CEF Digital
Connecting Europe

Working group meeting #6

Future of CEF eDelivery & API guidelines for government

28 June 2021
10.00 – 13.00 CET

Agenda

Time	Items	Speakers
10.00 - 10.10	Welcome and introduction	Maya Madrid (CEF eDelivery Business Owner, DG CNECT H4)
10.10 - 10.50	Update on ISA^2 IPS REST API profile: • Presentation of the changes since last meeting • Round table for feedback • Timeline and next steps	Bogdan Dumitriu, Jerry Dimitriou, Vlad Veduta (CEF eDelivery Technical team, DIGIT D3)
10.50 – 11.20	REST API Pilot: • Practical demonstration	Bogdan Dumitriu, Joze Rihtarsic (CEF eDelivery Technical team, DIGIT D3)
11.20 - 11.30	*Break (10 mins)*	
11.30 - 12.00	Pilot Domibus integration with CEF EBSI (blockchain): • Practical demonstration	Bogdan Dumitriu, Joze Rihtarsic (CEF eDelivery Technical team, DIGIT D3)
12.00 - 13.00	Update on JRC's work on API guidelines for government: • Security & Privacy essentials highlights • Empirical analysis contractual conditions of APIs	Monica Posada (JRC B6), Lorenzino Vaccari (Consultant)

Future of CEF eDelivery and API guidelines for government

Overview of actions and next steps

2020 ISA² Innovative Public Services (IPS) action

Action implementers | **DIGIT + JRC (oversight by DG CONNECT)**

	Objective	What was done so far?	What we will be doing next?
API guidelines for government	• Building on the APIs4DGov study, this work investigates API-related essentials relevant to gov. organisations implementing their digital agendas	• API management and discoverability solutions **interim reports** • Analysis of API-related **security** and **traceability** issues • Analysis of the **legal perspective** of APIs governments	• **Delivery of interim reports** describing API solutions to manage security, privacy and traceability (Q2 2021) • Outline of the Legal and organizational analysis (Q2 2021)
REST API profile	• Definition of a **REST-based profile** as a candidate profile for future inclusion in CEF eDelivery, in order to support new patterns of data access and data sharing	• Final draft of the **ISA² IPS REST API profile** completed and shared with the WG for feedback • **REST API PoC** 70% completed • **Analysis log document** 80% completed	• Create final version **ISA² IPS REST API profile** and **analysis log document** • Completion of the **REST API PoC**
Integration with CEF EBSI (blockchain)	• Piloting the use by CEF eDelivery of **blockchain services** offered by the CEF EBSI building block	• **Functional specifications** finalised • **Pilot** finalised	

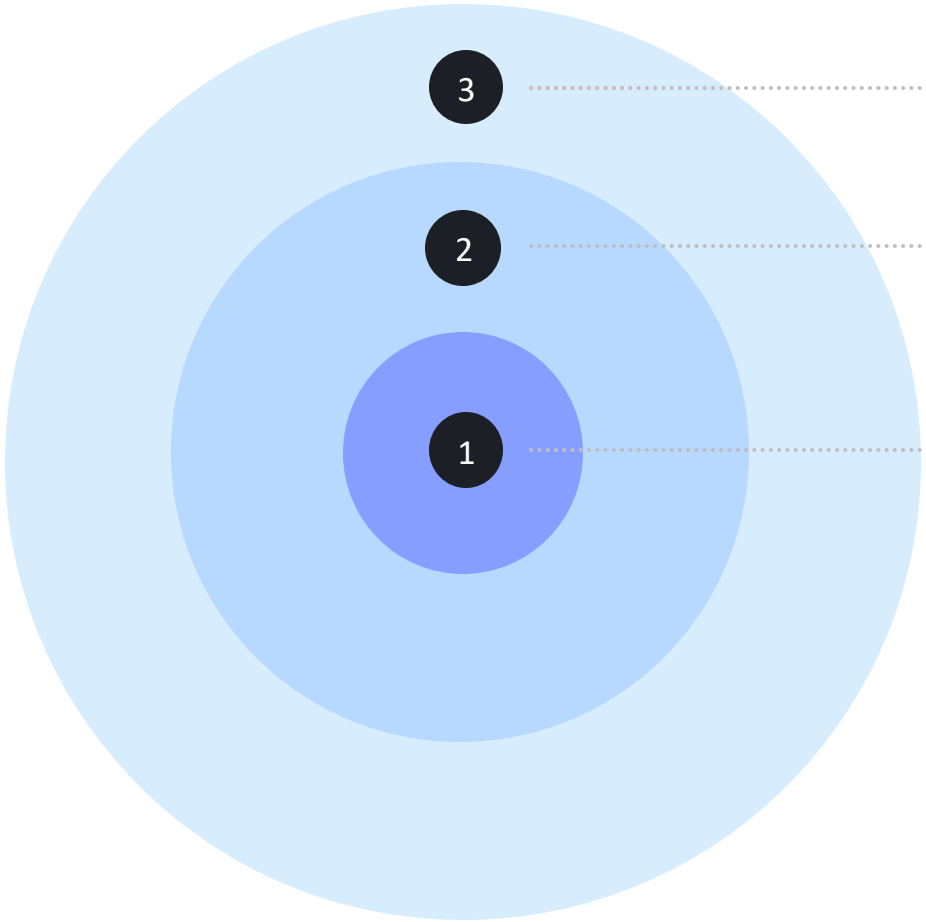
Update on ISA^2 IPS REST API profile

- Presentation of the changes since last meeting
- Round table for feedback
- Timeline and next steps

Bogdan Dumitriu, Jerry Dimitriou, Vlad Veduta - CEF eDelivery Technical team, DIGIT D3

REST API profile

Structure of the profile



Messaging API Specification

API Documentation

API Core Profile

Profile enhancement

- ✓ High-Security enhancement
- ✓ Discoverability enhancement

Changes Overview

- ✓ Comments provided by Member State stakeholders were taken into consideration
- ✓ Editorial changes, adding clarity and better structure where needed on all three layers
- ✓ Added new sections after evaluation of comments received in **API Core Profile**
  - ✓ Rest API design
  - ✓ Properly profiled JAdES for use for Message and Payload signature
  - ✓ Use of Multipart
- ✓ Updated **API Documentation** section following the changes in the core profile
- ✓ Updated **Messaging API Specification** section:
  - ✓ Many editorial updates, adding clarity and better structure
  - ✓ Definition of the User Message following Core Profile (Multipart)
  - ✓ Addition of schema definitions on payloads used in messaging API
  - ✓ Alignment with changes of the core profile on signatures

REST API Core Profile - Changes

Editorial and Structural Changes

- ✓ Editorial updates in the introduction section, adding clarity on what is profiled with this profile
- ✓ Re-order of Authorization Grants
  - ✓ Initially listing the ones that are recommended for use
  - ✓ Following by the listing of the ones that must not be used

REST API Core Profile - Changes

Signature support

- ✓ Payload and Message Signature are based on JAdES detached profile
- ✓ Defined a HTTP field for each type of signature
  - ✓ edel-payload-sig
  - ✓ edel-message-sig
- ✓ JAdES attribute profile for both types of signatures
  - ✓ HttpHeaders Mechanism of the JAdES Specification only
  - ✓ Minimum set of headers that must be used for the calculation of the signature
    - ✓ Different between types of signatures

REST API Core Profile - Changes

Signature support

- ✓ Payload Signature JAdES Profile:
  - ✓ The JWS content (Data to be Signed) MUST be detached from the signatures as defined in [RFC7515] Appendix F.
  - ✓ The signed **SigD** parameter object MUST be present in the JWS headers, denoting the use of the JAdES detached header profile.
  - ✓ The value of the **mId** parameter MUST be set to "http://uri.etsi.org/19182/HttpHeaders".
  - ✓ The **pars** array of the **SigD** MUST contain only the element **"digest"**, denoting that for the calculation of the signature only the digest of the HTTP payload must be taken into account, according to [RFC3230].
  - ✓ The **alg** parameter MUST be set to a value supported by the core profile
  - ✓ If the **alg** parameter is set to "EdDSA", the crv parameter MUST be set to the correct value (see above).
  - ✓ The JWS structure shall be carried in HTTP header field named "edel-payload-sig".

- ✓ Message Signature JAdES Profile:
  - ✓ Inherits from Payload signature profile, adding more headers in the pars array of SigD. The minimum headers are:
    - ✓ element **"(request-target)",** for containing the HTTP Request URI
    - ✓ element **"host"**, for containing the host the message was submitted to, if present
    - ✓ element **"content-encoding"**, if present
    - ✓ element **"digest"**, for taking into account the Digest header that contains the hash value of the HTTP payload.

REST API Core Profile - Changes

Common Semantics

- ✓ Added a new section in Common Semantics: REST API Design

- ✓ Patterns on the use of URIs as resources, queries, sub-resources etc.
- ✓ Rules on Resource Identifiers
- ✓ Rules on Resource Archetypes
  - ✓ Document, Collection, Store, Controller
- ✓ Rules on Query and Pagination parameters
  - ✓ Definition of specific keywords as predefined query types
- ✓ Rules on Projection

REST API Core Profile - Changes

Common Semantics – URI patterns

✓ Patterns on the use of URIs as resources, queries, sub-resources etc.

Term	Description	Formal Definition	Example
resource-path	The hierarchical URI sub structure defining an addresable resource	'/' base-url '/' version ('/' namespace)* ('/' resource)+	/archive-system/v2/messaging-service/messages/123-121
base-url	The base URL of the deployed API	ALPHA (ALPHA | DIGIT | '-')*	archive-system
version	The version of the deployed API	'v' (DIGIT)+	v2
namespace	Namespace identifiers are used to provide a context and scope for resources. They are determined by logical boundaries implemented by the API platform.	ALPHA (ALPHA | DIGIT | '-')*	messaging-service
resource	The hierarchical structure of a (sub)resource	resource-name ['/' resource-id]	messages/123-121
resource-name	The name of the resource	ALPHA (ALPHA | DIGIT | '-')*	messages
resource-id	The identifier of a resource	value	123-121
query	The filter string	name '=' value ('&' name = value)*	subject=report
name	The filter name	ALPHA (ALPHA | DIGIT | '_')*	subject
value	A value that is potentially URL encoded (used as a resource identifier or as a filter value)	URL-encoded value	report%202020

REST API Core Profile - Changes

Common Semantics – Resource Archetypes

- ✓ **Collection** Archetype
  - ✓ server-managed list of resources.
  - ✓ Up to the collection to choose to create a new resource or not.
  - ✓ A collection defines the URIs of each contained resource.
  - ✓ Plural nouns MUST be used to denote collection resource archetypes.
    - ✓ http://example.com/api/v1/library/books/

- ✓ **Store** Archetype
  - ✓ client-managed resource repository.
  - ✓ A store resource lets an API client put resources in, get them back out, and decide when to delete them.
  - ✓ The URI was chosen by a client when it was initially put into the store.\
  - ✓ Plural nouns MUST be used to denote collection resource archetypes.
    - ✓ http://example.com/api/v1/messaging/messages

REST API Core Profile - Changes

Common Semantics – Resource Archetypes

- ✓ **Document** Archetype
  - ✓ Singular concept that is akin to an object instance.
  - ✓ Document resources are usually inside collection resources.
  - ✓ Singular nouns or resource identifiers MUST be used to denote document resource archetypes
    - ✓ http://example.com/api/v1/messaging/messages/{message-id}
    - ✓ http://example.com/api/v1/user-management/users/admin

- ✓ **Controller** Archetype
  - ✓ Models a procedural concept.
  - ✓ Controller resources are like executable functions, with parameters and return values, inputs and outputs.
  - ✓ Controllers are the exception to the rule, using **verbs** to denote controller archetypes
  - ✓ They MUST appear as the last segment in a resource URI.
    - ✓ http://example.com/api/v1/cart-management/users/{user-id}/cart/**checkout**

REST API Core Profile - Changes

Common Semantics – Query and Pagination

- ✓ Pre-defined query and pagination parameters
  - ✓ **limit:** The number of resources of a collection to be returned from a request.
  - ✓ **offset:** The offset the response should start providing resources of the collection.
  - ✓ **q:** A generic query parameter used to express complex queries on the resource. The structure of the query string is resource specific and MUST be defined per resource
  - ✓ **sort**: Used to express the sorting order of resources in a collection.
    - ✓ It MUST follow the following regular expression: (-|+)<field-name> (',' (-|+)<field-name>)* where:
      - ✓ "+" denotes ascending order and "-" descending order,
      - ✓ <field-name> is a string representation of a field of the resource,
      - ✓ the sort order of resources should follow the order of the fields.

REST API Core Profile - Changes

Common Semantics – Projection

- ✓ The profile recommends the support of two different resource projections:
  - ✓ The complete one, returning the full structure of the resource representation of all the resources of a collection.
  - ✓ The minimal one, returning only the minimal required structure to distinguish a resource inside the collection. This can be a single Identifier or a collection of mandatory fields of the resource.

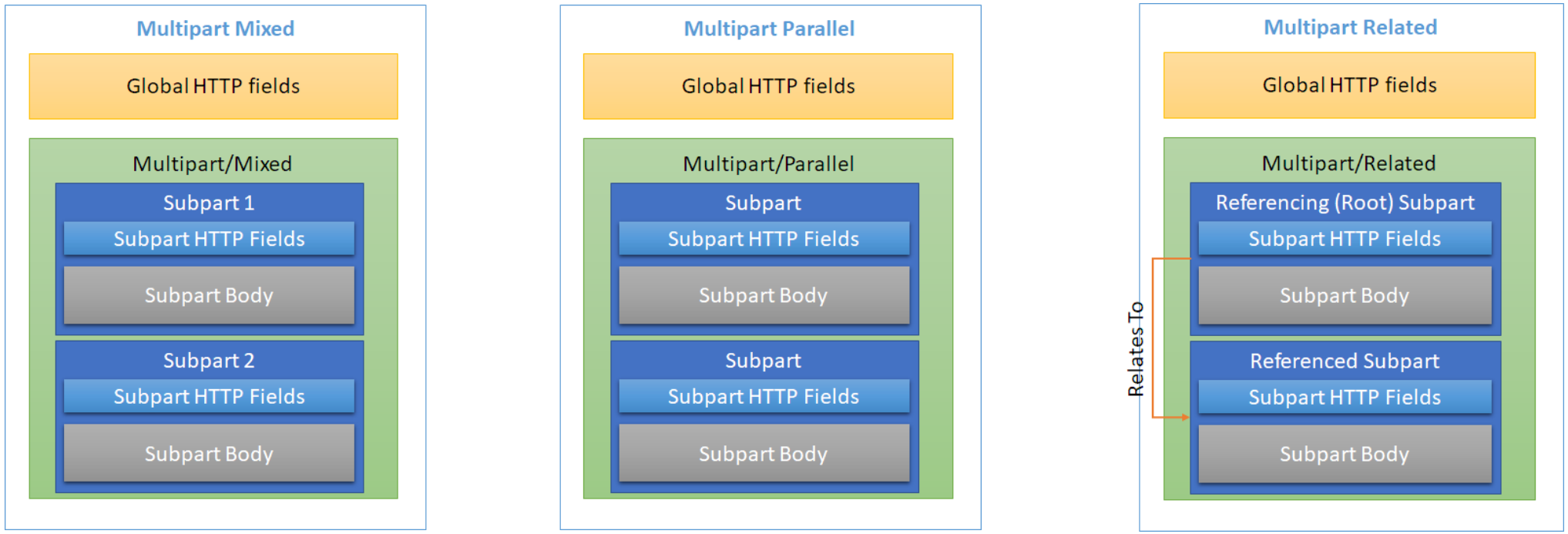
- ✓ Use of the **Prefer HTTP field** as defined in [RFC7240] for requesting the minimal or complete representation of resources.
  - ✓ Complete representation: Prefer header with the value "return=representation”
    - ✓ Prefer: return=representation
  - ✓ Minimal representation: Prefer header with the value "return=minimal“
    - ✓ Prefer: return=minimal

- ✓ As the "Prefer" header is optional, it MUST be declared in the operations of the OpenAPI Document when supported by the API.

REST API Core Profile - Changes

Multipart

- ✓ Elaborated on the use of Multipart in the Core Profile
- ✓ Rules on when and which multipart should be used.
  - ✓ Mixed, Parallel, Related



REST API Documentation - Changes

- ✓ Editorial updates, clarifications etc
- ✓ Updates on the signature documentation based on the core profile changes
- ✓ Additions and updates of json schemas
- ✓ Generation of the OpenAPI document of the REST API Profile
  - ✓ Contains defined objects like headers, signatures etc.
  - ✓ Should be referenced when creating compliant-based APIs

```
components:
  headers:
    edel-message-sig:
      description: The custom header used for carrying the detached signature
                   for signing the HTTP Message
      schema:
        $ref: '#/components/schemas/jws-compact-detached'

    edel-payload-sig:
      description: The custom header used for carrying the detached
                   signature for signing the payload
      schema:
        $ref: '#/components/schemas/jws-compact-detached'

    deprecation:
      description: Deprecation HTTP Response Header following the Internet-
                   Draft "The Deprecation HTTP Header Field
      schema:
        $ref: '#/components/schemas/date-value'

    sunset:
      description: Sunset HTTP Response Header following [RFC8594].
      schema:
        $ref: '#/components/schemas/date-value'
```

Messaging API - Changes

Editorial and structural changes

- ✓ Redefinition of the common message fields, using hyphen as a separator instead of camel case:
  - ✓ OriginalSender → Original-Sender
  - ✓ FinalRecipient → Final-Recipient
  - ✓ RefToMessageId removed
  - ✓ Edel-Message-Sig (New Field)
  - ✓ Edel-Payload-Sig (New Field)
  - ✓ ResponseWebhook → Response-Webhook
  - ✓ SignalWebhook → Signal-Webhook

Messaging API - Changes

API endpoint mapping with MEP

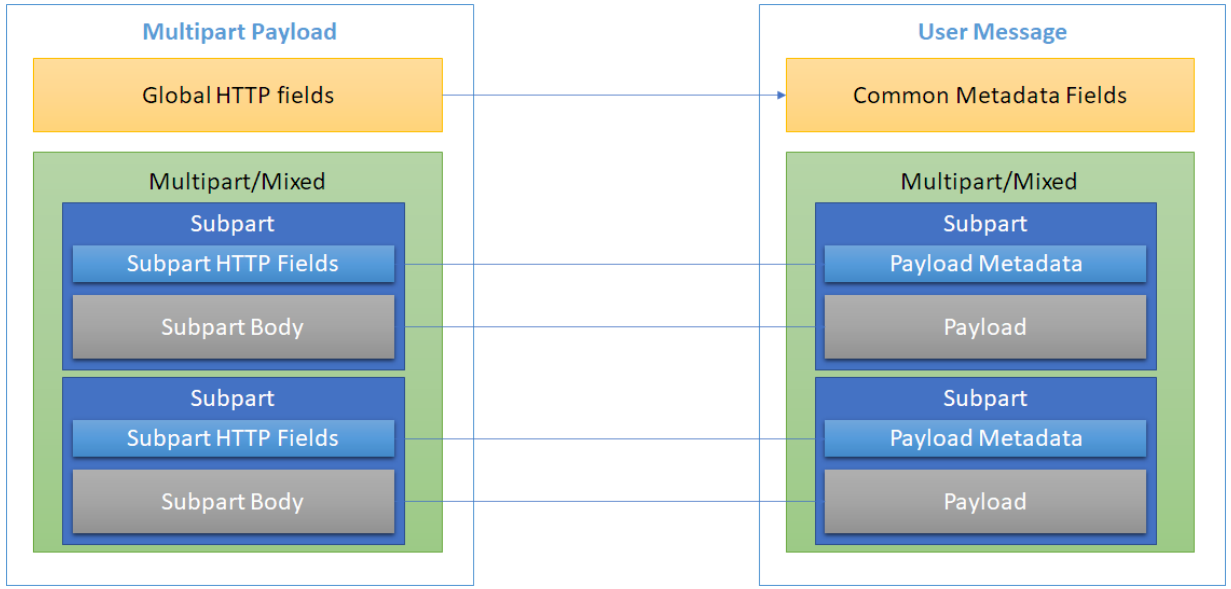
✓ API Endpoints updated to better describe their relation with the Message Exchange Patterns

Mapping: Message Exchange Patterns to Endpoints	Message Submission	Message Submission with Sync Response Message	Response Message Submission	Webhook Response Message Submission	Webhook Signal Submission	Get Message Reference List	Get Message	Get Response Message Reference List	Get Response Message
Send Message with No Response – Push	Message Receiver								
Send Message with No Response – Pull						Message Sender	Message Sender		
Send Message with Synchronous Response		Message Receiver							
Send Message with Asynchronous Response – Push and Pull	Message Receiver							Message Receiver	Message Receiver
Send Message with Asynchronous Response – Push and Webhook Pull	Message Receiver				Message Sender			Message Receiver	Message Receiver
Send Message with Asynchronous Response – Push and Webhook Push	Message Receiver			Message Sender					
Send Message with Asynchronous Response – Pull and Push			Message Sender			Message Sender	Message Sender		

Messaging API - Changes

User Message Definition

- ✓ Use of the Multipart section of the Core API profile



- ✓ Signature profile re-definition, based on the new Jades-based structure of the Core API Profile

Messaging API - Changes

Signal Message Definition

✓ Schema definition of the Signal Message derived from problem+json


```
{
   "$schema": "https://json-schema.org/draft-07/schema#",
   "$id": "https://joinup.ec.europa.eu/collection/api4dt/solution/.../messaging.schema.json",
   "title": "A Problem Details object (RFC 7807) defined by the ISA² IPS REST API Core Profile",
   "description": "A Problem Details object (RFC 7807) with ISA² IPS REST API extensions, used for signals (responses) to messages",
   "type": "object",
   "properties": {
      "type": {
         "type": "string",
         "format": "uri",
         "description": "An URI reference that identifies the problem type. When dereferenced, it SHOULD provide human-readable documentation for the problem type (e.g. using HTML
         "example": "https://joinup.ec.europa.eu/collection/api4dt/solution/.../messaging/signal/message-accepted"
      },
      "title": {
         "type": "string",
         "description": "A short summary of the problem type, written in English and readable for engineers (usually not suited for non technical stakeholders and not localized)."
         "example": "Message Accepted"
      },
      "status": {
         "type": "integer",
         "format": "int32",
         "description": "The HTTP status code generated by the origin server for this occurrence of the problem.",
         "minimum": 200,
         "exclusiveMaximum": 600,
         "example": 202
      },
      "detail": {
         "type": "string",
         "description": "A human-readable explanation specific to this occurrence of the problem."
      },
      "instance": {
         "type": "string",
         "format": "uri-reference",
         "description": "A URI reference that identifies the specific occurrence of the problem. It may or may not yield further information if dereferenced."
      },
      "digest": {
         "type": "string",
         "description": "The digest of the received message using the notation proposed in 'Digest Header' (https://datatracker.ietf.org/doc/html/draft-ietf-httpbis-digest-headers
         "example": "sha-256=4REjxQ4yrqUVicfSKYNO/cF9zNj5ANbzgDZt3/h3Qxo=,"
      }
   },
   "required": ["status", "type", "title", "instance"],
   "additionalProperties": false
}
```

Messaging API - Changes

Signal Message Definition

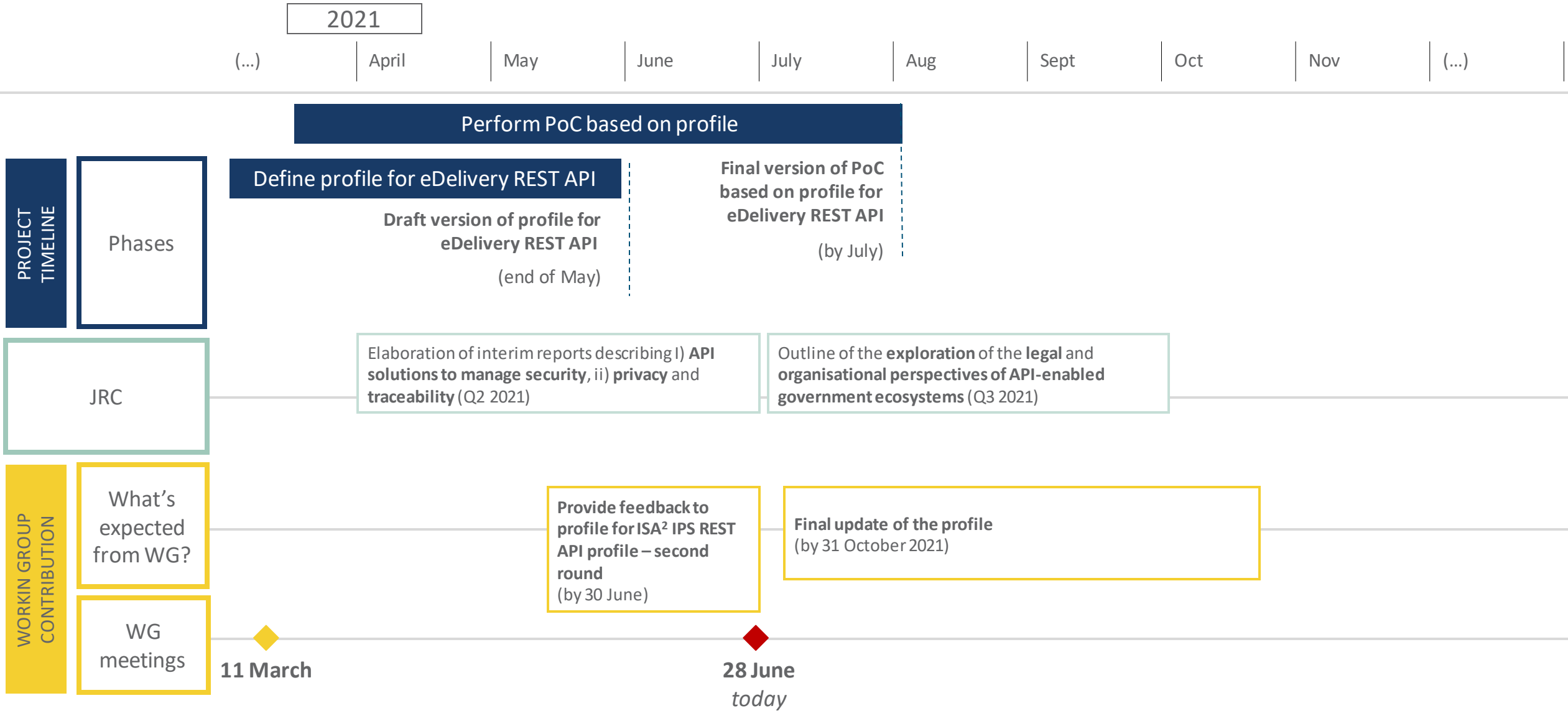
- ✓ List of pre-defined signals that are used in the MEPs
- ✓ Used as notifications or responses

No	Signal Title	Signal Status (code)	Signal Type	Signal Description
1	Message Accepted	202	https://joinup.ec.europa.eu/collection/api4dt/solution/.../messaging/signal/message-accepted	Sent when the message is properly validated. It may include a status monitor that can provide the user with an estimate of when the request will be fulfilled (see [RFC7231])
2	Validation Failed	400	https://joinup.ec.europa.eu/collection/api4dt/solution/.../messaging/signal/message-validation-failed	Sent when the message fails the validation process
3	Invalid or Duplicate Message ID	400	https://joinup.ec.europa.eu/collection/api4dt/solution/.../messaging/signal/invalid-message-id	Sent when the MessageId is not valid
4	Invalid Message Signature	400	https://joinup.ec.europa.eu/collection/api4dt/solution/.../messaging/signal/invalid-message-signature	Sent when the message signature cannot be verified
5	Invalid Addressing	400	https://joinup.ec.europa.eu/collection/api4dt/solution/.../messaging/signal/invalid-addressing	Sent when the Original Sender or Final Recipient(s) cannot be resolved
6	Invalid Message Format	400	https://joinup.ec.europa.eu/collection/api4dt/solution/.../messaging/signal/invalid-format	Sent when the message format does not adhere to the specification
7	Pull Error: No final recipient configured for the pulling user	400	https://joinup.ec.europa.eu/collection/api4dt/solution/.../messaging/signal/pull/no-final-recipient	Sent when the server cannot resolve/match the pulling user to a final recipient
8	Pull Error: No Message Found	404	https://joinup.ec.europa.eu/collection/api4dt/solution/.../messaging/signal/pull/no-message-found	Sent when no message is found that maps to the pull request
9	Pull Error: Unauthorized	401	https://joinup.ec.europa.eu/collection/api4dt/solution/.../messaging/signal/pull/unauthorized	Sent when the pull request is unauthorized
10	Message Response is ready	201	https://joinup.ec.europa.eu/collection/api4dt/solution/.../messaging/signal/message-ready	An HTTP Request following [RFC7807] MUST be sent when a message response is ready to be retrieve

Round table for feedback

REST API profile for eDelivery

Indicative timeline and next steps



REST API Pilot

- Practical demonstration

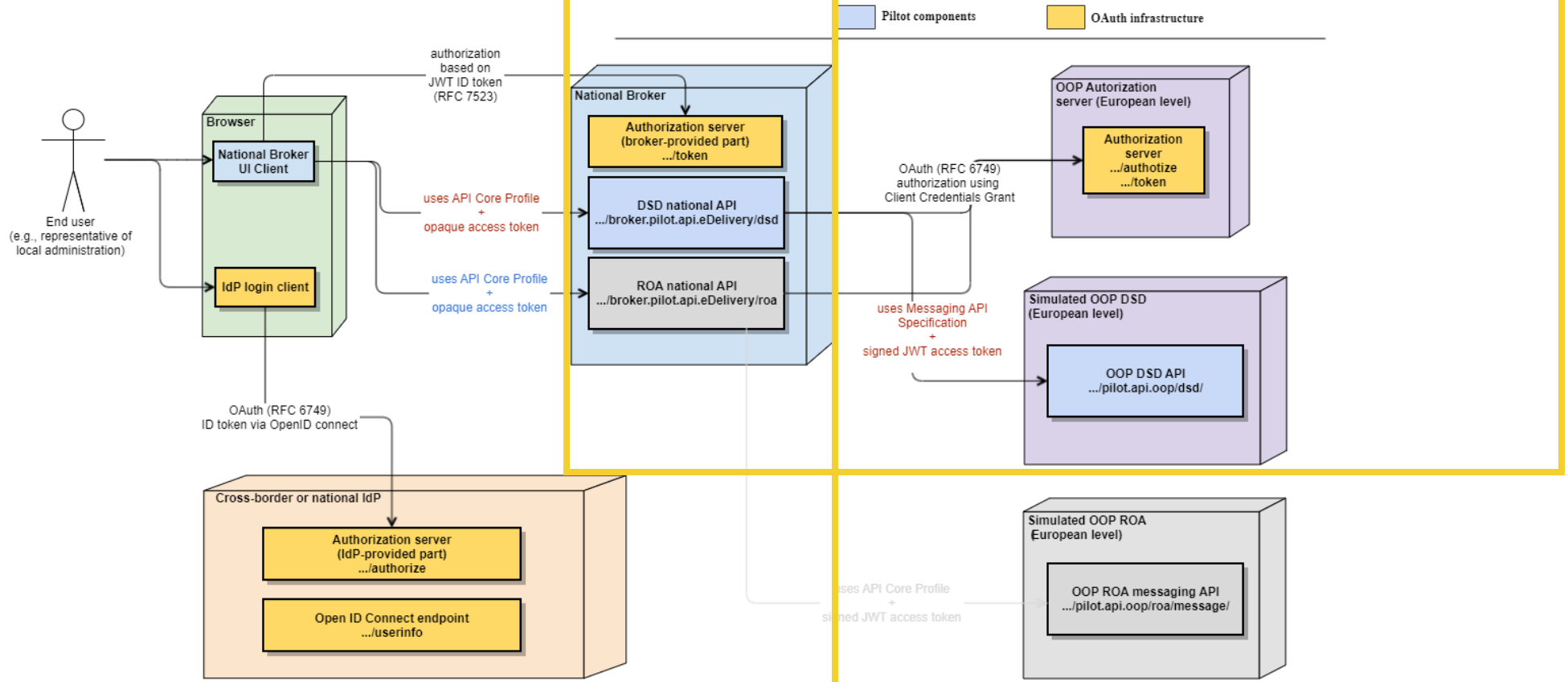
Bogdan Dumitriu, Joze Rihtarsic - CEF eDelivery Technical team, DIGIT D3

Authentication and authorization

Topology of client-server connections using OAuth 2 as framework for A&A:

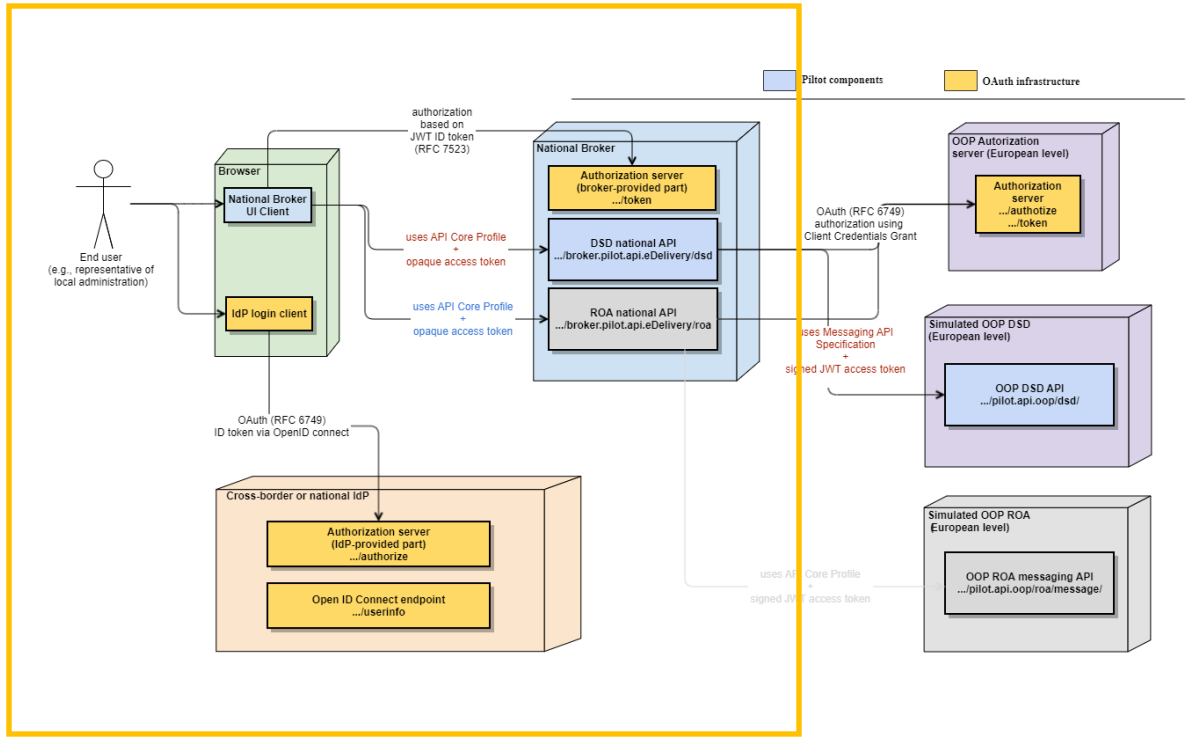
Internal AS with external IDP topology:
UI client connects to the National Broker

External AS with internal IDP topology:
National Broker connects to the DSD



UI client – National Broker

OAuth flow used with *IdP*: **authorization code with PKCE (RFC 7636)**
Client application type: **public (i.e., client is not trusted by AS)**
IdP scope: **openid (to indicate OpenID Connect is used)**
ID token format: **JWT (JSON Web Token)**
OAuth grant type with *AS*: **JWT bearer grant type (RFC 7523)**
Access Token format: **opaque**



Mapping of OAuth roles to pilot components:

OAuth role	Pilot component
Resource Server (RS)	National Broker
Authorization Server (AS)	National Broker (/token endpoint)
Client Application	UI client
End User	UI client end user
IdP	Auth0 (third-party service)

UI client – National Broker

OAuth flow used with *IdP*: **authorization code with PKCE (RFC 7636)**
Client application type: **public (i.e., client is not trusted by AS)**
IdP scope: **openid (to indicate OpenID Connect is used)**
ID token format: **JWT (JSON Web Token)**
OAuth grant type with *AS*: **JWT bearer grant type (RFC 7523)**
Access Token format: **opaque**

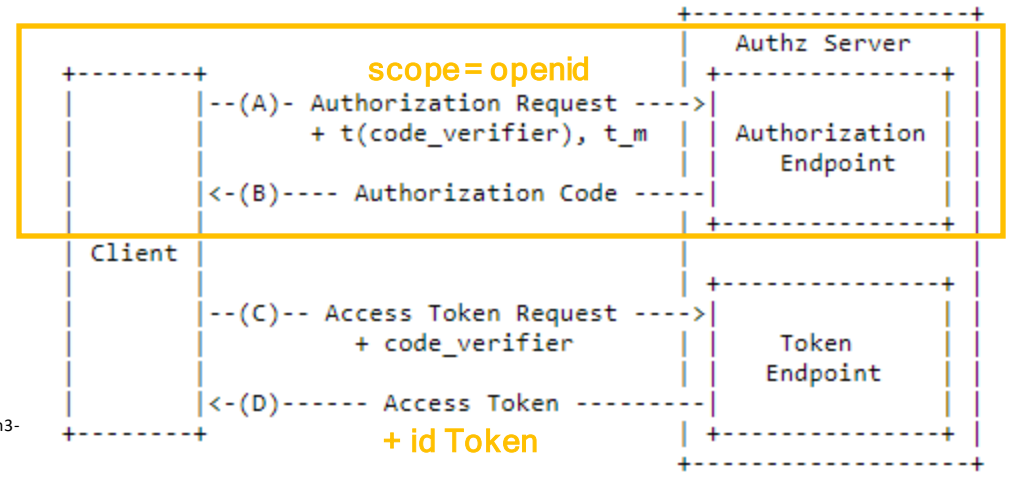
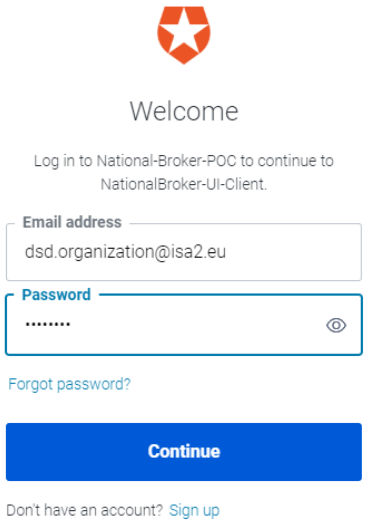


Figure 2: Abstract Protocol Flow



GET https://national-broker-poc.eu.auth0.com/authorize?**response_type=code**&state=xyzABC123&client_id=**6WfybVUzHbiLr**&**scope=openid**%20profile%20email&redirect_uri=http%3A%2F%2Fnational-broker-server%3A8080%2Fnational-broker%2Foauth%2Ftoken&**code_challenge=enVHTBRp_6NybyU891nIXwpc_WxlHS-1MrnbgyOZw4E**&**code_challenge_method=S256**:
Request Headers: {
"upgrade-insecure-requests": "1",
"user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Postman/8.6.2 Chrome/83.0.4103.122 Electron/9.4.3 Safari/537.36",
"accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9",
"sec-fetch-site": "none",
"sec-fetch-mode": "navigate",
"sec-fetch-user": "?1",
"sec-fetch-dest": "document",
"accept-encoding": "gzip, deflate, br",
"accept-language": "en-US"
}

"Response Headers": {
"alt-svc": "h3-27=\":443\"; ma=86400, h3-28=\":443\"; ma=86400, h3-29=\":443\"; ma=86400, h3=\":443\"; ma=86400",
"cache-control": "no-store, max-age=0, no-transform",
"cf-cache-status": "DYNAMIC",
"cf-ray": "66464e986a352e2c-BRU",
"cf-request-id": "0adfcf734400002e2c5f022000000001",
"content-length": "252",
"content-type": "text/html; charset=utf-8",
"date": "Thu, 24 Jun 2021 13:29:47 GMT",
"expect-ct": "max-age=604800, report-uri=\"https://report-uri.cloudflare.com/cdn-cgi/beacon/expect-ct\"",
"location": "http://national-broker-server:8080/national-broker/oauth/token?**code=RI1dgIPvQ6GCt-a_**&state=xyzABC123",
"ot-baggage-auth0-request-id": "66464e986a352e2c",
"ot-tracer-sampled": "true",
"ot-tracer-spanid": "6e731df32cc29bf6",
"ot-tracer-traceid": "2e4fa97947046606",
"pragma": "no-cache",
"server": "cloudflare",
"set-cookie": ["auth0=s%3AaK...hxI; Path=/; Expires=Sun, 27 Jun 2021 13:29:47 GMT; HttpOnly; Secure; SameSite=None", "auth0_compat=s%3AaK...0cohxI; Path=/; Expires=Sun, 27 Jun 2021 13:29:47 GMT; HttpOnly; Secure"
],
"status": "302",
"strict-transport-security": "max-age=31536000",
"vary": "Accept, Accept-Encoding",
"x-auth0-requestid": "f4a98caf2eb832d030c0",
"x-content-type-options": "nosniff",
"x-ratelimit-limit": "300",
"x-ratelimit-remaining": "299",
"x-ratelimit-reset": "1624541388"
}

UI client – National Broker

OAuth flow used with *IdP*: **authorization code with PKCE (RFC 7636)**
Client application type: **public (i.e., client is not trusted by AS)**
IdP scope: **openid (to indicate OpenID Connect is used)**
ID token format: **JWT (JSON Web Token)**
OAuth grant type with *AS*: **JWT bearer grant type (RFC 7523)**
Access Token format: **opaque**

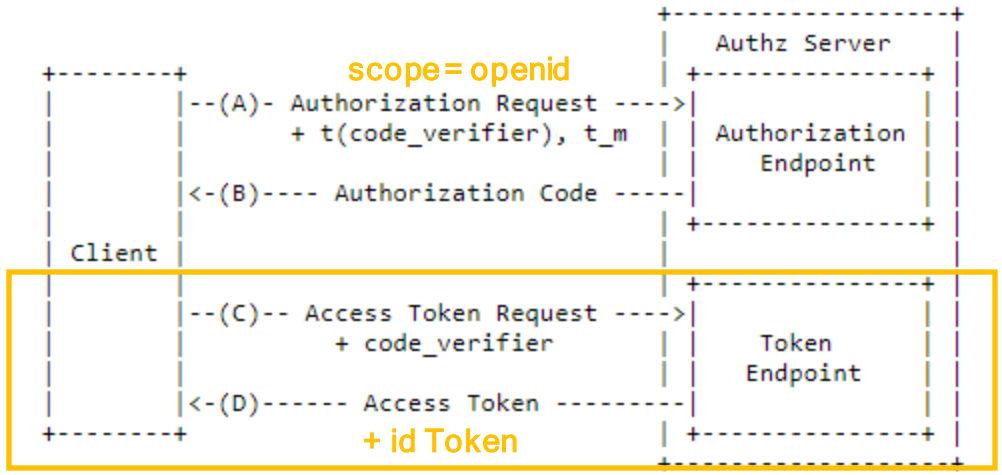


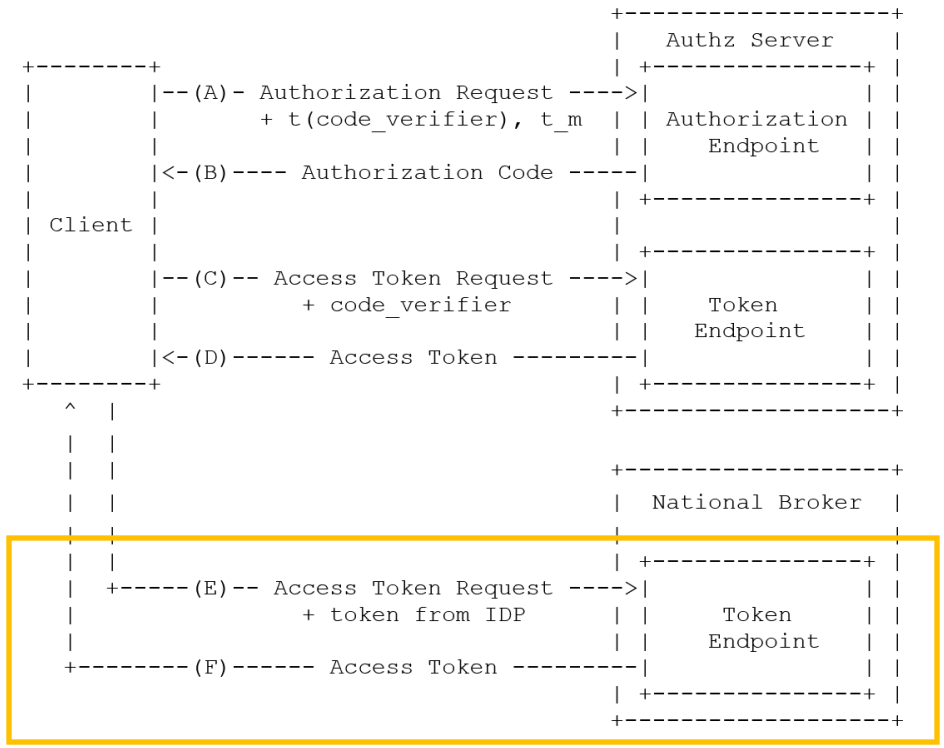
Figure 2: Abstract Protocol Flow

POST https://national-broker-poc.eu.auth0.com/**oauth/token**:
"Request Headers": {
"accept": "*/*",
"host": "national-broker-poc.eu.auth0.com",
"accept-encoding": "gzip, deflate, br",
"connection": "keep-alive",
"content-type": "application/x-www-form-urlencoded",
"content-length": "242",
"cookie": "did=s%3.....6g"
},
"Request Body": {
"grant_type": "**authorization_code**",
"client_id": "**6Wfyb...biLr**",
"code_verifier": "**CxzFs0k_Y8Mxte-NboFmBTn2oS5v6J5sSfOsLzDfCtM**",
"code": "**RI1dgIPvQ6GCt-a_**",
"redirect_uri": "http://national-broker-server:8080/national-broker/oauth/token" }

Response Headers": {
"date": "Thu, 24 Jun 2021 12:26:49 GMT",
"content-type": "application/json",
"transfer-encoding": "chunked",
"connection": "keep-alive",
"cf-ray": "6645f25cffa3d4a4-BRU",
"cache-control": "no-store",
"strict-transport-security": "max-age=31536000",
"vary": "Accept-Encoding, Origin",
"cf-cache-status": "DYNAMIC",
"cf-request-id": "0adf95ce190000d4a47333f000000001",
"expect-ct": "max-age=604800, report-uri=\"https://report-uri.cloudflare.com/cdn-cgi/beacon/expect-ct\"",
"pragma": "no-cache",
"x-auth0-requestid": "953a996ca9daf320fd0c",
"x-content-type-options": "nosniff",
"x-ratelimit-limit": "30",
"x-ratelimit-remaining": "29",
"x-ratelimit-reset": "1624537610",
"server": "cloudflare",
"content-encoding": "br",
},
"Response Body": "{
"access_token":"eyJh...BTMiJ9.eyJp...FpbCJ.U4...VDpIQ",
"id_token":"**ey...J9.ey...X0.By...NA**",
"scope":"**openid** profile email",
"expires_in":86400,
"token_type":"Bearer"}

UI client – National Broker

OAuth flow used with *IdP*: **authorization code with PKCE (RFC 7636)**
Client application type: **public (i.e., client is not trusted by AS)**
IdP scope: **openid (to indicate OpenID Connect is used)**
ID token format: **JWT (JSON Web Token)**
OAuth grant type with *AS*: **JWT bearer grant type (RFC 7523)**
Access Token format: **opaque**

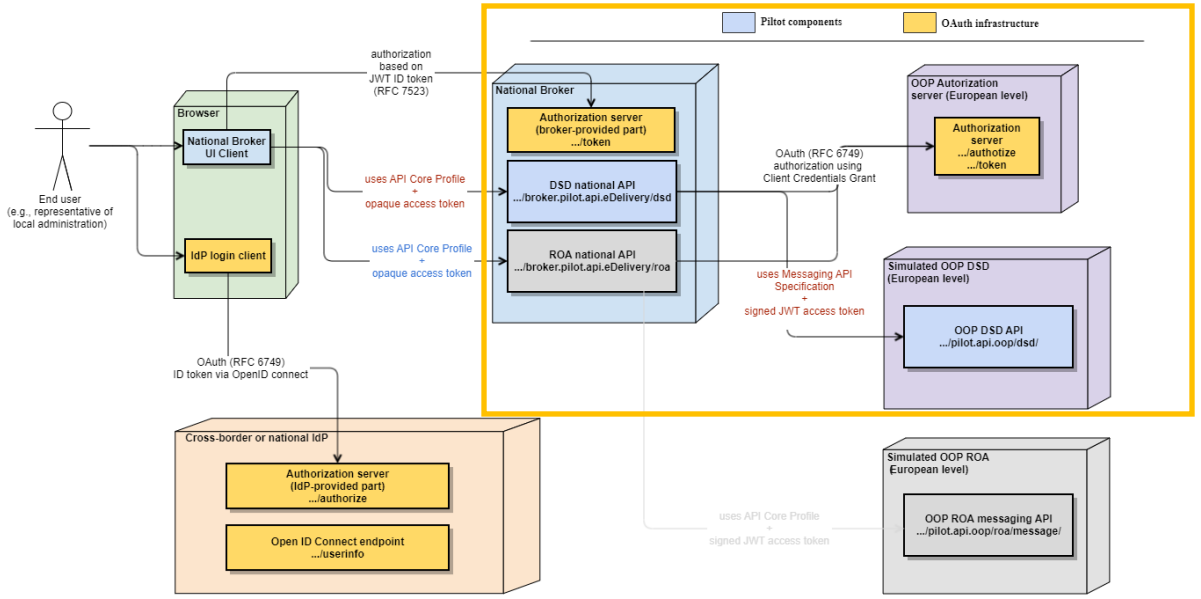


POST http://national-broker-server:8080/national-broker/**oauth/token**: {
"Request Headers": {
"content-type": "application/x-www-form-urlencoded",
"accept": "*/*",
"host": "national-broker-server:8080",
"accept-encoding": "gzip, deflate, br",
"connection": "keep-alive",
"content-length": "933",
"cookie": "JSESSIONID=_oiukU4DFmtOT2p.ord03332"
},
"Request Body": {
"grant_type":
"**urn%3Aietf%3Aparams%3Aoauth%3Agrant-type%3Ajwt-bearer**",
"assertion":"**eyJh…MiJ9.eyJp…bCJ9.U4xa…DpIQ**"
},

"Response Headers": {
"connection": "keep-alive",
"vary": [
"Origin",
"Access-Control-Request-Method",
"Access-Control-Request-Headers"
],
"content-type": "application/json",
"content-length": "116",
"date": "Thu, 24 Jun 2021 12:44:05 GMT"
},
"Response Body": "{
"access_token":"**88fdb060-7149-4871-ba49-563ae8d9a181**",
"expires_in":"3600",
"iat":"1624538645",
"token_type":"**Bearer**"}"
}

National broker – DSD

OAuth flow: **Client Credentials Grant**
Client application type: **confidential**
OAuth grant type: **client_credentials**
OAuth scope: **ROLE_UPDATE_DSD**
Access Token format: **JWT**



Mapping of oAuth roles to pilot components:

OAuth role	Pilot component
Resource Server (RS)	DSD
Authorization Server (AS)	OOP Authorization server (Okta - third party service)
Client Application	National Broker
End User	Not used
IdP	Internal to OOP authorization server

National broker – DSD

OAuth flow: **Client Credentials Grant**
Client application type: **confidential**
OAuth grant type: **client_credentials**
OAuth scope: **ROLE_UPDATE_DSD**
Access Token format: **JWT**

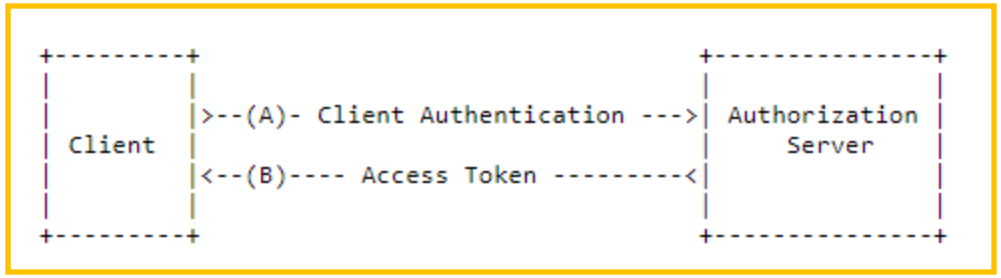


Figure 6: Client Credentials Flow

POST https://dev-24443841.okta.com/oauth2/aus1096gcr9r537Ut5d7/v1/**token:**


```
{
 "Request Headers": {
  "authorization": "Basic MG9hdD...NsQUU1dQ==",
  "accept": "*/*",
  "host": "dev-24443841.okta.com",
  "accept-encoding": "gzip, deflate, br",
  "connection": "keep-alive",
  "content-type": "application/x-www-form-urlencoded",
  "content-length": "44",
  "cookie": "DT=DI0w1S3PFeXRoulPfFk2EqbPA"
 },
 "Request Body": {
  "grant_type": "client_credentials",
  "scope": "ROLE_UPDATE_DSD"
 }
```



```
"Response Headers": {
  "date": "Thu, 24 Jun 2021 13:50:22 GMT",
  "content-type": "application/json",
  "transfer-encoding": "chunked",
  "connection": "keep-alive",
  "server": "nginx",
  "public-key-pins-report-only": "pin-sha256=\"r5EfzZx...2zt8=\"; pin-sha256=\"MaqI...vnQ=\"; pin-sha256=\"72G5IE...iyJI=\"; pin-sha256=\"rrV6...jCOg=\"; max-age=60",
  "x-okta-request-id": "YNSNnIXa3WffzsEWcYf4eAAADVw",
  "x-xss-protection": "0",
  "p3p": "CP=\"HONK\"",
  "x-rate-limit-limit": "300",
  "x-rate-limit-remaining": "299",
  "x-rate-limit-reset": "1624542682",
  "cache-control": "no-cache, no-store",
  "pragma": "no-cache",
  "expires": "0",
  "content-security-policy": "default-src 'self' dev-24443841.okta.com *.oktacdn.com;...",
  "expect-ct": "report-uri=\"https://oktaexpectct.report-uri.com/r/t/ct/reportOnly\", max-age=0",
  "x-content-type-options": "nosniff",
  "strict-transport-security": "max-age=315360000; includeSubDomains",
  "x-robots-tag": "none",
  "set-cookie": [
    "sid=\"\"; Expires=Thu, 01-Jan-1970 00:00:10 GMT; Path=/",
    "JSESSIONID=4160C25E67990291B0EE00F5D81585D5; Path=/; Secure; HttpOnly" ] },
 "Response Body": "{
"token_type":"Bearer",
"expires_in":3600,
"access_token":"eyJr...YifQ.eyJ2...NiJ9.APEu...mbrg",
"scope":"ROLE_UPDATE_DSD"}"}
```

REST API Implementation

Core API

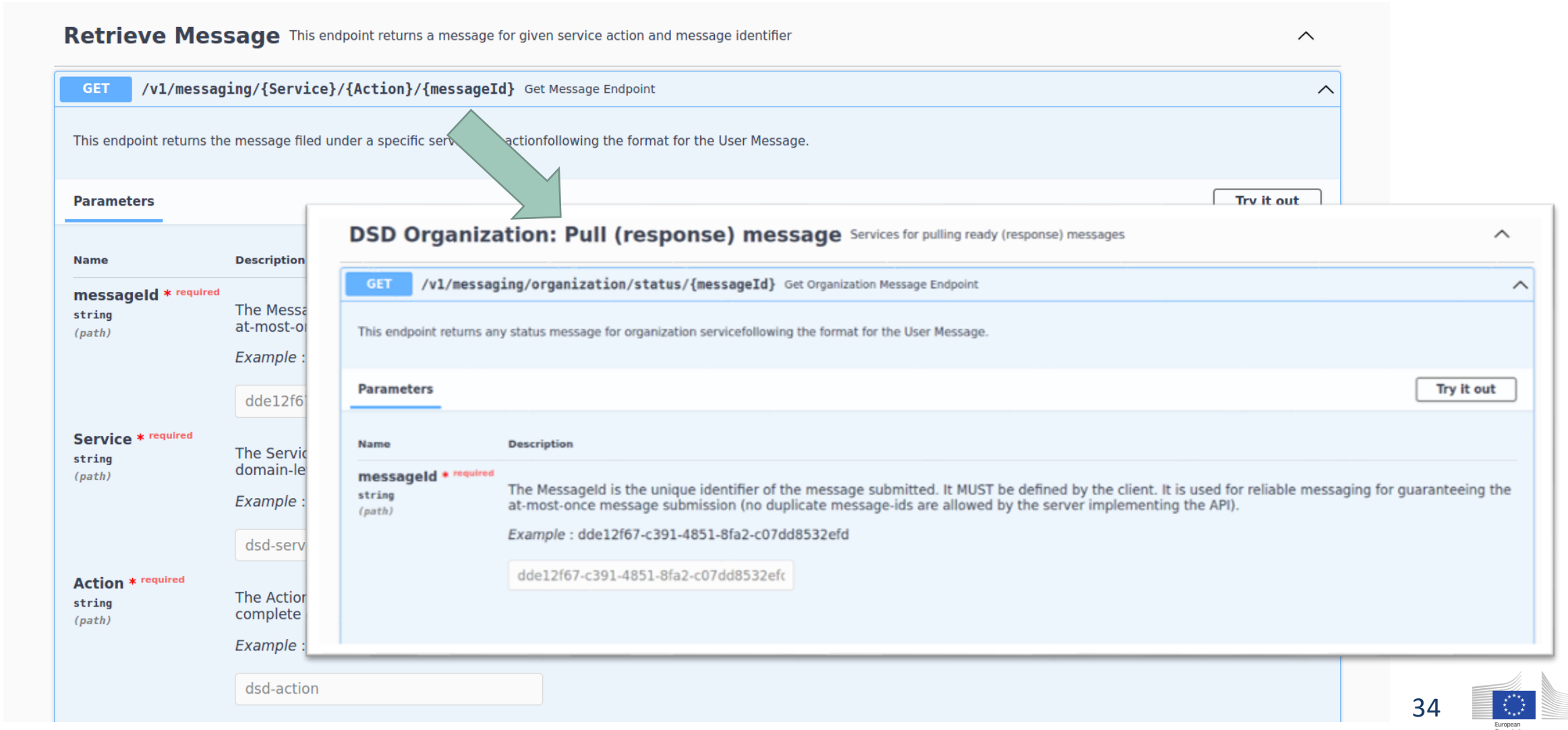
✓ Demo

Messaging API

✓ Demo

REST API Pilot

Generic Java framework for implementing Messaging API



REST API Pilot

Service, Action and Payload definitions can be added via annotations

Messaging operation type

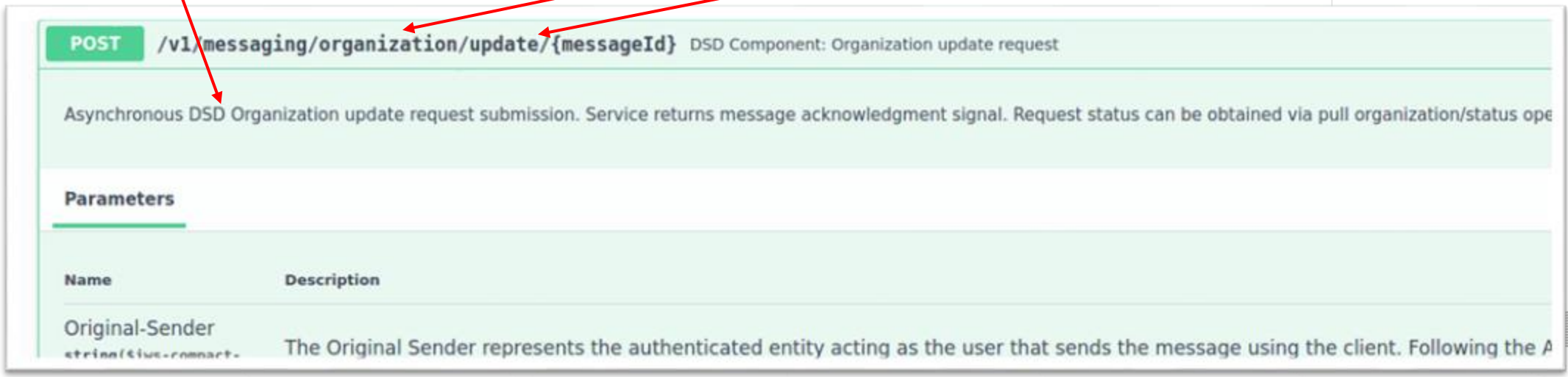
Operation description

Payload definition

Service

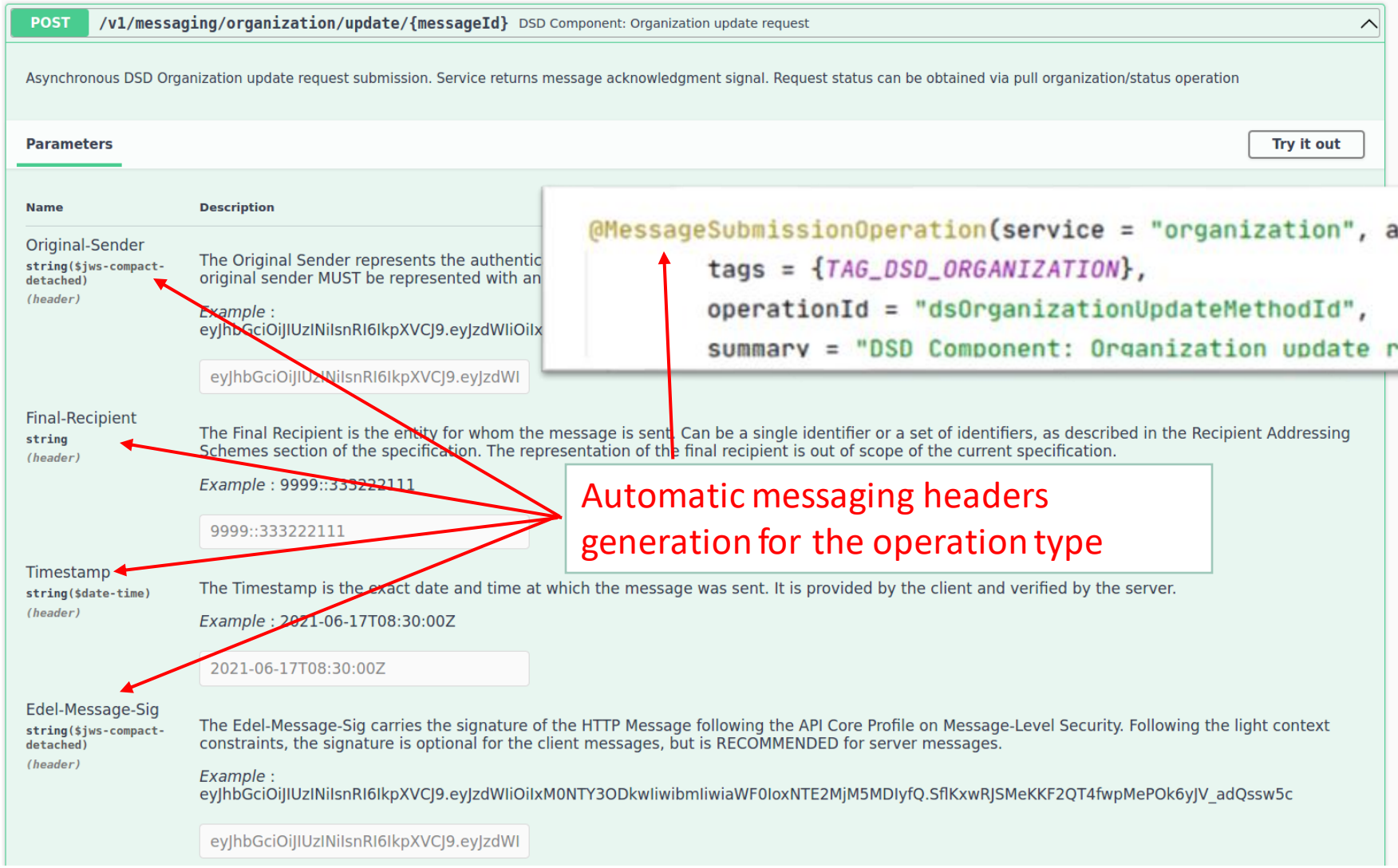
Action

```
@MessageSubmissionOperation(service = "organization", action = "update", sync = false,
        tags = {TAG_DSD_ORGANIZATION},
        operationId = "dsOrganizationUpdateMethodId",
        summary = "DSD Component: Organization update request ",
        description = "Asynchronous DSD Organization update request submission. Service returns message " +
                "acknowledgment signal. Request status can be obtained via pull organization/status operation",
                    requestTitle = "UpdateOrganizationRequest",
        requestDescription = "Massage request contains Organization objects with updated organization data!",
        requestPayloads = {
                @MultipartPayload(
                        name = "update-organization",
                        contentType = MediaType.APPLICATION_JSON_VALUE,
                        instance = OrganizationRO.class,
                        example = ORGANIZATION_UPDATE
                )
        })
void updateOrganization( String messageId, OrganizationRO organization);
```



REST API Pilot

Common HTTP headers (original sender, final recipient, timestamp, etc.)



REST API Pilot

Security: message signature and/or payload signature

```
# ---------------------------------- Keystore/Truststore ----------------------

#The location of the keystore
dsd.messaging.security.keystore.location=${dsd.config.location}/keystores/dsd_keystore.p12
#The type of the used keystore
dsd.messaging.security.keystore.type=pkcs12
#The password used to load the keystore
dsd.security.keystore.password=**************
#Private key
#The alias from the keystore of the message signing key
dsd.security.messaging.signature.key.alias=private_key_alias
#The private key password
dsd.security.messaging.signature.key.password=private_key_password

#Truststore
#The location of the truststore
dsd.security.messaging.truststore.location=${domibus.config.locatio
#Type of the used truststore
dsd.security.messaging.truststore.type=pkcs12
#The password used to load the trustStore
dsd.security.messaging.truststore.password=**************
```

DSD Organization: List of response messages to pull Services for returning list of ready resp

GET `/v1/messaging/organization/update/{messageId}/response` Get Response Message Reference List Endpoint

Responses

Server response

Code	Details
200 *Undocumented*	**Response body**

```
{
  "messageReferenceList": [],
  "count": 0,
  "limit": -1,
  "offset": 0
}
```

Response headers

```
connection: keep-alive
content-length: 62
content-type: application/json
date: Tue,22 Jun 2021 14:04:55 GMT
edel-message-sig: eyJhbGciOiJFUzIlNiJ9..4iiXu8qqjOCT8bm0yOkNcnzIXeI0oMM3FRbvHqe9jiikp2zpNnAucdyqCjLVdA0M36N9CA6bNFJlNCwVIp
```

REST API Pilot

Generates Open API document automatically conformant to the messaging API specification

```
openapi: "3.1.0"
info:
  title: "DSD Message service handler API"
  description: "This is a pilot project for  implementing isa2 messaging REST API"
  termsOfService: "https://www.eupl.eu/"
  contact:
  license:
    name: "EUPL 1.2"
    url: "https://www.eupl.eu/"
  version: "v1.0"
  summary: "DSD messaging rest service"
  x-edel-lifecycle:
    maturity: "development"
  x-edel-publisher:
    name: "isa2 messaging REST API pilot"
    url: "https://ec.europa.eu/cefdigital/wiki/display/EDELCOMMUNITY/API+Core+Profile?focusedCommentId=3851229
externalDocs: {…}
servers: […]
tags: […]
paths:
  /v1/messaging/organization/query/{messageId}/sync:
  /v1/messaging/{messageId}/response/signal:
  /v1/messaging/organization/update/{messageId}:
components:
  schemas:
  parameters:
  headers: {…}
webhooks: {…}
```

```
@MessageSubmissionOperation(service = "organization", action = "update", sync = false,
        tags = {TAG_DSD_ORGANIZATION},
        operationId = "dsOrganizationUpdateMethodId",
        summary = "DSD Component: Organization update request ".
```

Discussion

Pilot Domibus integration with CEF EBSI (blockchain)

- Practical demonstration

Bogdan Dumitriu, Joze Rihtarsic - CEF eDelivery Technical team, DIGIT D3

Scenarios

1. Timestamp the WS-Security signature of the AS4 message

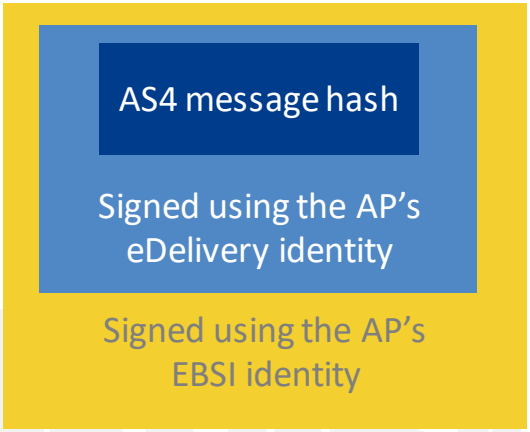
   Validate the timestamp on the receiving side

2. Use the EBSI Distributed Identity registry as an immutable whitelist for authorising participants in an eDelivery network (when using dynamic receiver)

Timestamp the WS-Security signature of the AS4 message

Register the signature of eDelivery AS4 message to the EBSI ledger, “notarising” the signature.

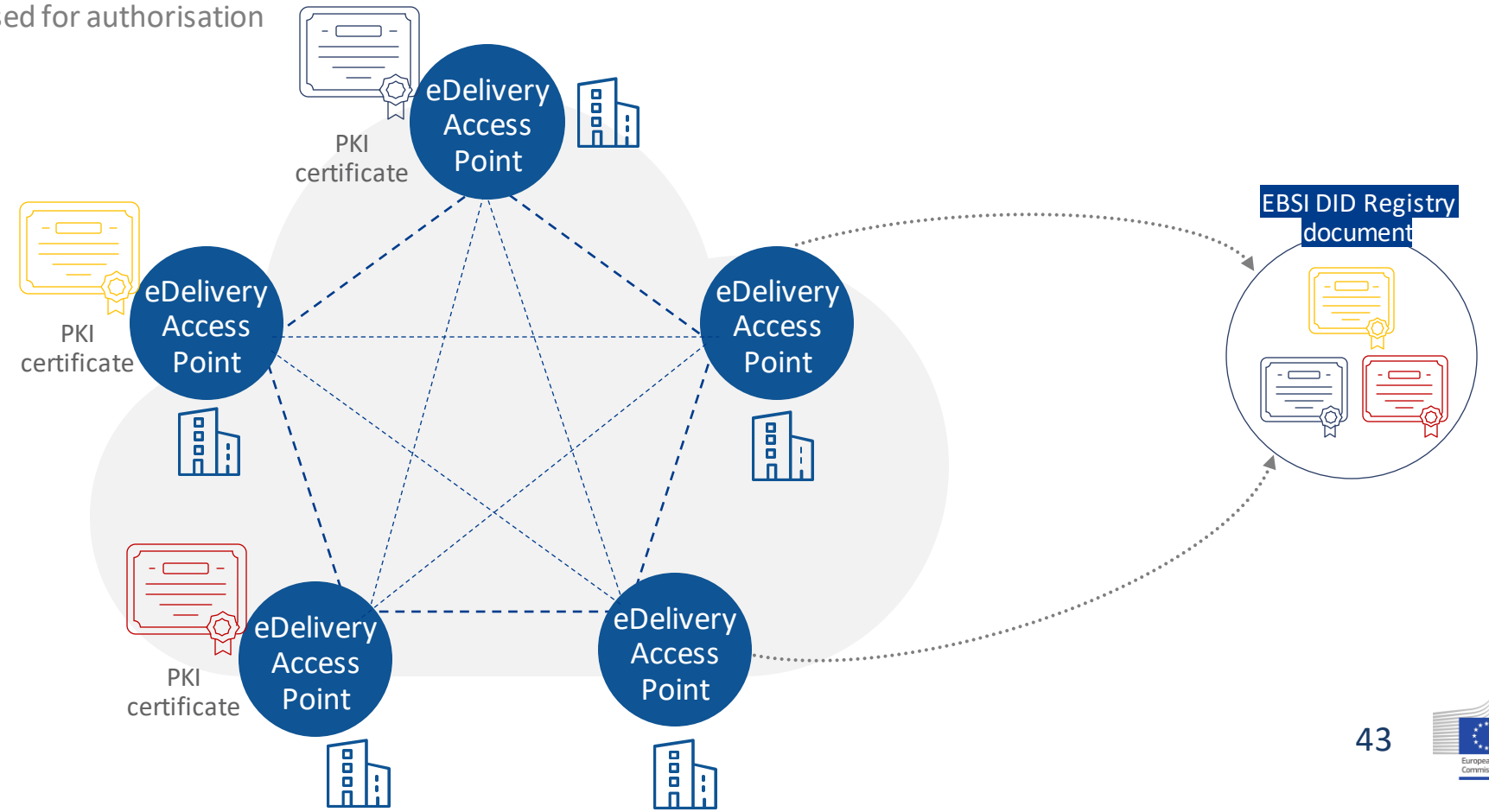
In order to write to the ledger, each AP will use its **identity (DID) in the EBSI world**.



EBSI DID Registry as an eDelivery authentication list

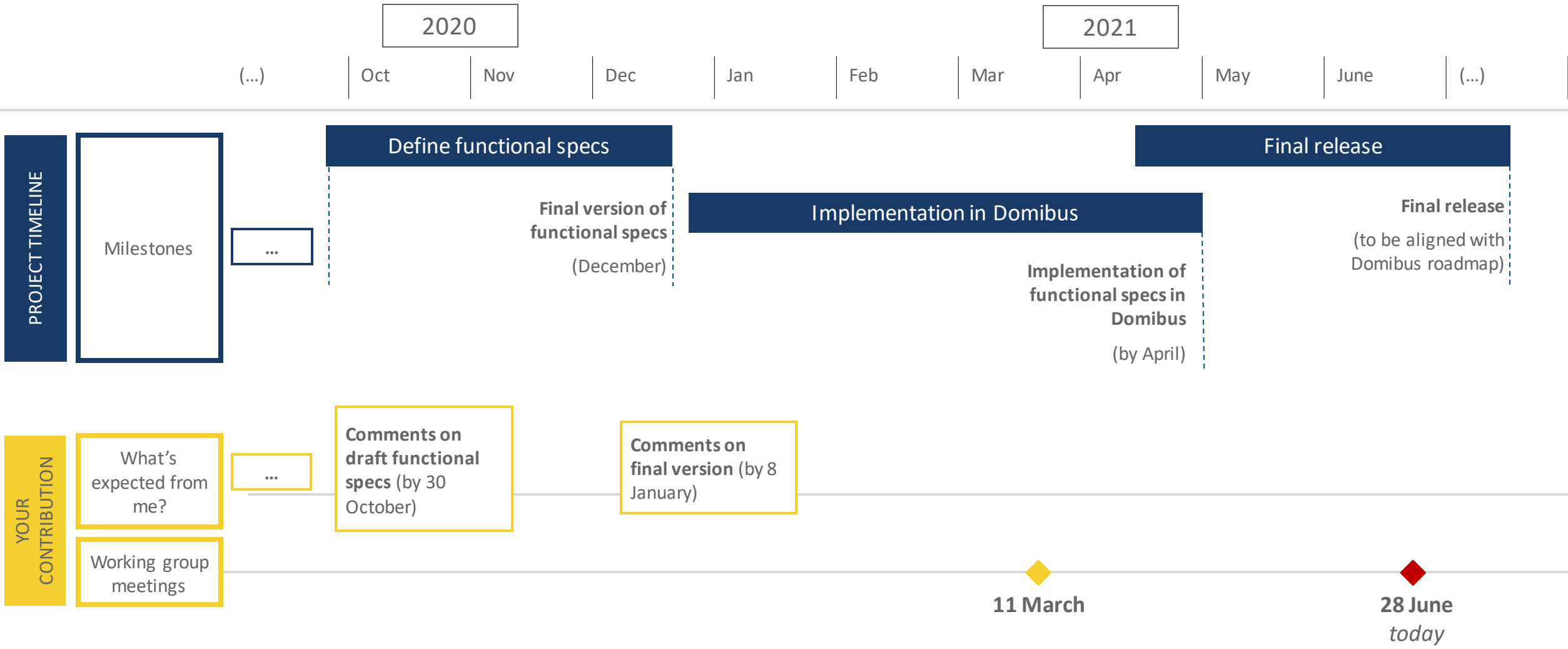
EBSI DID registry as a **certificate list** in a standard eDelivery scenario:

- The registry will store one DID document for **each domain**
- The document will contain the **list of the hashes** of the certificate + From/PartyId of authorised parties
  - Ensuring authentication only
- (Local) PMode configuration used for authorisation



Pilot Domibus integration with CEF EBSI (blockchain)

Indicative timeline

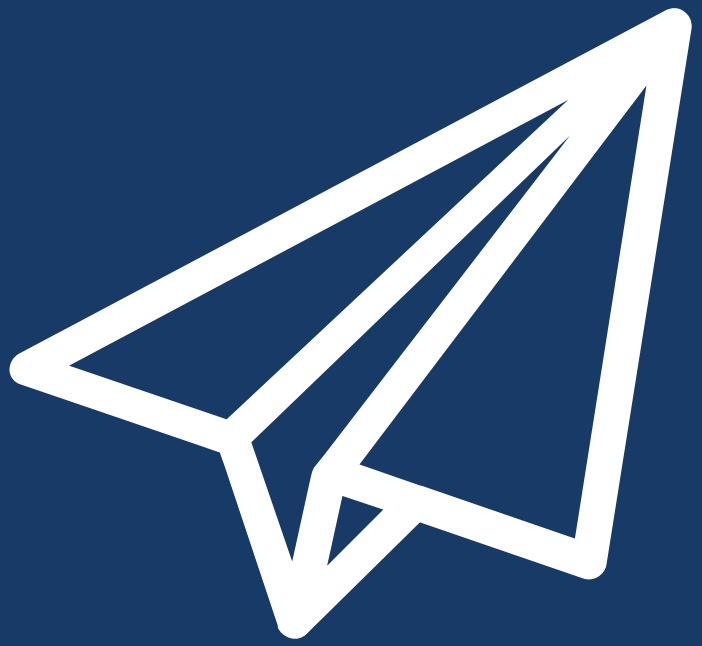


Update on JRC's work on API guidelines for government

- Security & Privacy essentials highlights
- Empirical analysis contractual conditions of APIs

Monica Posada - JRC B6, Lorenzino Vaccari (Consultant)

Thank you!



ec.europa.eu/cefdigital

CEF-BUILDING-BLOCKS@ec.europa.eu