:* The Connector DR sends the response to the Data Evaluator.
  - *Message type:* **ResponseLookupRoutingInformationType**. Detailed in Central IAL interface section.
- Error handling

- If Data Evaluator cannot be reached there will be a connection error. [DRE01]
- Mandatory fields must be present in the request. [DRE05]

- **2. Lookup routing information**

- Request
  - *Description:* The Connector DR sends a request to the Central IAL to get the routing information. The request received from the Data Evaluator is sent to the IAL.
  - *Message type:* Detailed in Central IAL interface section.
- Response
  - *Description:* The Central IAL sends a response to the Connector DR providing the routing information.
  - *Message type:* **ResponseLookupRoutingInformationType**. Detailed in Central IAL interface section.
- Error handling
  - If IAL cannot be reached there will be a communication error. [DRE06]
  - If request information does not obtain results, there will be a no data found error. [DRE02]

- **3. Subscription request**

- Request
  - *Description:* The Data Evaluator sends a subscription request to the Connector DR. In the message payload, Data Evaluator and Data Owner identify both components.
- *Message type:* **RequestEventSubscriptionType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If Connector DR cannot be reached there will be a connection error. [DRE01]

- **4. SMP addressing**
  - *Description:* The Connector DR sends a request to the SML to get the SMP address.
  - *Message type:* Detailed in eDelivery section.

- **5. Node information**

- Request
  - *Description:* The Connector DR sends a request to the SMP to get gateway information.
  - *Message type:* Detailed in eDelivery section.
- Response
  - *Description:* The SMP sends a response to the Connector DR providing the gateway information. The destination endpoint and the certificate will be provided in the response.

- *Message type:* Detailed in eDelivery section.

- Error handling
  - If the SMP cannot be reached there will be a connection error. [DRE01]
  - Mandatory fields must be present in the request. [DRE05]
  - If request information does not obtain results, there will be a no data found error. [DRE02].

- **6. Subscription request**

- Request
  - *Description:* The Connector DR sends a subscription request to the Connector DT. In the message payload, Data Evaluator and Data Owner identify both components.
- *Message type:* **RequestEventSubscriptionType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If Connector DT cannot be reached there will be a connection error. [DRE01]

- **7. Subscription request**

- Request
  - *Description*: The Connector DT sends a subscription request to the Data Owner.
- *Message type:* **RequestEventSubscriptionType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If the Data Owner cannot be reached there will be a connection error. [DTE01]
  - If request information does not obtain results, there will be a no data found error. [DTE02]

- **8. Subscription response**

- Request
  - *Description:* The Data Owner sends a response to the Connector DT confirming the subscription.

- *Message type:* **ResponseEventSubscriptionType**

- Response

- Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If the subscription cannot be created there will be an error. [DOE02]
  - If the subscription is not available when requested there will be an error. [DOE03]
  - If the identity cannot be matched there will be an identity matching error. [DOE04]
  - If the preview cannot be completed there will be an error. [DOE05]
  - If the user identity cannot be re-established there will be an error. [DOE06]
  - If the preview is rejected by the user there will be an error. [DOE07]

- **9. SMP addressing**

- *Description:* The Connector DR sends a request to the SML to get the SMP address. Detailed information is available in eDelivery section.
- *Message type:* Detailed in eDelivery section.

- **10. Node information**

- Request
  - *Description:* The Connector DT sends a request to the SMP to get gateway information.
  - *Message type:* Detailed in eDelivery section.
- Response
  - *Description:* The SMP sends a response to the Connector DT providing the gateway information. The destination endpoint and the certificate will be provided in the response.
  - *Message type:* Detailed in eDelivery section.
- Error handling
  - If the SMP cannot be reached there will be a connection error. [DTE01]
  - Mandatory fields must be present in the request. [DTE05]
  - If request information does not obtain results, there will be a no data found error. [DTE02].

- **11. Subscription response**

- Request
  - *Description:* The Connector DT sends a response to the Connector DR confirming the subscription.

▸ *Message type:* **ResponseEventSubscriptionType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling

- If Connector DR cannot be reached there will be a connection error. [DTE01]

- **12. Subscription response**

- Request
  - *Description:* The Connector DR sends a response to the Data Evaluator confirming the subscription.

▸ *Message type:* **ResponseEventSubscriptionType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.

- Message type: **ResponseErrorType**

- Error handling
  - If Connector DR cannot be reached there will be a connection error. [DRE01]

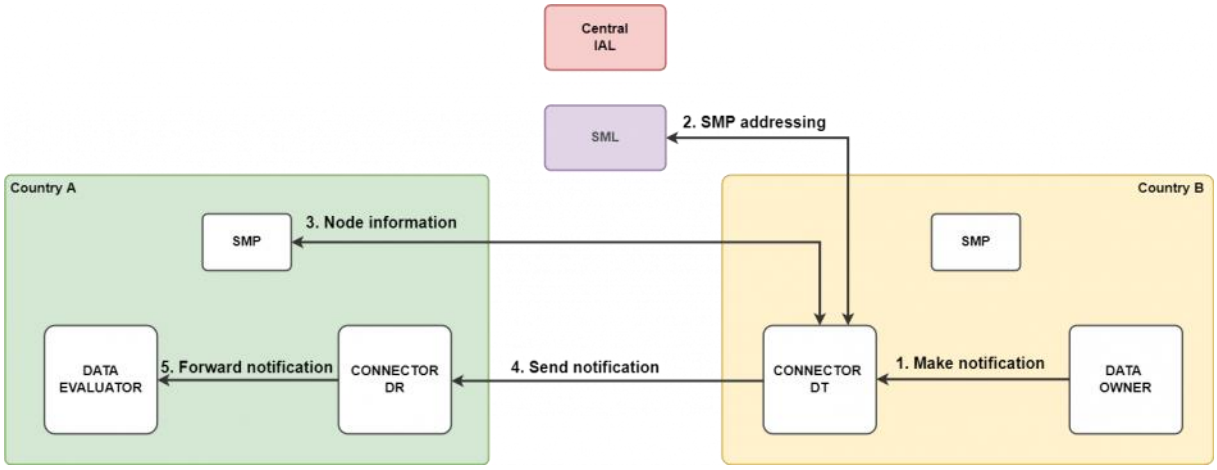

Figure 18: Notification pattern communication diagram

▸ Notification pattern sequence:

- **1. Make notification**

- Request
  - *Description*: The Data Owner sends a message to the Connector DT with the notification. In the message payload, Data Evaluator and Data Owner identify both components.

▸ *Message type*: **EventNotificationType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.

- Message type: **ResponseErrorType**

- Error handling
  - If the Connector DT cannot be reached there will be a connection error. [DOE01]

- **2. SMP addressing**

- *Description:* The Connector DT sends a request to the SML to get the SMP address.

Detailed information is available in eDelivery section.

- *Message type:* Detailed in eDelivery section.

- **3. Node information**

- Request
  - *Description:* The Connector DT sends a request to the SMP to get gateway information.
  - *Message type:* Detailed in eDelivery section.
- Response
  - *Description:* The SMP sends a response to the Connector DT providing the gateway information. The destination endpoint and the certificate will be provided in the response.
  - *Message type:* Detailed in eDelivery section.
- Error handling
  - If the SMP cannot be reached there will be a connection error. [DTE01]
  - Mandatory fields must be present in the request. [DTE05]
  - If request information does not obtain results, there will be a no data found error. [DTE02].

- **4. Send notification**

- *Description:* The Connector DT sends a message to the Connector DR with the notification.

▸ *Message type:* **EventNotificationType**

- Error handling
  - If the Connector DR cannot be reached there will be a connection error. [DTE01]

- **5. Forward notification**

- Request
  - *Description:* The Connector DR sends a message to the Data Evaluator with the notification.

▸ *Message type:* **EventNotificationType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling

- If Data Evaluator cannot be reached there will be a connection error. [DRE01]

For more detail about error codes go to section 7 DE4A Logs and error messages.

4.1.5 Lookup Pattern

The basic logic of the Lookup pattern is a simple Request-Response interaction between DC and DP without any user involvement. This is only applicable in cases where the exchange has a legal basis and can be executed without explicit request or consent from the User. Its main characteristic is online and near real-time (NRT) use of information.

Once is established that a lookup of the evidence is needed, e.g., via a notification from the DP to DC (see for instance the Subscription and Notification Pattern), the evidence can be retrieved in its entirety.

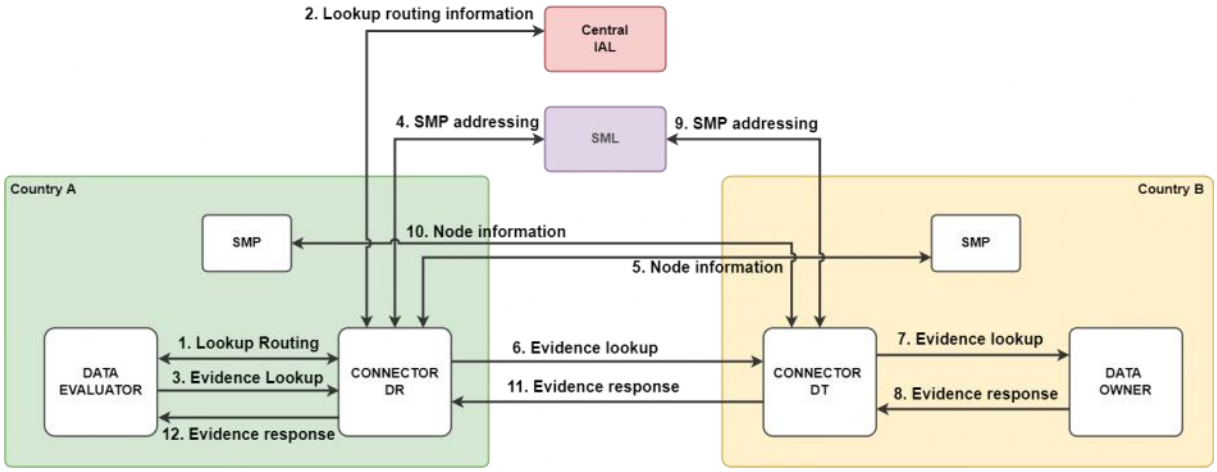

Figure 19: Lookup pattern communication diagram

- Lookup pattern sequence:

  - **1. Lookup routing information**

- Request
  - *Description*: The Data Evaluator sends a request to the Connector DR to obtain the Data Owner identifier. A canonical evidence type must be provided, also an optional country code can be specified to obtain the response.
  - *Message type:* Detailed in Central IAL interface section.
- Response
  - *Description:* The Connector DR sends the response to the Data Evaluator.
  - *Message type:* **ResponseLookupRoutingInformationType**. Detailed in Central IAL interface section.
- Error handling
  - If Data Evaluator cannot be reached there will be a connection error. [DRE01]
  - Mandatory fields must be present in the request. [DRE05]

  - **2. Lookup routing information**

- Request
  - *Description:* The Connector DR sends a request to the Central IAL to get the routing information. The request received from the Data Evaluator is sent to the IAL.
  - *Message type:* Detailed in Central IAL interface section.
- Response
  - *Description:* The Central IAL sends a response to the Connector DR providing the routing information.
  - *Message type:* **ResponseLookupRoutingInformationType**. Detailed in Central IAL interface section.
- Error handling
  - If IAL cannot be reached there will be a communication error. [DRE06]
  - If request information does not obtain results, there will be a no data found error. [DRE02]

- **3. Evidence lookup**

- Request
  - *Description:* The Data Evaluator sends a request to the Connector DR to get the updated evidence. In the message payload, Data Evaluator and Data Owner identify both components.

▸ *Message type:* **RequestExtractMultiEvidenceLUType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If Connector DR cannot be reached there will be a connection error. [DRE01]

- **4. SMP addressing**

- *Description:* The Connector DR sends a request to the SML to get the SMP address. Detailed information is available in eDelivery section.
- *Message type:* Detailed in eDelivery section.

- **5. Node information**

- Request
  - *Description:* The Connector DR sends a request to the SMP to get gateway information. Multi evidence approach is available in eDelivery section.
  - *Message type:* Detailed in eDelivery section.
- Response
  - *Description:* The SMP sends a response to the Connector DR providing the gateway information. The destination endpoint and the certificate will be provided in the response.
  - *Message type:* Detailed in eDelivery section.

- Error handling
  - If the SMP cannot be reached there will be a connection error. [DRE01]
  - Mandatory fields must be present in the request. [DRE05]
  - If request information does not obtain results, there will be a no data found error. [DRE02].

- **6. Evidence lookup**

- Request
  - *Description:* The Connector DR sends a request to the Connector DT to retrieve the updated evidence.

▸ *Message type:* **RequestExtractMultiEvidenceLUType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.

- Message type: **ResponseErrorType**

- Error handling
  - If Connector DT cannot be reached there will be a connection error. [DRE01]

- **7. Evidence lookup**

- Request
  - *Description*: The Connector DT sends a request to the Data Owner to retrieve the updated evidence.

▸ *Message type:* **RequestExtractMultiEvidenceLUType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.

- Message type: **ResponseErrorType**

- Error handling
  - If the Data Owner cannot be reached there will be a connection error. [DTE01]
  - If request information does not obtain results, there will be a no data found error. [DTE02]

- **8. Evidence response**

- Request
  - *Description:* The Data Owner sends a response to the Connector DT with the updated evidence.

▸ *Message type:* **ResponseExtractMultiEvidenceType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.

- Message type: **ResponseErrorType**

- Error handling
  - If the evidence cannot be created there will be an error extracting the evidence. [DOE01]
  - If the evidence is not available when requested there will be an error. [DOE02]
  - If the identity cannot be matched there will be an identity matching error. [DOE03]
  - If the preview cannot be completed there will be an error. [DOE04]
  - If the user identity cannot be re-established there will be an error. [DOE05]
  - If the preview is rejected by the user there will be an error. [DOE06]
  - If the canonical evidence cannot be generated there will be an error. [DOE07]

- **9. SMP addressing**

- *Description:* Connector DT sends a request to the SML to get the SMP address. Data Evaluator identifier will be sent as the participant identifier.

- **10. Node information**

- Request
  - *Description:* The Connector DT sends a request to the SMP to get gateway information. Multi evidence approach is available in eDelivery section.
  - *Message type:* Detailed in eDelivery section.
- Response
  - *Description:* The SMP sends a response to the Connector DT providing the gateway information. The destination endpoint and the certificate will be provided in the response.
  - *Message type:* Detailed in eDelivery section.
- Error handling
  - If the SMP cannot be reached there will be a connection error. [DRE01]
  - Mandatory fields must be present in the request. [DRE05]
  - If request information does not obtain results, there will be a no data found error. [DRE02].

Detailed information is available in eDelivery section.

- **11. Evidence response**

- Request
  - *Description:* The Connector DT sends a response to the Connector DR with the evidence information.

▸ *Message Type:* **ResponseExtractMultiEvidenceType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If the Connector DR cannot be reached there will be a connection error. [DTE01]

- **12. Evidence response**
- Request
  - *Description:* The Connector DR sends the response to the Data Evaluator with the evidence information.

▸ *Message Type:* **ResponseExtractMultiEvidenceType**

- Response
  - Description: The asynchronous message will have a response with the result of this communication.
- Message type: **ResponseErrorType**

- Error handling
  - If Data Evaluator cannot be reached there will be a connection error. [DRE01]

4.2 Data dictionary

This section provides details about the messages exchanged in every step of the patterns. More detailed information is available in the schema definition files.

▸ **RequestLookupRoutingInformationType**

All the necessary parameters will be in the URL. It will be a comma-separated canonical evidence identifier collection (up to 20) followed by the AtuCode.
Detailed in Central IAL interface section.

▸ **ResponseLookupRoutingInformationType**

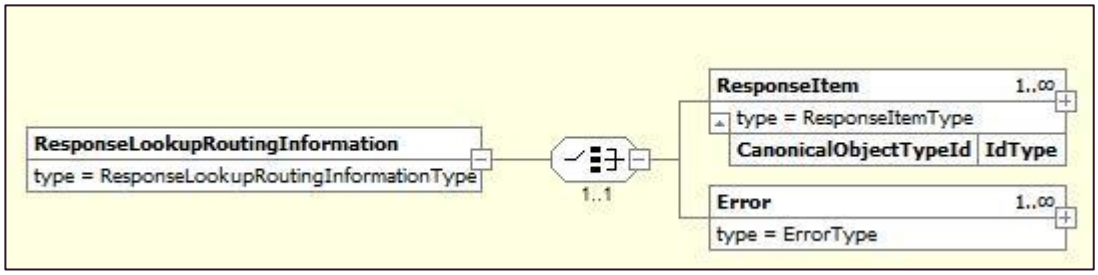

Figure 20: ResponseLookupRoutingInformationType diagram

Table 1: ResponseLookupRoutingInformationType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
ResponseItem	- **ResponseItemType**	1..n	Y/Y	Contains a list of available sources obtained.
Error	- **ErrorType**	1..n	Y/Y	Contains a list with the descriptions of the errors if one or more have occurred.

For more detail about error codes go to section 7 DE4A Logs and error messages.

- **ResponseItemType**

Location: //ResponseLookupRoutingInformation/ResponseItemType

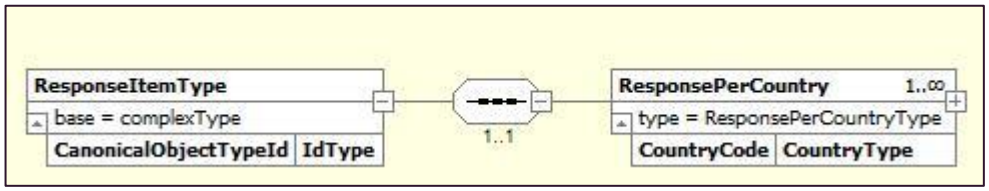

Figure 21: ResponseItemType diagram

Table 2: ResponseItemType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
ResponsePerCountry	- **ResponsePerCountryType**	1..n	Y/Y	Contains the data for each source obtained.

- **ResponsePerCountryType**

Location://ResponseLookupRoutingInformation/ResponseItemType/ResponsePerCountryType

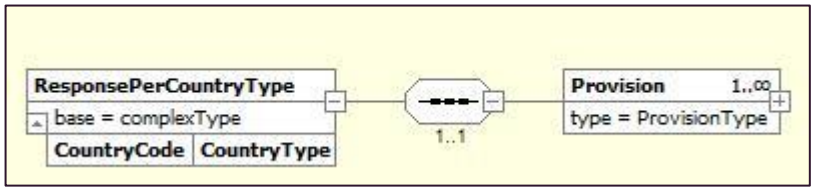

Figure 22: ResponsePerCountryType diagram

Table 3: ResponsePerCountryType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
CountryCode	String	1	Y/Y	The two-letter code of the DP country. Format: ISO 3166-1 alfa-2 https://www.iso.org/obp/ui/
Provision	- **ProvisionType**	1..n	Y/Y	Contains the data of each possible provision.

- **ProvisionType**

Location://ResponseLookupRoutingInformation/ResponseItemType/ResponsePerCountryType/Provision

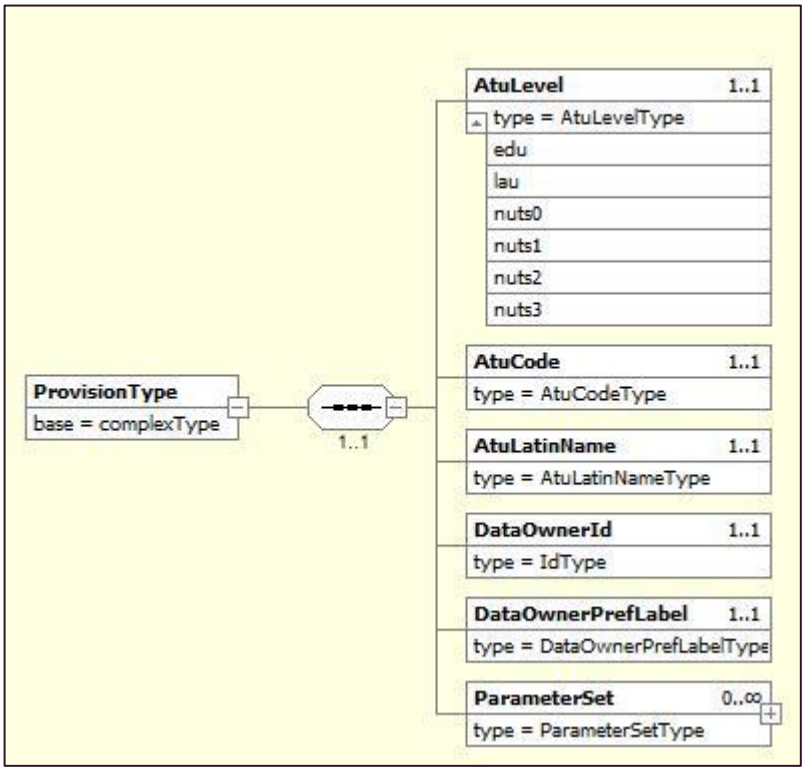

Figure 23: ProvisionType diagram

Table 4: ProvisionType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
AtuLevel	String	1	N/Y	Level of Administrative Territorial Unit. Allowed values: nuts0: National level nuts1: Regional/Province level nuts2: Regional/Province level nuts3: Regional/Province level lau: Local level edu: Educational level
AtuCode	String	1	N/Y	Code of the Administrative Territorial Unit. Allowed values: "ISO 3166-1 alfa-2" for nuts0 "Region" for nuts1-3 "Municipality" for lau

Name	Type	Cardinality	Mandatory Field / Label	Description
				"Educational Organization" for edu
AtuLatinName	String	1	N/Y	Name of the Administrative Territorial Unit.
DataOwnerId	anyURI	1	N/Y	Identifies the Data Owner in iso6523 format. It defines Party Id for all participants. It must be unique. **Format:** {PEPPOL ParticipantIdentifierScheme}::{Domain Scheme}:{Party Id} → iso6523-actorid-upis::[0-9][0-9][0-9][0-9]:[A-Z0-9]{1,15} **{PEPPOL ParticipantIdentifierScheme}:** Must be the constant "iso6523-actorid-upis" **{Domain Scheme}:** Identifies ICD value, must be a 4-digit UNIQUE number in PEPPOL **{Party Id}:** Identifies DE4A Partner, consisting of: 2 letters for country 2 digits for entity type → 00 if entity is a DE4A partner, 88 if not; for regional or local administrations, the entity type is 99 7 digits for DE4A Partner. For regional or local administrations, it is the NUTS/LAU code without country letters and starting with as many zeros on the left as necessary until reaching 7 digits. Example of the Spanish Municipality of San Sebastian with LAU code 20069: **ES990020069** Example of the Dutch Province of Groningen with NUTS code NL11: **NL990000011** https://docs.peppol.eu/edelivery/policies/PEPPOL-EDN-Policy-for-use-of-identifiers-4.0-2019-01-28.pdf For further information on the Domain Scheme values, see the Annex I. Lists of codes

Name	Type	Cardinality	Mandatory Field / Label	Description
DataOwnerPrefLabel	String	1	N/Y	Name of the issuing DO from which the evidence is obtained.
ParameterSet	- **ParameterSetType**	0..n	Y/Y	Contains a list of necessary parameters for the requested evidence.

- **ParameterSetType**

Location://ResponseLookupRoutingInformation/ResponseItemType/ResponsePerCountryType/Provision/ProvisionType

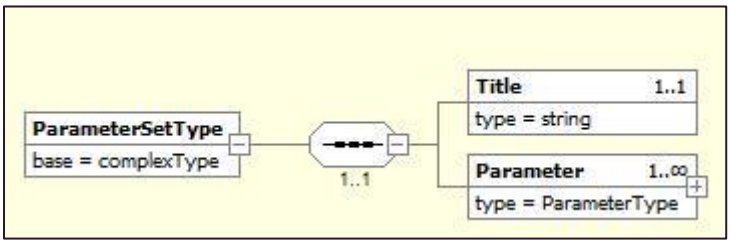

Figure 24: ParameterSetType diagram

Table 5: ParameterSetType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
Title	String	1	Y/Y	Name of the required set of parameters for the requested evidence.
Parameter	- **ParameterType**	1..n	Y/Y	Contains the data for each necessary parameter of the requested evidence.

- **ParameterType**

Location://ResponseLookupRoutingInformation/AvailableSources/Source/ProvisionItems/ProvisionItem/ParameterSet

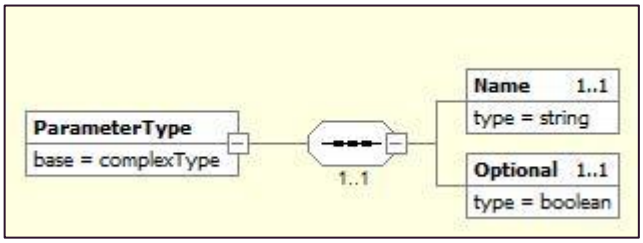

Figure 25: ParameterType diagram

Table 6: ParameterType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
Name	String	1	Y/Y	Name of the parameter
Optional	Boolean	1	Y/Y	Indicates whether the parameter is optional or not

- **ErrorType**

Location: //ResponseLookupRoutingInformation

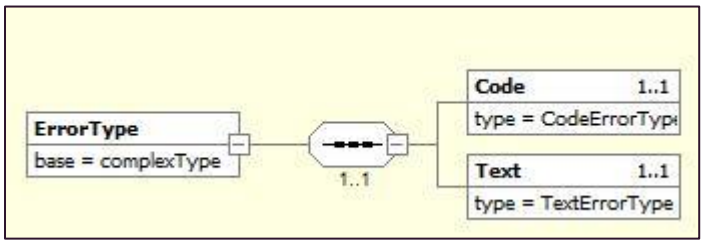

Figure 26: ErrorType diagram

Table 7: ErrorType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
Code	String	1	Y/Y	Code of the error occurred.
Text	String	1	Y/Y	Text defining the above code.

- **RequestExtractMultiEvidenceIMType**
  Location: //RequestExtractMultiEvidenceIMType

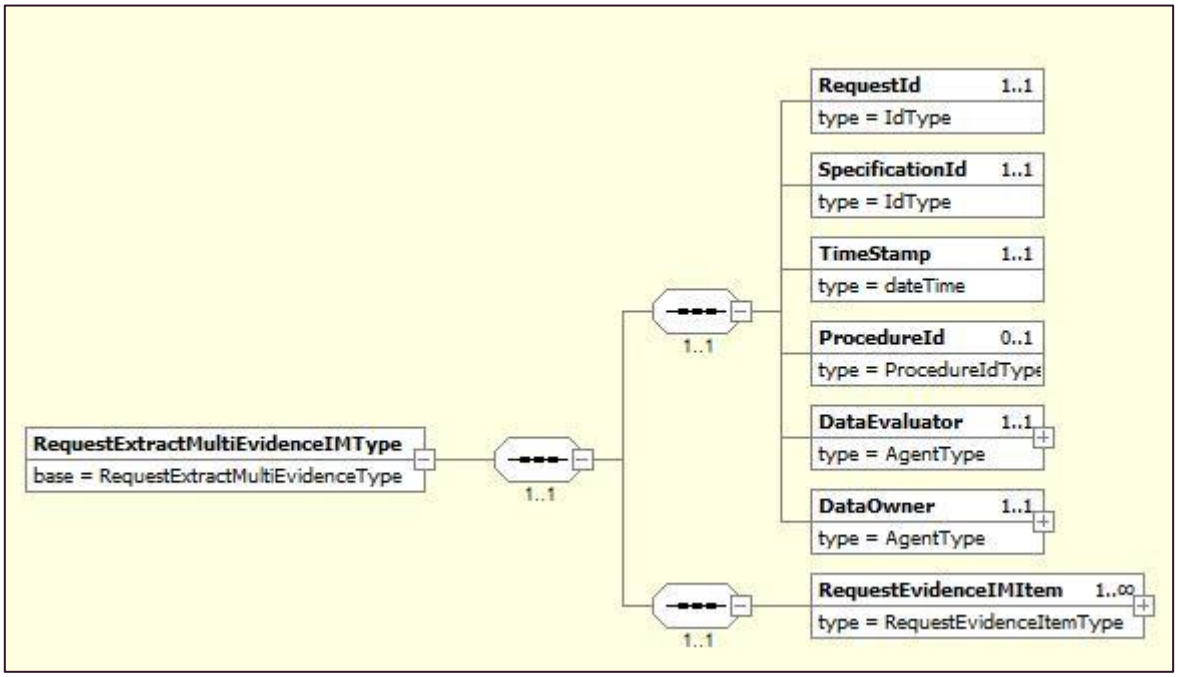

Figure 27: RequestExtractMultiEvidenceIMType diagram

Table 8: RequestExtractMultiEvidenceIMType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestId	String	1	Y/Y	Request identifier (UUID format). It is going to be the same during all interactions of this message exchange.
SpecificationId	String	1	Y/Y	IEM version used
Timestamp	Date	1	Y/Y	Date and time of creation of the message. Its value will remain unchanged as it passes through the intermediate components until it reaches its final destination.
ProcedureId	String	1	N/Y	Metadata about the procedure or public service the justifies the use of the DE4A OOTS.
DataEvaluator	**AgentType**	1	Y/Y	Identifies the Data Evaluator who is making the request for evidence. This information is also used to route the responses from the Data Owner to the Data Evaluator.
DataOwner	**AgentType**	1	Y/Y	Identifies the Data Owner to whom the request for evidence is sent.
RequestEvidenceIMItem	- **RequestEvid**	1..n	Y/Y	List of requested canonical evidence types

Name	Type	Cardinality	Mandatory Field / Label	Description
	enceItemType			

- **AgentType**

Location: //RequestExtractMultiEvidenceIMType/AgentType

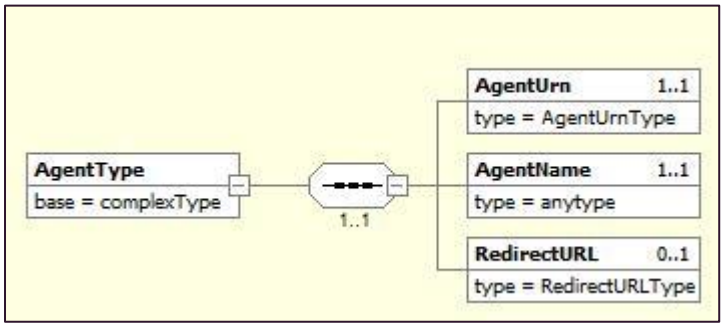

Figure 28: AgentType diagram

Table 9: AgentType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
AgentUrn	anyURI	1	Y/Y	Identifies the DE4A participant (Data Evaluator or Data Owner) in iso6523 format. It defines Party Id for all participants. It must be unique. **Format:** {PEPPOL ParticipantIdentifierScheme}::{Domain Scheme}:{Party Id} → iso6523-actorid-upis::[0-9][0-9][0-9][0-9]:[A-Z0-9]{1,15} **{PEPPOL ParticipantIdentifierScheme}:** Must be the constant "iso6523-actorid-upis" **{Domain Scheme}:** Identifies ICD value, must be a 4-digit UNIQUE number in PEPPOL **{Party Id}:** Identifies DE4A Partner, consisting of: 2 letters for country 2 digits for entity type → 00 if entity is a DE4A partner, 88 if not; for regional or local administrations, the entity type is 99 7 digits for DE4A Partner. For regional or local administrations, it is the NUTS/LAU code without country

Name	Type	Cardinality	Mandatory Field / Label	Description
				letters and starting with as many zeros on the left as necessary until reaching 7 digits. Example of the Spanish Municipality of San Sebastian with LAU code 20069: **ES990020069** Example of the Dutch Province of Groningen with NUTS code NL11: **NL990000011** https://docs.peppol.eu/edelivery/policies/PEPPOL-EDN-Policy-for-use-of-identifiers-4.0-2019-01-28.pdf For further information on the Domain Scheme values, see the Annex I. Lists of codes
AgentName	String	1	Y/N	Name of the DE4A participant (Data Evaluator or Data Owner). Based on the schema "https://data.europe.eu/semanticassets/ns/cv/common/dataTypes-2.0.0#"
RedirectUrl	String	0..1	N/Y	URL of the web page of origin. It is only filled in by the Data Evaluator in the context of the USI pattern, so that the Data Owner knows where to return the User once the preview has been completed in the DO portal.

- **RequestEvidenceItemType**

Location: // RequestExtractMultiEvidenceIMType/RequestEvidenceItemType

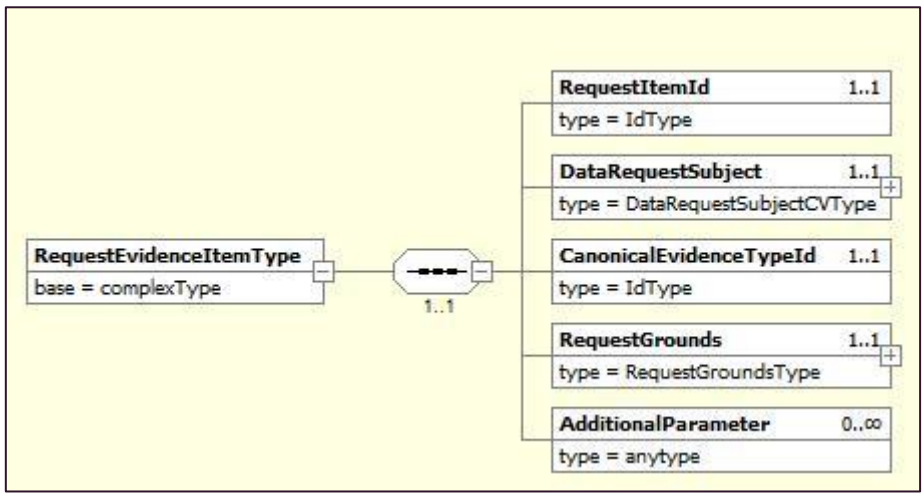

Figure 29: RequestEvidenceItemType diagram

Table 10. RequestEvidenceItemType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestItemId	String	1	Y/Y	Identifier of each item of the request (UUID format). It is going to be the same during all interactions of this message exchange.
DataRequestSubject	DataRequestSubjectCVType	1	Y/Y	Contains the identifying details of the person, company, or representative to whom the evidence relates.
CanonicalEvidenceTypeId	String	1	Y/Y	Identifier of the canonical evidence type. Format: urn:de4a-eu:CanonicalEvidenceType::*evidenceID* For further information on the Domain Scheme values, see the Annex I. Lists of codes
RequestGrounds	RequestGroundsType	1	Y/Y	Legal grounds justifying the request for evidence and supporting the applicant.
AdditionalParameter	AdditionalParameterType	0..n	N/Y	Contains each additional parameter necessary to request the evidence.

- **DataRequestSubjectCVType**

Location:

//RequestExtractMultiEvidenceIMType/RequestEvidenceItemType/DataRequestSubject

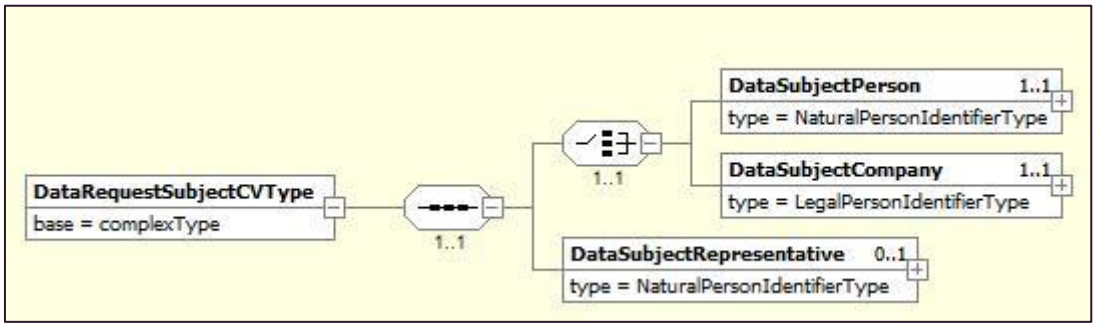

Figure 30: DataRequestSubjectCVType diagram

Table 11: DataRequestSubjectCVType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
DataSubjectPerson	- **NaturalPersonIdentifierType**	1	Y/Y	Contains the identifying details of the person to whom the evidence relates. Based on the schema "http://eidas.europa.eu/attributes/naturalperson".
DataSubjectCompany	- **LegalPersonIdentifierType**	1	Y/Y	Contains the identifying details of the company to which the evidence or event relates. Based on the schema "http://eidas.europa.eu/attributes/legalperson".
DataSubjectRepresentative	- **NaturalPersonIdentifierType**	0..1	N/Y	Contains the identifying details of the company representative to which the evidence or event relates. Based on the schema "http://eidas.europa.eu/attributes/naturalperson".

- **NaturalPersonIdentifierType**

Location:
//RequestExtractMultiEvidenceIMType/RequestEvidenceItemType/DataRequestSubject/NaturalPersonIdentifierType

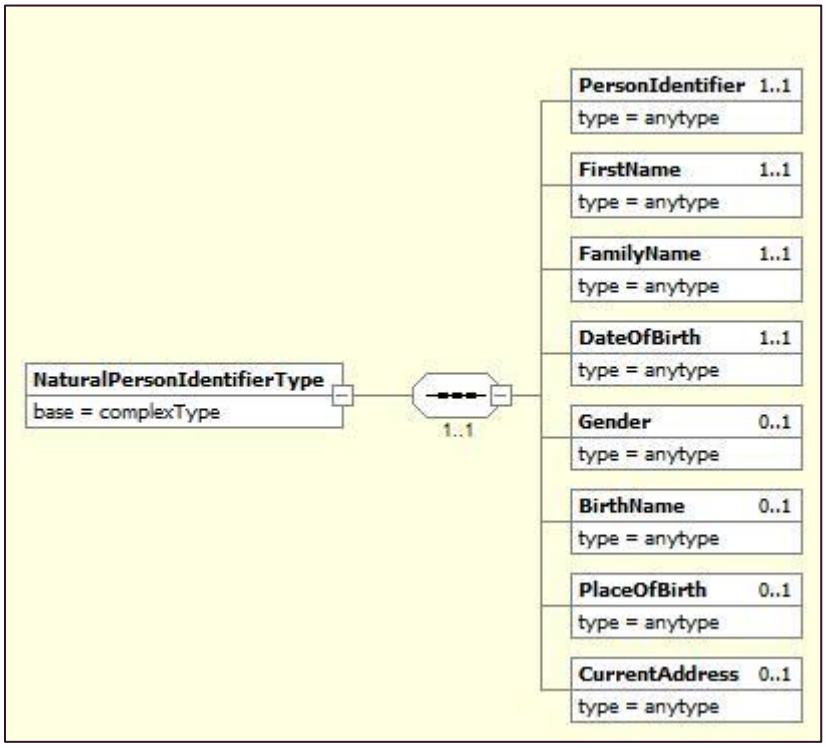

Figure 31: NaturalPersonIdentifierType diagram

Table 12: NaturalPersonIdentifierType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
PersonIdentifier	String	1	Y/N	Unique identifier for the natural person as defined by the eIDAS Regulation.
FirstName	String	1	Y/N	Current given names of the natural person.
FamilyName	String	1	Y/N	Current family name of the natural person.
DateOfBirth	Date	1	Y/Y	Date of birth of the natural person. Format: YYYY-MM-DD
Gender	String	0..1	N/Y	Gender of the natural person. Allowed values: Male Female Unspecified
BitrhName	String	0..1	N/N	First name(s) and family name(s) of the natural person at birth.
PlaceOfBirth	String	0..1	N/N	Place of birth of the natural person.
CurrentAddress	String	0..1	N/N	Current address of the natural person as a **base64 encoded string**.

- **LegalPersonIdentifierType**

Location:
//RequestExtractMultiEvidenceIMType/RequestEvidenceItemType/DataRequestSubject/LegalPersonIdentifierType

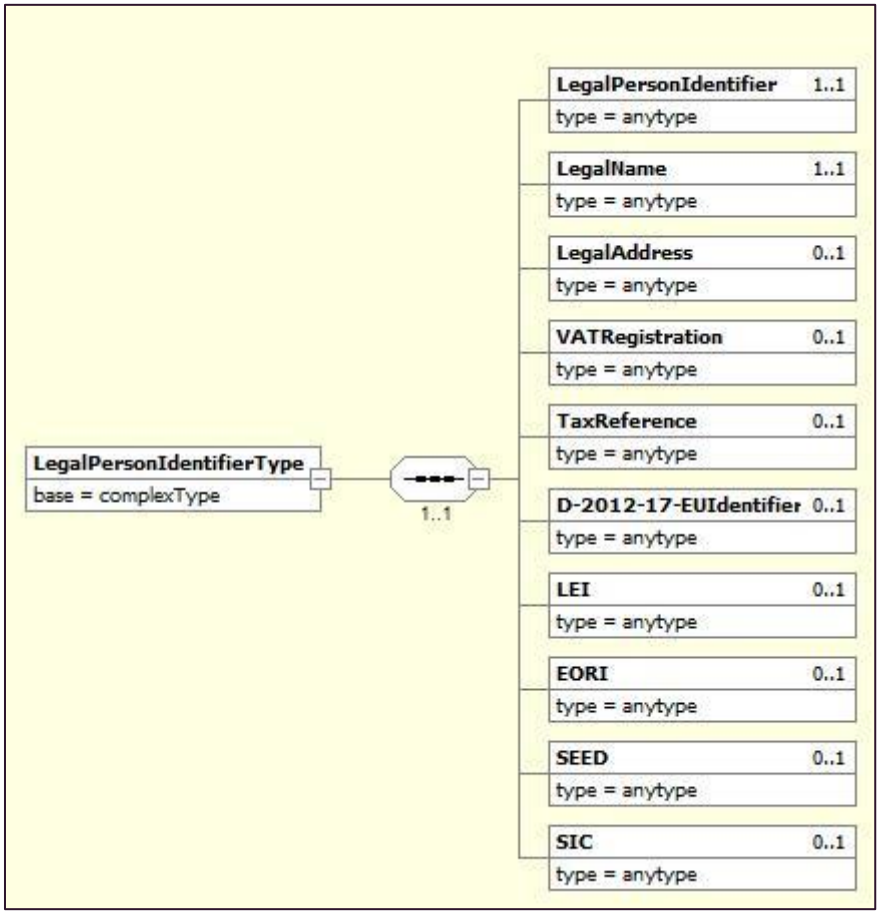

Figure 32: LegalPersonIdentifierType diagram

Table 13: LegalPersonIdentifierType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
LegalPersonIdentifier	String	1	Y/N	Unique identifier for the legal person as defined by the eIDAS Regulation.
LegalName	String	1	Y/N	Current legal name for the legal person or organisation.

Name	Type	Cardinality	Mandatory Field / Label	Description
LegalAddress	String	0..1	N/N	The address the legal person has registered with the MS authority or operating address if not registered. For a company this should be the registered address within the MS issuing the eID as a base64 encoded string.
VATRegistration	String	0..1	N/N	Value Addex Tax registration number
TaxReference	String	0..1	N/N	Tax reference number
D-2012-17-EUIdentifier	String	0..1	N/N	The identifier used under directive 2012/17/EU
LEI	String	0..1	N/N	Legal Entity Identifier
EORI	String	0..1	N/N	Economic Operator Registration and Identification
SEED	String	0..1	N/N	System for Exchange of Excise Data
SIC	String	0..1	N/N	Standard Industrial Classification

- **RequestGroundsType**

Location: //RequestExtractMultiEvidenceIMType/RequestEvidenceItemType/RequestGrounds

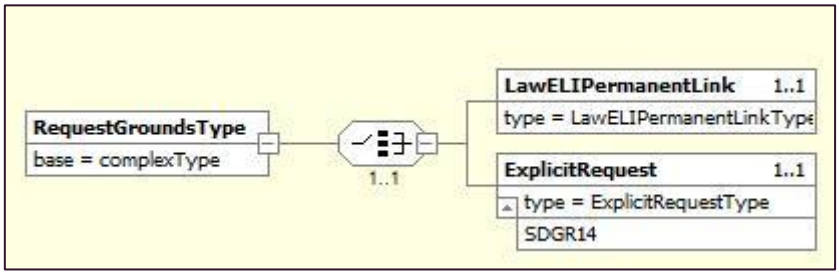

Figure 33: RequestGroundsType diagram

Table 14: RequestGroundsType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
LawELIPermanentLink	anyURI	1	Y/Y	Permanent link to the european law that allows a DE to request the corresponding evidence, identified by the European Legislation Identifier (ELI).
ExplicitRequest	token	1	Y/Y	Explicit consent made by the User to allow the DE to request the appropriate evidence through the DE4A system.

Name	Type	Cardinality	Mandatory Field / Label	Description
				Requirement imposed by article 14 of the SDG Regulation.

- **AdditionalParameterType**

Location:
//RequestExtractMultiEvidenceIMType/RequestEvidenceItemType/AdditionalPatameter

Table 15: AdditionalParameterType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
Label	String	1	Y/Y	Label identifying the required parameter.
Type	String	1	Y/Y	Type of the required parameter. Allowed values: Date → Date type InputText → Input Text type YesNo → Selector Yes/No type
Value	String	1	Y/Y	Value of the required parameter.

- **ResponseExtractMultiEvidenceType**

Location: //ResponseExtractMultiEvidenceType

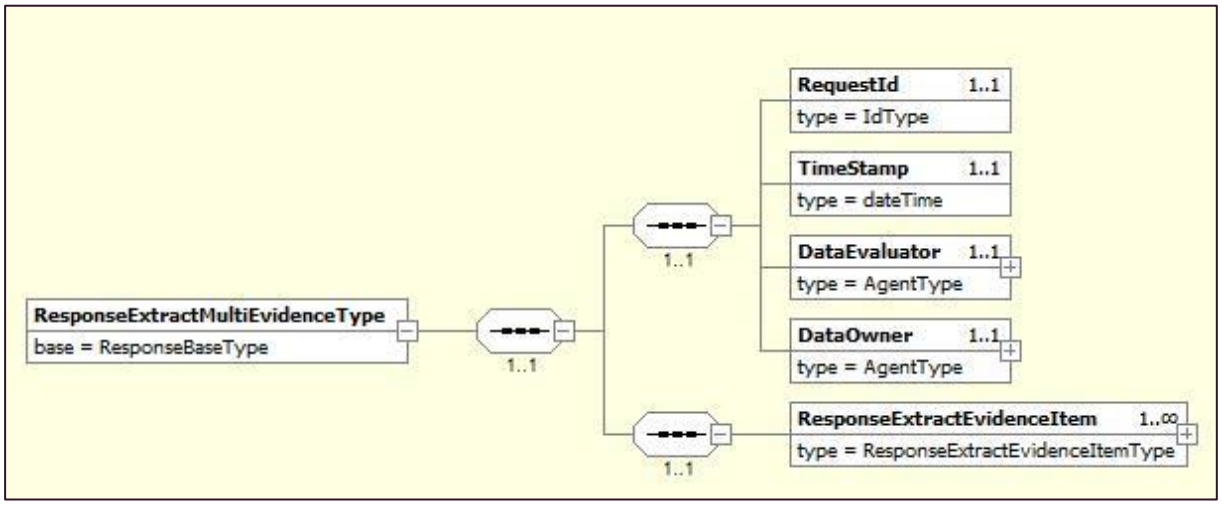

Figure 34: ResponseExtractMultiEvidenceType diagram

Table 16: ResponseExtractMultiEvidenceType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestId	String	1	Y/Y	Request identifier (UUID format). It is going to be the same during all interactions of this message exchange.
Timestamp	Date	1	Y/Y	Date and time of creation of the message. Its value will remain unchanged as it passes through the intermediate components until it reaches its final destination.
DataEvaluator	**AgentType**	1	Y/Y	Identifies the Data Evaluator who is making the request for evidence. This information is also used to route the responses from the Data Owner to the Data Evaluator.
DataOwner	**AgentType**	1	N/Y	Identifies the Data Owner to whom the request for evidence is sent.
ResponseExtractEvidenceItem	- **ResponseExtractEvidenceItemType**	1..n	Y/Y	List of requested canonical evidence types

- **ResponseExtractEvidenceItemType**

Location: //ResponseExtractMultiEvidenceType/ResponseExtractEvidenceItem

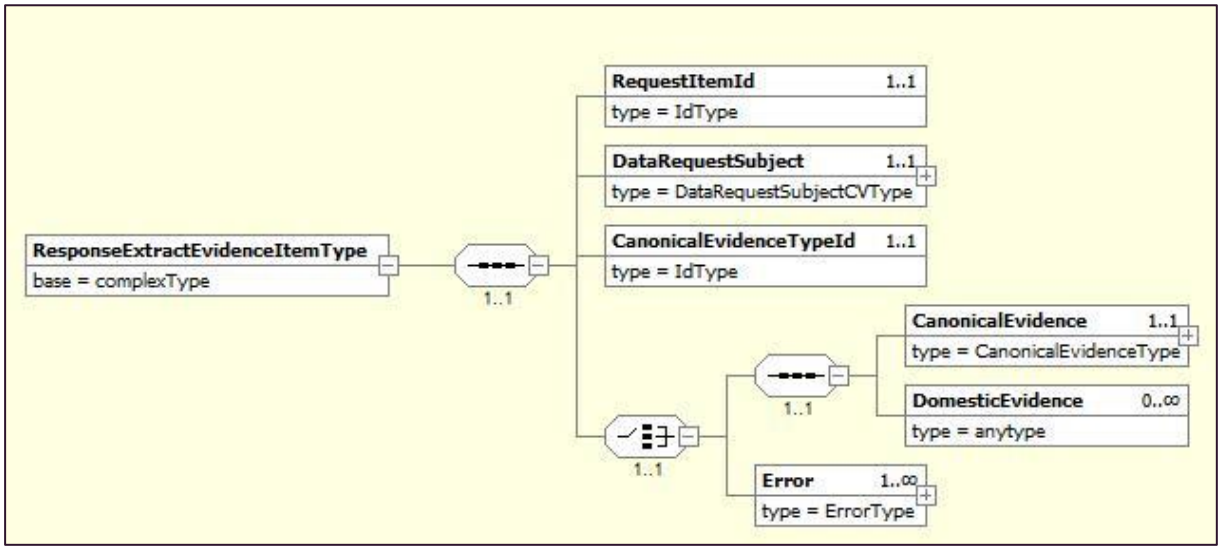

Figure 35: ResponseExtractEvidenceItemType diagram

Table 17: ResponseExtractEvidenceItemType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestItemId	String	1	Y/Y	Identifier of each item of the request (UUID format).
DataRequestSubject	DataRequestSubjectCVType	1	Y/Y	Contains the identifying details of the person, company or representative to whom the evidence relates.
CanonicalEvidenceTypeId	String	1	Y/Y	Identifier of the canonical evidence type. **Format:** urn:de4a-eu:CanonicalEvidenceType::*evidenceID* For further information on the Domain Scheme values, see the Annex I. Lists of codes
CanonicalEvidence	CanonicalEvidenceType	1	Y/Y	Evidence retrieved from the DO in its corresponding canonical format.
DomesticEvidence	**DomesticEvidenceType**	0..n	Y/Y	Contains the data for each piece of national evidence retrieved from the DO in the original format in which they are issued.
Error	ErrorType	1..n	N/Y	Contains a list with the descriptions of the errors if one or more have occurred.

- **CanonicalEvidenceType**

Location:
//ResponseExtractMultiEvidenceType/ResponseExtractEvidenceItem/CanonicalEvidence

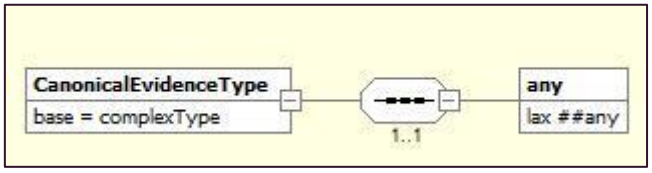

Figure 36: CanonicalEvidenceType diagram

- **DomesticEvidenceType**

Location:
//ResponseExtractMultiEvidenceType/ResponseExtractEvidenceItem/DomesticEvidence

Table 18: CanonicalEvidenceType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
IssuingType	Token	1	Y/Y	Indicates the format of the evidence: **OriginalIssuing:** Original format MultinlingualFormIssuing: Multilingual format
MimeType	normalizedString	1	Y/Y	Contains the MIME Type of the canonical evidence. https://docs.oasis-open.org/ubl/os-UBL-2.1/cva/UBL-DefaultDTQ-2.1.html#d4e1
DataLanguage	language	1	Y/N	Contains the language of the canonical evidence. Values conform to RFC 3066, allowed values: Conformity to ISO 639 Conformity to ISO 3166 Languages registered by the Internet Assigned Numbers Authority (IANA) Expression pattern: [a-zA-Z]{1,8}(-[a-zA-Z0-9]{1,8})*
EvidenceData	base64Binary	1	Y/Y	Contains the domestic evidence that the Data Owner system returns (in formats like XML, PDF, etc.) codified in B64.
AdditionalInfo	**TextType**	0..1	N/N	Contains additional information about the domestic evidence.

- **TextType**

Location:
//ResponseExtractMultiEvidenceType/ResponseExtractEvidenceItem/DomesticEvidence/AdditionalInfo

Table 19: AdditionalInfo specification

Name	Type	Cardinality	Mandatory Field / Label	Description
languageID	language	0..1	N/Y	Identifier of the language used in the content.
languageLocaleID	normalizedString	0..1	N/Y	Identification of the locale of the language.

- **RequestExtractMultiEvidenceUSIType**

Location: //ResponseExtractMultiEvidenceUSIType

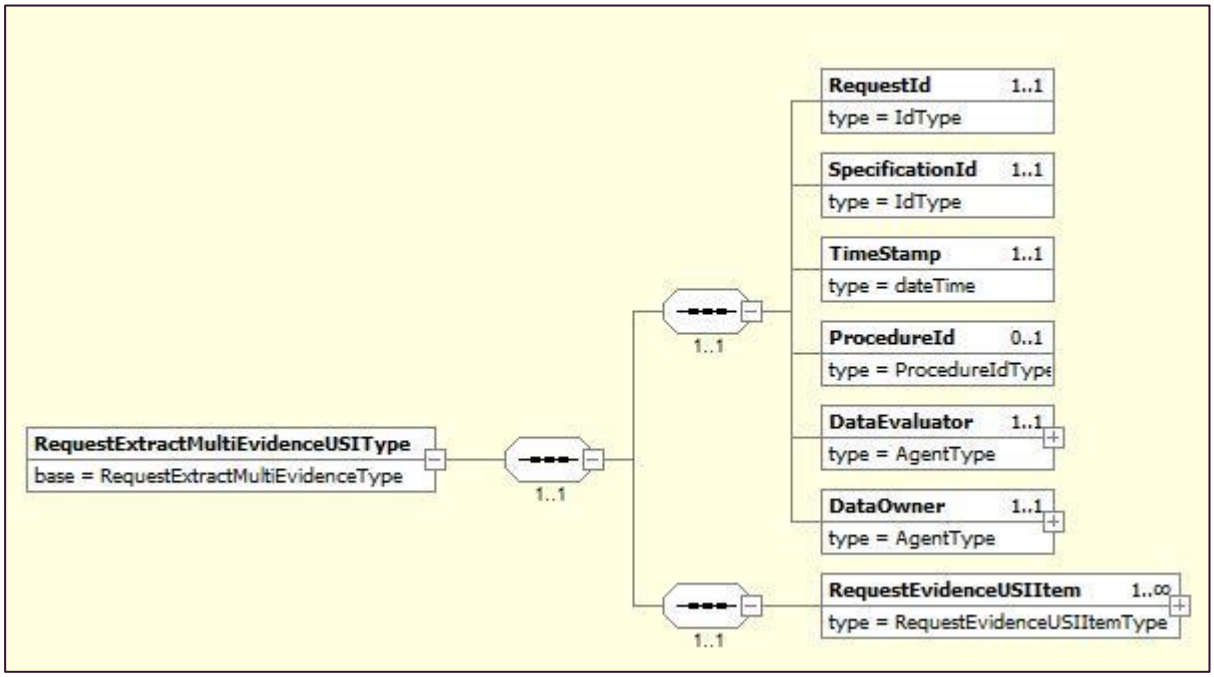

Figure 37: RequestExtractMultiEvidenceUSIType diagram

Table 20: RequestExtractMultiEvidenceUSIType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestId	String	1	Y/Y	Request identifier (UUID format). It is going to be the same during all interactions of this message exchange.
SpecificationId	String	1	Y/Y	IEM version used.
Timestamp	Date	1	Y/Y	Date and time of creation of the message. Its value will remain unchanged as it passes through the intermediate components until it reaches its final destination.
ProcedureId	String	0..1	N/Y	Metadata about the procedure or public service the justifies the use of the DE4A OOTS.
DataEvaluator	**AgentType**	1	Y/Y	Identifies the Data Evaluator who is making the request for evidence. This information is also used to route the responses from the Data Owner to the Data Evaluator.
DataOwner	**AgentType**	1	Y/Y	Identifies the Data Owner to whom the request for evidence is sent.

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestEvidenceUSIItem	RequestEvidenceUSIItemType	1..n	Y/Y	List of requested canonical evidence types

- **RequestExtractMultiEvidenceType**

Location: //ResponseExtractMultiEvidenceUSIType/RequestEvidenceUSIItem

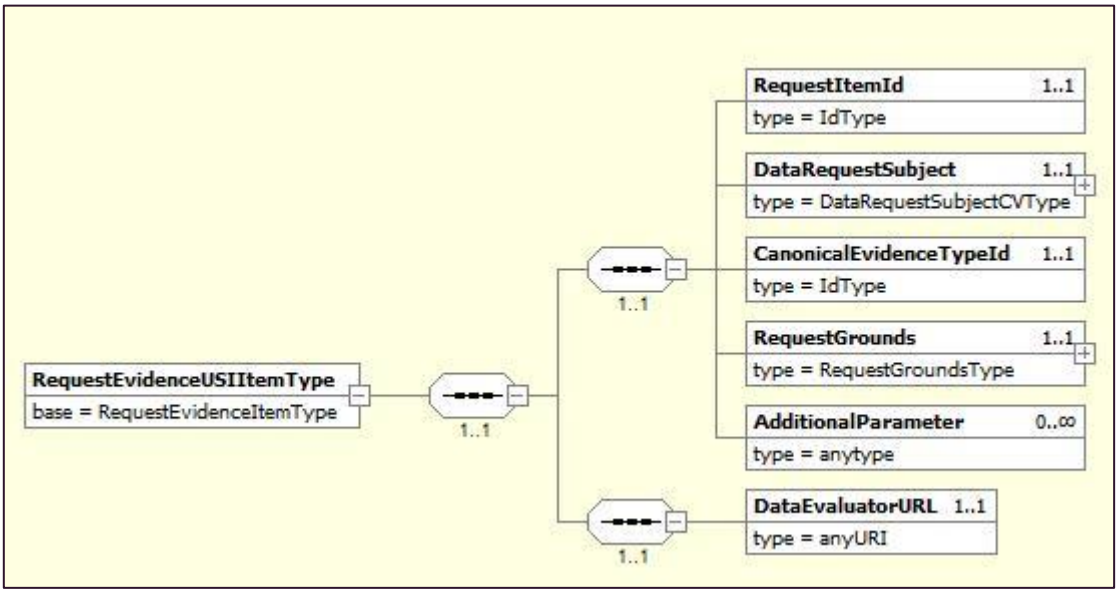

Figure 38: RequestEvidenceUSIItemType diagram

Table 21: RequestEvidenceUSIItemType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestItemId	String	1	Y/Y	Identifier of each item of the request (UUID format). It is going to be the same during all interactions of this message exchange.
DataRequestSubject	- **DataRequestSubjectCVType**	1	Y/Y	Contains the identifying details of the person, company or representative to whom the evidence relates.
CanonicalEvidenceTypeId	String	1	Y/Y	Identifier of the canonical evidence type. **Format:** urn:de4a-eu:CanonicalEvidenceType::*evidenceID*

Name	Type	Cardinality	Mandatory Field / Label	Description
				For further information on the Domain Scheme values, see the Annex I. Lists of codes
RequestGrounds	- **RequestGroundsType**	1	Y/Y	Legal grounds justifying the request for evidence and supporting the applicant.
AdditionalParameter	- **AdditionalParameterType**	0..n	N/Y	Contains each additional parameter necessary to request the evidence.
DataEvaluatorURL	anyUri	1	Y/Y	Indicates the redirection URL to return the User to the e-procedure portal of the Data Evaluator.

- **RedirectUserType**

Location: //RedirectUserType

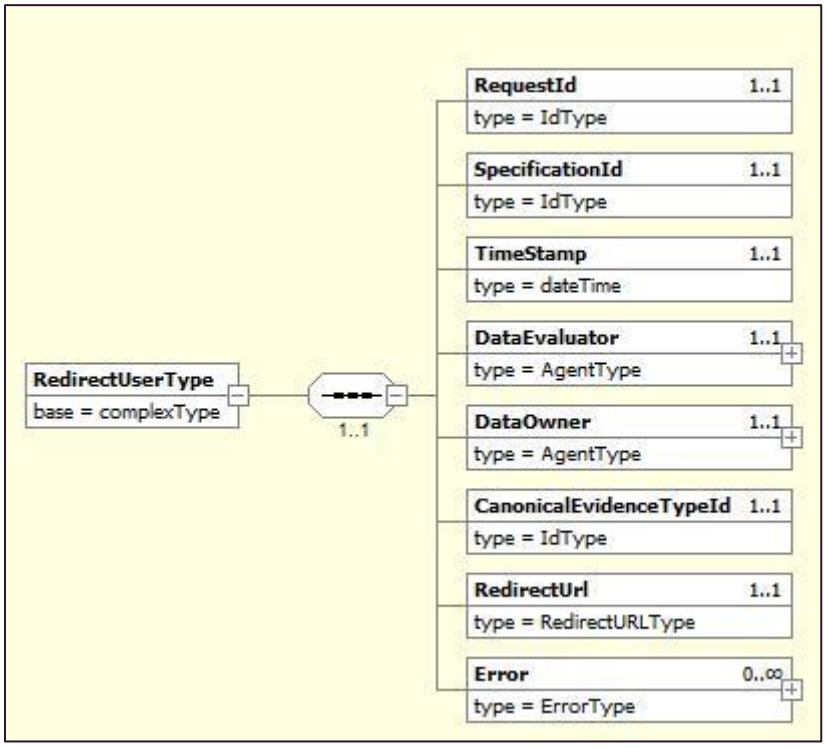

Figure 39: RedirectUserType diagram

Table 22: RedirectUserType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestId	String	1	Y/Y	Request identifier (UUID format). It is going to be the same during all interactions of this message exchange.
SpecificationId	String	1	Y	IEM version used
Timestamp	Date	1	Y/Y	Date and time of creation of the message. Its value will remain unchanged as it passes through the intermediate components until it reaches its final destination.
DataEvaluator	**AgentType**	1	Y/Y	Identifies the Data Evaluator who is making the request for evidence. This information is also used to route the responses from the Data Owner to the Data Evaluator.
DataOwner	**AgentType**	1	Y/Y	Identifies the Data Owner to whom the request for evidence is sent.
CanonicalEvidenceTypeId	String	1	Y/Y	Identifier of the canonical evidence type. **Format:** urn:de4a-eu:CanonicalEvidenceType::*evidenceID* For further information on the Domain Scheme values, see the Annex I. Lists of codes
RedirectURL	anyURI	1	Y/Y	Indicates the URL of the DO portal where to send the User to choose and preview the evidence.
Error	- **ErrorType**	0..n	N/Y	Contains the description of the error if one has occurred.

- **RequestEventSubscriptionType**

Location: //RequestEventSubscriptionType

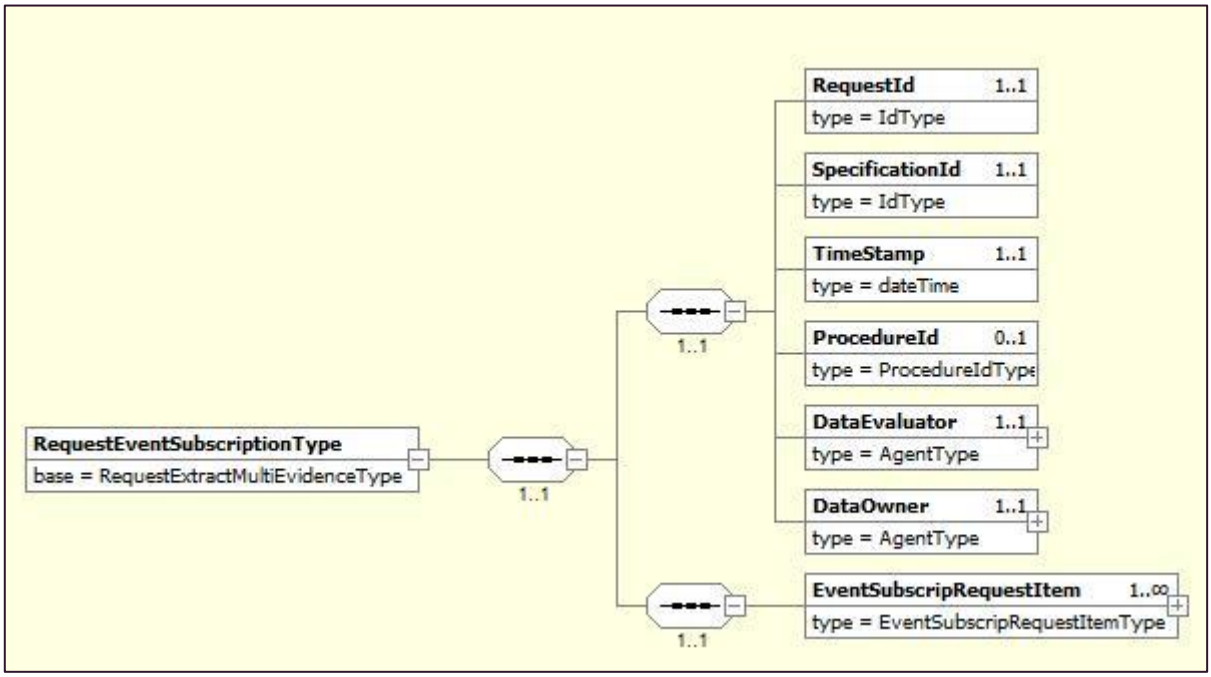

Figure 40: RequestEventSubscriptionType diagram

Table 23: RequestEventSubscriptionType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestId	String	1	Y/Y	Request identifier (UUID format). It is going to be the same during all interactions of this message exchange.
SpecificationId	String	1	Y/Y	IEM version used
Timestamp	Date	1	Y/Y	Date and time of creation of the message. Its value will remain unchanged as it passes through the intermediate components until it reaches its final destination.
ProcedureId	String	0..1	N/Y	Metadata about the procedure or public service the justifies the use of the DE4A OOTS.
DataEvaluator	**AgentType**	1	Y/Y	Identifies the Data Evaluator who is making the request for subscription. This information is also used to route the responses from the Data Owner to the Data Evaluator.
DataOwner	**AgentType**	1	Y/Y	Identifies the Data Owner to whom the request for subscription is sent.

Name	Type	Cardinality	Mandatory Field / Label	Description
EventSubscripRequestItem	- **EventSubscripRequestItemType**	1..n	Y/Y	List of subscriptions requested

- **EventSubscripRequestItemType**

Location: //RequestEventSubscriptionType/EventSubscripRequestItem

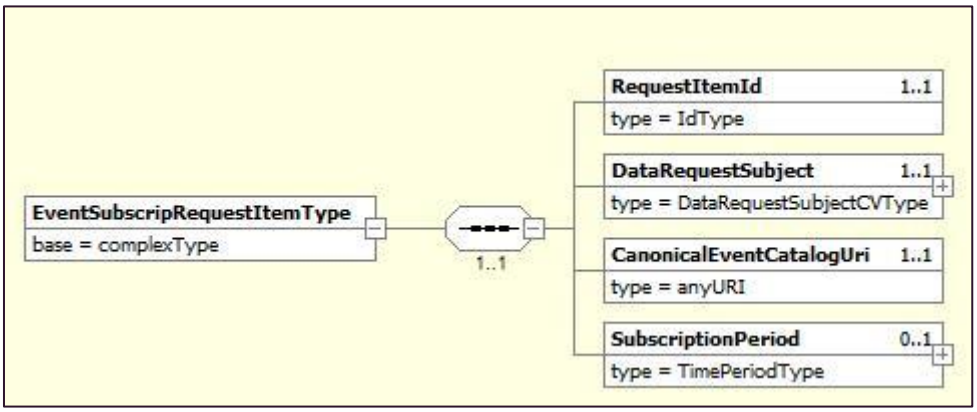

Figure 41: EventSubscripRequestItemType diagram

Table 24: EventSubscripRequestItemType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestItemId	string	1	Y/Y	Identifier of each item of the request (UUID format). It is going to be the same during all interactions of this message exchange.
DataRequestSubject	- **DataRequestSubjectCVType**	1	Y/Y	Contains the identifying details of the person, company or representative to whom the evidence relates.
CanonicalEventCatalogUri	anyUri	1	Y/Y	Identifier of the canonical event catalogue. **Format:**

Name	Type	Cardinality	Mandatory Field / Label	Description
				urn:de4a-eu:CanonicalEventCatalogueType::*catalogueID* For further information on the Domain Scheme values, see the Annex I. Lists of codes
SubscriptionPeriod	- **TimePeriodType**	0..1	Y/Y	Period for which the subscription will last.

- **TimePeriodType**

Location: //RequestEventSubscriptionType/EventSubscripRequestItem/SubscriptionPeriod

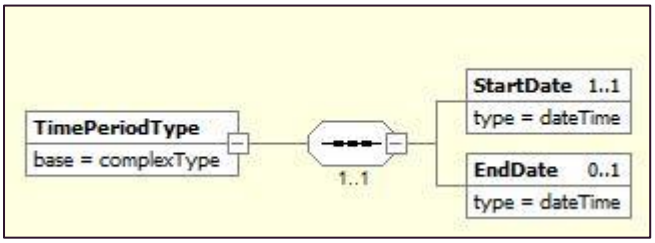

Figure 42: TimePeriodType diagram

Table 25: TimePeriodType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
StartDate	dateTime	1	Y/Y	Subscription start date
EndDate	dateTime	0..1	N/Y	Subscription end date

- **ResponseEventSubscriptionType**

Location: //ResponseEventSubscriptionType

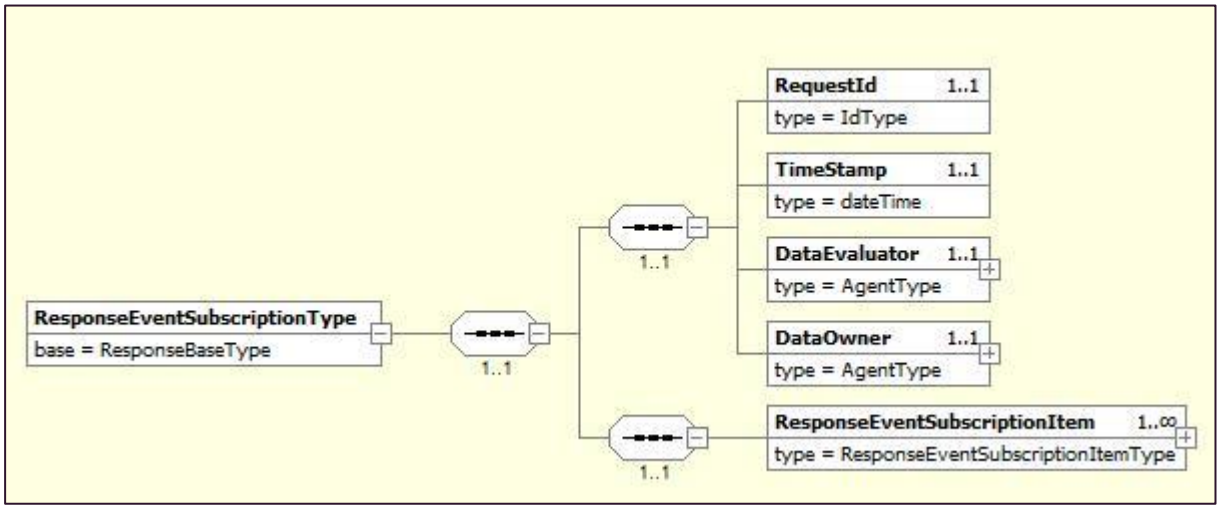

Figure 43: ResponseEventSubsciptionType diagram

Table 26: ResponseEventSubsciptionType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestId	String	1	Y	Request identifier (UUID format). It is going to be the same during all interactions of this message exchange.
Timestamp	Date	1	Y	Date and time of creation of the message. Its value will remain unchanged as it passes through the intermediate components until it reaches its destination.
DataEvaluator	**AgentType**	1	Y	Identifies the Data Evaluator who is making the request for subscription. This information is also used to route the responses from the Data Owner to the Data Evaluator.
DataOwner	**AgentType**	1	Y	Identifies the Data Owner to whom the request for subscription is sent.
ResponseEventSubscriptionItem	- **ResponseEventSubscriptionItemType**	1..n	Y	List of subscriptions requested

- **ResponseEventSubscriptionItemType**

Location: //ResponseEventSubscriptionType/ResponseEventSubscriptionItem

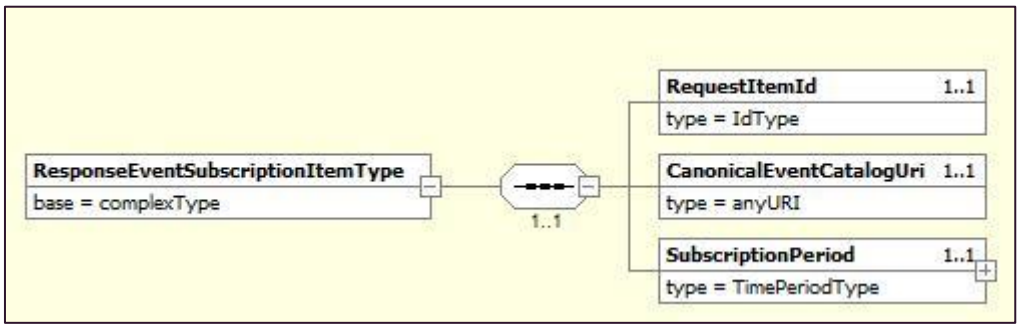

Figure 44: ResponseEventSubscriptionItemType diagram

Table 27: ResponseEventSubscriptionItemType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestItemId	string	1	Y	Identifier of each item of the request (UUID format). It is going to be the same during all interactions of this message exchange.
CanonicalEventCatalogUri	anyUri	1	Y	Identifier of the canonical event catalogue. Format: urn:de4a-eu:CanonicalEventCatalogueType::catalogueID For further information on the Domain Scheme values, see the Annex I. Lists of codes
SubscriptionPeriod	- **TimePeriodType**	1	Y	Confirmation of the period for which the subscription will last.

- **EventNotificationType**

Location: //EventNotificationType

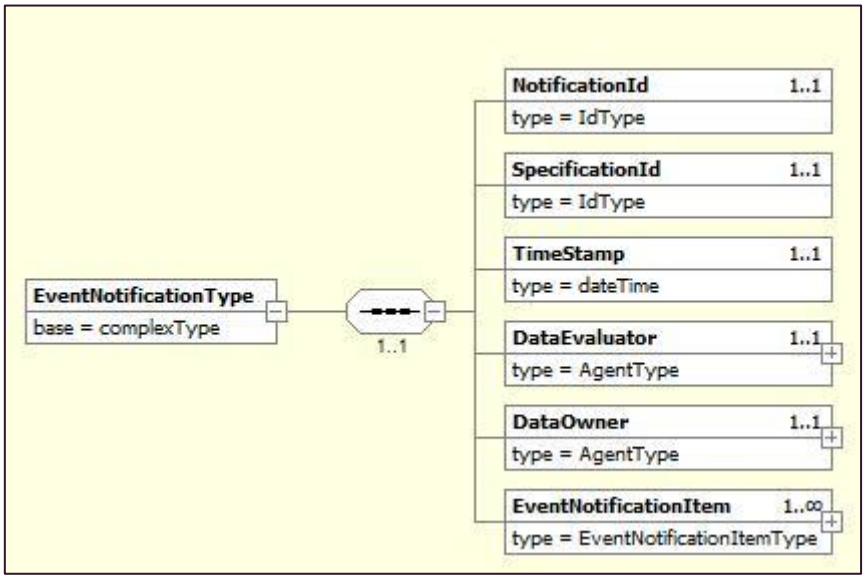

Figure 45: EventNotificationType diagram

Table 28: EventNotificationType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
NotificationId	String	1	Y	Event notification identifier (UUID format)
Timestamp	Date	1	Y	Date and time of creation of the message. Its value will remain unchanged as it passes through the intermediate components until it reaches its destination.
DataEvaluator	**AgentType**	1	Y	Identifies the Data Evaluator to whom the event notification is sent.
DataOwner	**AgentType**	1	Y	Identifies the Data Owner that sends the event notification message.
EventNotificationItem	**EventNotificationItemType**	1..n	Y	List of event notifications

- **EventNotificationItemType**

Location: //EventNotificationType/EventNotificationItem

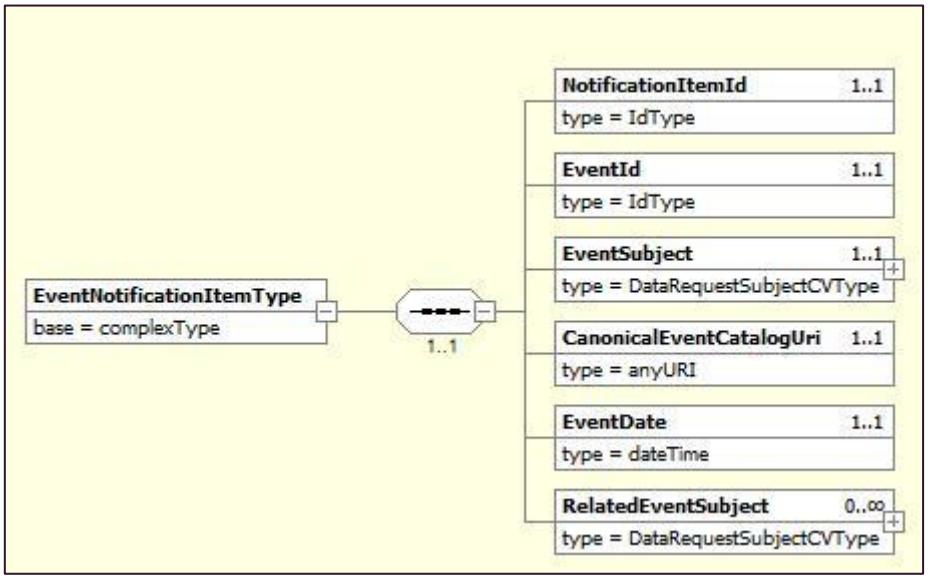

Figure 46: EventNotificationItemType diagram

Table 29: EventNotificationItemType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
NotificationItemId	String	1	Y	Identifier of each item of the notification (UUID format)
EventId	String	1	Y	Identifier of the event that triggered the notification in the DO system.
EventSubject	- **DataRequestSubjectCVType**	1	Y	Contains the identifying details of the company or representative to whom the event notification relates.
CanonicalEventCatalogUri	anyUri	1	Y	Identifier of the canonical event catalogue. **Format:** urn:de4a-eu:CanonicalEventCatalogueType::*catalogueID* For further information on the Domain Scheme values, see the Annex I. Lists of codes
EventDate	Datetime	1	Y	Date when the event occurred.

Name	Type	Cardinality	Mandatory Field / Label	Description
RelatedEventSubject	- **DataRequestSubjectCVType**	0..n	Y	Contains the identification data of possible subjects also related to or afected by this event.

- **RequestExtractMultiEvidenceLUType**

Location: //RequestExtractMultiEvidenceLUType

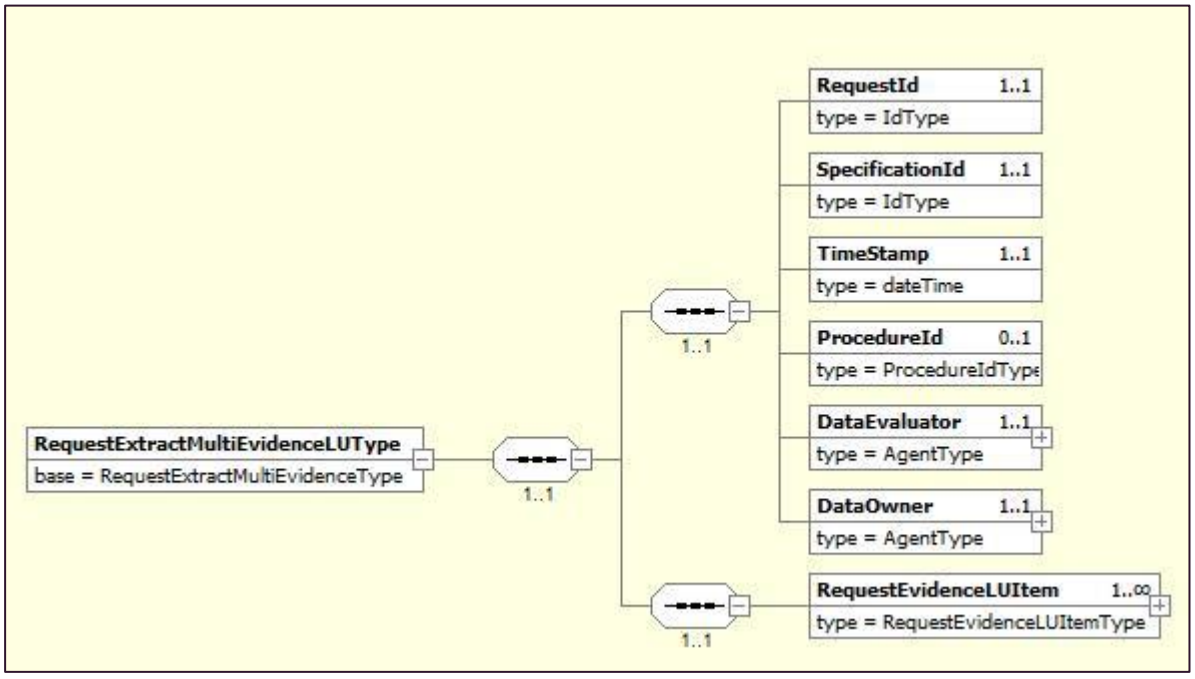

Figure 47: RequestExtractMultiEvidenceLUType diagram

Table 30: RequestExtractMultiEvidenceLUType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestId	String	1	Y	Request identifier (UUID format). It is going to be the same during all interactions of this message exchange.
SpecificationId	String	1	Y	IEM version used
Timestamp	Date	1	Y	Date and time of creation of the message. Its value will remain unchanged as it passes through the intermediate components until it reaches its destination.

Name	Type	Cardinality	Mandatory Field / Label	Description
ProcedureId	String	0..1	N	Metadata about the procedure or public service the justifies the use of the DE4A OOTS.
DataEvaluator	**AgentType**	1	Y	Identifies the Data Evaluator who is making the request for evidence. This information is also used to route the responses from the Data Owner to the Data Evaluator.
DataOwner	**AgentType**	1	Y	Identifies the Data Owner to whom the request for evidence is sent.
RequestEvidenceLUItem	**RequestEvidenceLUItemType**	1..n	Y	List of requested canonical evidence types

- **RequestEvidenceLUItemType**

Location: //RequestExtractMultiEvidenceLUType/RequestEvidenceLUItem

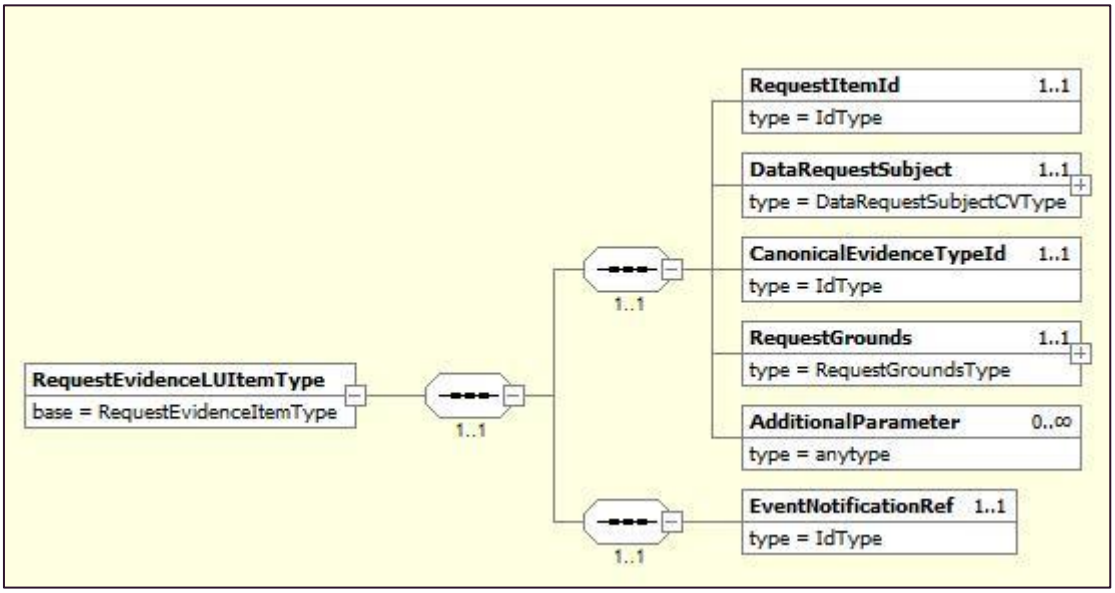

Figure 48: RequestEvidenceLUItemType diagram

Table 31: RequestEvidenceLUItemType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
RequestItemId	String	1	Y	Identifier of each item of the request (UUID format). It is going to be the same during all interactions of this message exchange.
DataRequestSubject	- **DataRequestSubjectCVType**	1	Y	Contains the identifying details of the person, company or representative to whom the evidence relates.
CanonicalEvidenceTypeId	String	1	Y	Identifier of the canonical evidence type. **Format:** urn:de4a-eu:CanonicalEvidenceType::*evidenceID* For further information on the Domain Scheme values, see the Annex I. Lists of codes
RequestGrounds	- **RequestGroundsType**	1	Y	Legal grounds justifying the request for evidence and supporting the applicant.
AdditionalParameter	- **AdditionalParameterType**	0..n	N/Y	Contains each additional parameter necessary to request the evidence.
EventNotificationRef	String	1	Y	NotificationId of the event notification this Lookup request is based on.

- **ResponseErrorType**

Location: //ResponseErrorType

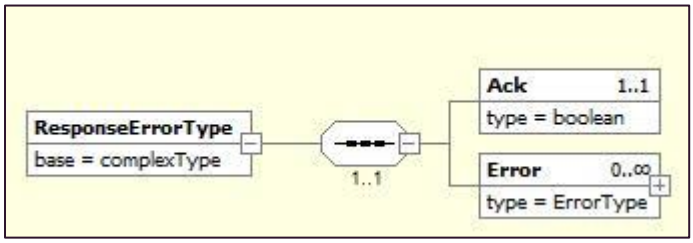

Figure 49: ResponseErrorType diagram

Table 32: ResponseErrorType specification

Name	Type	Cardinality	Mandatory Field / Label	Description
Ack	boolean	1	Y/Y	Value returned in case no error occurred.
Error	- **ErrorType**	0..n	N/Y	Data type returned in case of error.

5 Central IAL interface

The Central IAL Interface is provided by a separate component of the DE4A network called “ial-service”.

The service was developed in the WP5 GitHub repository and deployed at Aeioros premises (subcontractor of SGAD, WP5 leader) so that the rest of the partners could use it.

This chapter describes the public API of the IAL. Compared to the original WP3 design, the public API solely relies on the use of XML for communication. Hence no OpenAPI or similar document is available.

5.1 Data Owner queries

This chapter groups all the queries that can be used to query Data Owners (DOs). This functionality is called DSD (Data Services Directory) in TOOP and SDG-OOP.

5.1.1 /provision/{canonicalObjectTypeIDs}[/{atuCode}] API

Get all Data Owners that support the requested canonical object types. If an administrative territorial unit (ATU) is provided, it will also be taken into account.

Note: URLs are limited to 2048 characters. Therefore, only a maximum of 20 canonical object type IDs are safely supported.

Note: this API provides no special handling for the multi evidence case. It will only return the canonical evidence object types it was queried for.

Compatibility note: this interface replaces the following interfaces present in Iteration 1:

`/ial/{canonicalEvidenceTypeId}`

`/ial/{canonicalEvidenceTypeId}/{countryCode}`

The old APIs allowed only one canonical evidence type ID and optionally only a country code. This is a breaking change but can easily be hidden from implementers by keeping the Connector query API unchanged.

URL parameters:

- `{canonicalObjectTypeIDs}` – mandatory – one or more canonical object type identifiers, including the identifier schemes. Multiple IDs need to be separated by the comma (,) character. A maximum of 20 different values is allowed.
- `{atuCode}` – optional – an administrative territorial unit (ATU) code on an arbitrary level. This can be a country code, a NUTS 1, a NUTS 2, a NUTS 3 or a LAU code. If no ATU code is provided, results from all regions will be returned.

Query parameters:

- none

Request body:

- none

Response body:

- An XML document of type `ResponseLookupRoutingInformationType` (as defined by IAL.xsd). If nothing is found, this is indicated in this response. If the `Accept` HTTP header favours “application/json” over “application/xml”, a JSON representation will be returned.

Status codes:

- HTTP 200 Success
  - If the message was processed correctly
- HTTP 400 Bad Request
  - If the request payload does not match the requirements (e.g. invalid parameters)
- HTTP 500 Internal Server Error
  - If something unexpected happened

Examples calls (non-normative):

The examples are displayed without URL encoding. In reality e.g. all ":" characters must be replaced with "%3A" to work.

- `/provision/urn:de4a-eu:CanonicalEvidenceType::CompanyRegistration:1.0`
  - Search for all EPs that support the "Company Registration" evidence type, independent of the country
- `/provision/urn:de4a-eu:CanonicalEvidenceType::CompanyRegistration:1.0/AT`
  - Search for all EPs that support the "Company Registration" evidence type, limit to the matches in Austria
- `/provision/urn:de4a-eu:CanonicalEvidenceType::CompanyRegistration:1.0/AT130`
  - Search for all EPs that support the "Company Registration" evidence type, limit to the matches in Vienna, Austria (NUTS 3)
- `/provision/urn:de4a-eu:CanonicalEvidenceType::MarriageRegistration:1.0,urn:de4a-eu:CanonicalEvidenceType::BirthCertificate:1.0`
  - Search for all EPs that support the "Marriage Registration" or the "Birth Certificate" evidence type, independent of the country
    - `/provision/urn:de4a-eu:CanonicalEvidenceType::MarriageRegistration:1.0,urn:de4a-eu:CanonicalEvidenceType::BirthCertificate:1.0/SE`
  - Search for all EPs that support the "Marriage Registration" or the "Birth Certificate" evidence type, limit to the matches in Sweden

5.2 MOR APIs

The APIs of the Multilingual Ontology Repository (MOR) designed by WP3 have been added to the Central IAL Service, since it needed a central service to provide the resulting ontologies. The API is based on the pattern `/mor/xx.json` to retrieve the MOR data for the provided country code `xx`. Available country codes are (case sensitive):

- `en` - English
- `es` - Spanish
- `fr` - French
- `pt` - Portuguese
- `ro` - Romanian
- `sl` - Slovenian

6 Summary of software interfaces

Table 33: Software interfaces

Component	Pattern	Service	Referenced API	Interface type	Communication	Input message	Output
Data Evaluator	USI, IM, LU	Response transfer evidence	/response/evidence	XML REST	Asynchronous	**ResponseExtractMultiEvidenceType**	ACK
	USI	Response user redirection	/response/usi/redirectUser	XML REST	Asynchronous	**RedirectUserType**	HTTP Status 302/303 (Http/1 Http/2)
	S&N	Event Subscription	/response/subscription	XML REST	Asynchronous	**ResponseEventSubscriptionType**	ACK
	S&N	Event Notification	/notificationEvent	XML REST	Asynchronous	**EventNotificationType**	ACK
Data Owner	IM	Extract evidence IM	/request/im/	XML REST	Asynchronous	**RequestExtractMultiEvidenceIMType**	ACK
	USI	Extract evidence USI	/request/usi/	XML REST	Asynchronous	**RequestExtractMultiEvidenceUSIType**	ACK
	LU	Lookup	/requestExtractMultiEvidenceLU	XML REST	Asynchronous	**RequestExtractMultiEvidenceLUType**	ACK

Component	Pattern	Service	Referenced API	Interface type	Communication	Input message	Output
	S&N	Event Subscription	/requestEventSubscription	XML REST	Asynchronous	**RequestEventSubscriptionType**	ACK
Data Requestor (Connector)	IM, USI	Lookup Routing Information	/service/ial/	XML REST	Synchronous	All the necessary parameters will be in the URL. It will be a comma-separated canonical evidence identifier collection (up to 20) followed by the AtuCode.	**ResponseLookupRoutingInformationType**
	IM	Extract evidence IM	/request/im/	XML REST	Asynchronous	**RequestExtractMultiEvidenceIMType**	ACK
	USI	Extract evidence USI	/request/usi/	XML REST	Asynchronous	**RequestExtractMultiEvidenceUSIType**	ACK
	LU	Lookup	/requestTransferEvidenceLU	XML REST	Asynchronous	**RequestExtractMultiEvidenceLUType**	ACK
	S&N	Event Subscription	/request/subscription	XML REST	Asynchronous	**RequestEventSubscriptionType**	ACK
Data Transferor	USI, IM, LU	Response transfer evidence	/response/evidence	XML REST	Asynchronous	**ResponseExtractMultiEvidenceType**	ACK

Component	Pattern	Service	Referenced API	Interface type	Communication	Input message	Output
(Connector)	USI	Response user redirection	/response/usi/redirectUser	XML REST	Asynchronous	**RedirectUserType**	ACK
	S&N	Event Subscription	/response/subscription	XML REST	Asynchronous	**ResponseEventSubscriptionType**	ACK
	S&N	Event Notification	/event/notification	XML REST	Asynchronous	**EventNotificationType**	ACK

7 DE4A Logs and error messages

7.1 General structure

The log message follows the structure below:

[timestamp] [level] [message code] [logging participant ID] + Specific text message

Where:

- **[timestamp]:** when the log message is written. ISO 8601 format with UTC time, e.g. [2022-07-26T12:41:13.666Z]
- **[level]:** INFO, ERROR, WARN or DEBUG
- **[message code]:** 5-character string consisting of:
  Component code (2 letters) + Level code (1 letter) + Template (2 digits)
- [Logging participant ID]: composed of:
  Acronym of the participant's name + '-' + Two-letter country code
- **Specific text message:** instance of the text message template that is referred to in the message code. In the case of messages logged by Data Evaluators and Data Owners (codes starting with DE and DO respectively), the text is preceded by [Use Case] that identifies the running use case.

Details of the above-mentioned parts of the log message are included below the following examples.

Example of a log message (*response* message, *INFO* level, by a *DT*):

```
[2021-07-26T12:41:13.657Z] [INFO] [DTI07] [SGAD-ES] Evidence
Response message received: ResponseExtractMultiEvidenceType -
2ecda60e-3824-4e92-a459-7c4a81258952 - iso6523-actorid-
upis::9991:pt990000101 - iso6523-actorid-upis::9920:ess2833002e -
(09754288-72b9-41d0-92b2-59f02506891c:urn:de4a-
eu:CanonicalEvidenceType::HigherEducationDiploma:2.0, f0cbbb5f-45d6-
4d1e-8be4-047c1ce2ae8b:DOE01)
```

ERROR level log messages should contain the same error code and text than the corresponding ErrorResponseItem within IEM response message:

Table 34: General structure for logging

Log message	IEM Response message	Example
Message code	ErrorResponseItem/Code	DRE05
Specific text	ErrorResponseItem/Message	Missing or invalid arguments at {service requested}: {missing arguments}

7.1.1 Logging components and their codes

Specifies which component is logging the message.

Table 35: Codes for logging components

Logging component	Component code
Data Requestor	DR
Data Transferor	DT
SSI Authority Agent	AA
Data Evaluator	DE
Data Owner	DO

7.1.2 Message severity and level code

Besides the severity of the message, the level determines where the messages SHOULD be sent. In Warning and Debug level messages, the "Specific text message" is freely defined.

Table 36: Severity level codes

Level	Level code	Meaning	Destination
INFO	I	To inform on an event relevant for the system	Logs for collection Kafka server
ERROR	E	To inform on an error that interrupt the proper functioning	DE4A response messages Logs for collection Kafka server
WARN	W	To point out something unexpected that does not interrupt the functioning	Internal logs (not to be collected)
DEBUG	D	To help diagnose issues	Internal logs (not to be collected)

7.1.3 Logging participant ID

Identifies the specific participant that is logging the message. It is compound of the following elements:

Acronym of the participant's name + '-' + Two-letter country code'

The acronym refers to the participant that has deployed and is operating the corresponding component. In the case of the Connector, this is a configurable property whose value is set by the operator of the Connector themselves.

7.1.4 Use case

Specific text in messages logged by Data Evaluator and Data Owner components (log messages with codes starting by "DE" and "DO" respectively) should be preceded by [*Use Case*], which corresponds to the specific DE4A use case running by such components:

Table 37: Use case codes

Studying Abroad pilot	Doing Business Abroad pilot	Moving Abroad pilot
UC#1.1	UC#2.1	UC#3.1
UC#1.2	UC#2.2	UC#3.2
UC#1.3		

7.2 Message catalogue

7.2.1 Data Requestor

Table 38: Connector Data Requestor logging codes

Component	Level	Template	Specific text message
DR	I	01	Legacy IM Request message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - {CanonicalEvidenceTypeUri}

Component	Level	Template	Specific text message
DR	I	02	IM Request message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (RequestItemID):(CanonicalEvidenceTypeUri)})
DR	I	03	USI Request message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (RequestItemID):(CanonicalEvidenceTypeUri)})
DR	I	04	Subscription Request message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (RequestItemID):(CanonicalEventCatalogUri)})
DR	I	05	LU Request message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (RequestItemID):(CanonicalEvidenceTypeUri)})
DR	I	06	Legacy IM Response message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - {CanonicalEvidenceTypeUri/ErrorCode}
DR	I	07	Evidence Response message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (RequestItemID):(CanonicalEvidenceTypeUri/ErrorCode)})
DR	I	08	Redirect User message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - {CanonicalEvidenceTypeUri} - {RedirectURL/ErrorCode}
DR	I	09	Subscription Response message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (RequestItemID):(CanonicalEventCatalogUri/ErrorCode)})
DR	I	10	Event Notification message received: {MessageType} - {NotificationID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (NotificationItemID):(CanonicalEventCatalogUri):(EventID)})
DR	I	11	Querying SML: {ParticipantID}
DR	I	12	Querying SMP: {ParticipantID} - {DocumentTypeID} - {ProcessID}
DR	I	13	Querying Central IAL: {query url}
DR	E	01	Connection error with {destination component}: {inherited error message}
DR	E	02	Error on response from {error source component}: {inherited error message}
DR	E	03	Object conversion error in {error source component}: {inherited error message}

Component	Level	Template	Specific text message
DR	E	04	Error accessing/saving data on {error source component}: {inherited error message}
DR	E	05	Missing or invalid arguments at {service requested}: {missing arguments}
DR	E	06	Service requested {REST service requested} to {destination component} not found
DR	E	07	Scheme validation failed for {MessageType}: {inherited error message}
DR	E	08	Error occurred on AS4 Gateway to {destination component}: {inherited error message}

7.2.2 Data Transferor

Table 39: Connector Data Transferor logging codes

Component	Level	Template	Specific text message
DT	I	01	Legacy IM Request message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - {CanonicalEvidenceTypeUri}
DT	I	02	IM Request message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (RequestItemID):(CanonicalEvidenceTypeUri)})
DT	I	03	USI Request message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (RequestItemID):(CanonicalEvidenceTypeUri)})
DT	I	04	Subscription Request message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (RequestItemID):(CanonicalEventCatalogUri)})
DT	I	05	LU Request message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (RequestItemID):(CanonicalEvidenceTypeUri)})
DT	I	06	Legacy IM Response message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - {CanonicalEvidenceTypeUri/ErrorCode}
DT	I	07	Evidence Response message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (RequestItemID):(CanonicalEvidenceTypeUri/ErrorCode)})
DT	I	08	Redirect User message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - {CanonicalEvidenceTypeUri} - {RedirectURL/ErrorCode}

Component	Level	Template	Specific text message
DT	I	09	Subscription Response message received: {MessageType} - {RequestID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (RequestItemID):(CanonicalEventCatalogUri/ErrorCode)})
DT	I	10	Event Notification message received: {MessageType} - {NotificationID} - {DataEvaluator (ParticipantID)} - {DataOwner (ParticipantID)} - ({list (NotificationItemID):(CanonicalEventCatalogUri):(EventID)})
DT	I	11	Querying SML: {ParticipantID}
DT	I	12	Querying SMP: {ParticipantID} - {DocumentTypeID} - {ProcessID}
DT	E	01	Connection error with {destination component}: {inherited error message}
DT	E	02	Error on response from {error source component}: {inherited error message}
DT	E	03	Object conversion error in {error source component}: {inherited error message}
DT	E	04	Error accessing/saving data on {error source component}: {inherited error message}
DT	E	05	Missing or invalid arguments at {service requested}: {missing arguments}
DT	E	06	Service requested {REST service requested} to {destination component} not found
DT	E	07	Scheme validation failed for {MessageType}: {inherited error message}
DT	E	08	Error occurred on AS4 Gateway to {destination component}: {inherited error message}

7.2.3 SSI Authority Agent

Table 40: Authority Agent logging codes

Component	Level	Template	Specific text message
AA	I	01	{APIMethod}: Received input eIDAS user data.
AA	I	02	{APIMethod}: Received input userId data.
AA	I	03	{APIMethod}: Generated DID invitation for edge agent with ID {UUID}.
AA	I	04	{APIMethod}: Received input evidence data.
AA	I	05	{APIMethod}: Signed a Verifiable Credential.
AA	I	06	{APIMethod}: Sent Verifiable Credential {VC Id} to the edge agent under invitation {UUID} from {DO URI}.
AA	I	07	{APIMethod}: Accepted a submitted Verifiable Presentation.

Component	Level	Template	Specific text message
AA	I	08	{APIMethod}: Decoded input eIDAS user data.
AA	I	09	{APIMethod}: Validated the digital signature of the submitted VP.
AA	I	10	{APIMethod}: Validated the subject of the submitted VP.
AA	I	11	{APIMethod}: Received HTTP response code: {ResponseCode} from endpoint: {Endpoint}.
AA	I	12	{APIMethod}: Processing the JSON response received from /{AriesAPIMethod}.
AA	I	13	{APIMethod}: Stored current state in {Component} internal database.
AA	I	14	{APIMethod}: Received user {DataObject} status data.
AA	I	15	{APIMethod}: Converted input evidence in format: {Format} to format: {Format}.
AA	I	16	{APIMethod}: Received response data PIID: {PIID}.
AA	I	17	{APIMethod}: Found a {DataObject} action match with PIID: {PIID}.
AA	I	18	{APIMethod}: Found a Verifiable Presentation with name: {VPName}.
AA	I	19	{APIMethod}: Found a Verifiable Presentation with ID: {VPID}.
AA	I	20	{APIMethod}: Received Verifiable Credential {VC Id} at the verifier {DE URI} under invitation {UUID}
AA	I	21	{APIMethod}: Validated Verifiable Credential {VC Id} of type {CanonicalEvidenceTypeUri} under invitation {UUID} for {DE URI}.
AA	I	22	{APIMethod}: Sent Offer for Verifiable Credential {VC Id} of type {CanonicalEvidenceTypeUri} under invitation {UUID} from {DO URI}.
AA	I	23	{Component}: Issuer DID has already been generated and registered in EBSI.
AA	I	24	{Component}: Successfully created files for EBSI integration.
AA	I	25	{Component}: Successfully generated key: {KeyType}, value: {Value}.
AA	I	26	{Component}: Successfully exported JWK private key.
AA	I	27	{Component}: Successfully imported DID document into Aries.
AA	I	28	{APIMethod}: Overwriting DID connection with ID {old invitation ID} to {new invitation ID} .
AA	I	29	{APIMethod}: DID connection has been established for invitation ID {invitation ID}.
AA	I	30	{APIMethod}: Validated the issuer of the submitted VP.
AA	I	31	{APIMethod}: Validated the schema of the submitted VP.

Component	Level	Template	Specific text message
AA	I	32	{APIMethod}: Event notification of type '{event type}' has been successfully sent to {frontend URI}. Received HTTP response code: {response code}.
AA	I	33	{Component}: Successfully anchored DID {did} for organization {organization alias} into the EBSI DID Registry.
AA	E	01	Connection error with {destination component}: {inherited error message}
AA	E	02	Error on response from {error source component}: {inherited error message}
AA	E	03	Object conversion error in {error source component}: {inherited error message}
AA	E	04	Error accessing/saving data on {error source component}: {inherited error message}
AA	E	05	Missing or invalid arguments at {service requested}: {missing arguments}
AA	E	06	Error generating {key} key.
AA	E	07	Error exporting JWK private key.
AA	E	08	Error importing DID document into Aries.
AA	E	09	Configuration error occurred on {error source component}
AA	E	10	Event notification of type '{event type}' could not be sent: {inherited error message}
AA	E	11	Event notification of type '{event type}' could not be sent: {inherited error message}

7.2.4 Data Evaluator

Table 41: Data Evaluator logging codes

Component	Level	Template	Specific text message
DE	I	01	[UseCase] DE4A pilot process started
DE	I	02	[UseCase] DE4A pilot process successfully finished
DE	E	03	[UseCase] Failed to establish user identity
DE	E	04	[UseCase] Preview rejected by the User

7.2.5 Data Owner

Table 42: Data Owner logging codes

Component	Level	Template	Specific text message
DO	E	01	[UseCase] Error extracting evidence/subscription
DO	E	02	[UseCase] Evidence/Subscription not available yet (delayed)

Component	Level	Template	Specific text message
DO	E	03	[UseCase] Error in identity matching
DO	E	04	[UseCase] Unsuccessful completion of preview
DO	E	05	[UseCase] Failed to re-establish user identity
DO	E	06	[UseCase] Preview rejected by the User
DO	E	07	[UseCase] Unsuccessful generation of the canonical evidence/subscription
DO	E	08	Connection error with {destination component}: {inherited error message}

8 Conclusions

As a starting point for the development of the common DE4A components, this deliverable has addressed the definition of the interfaces those components are going to use to share information. These interfaces meet the requirements and functionalities defined by the reference architecture of WP2 and the semantic tools of WP3.

The interfaces relate not only to the common components, but to the pilot components too. Therefore, this document is crucial for all development teams of DE4A, since their components must comply with the interfaces specified here.

Along with the interfaces, some common specifications for the whole project have been set out: the concrete way eDelivery is used in DE4A, the behaviour and specific interactions between components (low-level design) and the logs and error messages shared by all components.

Starting from deliverable D5.3 with the initial design of interfaces and common specifications, WP5 has taken advantage of the insights gained during the first iteration to refactor and improve the IEM data model and the way the components interact between them. Additionally, the interfaces designed implement two new functionalities introduced during the second iteration of the project:

- Two new interaction patterns: Subscription and Notification pattern and Lookup pattern
- Support for the exchange of multiple pieces of evidence in only one request/respond.

A new component has been also implemented to replace the Mocked IDK of iteration 1: the Central IAL.

Now, the next step is, for WP5, to implement the common components that realise these functionalities, and for WP4, to implement the pilot component that complies with the interfaces and specification outlined here.

References

[1] 2021-06-23 DE4A Policy for use of identifiers 1.0. Available at the DE4A Wiki: https://wiki.de4a.eu/images/1/19/DE4A_Policy_for_use_of_identifiers.pdf, retrieved on 09/02/2023

[2] DE4A D2.5 Project Start Architecture second iteration v1.0. Available at the DE4A website: https://www.de4a.eu/_files/ugd/2844e6_062d529331194f4a8e98f9edea9e701b.pdf , retrieved on 09/02/2023

[3] DE4A D3.6 Semantic Toolkit- Final Version v1.0. Available at the DE4A website: https://www.de4a.eu/_files/ugd/2844e6_f84ca136597f40c2922e9a2cfe56f65e.pdf , retrieved on 09/02/2023

[4] DE4A D4.1 Studying abroad - Use Case Definition &Requirements. Available at the DE4A website: https://www.de4a.eu/_files/ugd/2844e6_cd8abc5ce50048fa8fc32bb283775bc9.pdf , retrieved on 09/02/2023

[5] DE4A D4.5 Moving abroad - Use Case Definition &Requirements. Available at the DE4A website: https://www.de4a.eu/_files/ugd/2844e6_06009964818747a0824f7dd2f9f75fde.pdf , retrieved on 09/02/2023

[6] DE4A D4.9 Studying Abroad - Use Case Definition &Requirements. Available at the DE4A website: https://www.de4a.eu/_files/ugd/2844e6_a07e5bbeac0f404999beb323c82b1416.pdf , retrieved on 09/02/2023

[7] DE4A D5.2 Final inventory of features for products- components. Available at the DE4A website: https://www.de4a.eu/_files/ugd/2844e6_f93c452f9dc5483482da64310b1d5696.pdf , retrieved on 09/02/2023

[8] DE4A D5.8 Final Release of DE4A Self-Sovereign Identity Supporting Framework v1.0. Available at the DE4A website: https://www.de4a.eu/_files/ugd/2844e6_9bca028dda934be5ad5ea90a13233d8e.pdf , retrieved on 09/02/2023

[9] eDelivery building block portal of the European Commission, https://ec.europa.eu/digital-building-blocks/wikis/display/DIGITAL/eDelivery, retrieved on 09/02/2023

[10] RFC 2915 The Naming Authority Pointer (NAPTR) DNS Resource Record, https://www.ietf.org/rfc/rfc2915.txt, retrieved on 09/02/2023

[11] phoss Business Cards GitHub directory, https://github.com/phax/phoss-directory/tree/master/phoss-directory-businesscard/src/main/resources/schemas, retrieved on 09/02/2023

[12] phoss SMP GitHub directory, https://github.com/phax/phoss-smp, retrieved on 09/02/2023

[13]OASIS ebXML Messaging Services Version 3.0: Part 1, Core Features, http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/ebms_core-3.0-spec.html, retrieved on 09/02/2023

[14]phoss Directory GitHub directory, https://github.com/phax/phoss-directory, retrieved on 09/02/2023

[15]Apache Lucene webpage, https://lucene.apache.org/, retrieved on 09/02/2023

[16]IAL service in the WP5 GitHub directory, https://github.com/de4a-wp5/ial-service, retrieved on 09/02/2023

[17]OASIS ebXML RegRep Version 4.0, http://docs.oasis-open.org/regrep/regrep-core/v4.0/os/regrep-core-overview-v4.0-os.html, retrieved on 09/02/2023

Annexes

Annex I. Lists of codes

This section includes information regarding the list of codes used in DE4A project.

Table 43: Document Type identifiers

ID Name	ID Scheme	ID Value	Initial iteration	State	Note
DBA Company Registration	urn:de4a-eu:CanonicalEvidenceType	CompanyRegistration:1.0	1	active	Iteration 2 identifier
DBA Business events	urn:de4a-eu:CanonicalEventCatalogueType	BusinessEvents:1.0	2	active	Iteration 2 identifier
SA Higher Education Diploma	urn:de4a-eu:CanonicalEvidenceType	HigherEducationDiploma:1.0	1	active	Iteration 2 identifier
SA Higher Education Diploma	urn:de4a-eu:CanonicalEvidenceType	HigherEducationDiploma:2.0	2	active	Iteration 2 identifier
SA Secondary Education Diploma	urn:de4a-eu:CanonicalEvidenceType	SecondaryEducationDiploma:1.0	2	active	Iteration 2 identifier
SA Large Family Evidence	urn:de4a-eu:CanonicalEvidenceType	LargeFamilyEvidence:1.0	2	active	Iteration 2 identifier
SA Disability Evidence	urn:de4a-eu:CanonicalEvidenceType	DisabilityEvidence:1.0	2	active	Iteration 2 identifier
MA Residence Registration	urn:de4a-eu:CanonicalEvidenceType	ResidenceRegistration:1.0	1	active	Iteration 2 identifier
MA Marriage Registration	urn:de4a-eu:CanonicalEvidenceType	MarriageEvidence:1.0	1	active	Iteration 2 identifier
MA Birth Certificate	urn:de4a-eu:CanonicalEvidenceType	BirthCertificate:1.0	1	active	Iteration 2 identifier
Multi Item Evidence Request	urn:de4a-eu:CanonicalEvidenceType	MultiItem:1.0	2	active	Iteration 2 identifier
Multi Item Subscription Request	urn:de4a-eu:CanonicalEventCatalogueType	MultiItem:1.0	2	active	Iteration 2 identifier

Table 44: Process identifiers

ID Name	ID Scheme	ID Value	Initial iteration	State	Note
DE4A Request	urn:de4a-eu:MessageType	request	1	active	Iteration 2 identifier
DE4A Response	urn:de4a-eu:MessageType	response	1	active	Iteration 2 identifier
DE4A Notification	urn:de4a-eu:MessageType	notification	2	active	Iteration 2 identifier

Table 45: AS4 Transport profile

Protocol name	Profile version	Profile ID	Initial version	State
CEF AS4	1.0	bdxr-transport-ebms3-as4-v1p0	1	active