DomiSMP

Contents

Other Releases

- **DomiSMP 5.1.1**
- **DomiSMP 5.1**

See All → DomiSMP Releases

I am looking to...

Install
→ Install DomiSMP
→ Download DomiSMP

Configure
→ SMP Configuration

Get Started
→ SML/SMP Services How-to Guides

Chapter 1. Documentation Contents

Here's a summary of the contents you can find here about DomiSMP.

▼ **DomiSMP Contents**

Chapter 2. About DomiSMP

An eDelivery Product

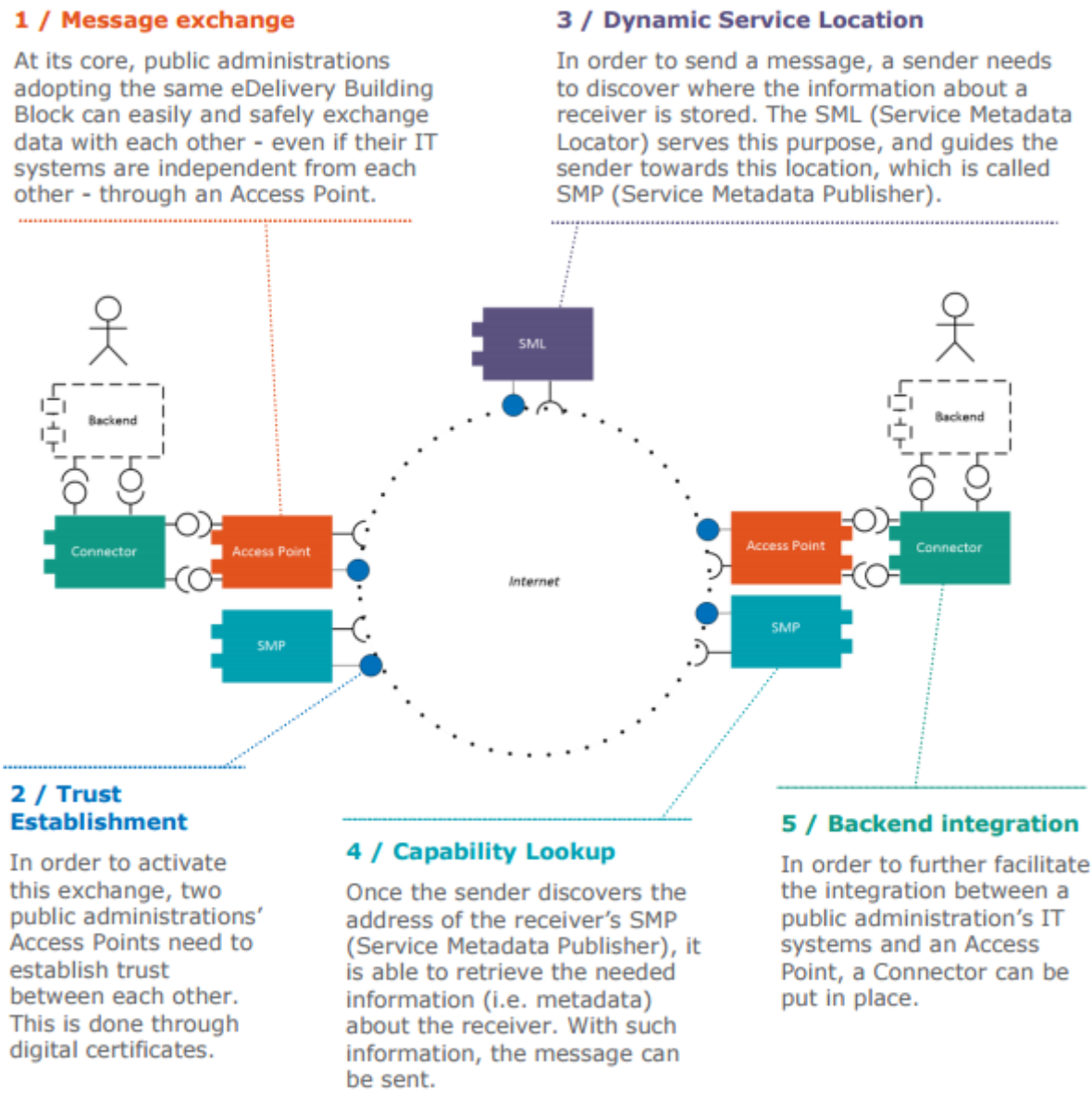
DomiSMP is one eDelivery's products. Besides DomiSMP we have Domibus and DomiSML.

eDelivery Positioning

eDelivery products helps public administrations exchange electronic data and documents with their counterparts, as well as businesses and citizens, in an interoperable, secure, reliable and trusted way.

By using this building block, every participant becomes a node in the network using standard transport protocols and security policies. eDelivery is based on a distributed model, allowing direct communication between participants without the need to set up bilateral channels.

NOTE See also European Comission's Building Blocks for a broader context of eDelivery.



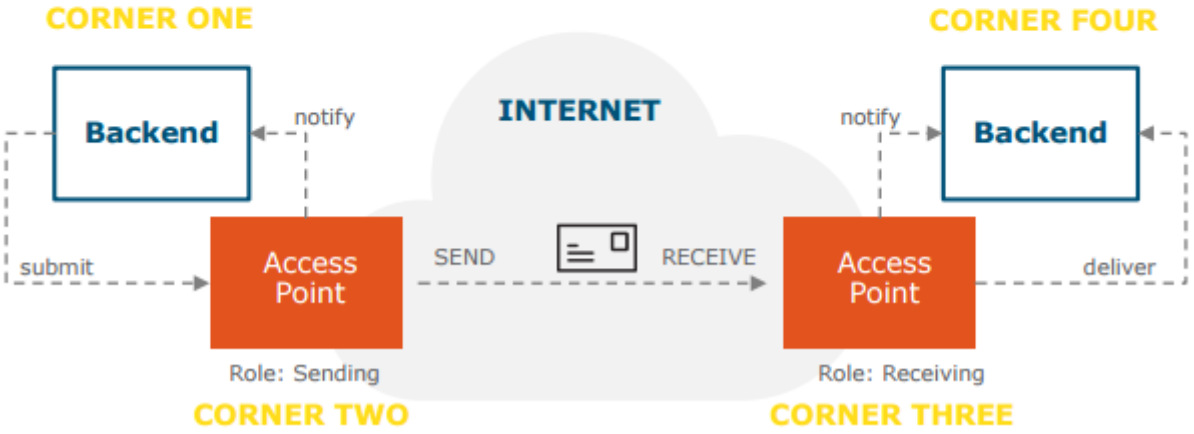
eDelivery in a nutshell

The technical architecture of eDelivery is based on a conceptual model called **'four-corner model'.** This means that Backend systems (corners one and four) do not exchange messages directly with each other but via Access Points (corners two and three) that, in any given exchange, play the sender or receiver role.

The Access Points of eDelivery are not operated centrally, instead they are deployed in the Member States under the responsibility of a public or private sector service provider.

The users of the Access Points are the Backend systems that need to exchange information with other administrations or businesses across borders.

During the exchange, the data and documents are secured by eDelivery's trust establishment mechanisms. This implies a choice of trust establishment model.



Chapter 3. Architecture

Service metadata publishing (SMP) was introduced to eDelivery network by PEPPOL's project, PEPPOL Transport Infrastructure Service Metadata Publishing. The purpose of the SMP is similar to an address book or business registry. eDelivery participants (message senders and receivers) use SMP to publish their transport/service capabilities and to discover partner's transport/service capabilities as: delivery addresses, supported business processes and document types, etc.

The PEPPOL's SMP specification was submitted as input to the (Business Document Exchange Technical Committee (OASIS BDXR TC) with the intent of defining a standardized and federated document transport infrastructure for business document exchange. It resulted into a new specification: OASIS Service Metadata Publishing Specification, OASIS SMP specification.

The eDelivery SMP profile, provides a set of implementation guidelines for the OASIS SMP specification.

- It is designed to be used in eDelivery with the dynamic receiver (and sender) discovery functionality.

The eDelivery Service Metadata Publisher application (DomiSMP) is the sample implementation of the eDelivery SMP profile (thus OASIS SMP spec as well).

▼ **Purpose**

This document is the Software Architecture Document of the DomiSMP application. It is intended to provide detailed information about the project:

- An overview of the solution
- A description of business and administration functions implemented in the DomiSMP
- A description of the application architecture and its modules
- An overview of code organization and code quality measurements
- An overview of technical requirements

▼ **References**

- **OASIS SMP Specification**, version 1.0. This specification defines documents and REST binding of SMP public interface.
- **eDelivery SMP Profile**, the eDelivery profile based on the OASIS SMP Specification.
- **eDelivery SMP Administration** guide
- **eDelivery SMP Interface Description** guide. Defines interface of eDelivery SMP – extends OASIS SMP specification.
- **eDelivery SML Administration** guide (pdf), in the Documentation section of SML Software.Provides comprehensive details on eDelivery SML installation, configuration and maintenance.
- **eDelivery BDMSL (SML)**. Application offered by eDelivery in SaaS model. Facilitates write

access to the DNS zone needed for dynamic discovery of Participants. Exposes SOAP interface that is consumed by SMP in order to (un)register participant DNS entries.

- **PEPPOL** The Pan-European Public Procurement On-Line (PEPPOL) project was a pilot project funded jointly by the European Commission and the PEPPOL Consortium members. After successful completion of the project new organization OpenPEPPOL Association was established. The organization is now responsible for the governance and maintenance of the PEPPOL specifications.
- **OASIS Service Metadata Publishing (SMP) specification**, version 2.0. This document describes the version 2.0 of the Oasis SMP standard.
- **OASIS ebCore Party Id Type TS**, version 1.0. This document describes the OASIS ebCore Party Id Type.

▼ **Definitions**

Service Metadata Publisher (SMP)

REST service application providing set of CRUD operations for two web resources: ServiceGroup and ServiceMetadata.

SMP is eDelivery's implementation of what's defined in OASIS SMP Specification and in SMP's Interface Document.

Identifier

The identifier uniquely identifies DomiSMP entities such as resources and subresources. An identifier consists of a schema (namespace) and a value. An identifier has rules about how it is represented in the URL (concatenated format) and how it is written in the resource document.
See Identifiers, in the Functional View chapter.

ParticipantIdentifier

The ParticipantIdentifier is an entity that uniquely identifies receiver or sender (participants) in eDelivery process. Examples of identifiers are company registration and VAT numbers, DUNS numbers, GLN numbers, email addresses etc.

Resource

The DomiSMP URL resource is associated with a specific Participant Identifier. The resource can be Service Group (see below) or any other document type supported by the DomiSMP extensions.

ServiceGroup

The ServiceGroup contains list of services associated with a specific Participant Identifier that is handled by a Service Metadata Publisher.
ServiceGroup XML representation is defined by XML Schema attached to OASIS SMP Specification.

Subresource

The DomiSMP URL (sub)resource is the sub-document of the resource. The resource can be ServiceMetadata (see below or any other document type supported by the DomiSMP extensions).

ServiceMetadata

The ServiceMetadata contains all necessary metadata (endpoint URLs, certificate for encryption, document types, etc.) about a specific service that a participant (service requester) needs to know in order to send a message to that service. ServiceMetadata XML representation is defined by an XML Schema included into OASIS SMP Specification.

SignedServiceMetadata

ServiceMetadata signed by Service Metadata Publisher (SMP).

DocumentIdentifier

represents document types in a service. It also contains scheme type which represents format of the identifier itself. XML representation is defined by an XML Schema included into OASIS SMP Specification as part of ServiceMetadata.

BDMSL (SML)

Application offered by eDelivery in SaaS model. Facilitates write access to the DNS zone needed for dynamic discovery of Participants. Exposes a WSDL interface that is consumed by SMP in order to (un)register participants' DNS entries.

Domain

The Domain indicates the purpose of the exchange network, such as the E-Invoice exchange, eHealth record exchange, etc.
If the domain network uses the delegated dynamic discovery service, the domain has its own DNS zone handled by the BDSML application. For eDelivery SML the domains are:

- acc.edelivery.tech.ec.europa.eu: acceptance domain for testing SMP instances and subdomains.
- delivery.tech.ec.europa.eu: production domain.

Group

The domain participant group. The Domain can have one or more groups where the Group admin is responsible for the particular group of participants for creating and deleting the domain resources. For example, the domain groups allow the Domain’s resources (e.g., service groups) to be segmented into different countries, regions, etc. and managed by the responsible group admin

Subdomain

Subdomain defines business domains handled by BDSML application in particular DNS zone. Examples of subdomain (business domain) are: peppol, ehealth, generalerds and they are all in part of domain (DNS zone) edelivery.tech.ec.europa.eu domain.

Dynamic Discovery

Dynamic Discovery is the process of discovering participants' service metadata.

3.1. Solution Overview

The eDelivery Service Metadata Publisher (DomiSMP) enables the participants of an eDelivery Messaging Infrastructure network to dynamically discover each other’s capabilities (Legal,

Organisational, and Technical). For this to happen, each participant must publish into an SMP its capabilities and settings (including but not limited to):

- business processes that the participant supports
- the security setup (public key certificate)
- the transport protocol (AS2 or AS4)
- the location of the receiver's access point

The SMP usually serves multiple participants to publish theirs exchange capabilities. But in eDelivery network/business domain can coexist in multiple SMPs. Because of this distributed architecture, each participant must have a unique ID in a particular subdomain.

Business Document Metadata Service Location (BDMSL), a central component, uses these IDs to create URLs that, when resolved, direct the eDelivery Access Points towards the specific SMP of the participant.

The SMP software component described in this document implements the OASIS Service Metadata Publishing eDelivery SMP profile based on the BDX SMP specifications.

3.2. Functional View

This section describes interactions, data flows and dependencies between SMP and other integrated applications in dynamic discovery process. All use cases refer to the SMP Interface Description guide, where they are presented with more interface-specific details.

The use cases UC06 GET ServiceGroup and UC07 GET ServiceMetadata are implementations of the service defined in OASIS SMP Specification. All other use cases cover administration/maintenance services which are not part of the specifications.

The use cases cover RESTful CRUD operations for following SMP business objects:

ServiceGroup, under relative URL:

```
/{ParticipantIdentifierScheme}::{ParticipantIdentifierValue}
```

ServiceMetadata, under relative URL:

```
/{ParticipantIdentifierScheme}::{ParticipantIdentifierValue}/services/{DocTypeIdentifi
erSche me}::{DocTypeIdentifierValue}
```

3.2.1. Identifiers

The identifier uniquely identifies DomiSMP entities such as resources (e.g., ServiceGroups) and subresources (e.g., ServiceMetadata). The identifiers are being used in the URL requests as part of the URL request path segment, and also in the (sub)resource documents.

Example of the URL request (scheme:oasis:names:tc:ebcore:partyid-type:iso6523:0088, value:4035811991021) concatenated with single colon, :.

```
http://my-app.example.eu/smp/urn:oasis:names:tc:ebcore:partyid-
type:iso6523:0088:4035811991021
```

Example of the document element with the participant identifier split to scheme attribute and element value:

```
<ParticipantIdentifier scheme=urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088>
    4035811991021
</ParticipantIdentifier>
```

▼ **Identifiers encoding**

According to the OASIS SMP Specification and OASIS SMP v2.0 specification above, SMP deals with two types of identifiers: participant and document identifier. The OASIS specification prescribes that both are built out of scheme and value, delimited by special character(s) such a double colon separator :: or the single colon separator : as defined in the OASIS ebCoreParty Id, see SMP Administration Guide and OASIS ebCore Party Id Type Technical, v1.

ServiceGroup identifier, from business perspective known as Participant Identifier

```
ServiceGroup identifier := {ParticipantIdentifierScheme}::{ParticipantIdentifier}
```

ServiceMetadata Identifier, from business perspective known as Document Type Identifier:

```
ServiceMetadata identifier := {DocTypeIdentifierScheme}::{DocTypeIdentifier}
```

All identifiers that are included in the URL of the REST request must be URL-encoded (note also the double colon separator ::).

Example: the participant identifier (ServiceGroup identifier) built out of:

- ParticipantIdentifierScheme = "participant#domain#scheme"
- ParticipantIdentifier = "participant#id"

must be encoded in the URL request as:

- participant%23domain%23scheme%3A%3Aparticipant%23id

Also, in some cases (all PUT requests), the identifiers are present in the URL and in the XML body of the request. In these cases, only identifiers in URL must be URL-encoded.

▼ **ebCore party identifier**

The eDelivery SMP has the feature to support handling participant identifiers as described in Use

with eDelivery ebCore Party Identifiers, in the eDelivery SMP profile.

In this case, the participant starts with the: `urn:oasis:names:tc:ebcore:partyid-type:` following by the words: `unregistered` or `iso6523`

All ebCore party identifiers in the REST request must be URL-encoded using only one double colon separator `:`, as in the example below:

- `urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088:4035811991021`

URL-encoded example:

- `urn%23oasis%23names%23tc%23ebcore%23partyid-type%23iso6523%230088%234035811991021`

The eDelivery SMP has the option to serialize ebCore party Id to XML according to the OASIS SMP Specification as separate values, as in the example below:

```
<ParticipantIdentifier
    scheme="urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088">
    4035811991021
</ParticipantIdentifier>
```

or according to the eDelivery SMP profile as concatenated value:

```
<ParticipantIdentifier>
 urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088:4035811991021
</ParticipantIdentifier>
```

See Configuration for more info on the configuration of this behaviour.

▼ **Identifier's case sensitivity**

SMP can handle identifiers (scheme and value) in case-sensitive or in a case-insensitive way.
See Configuration for more info on the configuration of this behaviour.

When the SMP is configured as case-insensitive the SMP normalizes the identifiers extracted from the requests. Identifiers within incoming requests are considered as case-insensitive and converted to lowercase. Further processing like the storage and querying in the database is performed using lowercase letters only. If the case sensitivity configuration is modified, the database records must be updated manually.

When the SMP is configured as case-sensitive, then Identifiers are not modified during the whole request processing.

3.2.2. BDMSL Integration

Creation or removal of ServiceGroup within SMP triggers a synchronous (un)registration of relevant record(s) in DNS. This process is required to allow Dynamic Discovery of SMPs to store Participant's metadata.

Write access to DNS zone is facilitated by BDMSL (SML), a centralized application that exposes a SOAP interface for that purpose (see eDelivery BDMSL (SML). SMP is a consumer of the SML services. SML authorization of SMP is based on mutual HTTPS authentication. Therefore, SMP client TLS certificate with private key needs to be configured on SMP side.

If SMP serves data in only one domain, then a single certificate is needed. Otherwise, if the SMP is configured to work in multi-domain mode, the System Administrator will need to set up one certificate per subdomain.
See also Domain Multitenancy and Configuration, for more details.

3.2.3. Domain Multitenancy

An SML subdomain can be considered as a set of an inter-network of eDelivery components: SML, SMPs and Access Points for a business domain. All these members communicate with each other within that subdomain and exchange messages according to the strict rules defined for that business domain. One network can be used to exchange invoices between participants, another one could exchange health information between hospitals and insurance companies, etc.

In most scenarios there will be multiple SMPs in a single business domain and each of them will handle ServiceMetadata sets of multiple participants from the same subdomain. The business domain authority can set its own SMP to administrate its participants and the SMP is used only in one domain. But an SMP could be used in more than one business domain at the same time. Because of SML restrictions such setup implies the following SMP functionality:

- The SMP must use a different SMP ID and a different certificate to authenticate for a particular SML subdomain.
- The SMP must be able to sign ServiceMetadata responses using a different certificate for each domain (one certificate per domain).

3.2.4. Roles

Roles are documented with more details in the SMP Interface Description guide. The table below explains their meaning from a functional perspective:

Role Alias	Description
Anonymous	Any user that has not provided any authentication details. This user can query for public resources e.g.: `ServiceGroup` and sub resources, for example, `ServiceMetadata`.

Role Alias	Description
User	User with the role can log in to the DomiSMP and has access and edit rights to resources according to memberships on resources, groups and domain. For example, user who is a member of the resource with Admin membership can perform administrative actions update service group extension data and add/update/delete service metadata for the service group. User who is member of the Group with Group Admin membership role is allowed to execute create and resource for the group. User who is member of the Domain with Domain Admin membership role, can create/delete groups for Domain and manage the domain memberships.
System Admin	System user who can administer domains, users, application properties, truststore and keystore on the DomiSMP.

3.2.5. Domain, Group and Resources

The DomiSMP supports 3-layer security realms.

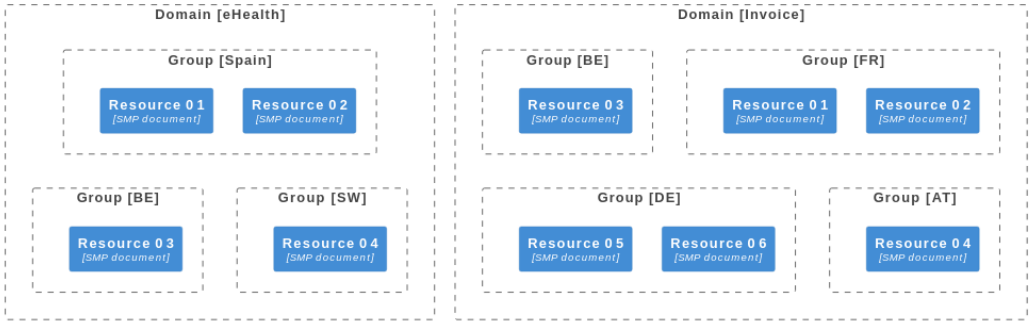
- The most basic unit is the **Resource**. The Resource is identified by the unique ID, which is part of the URL of the resource as example:

  http://localhost/smp/resource-identifier

An example of the Resource is the “Service Group” document from the Oasis SMP specification.

The user can be a Resource member with **Admin** or **Viewer** membership roles. If the user has an Admin membership role, it can modify resource document(s) and manage the resource memberships. If the user has role Viewer, it can view/read the Resource if the Resource has visibility set to: “Private”.

- The **Group** is a cluster of resources managed by the dedicated group administrators. The group admin(s) can create and delete the resource, but **only** the resource admins can modify data/documents for the resource. The user can be a Group member with **Admin** or **Viewer** membership roles. With Admin group membership, the user can create and delete group resources. If the user has group role Viewer, it can view/read the Resources if the Group has visibility set to: “Private”.
- The top layer is the **Domain**. It indicates the business purpose of the network of participants, such as invoice exchange, Health Records message exchanges, etc. The Domain usually has a domain owner who handles participant interoperability, defining message types, network authentication, and authorization methods such as Certificate PKI, Identity Service providers, etc. In DomiSMP 5.0, the user with a Domain Admin role can create domain groups and assign users to them.



3.2.6. Extensions

One of the main DomiSMP sample implementation purpose is enabling the setup (and testing) of various network configurations. Designed with flexibility in mind, DomiSMP allows the implementation of custom logic of the document processing. To achieve this, developers can create custom extensions. These extensions are packaged as JAR files and extend one or more interface classes from the DomiSML module `eu.europa.ec.edelivery:smp-spi`.

Example of how to include an extension in your custom extension project:

```
<dependency>
 <groupId>eu.europa.ec.edelivery</groupId>
 <artifactId>smp-spi</artifactId>
 <version>${project.version}</version>
</dependency>
```

The extension JAR must be added to the DomiSMP extension library path before starting up the application. The path where the extensions must be deployed is defined with file property: `smp.libraries.folder`.

An example:

Extension folder
path where SMP extensions are located. The Folder is loaded by the SMP classloader at startup.

```
smp.libraries.folder=/cef/test/smp/apache-tomcat-8.5.73/smp/ext-lib
```

DomiSMP 5.0.x supports two types of extensions:

Resource Handling Extension

DomiSMP supports various document types via custom designed extension. This extension handles/processes the resource and sub resource documents.

Payload Validation Extension

with this extension, users can validate payloads/documents according to specific rules. Users can develop custom security scanning of all payload uploaded to the DomiSMP.

In the next section we describe both extensions in more detail.

When the DomiSMP library is loaded by the class loader, the extension registrar searches for Spring

beans that implement the “ExtensionInfo” interface. These beans provide essential information about the extensions.

Key parameters

Identifier

The unique identifier represents the extension. If an extension is upgraded, its identifier must remain the same to ensure proper handling of existing extension data.
Example identifier: edelivery-oasis-smp-extension.

Name

The human-readable name of the extension. This name helps users understand the purpose and functionality of the extension.

- **Description**: A brief description of the extension, providing purpose and extension details.
- **Version**: The version number of the extension, indicating its release or revision.
- **ResourceTypes**: List of resource handling extensions.
- **PayloadValidators**: List of Payload validator extensions

```
import eu.europa.ec.smp.spi.resource.ResourceDefinitionSpi;
import java.util.List;
/**
* DomiSMP extension information. When updating the extension it
must have the same Name for DomiSMP to handle the upgrade correctly. */
public interface ExtensionInfo {
  String identifier();
  String name();
  String description();
  String version();
  List < ResourceDefinitionSpi > resourceTypes();
  List < PayloadValidatorSpi > payloadValidators();
}
```

Resource Handling Extension

One of the most important purposes of DomiSMP is to allow users to publish various connectivity capability documents for serving or business message exchange. Examples of these documents include OASIS SMP 1.0 and OASIS SMP 2.0 documents, as well as ServiceGroup and ServiceMetadata documents, CPP from Oasis CPPA3, and any other text-based custom documents.

The resource extension enables the following tasks:

- Automatic Document Generation: the extension automatically generates sample documents for an extension. For example, Oasis SMP 1.0 extension can generate sample document for ServiceGroup and ServiceMetadata which are used by the DomiSMP User interface when creating new resources.
- Document Validation During Registration: when a new document is registered, the extension validates its structure, metadata, and content. This ensures that only valid documents are

accepted and published on the DomiSMP.

- Document Management: the extension can validate, modify, and update documents on read, store, and validate action.

When creating a resource handling extension, developers must implement the ResourceDefinitionSpi and optionally the SubresourceDefinitionSpi. These interfaces contain the resource definition such as identifier, version name, document mimetype, etc.

The `ResourceDefinitionSpi` and `SubresourceDefinitionSpi` contain the resource handling extension metadata for the Resources/Subresources. To process the document, the resource and subresource implementation must also contain the implementation of the `ResourceHandlerSpi` which handles the document processing such as: generation of empty document, validation of the document, and methods which are invoked while reading and storing the document.

Resource Handler Interface definition

```
package eu.europa.ec.smp.spi.resource;

import eu.europa.ec.smp.spi.api.model.RequestData;
import eu.europa.ec.smp.spi.api.model.ResponseData;
import eu.europa.ec.smp.spi.exceptions.ResourceException;
import java.util.List;

/**
* The class implementing the ResourceHandlerSpi must support read
* transformation, store transformation, and
* validation methods for the particular resource type, such as
* Oasis SMP 1.0 document, CPP document, etc.
*
*/
public interface ResourceHandlerSpi {

    /**
    * Method get data from the resource in the input stream, and it
    * writes transformation of the data as they are returned to
    * @param resourceData the resource data
    * @param responseData the date object for setting the response */
    void readResource(RequestData resourceData, ResponseData responseData) throws
ResourceException;
    void storeResource(RequestData resourceData, ResponseData responseData) throws
ResourceException;

    /**
    * Validate resource schema and data. if resource is invalid the error is thrown
    * @param resourceData the resource data */
    void validateResource(RequestData resourceData) throws ResourceException;

    /**
    * Validate resource schema and data. if resource is invalid the error is thrown
    * @param resourceData the resource data */
```

```
    void generateResource(RequestData resourceData, ResponseData responseData, List <
String > fields) throws ResourceException;
}
```

SEE ALSO

For more details, see in the DomiSMP code repository:

- the maven module `smp-resource-extensions` containing sample implementations of the Oasis SMP 1.0 and Oasis SMP 2.0 documents and basic Oasis CPPA3-CPP document.
- the maven module `smp-examples/resource-spi-example` containing additional examples (JSON and property files) of DomiSMP Resource extensions.

Payload Validation Extension

To increase security, the eDelivery SMP offers the possibility of registering custom extensions for security scanning/validations of all binary documents such as the certificates and the keystores. The certificates can be uploaded by the users when setting the user certificate for authentication. The keystores binaries can be uploaded by the System Administrators when managing the SMP keystore.

When the user loads one of the mentioned payloads, the eDelivery SMP validation framework is activated. At this point, the payload binary data is passed to all registered spring beans, which implement the `PayloadValidatorSpi` interface below.

`PayloadValidatorSpi` *interface*

```
package eu.europa.ec.smp.spi;

import eu.europa.ec.smp.spi.exceptions.PayloadValidatorSpiException;
import java.io.InputStream;

/**
 * SMP Service provider interface (SPI) for uploaded payload validation.
 * This SPI interface allows antivirus validation using third-party antivirus
software.
 */
public interface PayloadValidatorSpi {

    /**
     * Validates the SMP payload. If the payload is invalid the method MUST
     * throw PayloadValidatorSpiException
     *
     * @param payload The payload data to be validated
     * @param mimeType The payload mime type
     * @throws PayloadValidatorSpiException in case the validation does not pass
     */
    void validatePayload(InputStream payload, String mimeType) throws
    PayloadValidatorSpiException;
```

```
}
```

The implementers of the extension must implement the method `validatePayload` for payload validation. In the event of malware detection, the method MUST throw the `PayloadValidatorSpiException` to terminate the future payload handling by the eDelivery SMP.

A simple example of the `PayloadValidatorSpi` implementation can be found in the SMP project module `smp-examples/smp-spi-example/`. See Souce Code Overview.

To register the extension in the eDelivery SMP, the interface implementation class must be

- located under the java package `eu.europa.ec.smp.spi`,
- tagged with spring bean annotation `@Component` or `@Service`,

as in the example below:

`PayloadValidatorSpi` *Implementation Example*

```
package eu.europa.ec.smp.spi.example;

import eu.europa.ec.smp.spi.exceptions.PayloadValidatorSpiException;
import org.springframework.stereotype.Service;
import java.io.InputStream;

@Service
public class ExamplePayloadValidatorSpiImpl implements PayloadValidatorSpi {
    public void validatePayload(InputStream payload, String mimeType) throws
    PayloadValidatorSpiException {
        // ...
    }
}
```

To prepare the extension for the deployment in the eDelivery SMP, the code must be compiled and stored in the java archive file format known as the JAR.

In the eDelivery SMP, the property `libraries.folder` must be configured in the `smp.config.properties` to point to the folder where extension libraries are located.
The SMP classloader loads the libraries in the folder at the startup of the SMP and registers the `PayloadValidatorSpi` beans.

3.3. Use Case Details

Here you can find a description of each Use Case.

3.3.1. UC01 Manage Administrators

▼ *UC01 Manage Administrators details*

Prerequisites

- User (system admin) has rights to modify content of SMP configuration tables.

Description

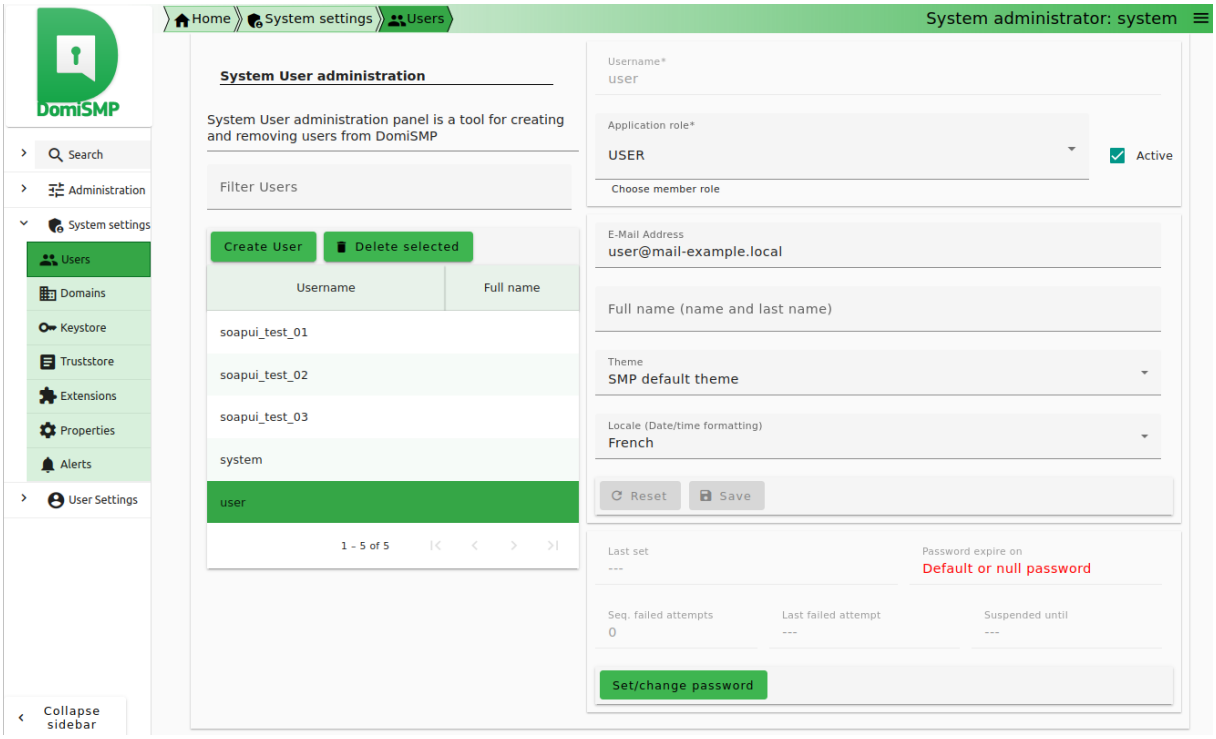
This use case does not involve SMP application, instead the user's management is implemented as a simple manual SQL queries. Users and its roles are not cached by the SMP, so they can be used immediately after the corresponding SQL transaction is committed. Sample SQLs inserting users authenticated by password or certificate are presented below.

For more details on users, see also Data Layer and Security.

SQL Sample for User Creation

```
-- user authenticated with password (Oracle dialect)
INSERT INTO
    smp_user ( id, username, active, application_role, email, created_on,
last_updated_on )
VALUES
    (
        smp_user_seq.nextval, 'smp_admin', 1, 'SYSTEM_ADMIN', 'system@mail-
example.local', sysdate, sysdate
    )
;
INSERT INTO
    smp_credential ( fk_user_id, credential_active, credential_name,
credential_value, credential_type, credential_target, created_on, last_updated_on )
VALUES
    (
(
        SELECT
            id
        FROM
            smp_user
        WHERE
            username = 'smp_admin' ),
            1,
            'smp_admin',
            '$2a$10$olcGeWKGEoRia2DPuFqRNeca0IEdRSmOrljLz57BAjf1jlC9SohrS',
            'USERNAME_PASSWORD',
            'UI',
            sysdate,
            sysdate
    )
;
```

If the system administrator user is already configured, the system administrator can use the eDelivery SMP UI tool to further manage users.



NOTE

For invoking the PUT or DELETE Use cases described in the sections below, credentials such as Access token or Client certificate must be used for the authentication.
See Security (DomiSMP Architecture).

3.3.2. UC02 PUT ServiceGroup

▼ *UC02 PUT ServiceGroup details*

Prerequisites

PUT ServiceGroup (Create or Update):

- The authenticated user has the role of `Admin SMP`.
- If the ServiceGroup is managed remotely, the `Resource Admin` must have been created before in the `Administrator` table.
- If the SMP is serving multiple domains, the header field "Domain" must be populated and refer to one of the domains served by the SMP.

Description

`PUT ServiceGroup` is an idempotent create/update REST action.

NOTE

Idempotence is the property of certain operations in whereby they can be applied multiple times without changing the result beyond the initial application.

If the SMP is configured to be integrated with BDMSL, then additional synchronous request is performed to register the newly created Participant in the DNS. A sample request is presented below, with the following conventions:

Sample PUT ServiceGroup Request

Request's Headers

```
PUT http://smp.eu/participant-domain-scheme%3A%3Aparticipant-id ①
HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: text/xml;charset=UTF-8
Authorization: Basic c21wX2FkbWluOmNoYW5nZWl0
ServiceGroup-Owner: anotherownerusername ②
Domain: domain2 ②
Content-Length: 284
```

Where:

① The identifiers: `participant-domain-scheme` and `participant-id` must match the equivalent identifiers used in the request's body (see Request's Body below).

② Optional headers.

Request's Body

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<ServiceGroup xmlns="http://docs.oasis-open.org/bdxr/ns/SMP/2016/05">
    <ParticipantIdentifier scheme="participant-domain-scheme"> ①
        participant-id ①
    </ParticipantIdentifier>
    <ServiceMetadataReferenceCollection/>
</ServiceGroup>
```

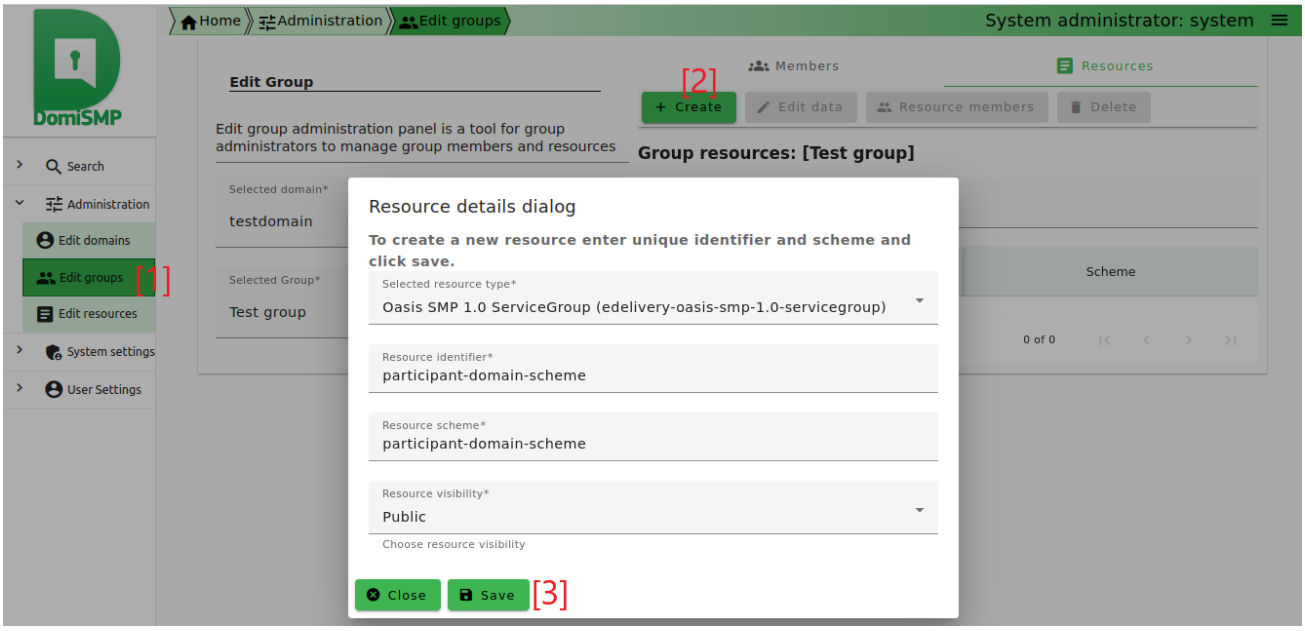
Where:

① The identifiers: `participant-domain-scheme` and `participant-id` in the body must match identifiers used in the request's header (see Request's Headers above).

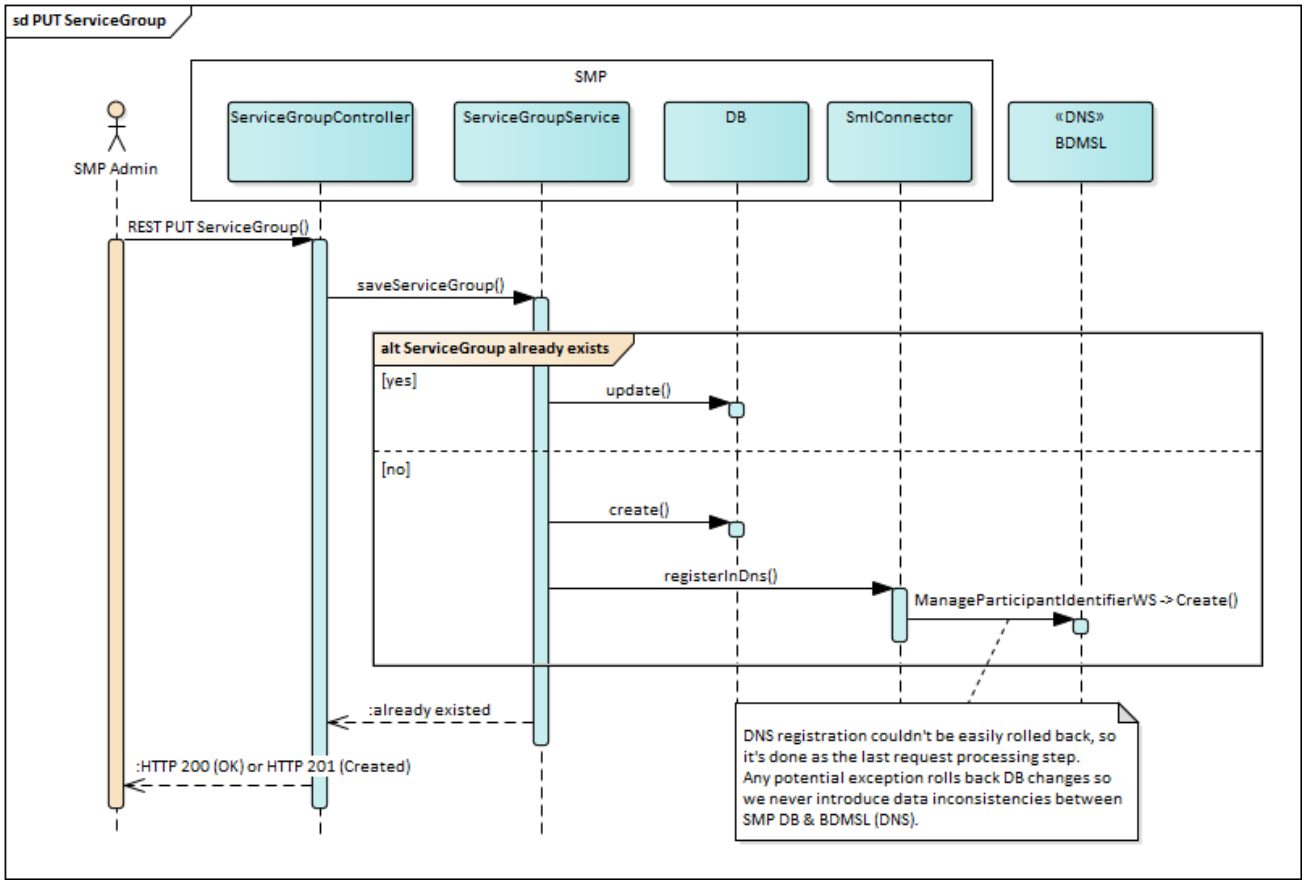
Successful responses

- `HTTP 200 (OK)` – ServiceGroup was updated
- `HTTP 201 (Created)` – New ServiceGroup was created

The DomiSMP group administrator can also register a ServiceGroup with the DomiSMP UI tool for Service group management (see [1] on the picture below). The ServiceGroup is registered by activating/clicking the save button (see [3] on the picture below) after all the necessary data are entered.



If BDMSL integration is enabled and configured for the selected domain, the SML request is submitted when the ServiceGroup is created.



ServiceGroup-Owner HTTP header

Specifying Owners

Only the DomiSMP Group administrator has permission to register (or delete) the ServiceGroup. The Group administrator usually creates a ServiceGroup for the end-user with the `Resource Admin` role, which has only the permission to update the ServiceGroup service metadata.

By default, the Admin of the ServiceGroup is the user who created the ServiceGroup. But this can

be changed at creation time by setting the `ServiceGroup-Owner` HTTP header with a different owner's identifier. The identifier of the service owner can be the username, the users access token identifier, or the certificate identifier.

Below are examples of HTTP header `ServiceGroup-Owner`:

```
ServiceGroup-Owner: anotherownerusername
```

Non-ASCII characters must be URL-encoded. For example, a username such as **Żółty Jérôme** should be encoded as:

```
ServiceGroup-Owner: %C5%BB%C3%B3%C5%82ty%20J%C3%A9r%C3%B4me
```

Users authenticated by certificate can become owners as well, for example, a user `CN=new owner,O=EC,C=BE:000000000000100f` should be encoded as:

```
ServiceGroup-Owner: CN%3Dnew%20owner,O%3DEC,C%3DBE%3A000000000000100f
```

Domain HTTP header

Specifying a Domain

This feature is used only when the SMP is set up in multi-tenant mode. When creating new ServiceGroup the Domain HTTP header must be specified in the PUT ServiceGroup request

```
Domain: domain2
```

See also Domain Multitenancy.

3.3.3. UC03 DELETE ServiceGroup

▼ *UC03 DELETE ServiceGroup details*

Prerequisites

- The authenticated user has the role of `Admin SMP`.
- If the ServiceGroup is managed remotely, the `Resource Admin` must have been created before in the `Administrator` table.
- If the SMP is serving multiple domains, the header field "Domain" must be populated and refer to one of the domains served by the SMP.

Description

This action removes the specified ServiceGroup from SMP's database **including all related ServiceMetadata**.

If the SMP is configured to integrate the BDMSL, then an additional synchronous request is issued in order to unregister the Participant from the DNS.

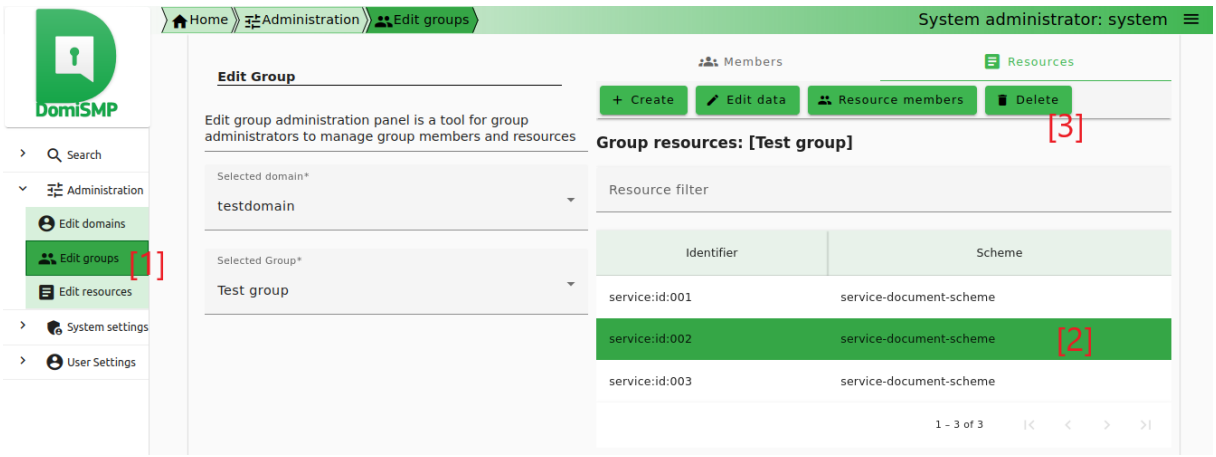
Request's Header

```
DELETE http://smp.eu/participant-domain-scheme%3A%3Aparticipant-id
HTTP/1.1
Accept-Encoding: gzip,deflate
Authorization: Basic c21wX2FkbWluOmNoYW5nZWl0
Content-Length: 0
```

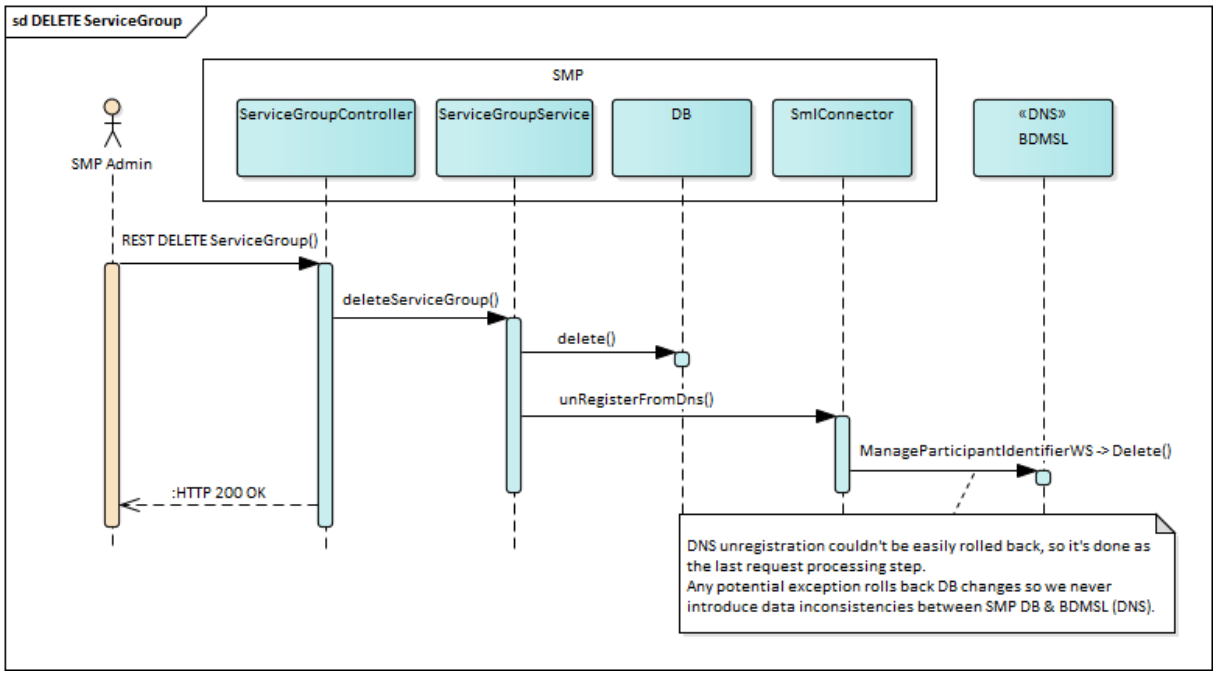
Successful responses

- `HTTP 200 (OK)` – ServiceGroup was removed.

The Group Admin can delete a ServiceGroup with the DomiSMP UI tool for group administration [1] management. The ServiceGroup can be deleted by selecting the ServiceGroup row [2], clicking the Delete button (see [3] in the figure below).



If BDMSL integration is enabled and configured for the selected domain, the SML delete request is submitted when the ServiceGroup is deleted.



3.3.4. UC04 PUT ServiceMetadata

▼ *UC04 PUT ServiceMetadata details*

Create or Update `ServiceMetadata`.

Prerequisites

- The authenticated user has the role `Resource Admin` (or `Admin SMP`).
- The `Resource Admin` user initiating the request is linked to the specified `ServiceGroup`
- `Resource Admin's certificate is valid.
- The certificate information of the `Resource Admin` was previously stored in the configuration.

Description

`PUT ServiceMetadata` is an idempotent create/update REST action.
A sample request is presented below.

> **NOTE** Identifiers present in the body of the request and in the URL marked in yellow must match.

`ServiceMetadata` is processed and stored as the whole unaltered XML document represented as string (including original whitespaces and comments between nodes). ServiceMetadata can be signed by ServiceGroup owner and e-signature can be placed in <Extension> node. To preserve integrity of signed metadata, SMP does not perform any transformation, canonicalization, or decomposing XML document into separate database records. While querying for the metadata (UC07 GET ServiceMetadata) original XML document is returned.

Request's Header

```
PUT http://smp.eu/participant-domain-scheme#%3A%3Aparticipant-id/services/doc-type-
scheme#%3A%3Adoc-type-id ①

HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: text/xml;charset=UTF-8
Authorization: Basic c21wX2FkbWluOmNoYW5nZWl0
Content-Length: 2152
```

Where:

① The identifiers: `participant-domain-scheme`,`participant-id`, `doc-type-scheme` and `doc-type-id` must match the equivalent identifiers used in the request's body (see Request's Body below).

Request's Body

```
<ServiceMetadata
    xmlns="http://docs.oasis-open.org/bdxr/ns/SMP/2016/05">
    <ServiceInformation>
        <ParticipantIdentifier scheme="participant-domain-scheme"> ①
            participant-id ①
```

```
        </ParticipantIdentifier>
        <DocumentIdentifier scheme="doc-type-scheme"> ①
            doc-type-id ①
        </DocumentIdentifier>
        <ProcessList>
            <Process>
                <ProcessIdentifier scheme="process-scheme">process-
id</ProcessIdentifier>
                <ServiceEndpointList>
                    <Endpoint transportProfile="busdox-transport-start">
                        <EndpointURI>https://poland.pl/theService</EndpointURI>
                        <RequireBusinessLevelSignature>true
</RequireBusinessLevelSignature>
                        <ServiceActivationDate>2003-01-
01T00:00:00</ServiceActivationDate>
                        <ServiceExpirationDate>2020-05-
01T00:00:00</ServiceExpirationDate>
                        <Certificate>SAMPLEBASE64ENCODEDCERT</Certificate>
                        <ServiceDescription>
                            Sample description of invoicing service
                        </ServiceDescription>
                        <TechnicalContactUrl>https://example.com
</TechnicalContactUrl>
                    </Endpoint>
                </ServiceEndpointList>
            </Process>
        </ProcessList>
    </ServiceInformation>
</ServiceMetadata>
```

Where:

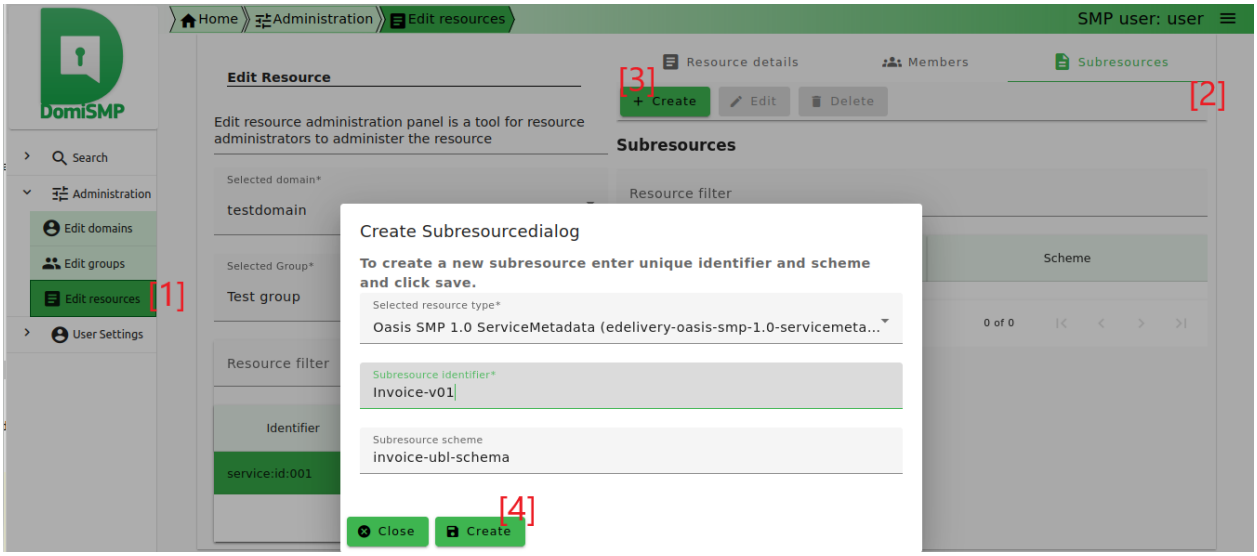
① The identifiers: `participant-domain-scheme`,`participant-id`, `doc-type-scheme` and `doc-type-id` must match the equivalent identifiers used in the request's body (see Request's Body below).

Successful responses

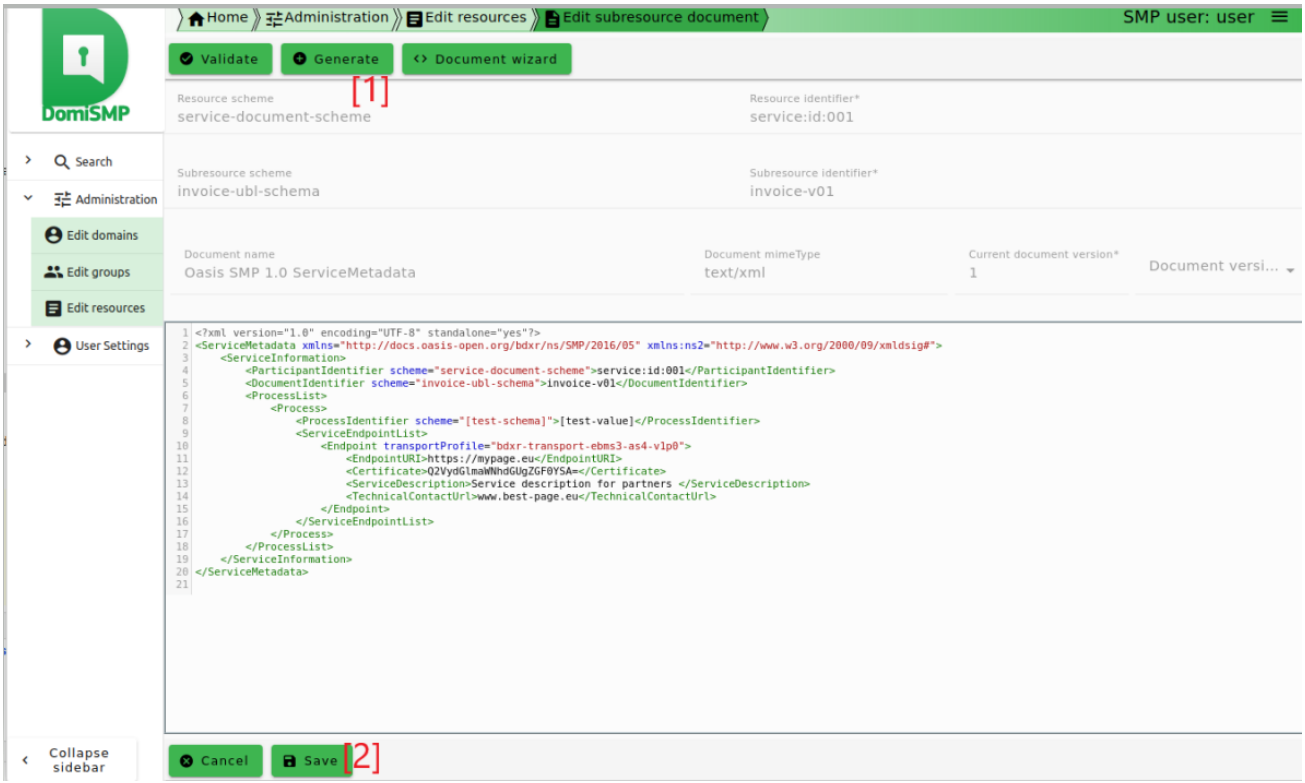
- `HTTP 200 (OK)` – ServiceMetadata was updated.
- `HTTP 201 (Created)` – New ServiceMetadata was created.

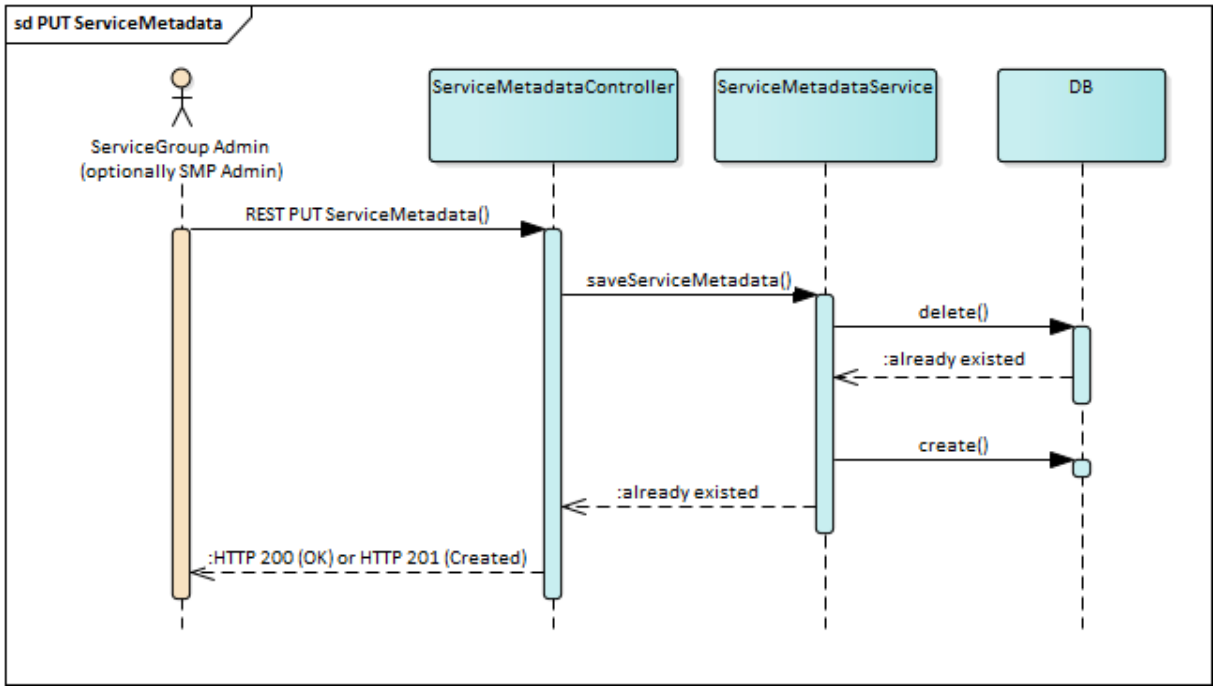
More about the request

- The Resource Admin, can register a `ServiceMetadata` with the DomiSMP UI tool for Service group management (see [1] in the picture below).
- To add `ServiceMetadata`, click first on the **Edit Resources** tool (see [1] in picture below), choose the resource and select tab **Subresources** [2].
- Click Create [3] and enter the `ServiceMetadata` identifiers in the dialog and click on **Create** [4].



- Once the record is created, click the edit button to enter the document editor for adding the ServiceMetadata XML (see the image below). To generate the ServiceMetadata click the button Generate [1] and then save button [3] below.





3.3.5. UC05 DELETE ServiceMetadata

▼ *UC05 DELETE ServiceMetadata details*

Prerequisites

- The `Resource Admin` initiating the request is linked to the specified `ServiceGroup` (or is `Admin SMP`).
- The authenticated user has the `Resource Admin` role.
- The referenced `ServiceMetadata` exists.

Description

This action removes the specified `ServiceMetadata` from the SMP's database. The SMP validates the request and deletes corresponding records.

Request's Header

```
DELETE http://smp.eu/participant-domain-scheme%3A%3Aparticipant-id/services/doc-
type-scheme%3A%3Adoc-type-id

HTTP/1.1
Accept-Encoding: gzip,deflate
Authorization: Basic c21wX2FkbWluOmNoYW5nZWl0
Content-Length: 0
```

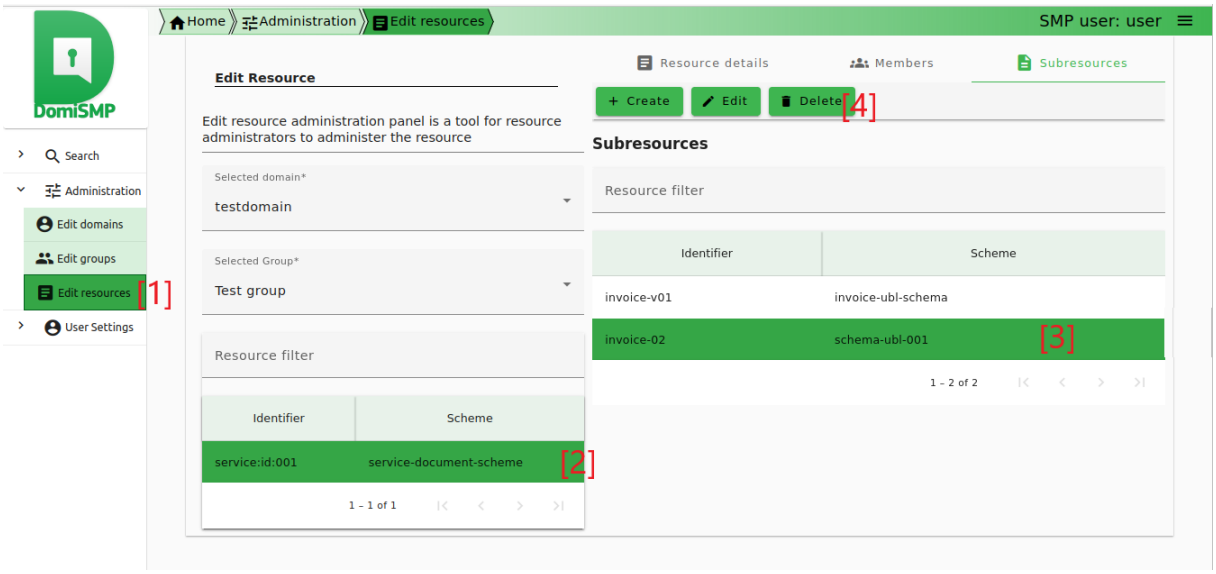
Successful responses:

- `HTTP 200 (OK)` – ServiceGroup was removed.

More about the request

- The `Resource Admin`, can delete a ServiceMetadata in the Admin Console using the **Editing the Resources** tool (see [1] in picture below).

- To delete a ServiceMetadata, select the resource which contains the ServiceMetadata [2].
- Then, in **subresources** table, select the service metadata for the deletion [3].
- Finally, click the delete button [4].



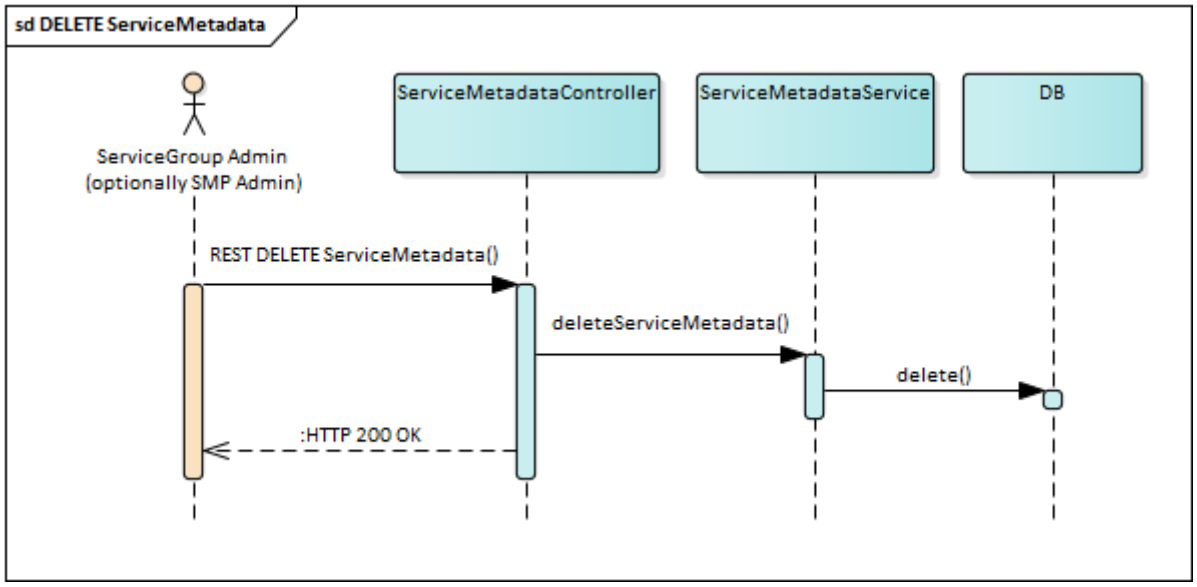


Figure 1. ServiceMedatada Delete Flow:

3.3.6. UC06 GET ServiceGroup

▼ *UC06 GET ServiceGroup details*

Prerequisites

- ServiceGroup exists.

Description

The SMP retrieves the details of the specified ServiceGroup from its database including references to all associated ServiceMetadata and returns them in XML format.

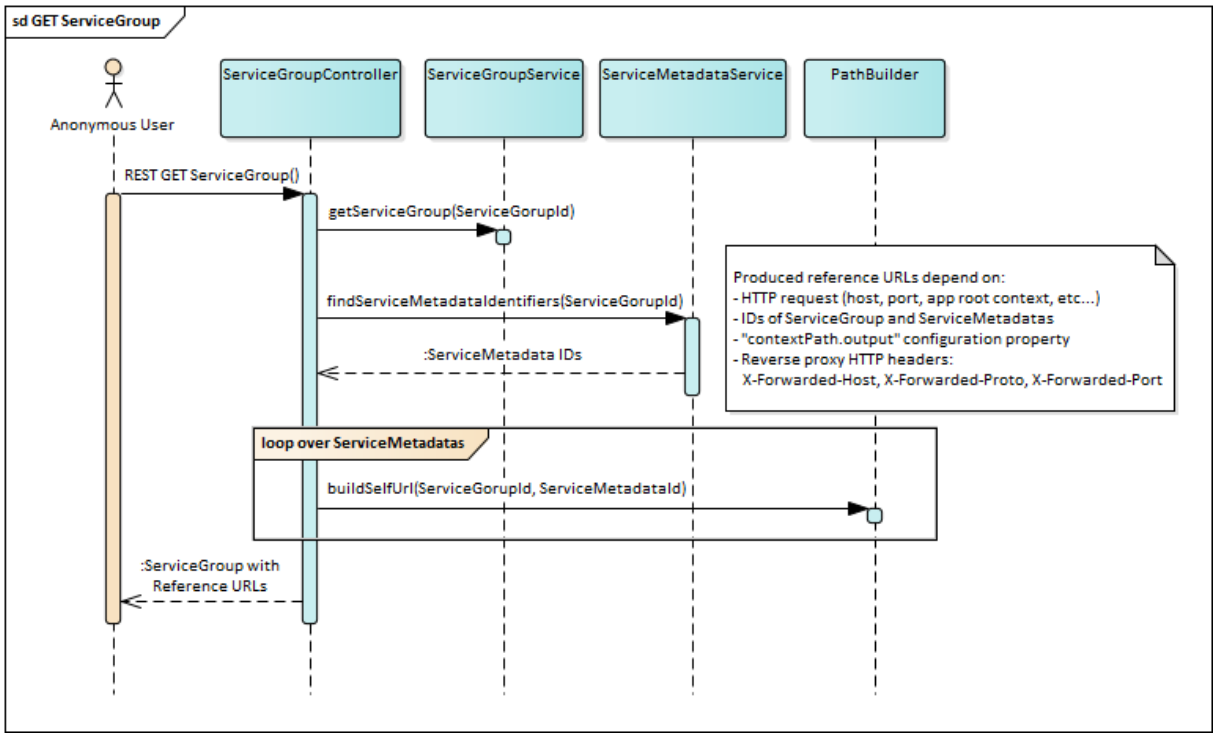
Request's Header

```
GET http://smp.eu/participant-domain-scheme%3A%3Aparticipant-id
```

```
HTTP/1.1
Accept-Encoding: gzip,deflate
Successful response: HTTP/1.1 200
Content-Type: text/xml;charset=UTF-8`
Content-Length: 496
```

Request's Body

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServiceGroup xmlns="http://docs.oasis-open.org/bdxr/ns/SMP/2016/05">
    <ParticipantIdentifier scheme="participant-domain-scheme">
        participant-id
    </ParticipantIdentifier>
    <ServiceMetadataReferenceCollection>
        <ServiceMetadataReference href="http://smp.eu/participant-domain-
scheme%3A%3Aparticipant-id/services/doc-type-scheme%3A%3Adoc-type-id"/>
    </ServiceMetadataReferenceCollection>
</ServiceGroup>
```



URL References

The URL references inside the `<ServiceMetadataReferenceCollection>` node refers to the same SMP and can be immediately used by the client to retrieve ServiceMetadata details. Because SMP is usually deployed behind a ReverseProxy, when the load balancer or the router redirects the request to the backend system, it adds below listed `X-Forwarded` parameters when constructing the URLs:

`X-Forwarded-Host`

identifies the original host requested by the client in theHostHTTP request header, since the host name and/or port of the reverse proxy (load balancer) may differ from the origin server

handling the request.

`X-Forwarded-Proto`

identifies the originating protocol of an HTTP request, since a reverse proxy (or a load balancer) may communicate with a web server using HTTP even if the request to the reverse proxy is HTTPS.

The ReverseProxy can also hide application root context, for instance, if the application is deployed on the server: `http://localhost/smp/`. Depending on the ReverseProxy configuration, the application can be accessed from internet without root context: http://smp.eu/ or with root context: http://smp.eu/smp/. To properly build the URL, the parameter `contextPath.output` must be set accordingly
See <<smp_arch_config, Configuration>.

3.3.7. UC07 GET ServiceMetadata

▼ *UC07 GET ServiceMetadata details*

Prerequisites

ServiceMetadata exists in the database.

Description

Service returns details of specified ServiceMetadata from the database. ServiceMetadata is signed and wrapped into the SignedServiceMetadata node.

Request's Header

```
GET http://smp.eu/participant-domain-scheme%3A%3Aparticipant-id/services/doc-type-
scheme/%3A%3Adoc-type-id ①

HTTP/1.1
participant-domain-scheme
doc-type-id
participant-id
Accept-Encoding: gzip,deflate
```

Where:

① The identifiers `participant-domain-scheme`, `participant-id`, `doc-type-scheme` are to be replaced in accordance with the specific call you wish to perform.

Successful sample response with SMP XMLDSIG signature

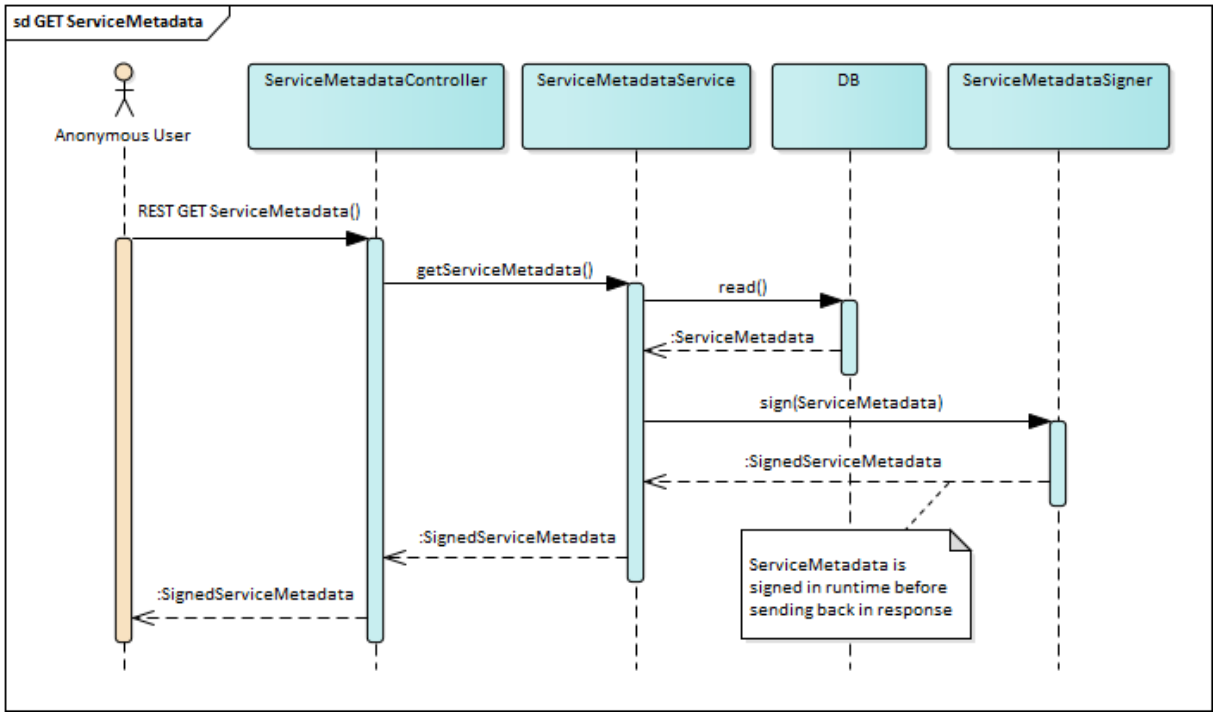
```
1 HTTP/1.1 200
2 Content-Type: text/xml;charset=UTF-8
3 Content-Length: 4939
4
5 <?xml version="1.0" encoding="UTF-8" standalone="no"?>
6 <SignedServiceMetadata
7     xmlns="http://docs.oasis-open.org/bdxr/ns/SMP/2016/05">
```

```
    <ServiceMetadata>
        <ServiceInformation>
            <ParticipantIdentifier scheme="participant-domain-scheme">
                participant-id
            </ParticipantIdentifier>
            <DocumentIdentifier scheme="doc-type-scheme">
                doc-type-id
            </DocumentIdentifier>
            <ProcessList>
                <Process>
                    <ProcessIdentifier scheme="cenbii-procid-ubl">
                        urn:www.cenbii.eu:profile:bii04:ver1.0
                    </ProcessIdentifier>
                    <ServiceEndpointList>
                        <Endpoint transportProfile="busdox-transport-start">
                            <EndpointURI>
https://poland.pl/theService</EndpointURI>
                            <RequireBusinessLevelSignature>
                                true
                            </RequireBusinessLevelSignature>
                            <ServiceActivationDate>
                                2003-01-01T00:00:00
                            </ServiceActivationDate>
                            <ServiceExpirationDate>
                                2020-05-01T00:00:00
                            </ServiceExpirationDate>
                            <Certificate>BASE64ENCODEDSAMPLECERT</Certificate>
                            <ServiceDescription>
                                Sample description of invoicing service
                            </ServiceDescription>
                            <TechnicalContactUrl>
https://example.com</TechnicalContactUrl>
                        </Endpoint>
                    </ServiceEndpointList>
                </Process>
            </ProcessList>
        </ServiceInformation>
    </ServiceMetadata>
    <Signature xmlns="http://www.w3.org/2000/09/xmldsig#"> ①
        <SignedInfo>
            <CanonicalizationMethod
                Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
            <SignatureMethod
                Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
            <Reference URI="">
                <Transforms>
                    <Transform
                        Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature"/>
                </Transforms>
                <DigestMethod
```

```
56                          Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
57                  <DigestValue>BASE64SAMPLEDIGEST</DigestValue>
58              </Reference>
59          </SignedInfo>
60          <SignatureValue>BASE64SAMPLESIGNATUREVALUE</SignatureValue>
61          <KeyInfo>
62              <X509Data>
63                  <X509SubjectName>Certificate subject name</X509SubjectName>
64                  <X509Certificate>BASE64CERTUSEDFORSIGNING</X509Certificate>
65              </X509Data>
66          </KeyInfo>
67      </Signature>undefined</SignedServiceMetadata>
```

Where:

① SMP's XMLDSIG signature is featured from line 44| onwards.



3.4. Implementation View

3.4.1. Source Code Overview

SMP is a Java REST application shipped and packaged as a `.war` file.

The SMP project uses Maven 3 for its build process and dependency management.
Below is a description of SMP's Maven project structure.

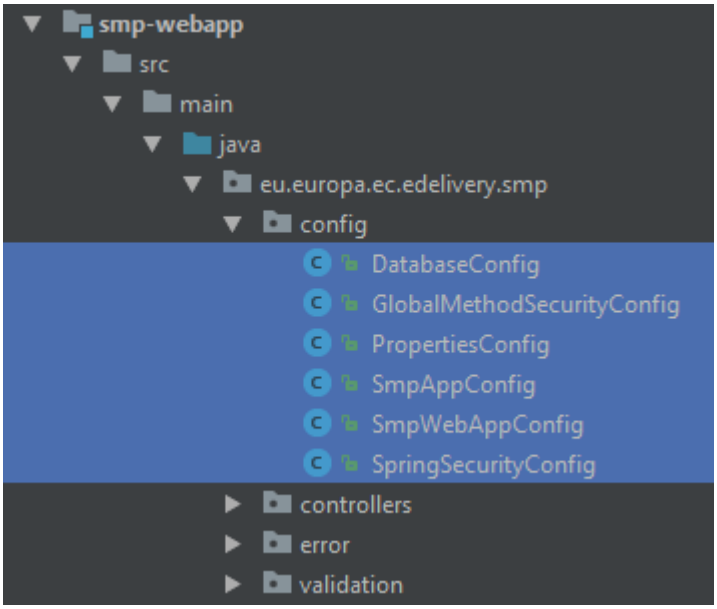
▼ *SMP Project Modules*

Module	Description
`smp-api`	Module contains OASIS SMP response schemas and administration API schemas. Module purpose is to generate java API classes from predefined XML schemas. Module also contains utility classes used for conversion and validation. This module is used by the SMP REST service implementation and can also be used for building SMP client.
`smp-parent-pom`	Parent POM contains dependency and plugin management used in sub-modules.
`smp-angular`	Angular web fragment for UI.
`smp-server-library`	SMP core library. Covers database access and business logic. This module does not have any HTTP/REST dependencies.
`smp-resource-extension`	The module contains the default resource extensions for Oasis SMP 1.0 and Oasis SMP 2.0 standard.
`smp-soapui-tests`	Module contains Soap UI tests for regression testing in the CI server.
`smp-ui-tests`	Module contains UI regression tests.
`smp-webapp`	REST interface over the core library. Defines REST binding, adds web-specific validations and security. Module also build SMP artefact for deploying to application server and package SMP setting examples, its output is WAR application and ZIP file smp_setup.zip with configuration files and Soap UI test project.
`smp-docker`	This module provides Dockerfiles for building Oracle/Tomcat and MySQL/Tomcat setups, along with Docker Compose files to launch the environments. Their primary purpose is to prepare a testing environment for API and UI integration. **IMPORTANT** These images are not production-ready and should not be used in live environments.
`smp-examples`	The module contains SMP examples of API and SPI implementations. Currently, SPI payload validation example.

3.4.2. Application Skeleton

Spring annotations context setup

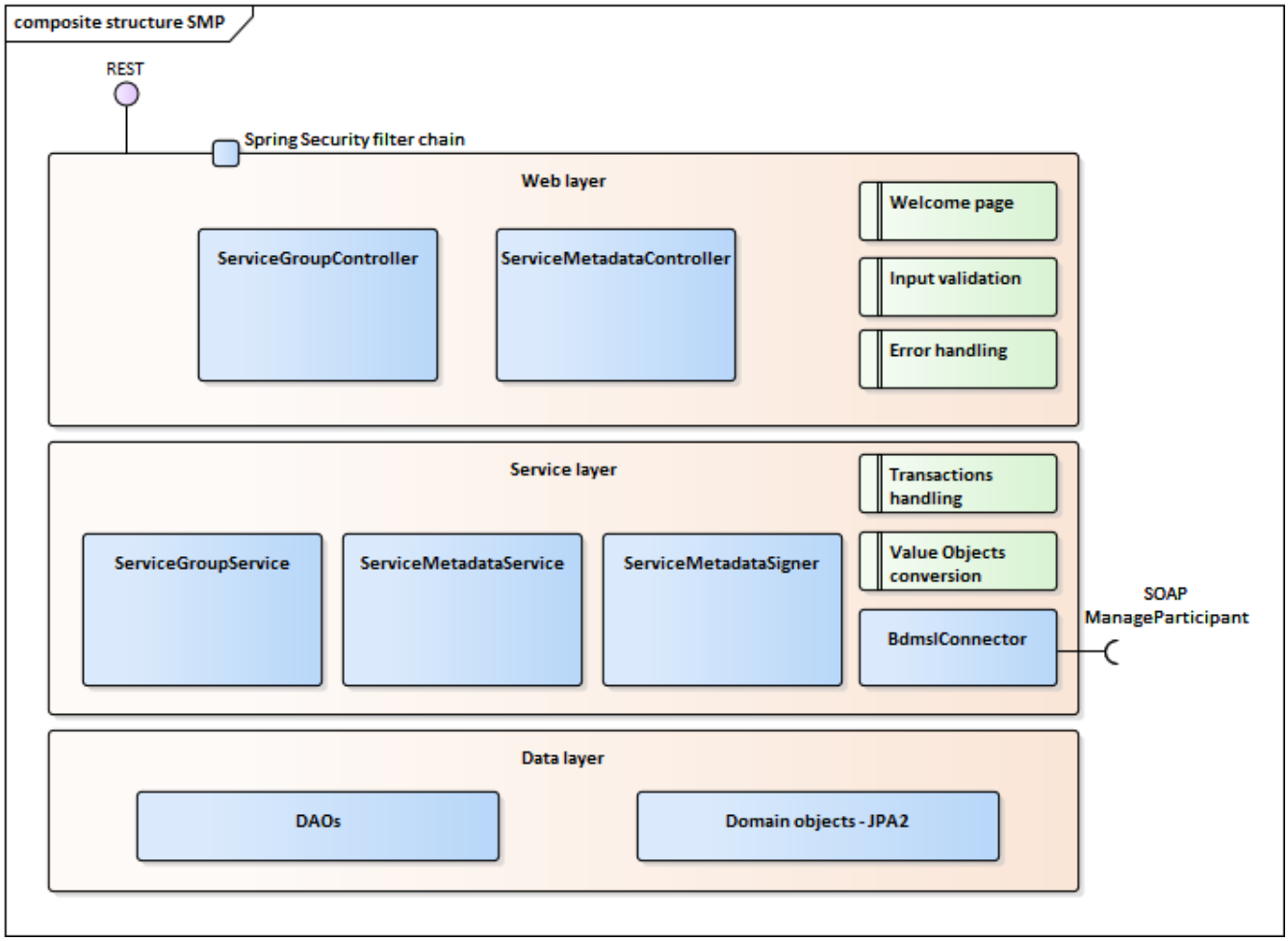
The SMP application is built with SpringFramework, the context is set up by classes with `@Configuration` annotations which are organized hierarchically. List of configuration classes, sample classes defining dependencies, scanning rules in packages and importing another context configuration are presented below.




```
Configuration
@ComponentScan(basePackages = {
  "eu.europa.ec.edelivery.smp.validation",
  "eu.europa.ec.edelivery.smp.services",
  "eu.europa.ec.edelivery.smp.sml",
  "eu.europa.ec.edelivery.smp.conversion"
})
@Import(DatabaseConfig.class)
public class SmpAppConfig {}
```

3.4.3. Layers Overview

- Spring MVC
- Business Services Layer
- Data Layer



Spring MVC

REST interface layer

The top layer, implemented within the smp-webapp module, uses Spring MVC's framework. Both resources (ServiceGroup, ServiceMetadata) have a dedicated Controller implementation. Each controller has 3 public methods (GET, PUT, and DELETE) which share the same URL defined by *@RequestMapping* annotation at the Controller class level.

A sample method definition, utilizing also metadata transferred in the request headers is presented below.

This layer is responsible for: REST binding, security validation, request data validation, forwarding request to services layer and forwarding response back to the caller and for error handling.
See more details in the Security section.

Sample method implementing REST action

```
@RestController @RequestMapping("/{serviceGroupId}") public class
ServiceGroupController {
    @PutMapping
    @Secured("ROLE_SMP_ADMIN")
    public ResponseEntity saveServiceGroup(
      @PathVariable String serviceGroupId,
      @RequestHeader(name = "ServiceGroup-Owner", required = false) String
serviceGroupOwner,
```

```
      @RequestHeader(name = "Domain", required = false) String domain,
      @RequestBody String body) throws XmlInvalidAgainstSchemaException,
UnsupportedEncodingException {
      /* . . . */ }
```

Business Services Layer

The business logic is implemented within the *smp-server-library module*. Business logic is implemented as ServiceGroup and ServiceMetadata Services. Module contains additional classes for Integration with BDMSL, signing messages and transaction handling with use of Spring *@Transactional* annotation and TransactionManager.

Because the SMP is a small application without need of polymorphism, the implementation does not use interface patterns for its services.

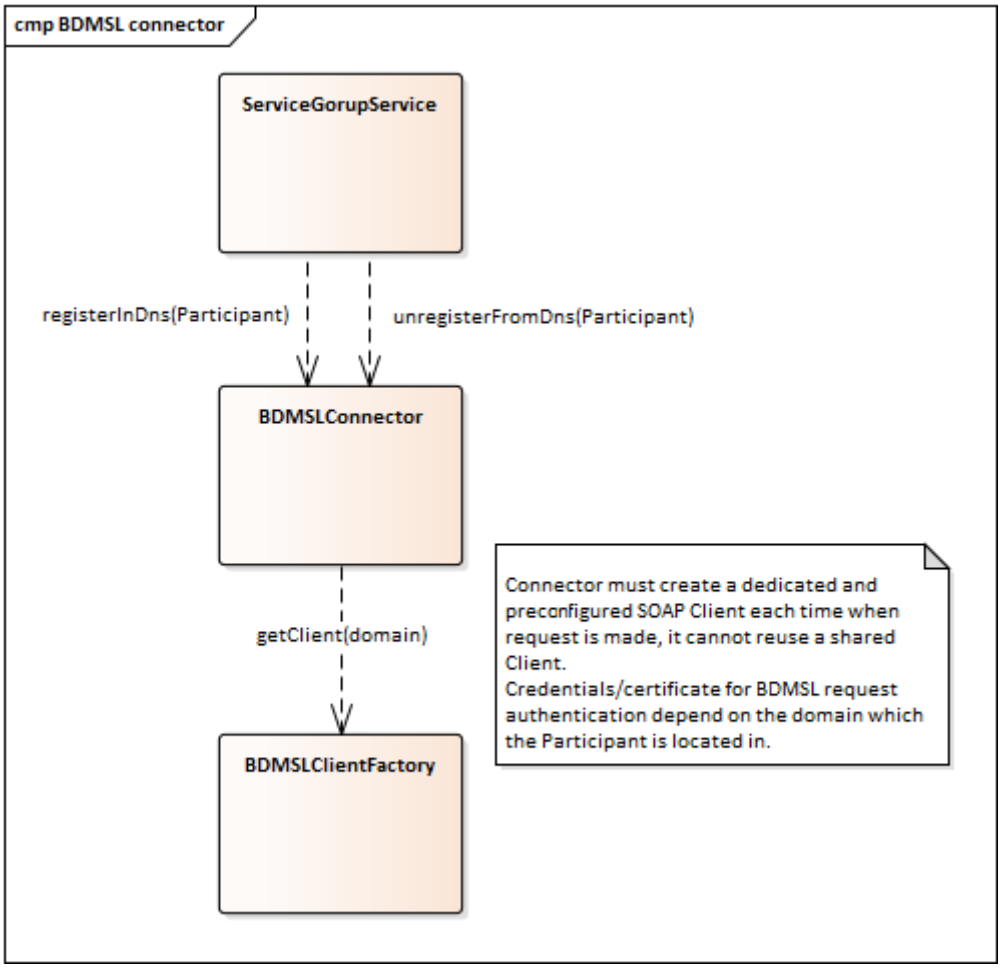
Sample Service method definition

```
@Service
public class ServiceMetadataService {
  @Transactional
  public boolean saveServiceMetadata(ParticipantIdentifierType serviceGroupId,
DocumentIdentifier documentId, String xmlContent) {
    /* . . . */
  }
```

BDMSL Integration

The BDMSL integration used by `ServiceGroupService` is implemented by `BDMSLConnector`. Participant's (un)registration is called synchronously as the last action Service's method to make sure that any potential `RuntimeException` causes rollback of the whole transaction, including database changes.

To support multiple domains functionality `BDMSLClientFactory` was introduced. Its responsibility is to create and preconfigure client (`BDMSLConnector`) to set up needed HTTP headers, configure proxy, manage client X509 Certificate, for each domain.
See also Domain Multitenancy.



Case Sensitivity Normalisation

Case Sensitivity Support

As functionally described in ebCore party identifier:

The eDelivery SMP has the feature to support handling participant identifiers as described in Use with eDelivery ebCore Party Identifiers, in the eDelivery SMP profile. In this case, the participant starts with the: `urn:oasis:names:tc:ebcore:partyid-type:` following by the words: `unregistered` or `iso6523`.

All ebCore party identifiers in the REST request must be URL-encoded using only one double colon separator `:`, as in the example below:

`urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088:4035811991021URL-encoded`

example:

`urn%23oasis%23names%23tc%23ebcore%23partyid-type%23iso6523%230088%234035811991021`

The eDelivery SMP has the option to serialize ebCore party Id to XML according to the OASIS SMP Specification as separate values, as in the example below:

```
<ParticipantIdentifier scheme="urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088">
    4035811991021
```

```
</ParticipantIdentifier>
```

or according to the eDelivery SMP profile as concatenated value:

```
<ParticipantIdentifier>
    urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088:4035811991021
</ParticipantIdentifier>
```

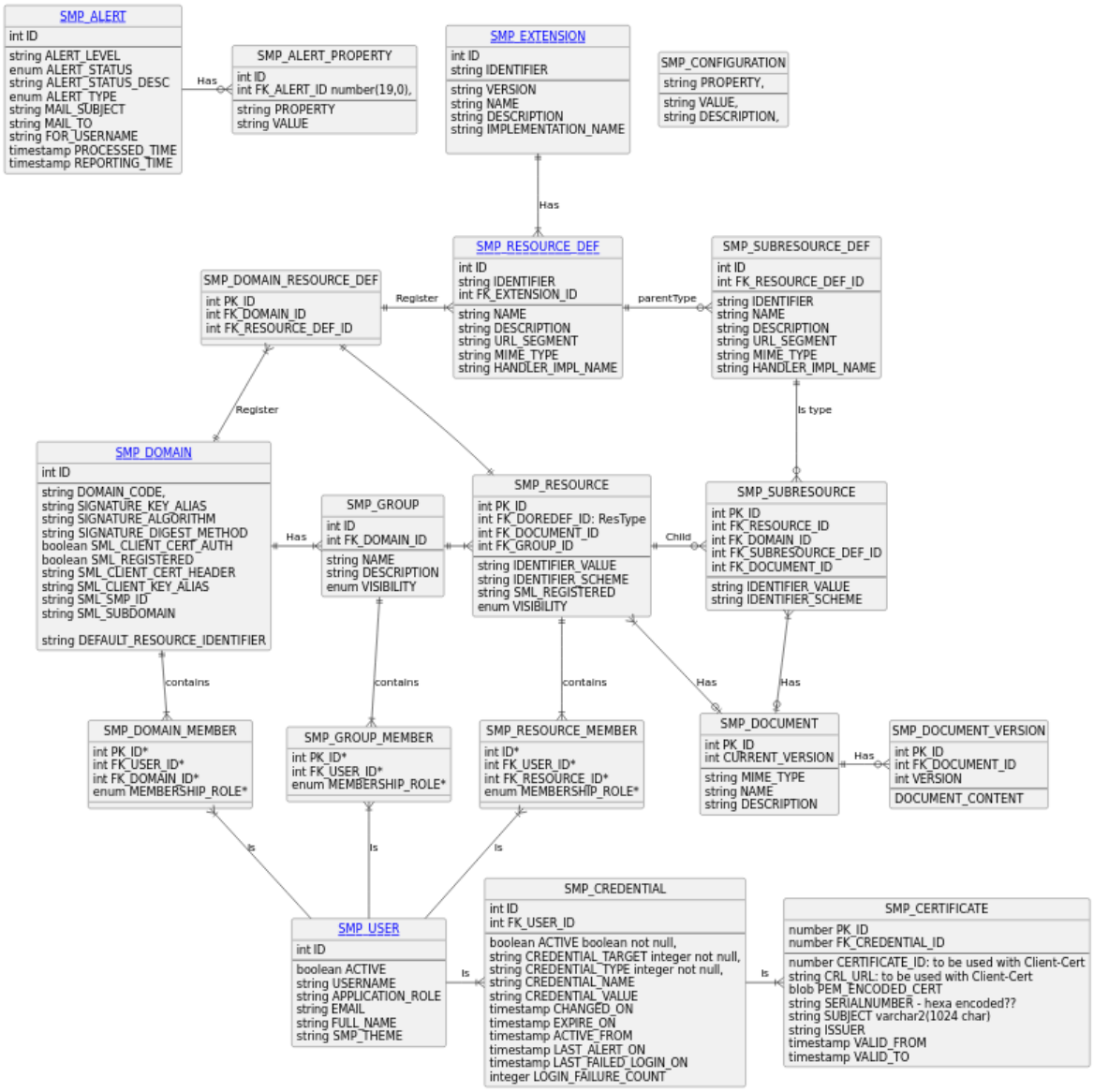
See Configuration for more info on the configuration of this behaviour.

Identifier's case sensitivity" and Configuration is implemented by the `CaseSensitivityNormalizer` bean. Normalization is performed at the very beginning of each service method processing. Also, by separating this to a dedicated bean, normalization can be used as well for permissions verification in connection with Spring Security's `@PreAuthorize` annotation:

```
@PreAuthorize("hasAnyAuthority('ROLE_SMP_ADMIN',
@caseSensitivityNormalizer.normalizeParticipantId(#serviceGroupId))")
```

Data Layer

The SMP stores data in a relational database. MySQL and Oracle DDL scripts are released with the application in `smp-setup.zip` file. The database object relations are presented in the following figure:



Besides all the necessary metadata used by the DomiSMP business logic, the database is also used to store XML documents in table (oracle: blob, mysql: TEXT type). The Resources and Subresources store versions of the document into the table `SMP_DOCUMENT_VERSION`. The documents are stored as a binary data because it could be electronically signed by Resource owner. Decomposing and composing XML could compromise the xml signature. When a user is querying for the resource/subresource, the original xml is returned with a valid xml signature.

The Java data access layer is implemented within the `smp-server-library` module. `DataSource`, `EntityManager` and `TransactionManager` are configured and registered into Spring context in the `DatabaseConfig` class.

Java classes located in `eu.europa.ec.edelivery.smp.data.model` package define the Model with the use of JPA2 annotations. All model classes implement the `BaseEntity` interface. Separate @Embeddable classes are defined for all composite primary keys:

Sample(part) of JPA2 Model class with embedded composite PK

```
@Entity
@Table(name = "smp_service_group")
public class DBServiceGroup implements BaseEntity {
```

```
  @EmbeddedId
  @Override
  public DBServiceGroupId getId() {
    return serviceGroupId;
  }
  /* . . . */
}
```

Sample(part) @Embeddable composite PK

```
@Embeddable
public class DBServiceGroupId implements Serializable {
  @Column(name = "businessIdentifierScheme", nullable = false, length =
MAX_IDENTIFIER_SCHEME_LENGTH)
  public String getBusinessIdentifierScheme() {
    return participantIdScheme;
  }

  @Column(name = "businessIdentifier", nullable = false, length =
MAX_IDENTIFIER_VALUE_LENGTH)
  public String getBusinessIdentifier() {
    return participantIdValue;
  }
  /* . . . */
}
```

All DAO classes located in the `eu.europa.ec.edelivery.smp.data.dao` package extend the `BaseDao` generic abstract class that already provides most common DAO operations (find, remove, etc.).

See some samples below:

Simplest DAO not requiring to provide additional methods

```
@Repository
public class ServiceGroupDao extends BaseDao<DBServiceGroup> {}
```

Significant part of the generic BaseDao

```
@Repository
public class ServiceGroupDao extends BaseDao < DBServiceGroup > {}
public abstract class BaseDao < E extends BaseEntity > {
  @PersistenceContext
  protected EntityManager em;

  private final Class < E > entityClass;

  public BaseDao() {
    entityClass = (Class < E > ) GenericTypeResolver.resolveTypeArgument(getClass(),
BaseDao.class);
```

```
  }

  public E find(Object primaryKey) {
    return em.find(entityClass, primaryKey);
  }
  /* . . . */
}
```

Exception Handling

Detailed functional description of all errors that might occur is presented in the SMP Interface Description guide. This section presents a generalized view on error groups and focuses on implementation perspective.

eDelivery SMP utilizes HTTP error codes according to the best RESTful recommendations, i.e., given codes are always returned for:

- **200 (OK)** or **201 (Created)** – Successful responses
  (Resource was created/updated/retrieved/deleted).
- **4xx (Bad request)** – Invalid or unauthenticated request.
- **5xx (Server Error)** – SMP technical issue, could be related to configuration, internal networking, integration with BDMSL or DB, etc.

The OASIS SMP Specification does not specify error messages, so eDelivery SMP introduces its own simple XSD with XML namespace: `ec:services:SMP:1.0`.

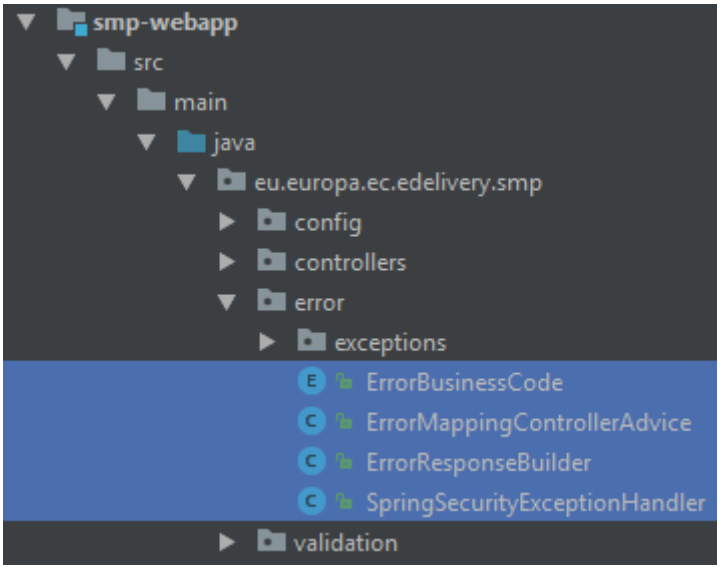
This one describes the structure of error response messages as in sample below:

Sample error response

```
<ErrorResponse
    xmlns="ec : services:SMP:1.0">
    <BusinessCode>NOT_FOUND</BusinessCode>
    <ErrorDescription>
        ServiceMetadata not found, ServiceGroupID: 'x ::y', DocumentID: 'a::b'
    </ErrorDescription>
    <ErrorUniqueId>2018-03-27T15 :07 :35.470CEST :d3ba543a-7233-4e69-9f34-
655e3998cb3c</ErrorUniqueId>
</ErrorResponse>
```

▼ Error Handling Mechanism Implementation

All classes for processing errors are located in package `eu.europa.ec.edelivery.smp.error`:



▼ **ErrorMappingControllerAdvice**

All backend exceptions are mapped to REST responses within one single class registered in Spring context with `@RestControllerAdvice` and by its many handler-methods annotated with `@ExceptionHandler`. The class uses `ErrorResponseBuilder` and is responsible for:

- mapping exceptions to HTTP response codes and `ErrorBusinessCodes`;
- logging user errors as WARN level and technical errors as ERROR level including `uniqueErrorId` for easier maintenance and debugging.

Class declaration, sample handler-method (one of many) and internally reused `buildAndWarn` method:

```
RestControllerAdvice
public class ErrorMappingControllerAdvice {

  @ExceptionHandler(NotFoundException.class)
  public ResponseEntity handleNotFoundException(NotFoundException ex) {
    return buildAndWarn(NOT_FOUND, ErrorBusinessCode.NOT_FOUND, ex.getMessage(),
ex);
  }
/* . . . */
private ResponseEntity buildAndWarn(HttpStatus status, ErrorBusinessCode
businessCode, String msg, Exception exception) {
/* . . . */ }
}
```

▼ **ErrorResponseBuilder**

`ErrorResponseBuilder` implementing builder pattern is responsible for building Spring's `ResponseEntity`, based on provided HTTP status code, `ErrorBusinessCode` and text message. Produced response not only is compliant with introduced dedicated XSD, but contains a `uniqueErrorId` that in future problem investigation can be easily found out in log files once user provides error message details.

Every `uniqueErrorId` is built out of:

- **Timestamp** – this information facilitates support and development by specifying when the error occurred and in which rolled log file more details can be found.
- **UUID** – helps in uniquely locating the error stack trace.

`2018-03-27T15:07:35.470CEST:d3ba543a-7233-4e69-9f34-655e3998cb3c`

▼ **ErrorBusinessCode**

`ErrorBusinessCode` is a simple `Enum` with given values, used by other error-handling classes:

Business error code	Description
`XSD_INVALID`	Bad request, XML document provided by the user does not pass schema validation
`WRONG_FIELD`	Bad request, one of the request fields is wrong, e.g., specified Domain does not exist.
`OUT_OF_RANGE`	Bad request, e.g., specified dates from-to are overlapped.
`FORMAT_ERROR`	Bad request, e.g., provided identifier format does not comply with OASIS SMP Specifications.
`UNAUTHORIZED`	Unauthorized (`HTTP 401`), the user has no permission to access requested resource.
`NOT_FOUND`	Bad request, the requested resource does not exist (`GET` or `DELETE`).
`USER_NOT_FOUND`	Bad request, e.g., the newly created ServiceGroup cannot be owned by a user that does not exist.
`TECHNICAL`	Technical problem on SMP or infrastructure side (BDMSL integration, database etc). This error is always returned with `HTTP 500`, "Internal Server Error" code. The specific cause of this error is not communicated in the response since Exceptions' messages might eventually reveal sensitive information.

SpringSecurityExceptionHandler

`SpringSecurityExceptionHandler` is a glue code that allows exceptions thrown by SpringSecurity to be processed by a common exception-handling mechanism. As a result, all security error responses follow the same pattern as other error responses.

SpringSecurity is implemented as a filter chain at the very beginning of the processing of HTTP requests.

3.5. Configuration

SMP configuration (database, keystore, authentication type ...) is placed in the property file `smp.config.properties`. File with default values is already included in deployment war package. To override custom values the copy of `smp.config.properties` with updated values must be placed in the application server classpath. More details on configuring classpath can be found in the SMP

Administration Guide and in Environment specific configuration.

When the SMP is used in multi-tenancy as described in section Domain Multitenancy, the configuration properties for domain (SMP ID, BDMSL authentication data) are located in database table: `SMP_DOMAIN`. One record represents one domain, columns represent configuration parameters which are applied for that specific domain. See more about Domain configuring in the SMP Administration Guide.

Environment specific configuration

Detailed configuration steps for Windows and UNIX systems are covered in the SMP Administration Guide. This section is focused explaining the motivation behind particular configuration rather than configuration steps themselves.

3.5.1. Tomcat

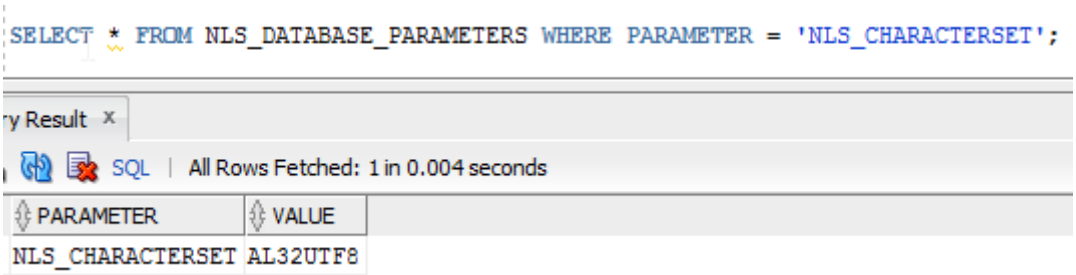
Classpath:

The SMP requires configuration file: **smp.config.properties** to be placed in the classpath. On tomcat server custom classpath folder (e.g. /conf_dir_path) can be set by modifying the starting scripts in the same way as for WebLogic, or by adding this entry in `context.xml` file:

```
<Resources
 className="org.apache.catalina.webresources.StandardRoot"
 cachingAllowed="true" cacheMaxSize="100000">
 <PreResources
   className="org.apache.catalina.webresources.DirResourceSet"
   base="/conf_dir_path"
   internalPath="/"
   webAppMount="/WEB-INF/classes" />
</Resources>
```

3.5.2. Oracle

`NLS_CHARACTERSET` must be set to `AL32UTF8`, otherwise SMP will face issues with non-ASCII characters.



3.5.3. MySql

Character set, collation and especially JDBC connection protocol encoding – all must be set to UTF-8, otherwise SMP will face issues with non-ASCII characters.

```
1  SHOW VARIABLES WHERE Variable_name
2  LIKE 'character\_set\_%' OR Variable_name LIKE 'collation%';
```

sult Grid | Filter Rows: | Export: | Wrap Cell Content:

Variable_name	Value
character_set_client	utf8
character_set_connection	utf8
character_set_database	utf8
character_set_filesystem	binary
character_set_results	utf8
character_set_server	utf8
character_set_system	utf8
collation_connection	utf8_general_ci
collation_database	utf8_bin
collation_server	utf8_general_ci

3.6. Security

The SMP is secured with the SpringSecurity. The spring security configuration is executed at the eDelivery startup in the following classes:

- `WSSecurityConfigurerAdapter.java`: class that handles the webservice endpoint security configuration;
- `UISecurityConfigurerAdapter.java`: class that handles the UI endpoint security configuration;
- `SMPCasConfigurer.java`: class that handles the UI Cas configuration.

3.6.1. Authentication

The Authentication Manager (id = `smpAuthenticationManager`) utilizes two Authentication One handles basic username/ password authentication and the second is SpringSecurity implementation `PreAuthenticatedAuthenticationProvider` class configured to handle `X509Certificate` and `BlueCoat` authentication. The pre-authenticated scenarios take precedence over basic authentication. That means if a client provided a valid certificate and also valid username and password, then he is logged in using his certificate and username/password is ignored.

Username and password authentication (Basic Authentication forUI)

Standard SpringSecurity mechanism is used to verify username and BCrypt hashed passwords using the SMPAuthenticationProvider. Username/Password authentication can be used for the UI authentication.

Access token authentication (Basic Authentication for web-services)

eDelivery SMP uses different credentials for UI and for WebService authentication.
The access token is randomly generated access token id and access token value. Together they are used as HTTP basic authentication when invoking the web-services.

Client certificate authentication

Client Certificate authentication can be used only for authentication when invoking the REST API services. The purpose of the certificate authentication is to support mutual 2-way TLS authentication for machine-to-machine integration.

SMP supports two types of Client Certificate authentications: X509 certificate authentication and Authentication behind Reverse Proxy. Both scenarios are performed in 2 steps:

1. Certificate details are extracted to the eDelivery-specific text format. This step is handled by two custom filters: `x509AuthFilter` and `blueCoatReverseProxyAuthFilter`, separately for both scenarios.
2. `PreauthAuthProvider` verifies that if certificate-defined user exists in the database.

`X509Certificate` and Certificates HTTP Client-Cert header are validated with the following attributes:

- Valid from: if “current date” is smaller than “valid from” date, then authentication is rejected
- Valid to: if “current date” is greater than “certificates valid to” date, then authentication is rejected
- Revocation List: certificates are validated by CRL which is downloaded and cached till the CRL “valid to” date. CRL URL endpoint is defined in `SMP_CERTIFICATE.CRL_URL` column and is used for HTTP Client-Cert authentication and for X509Certificate authentication. If the CRL is not reachable, SMP silently ignores the CRL verification, if the configuration attribute “smp.certificate.crl.force” is set to false. If the attribute is set to true, then Client is not authenticated due to technical issues.
- Truststore: If the SMP truststore is not empty, then formatted issuer or subject is verified if it exists in the truststore. If none of the values exists in the truststore, then certificate authentication is rejected.

Users that are authenticated by certificate are stored in the SMP_USER table, together with users authenticated by password. The USERNAME value of certificate authenticated users is a string value created from parts of certificate distinguish name (DN) and serial number by the following pattern (eDelivery format):

```
CN={common name},O={organisation},C={country}:{16-digit-zero-padded-hex-serial}
```

Example:

```
CN=CEF eDelivery,O=European Commission,C=BE:000000000000c41f
```

Application distinguished certificate authenticated users from password-authenticated user by an empty PASSWORD column.

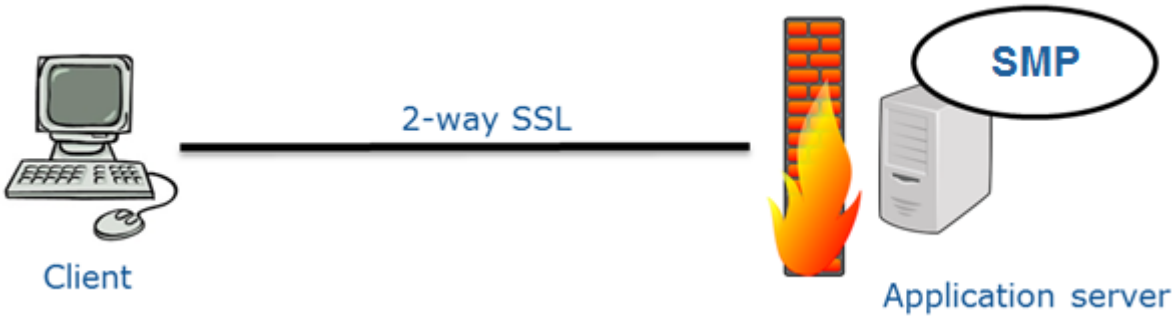
Most eDelivery projects supporting client certificate authentication, utilize the same client certificate text representation and BlueCoat Client-Cert HTTP header patterns. For this reason, custom Java code responsible for client certificate authentication has been extracted and released within a separate JAR library; maven dependency gropuId/artifactId:

eu.europa.ec.edelivery/edelivery-springsecurity-2-way-ssl-auth.

X509 certificate authentication

The client X509 certificate authentication uses server's (Tomcat or WebLogic) certificate authentication settings. After the request passes the server validation successfully, *x509AuthFilter* extract certificate details and then authentication proceeds in the way as described above.

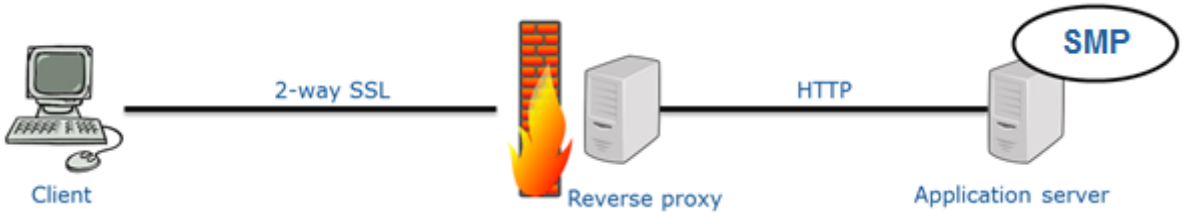
The filter itself (class *EDeliveryX509AuthenticationFilter*) is a simple extension of SpringSecurity's *X509AuthenticationFilter* class, which is a ready-to-use implementation handling *java.security.cert.X509Certificate*.



Authentication behind Reverse Proxy

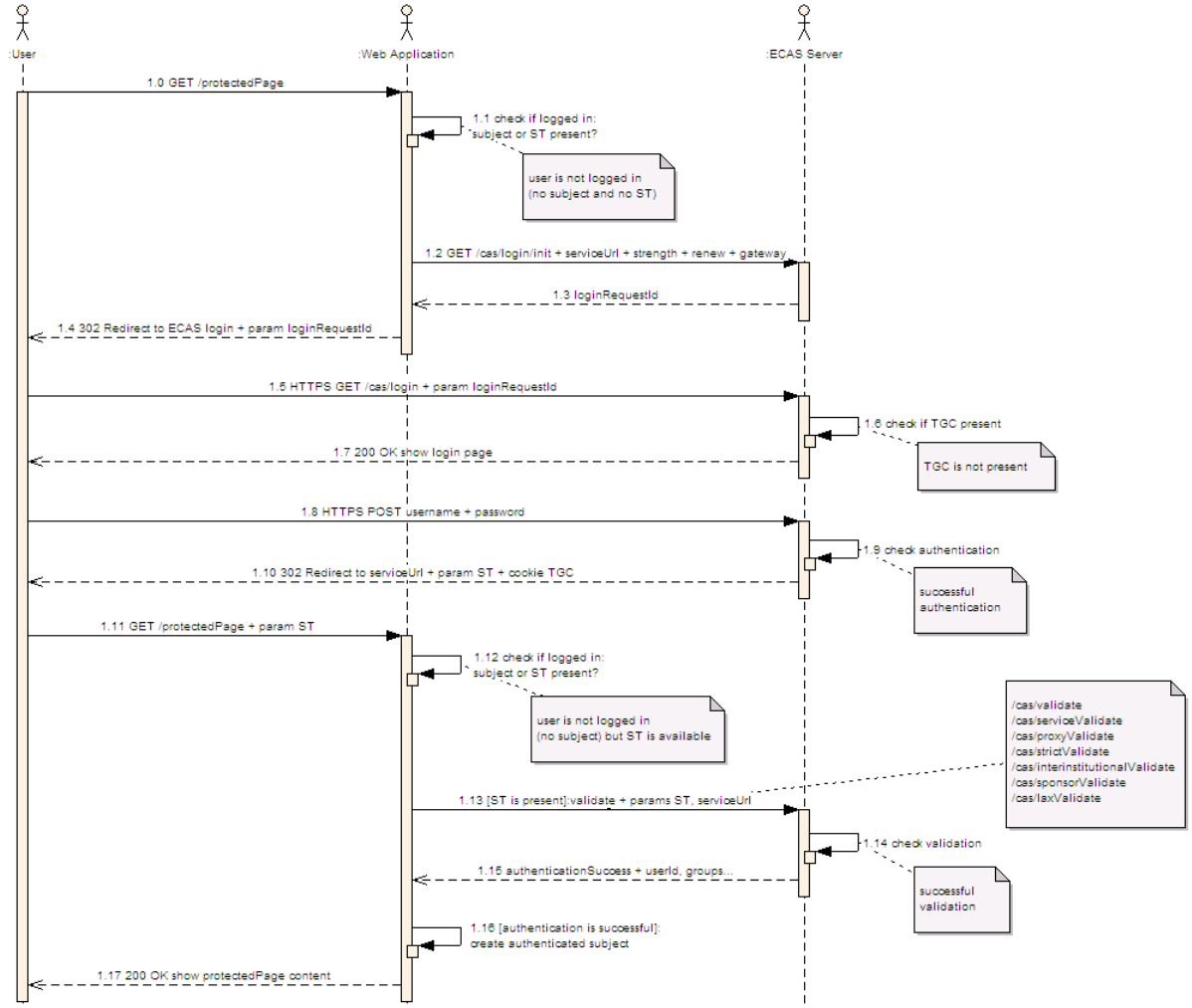
In this setup the basic certificate validation is configured in the BlueCoat reverse proxy. After certificate validation passed successfully, the BlueCoat reverse proxy adds a "Client-Cert" HTTP header and forwards the request to the SMP over HTTP(S). The spring filter *blueCoatReverseProxyAuthFilter* extracts the header, converts it from Bluecoat's to the eDelivery format specified above and then authentication proceeds in the way as described above.

The filter itself (class *BlueCoatAuthenticationFilter*) is based on the SpringSecurity's *RequestHeaderAuthenticationFilter*, dedicated for similar scenarios.



SSO Central Authentication service with EU-LOGIN

CAS authentication can be used only for the UI authentication, and it was made with intention to integrate with ECAS also called EU-Login. ECAS is based on the Central Authentication Service (CAS) version 2 developed at Yale University1. It is an authentication service to protect Web-based applications. SMP was tested only with ECAS, but it should also work with any CAS 2.0 implementation,



When the SMP does not find a service ticket granting access it redirects to EUs login page for user authentication. After user authenticates via the EU login, the response redirects the page back to the SMP UI page with granting ticket.

SMP validates ticket with ECAS. If validation is successful, the SMP authorize access to the user according to user authorization defined on SMP user configuration.

3.6.2. Authorization

Authorities and Roles

Authorities

Authorities in SMP are organized into a two-dimensional space, with Roles as first dimension and **Error! Reference source not found.** as the second one.

Roles

Roles are documented with more details in the SMP Interface Description guide. The table below explains their meaning from the implementation perspective:

Role alias	Description
ROLE_ANONYMOUS	Any user that has not provided any authentication details.

Role alias	Description
ROLE_USER	Any authenticated user existing in the database and who doesn't have system admin permissions. Such user is supposed to be a member of the Domain, Group or Resource.
ROLE_SYSTEM_ADMIN	Role for UI enables administration of domains and users.

ICD mentions "System Admin" role, but it's rather a sysadmin, not the business role to be considered in SMP source code.

Authorities execution

Authorities' verification is very flexible thanks to loading all granted authorities to the security context.

HTTP methods: GET/PUT/DELETE

The first level of verification is made on HTTP method level. GET is allowed to everybody, while all modifying actions are allowed only to authenticated users, which is configured in `spring-security.xml` file:

```
<intercept-url method="PUT" access=" ! isAnonymous()" pattern="/*"/>
<intercept-url method="DELETE" access=" ! isAnonymous()" pattern="/*"/>
```

Business object and action level

Once all granted authorities are present in the security context, they are validated at the business methods level with SpringSecurity's annotations and Spring Expression Language (SpEL):

```
@Secured("ROLE_SMP_ADMIN")
```

action allowed only for Group Admin, or:

```
@PreAuthorize("hasAnyAuthority('ROLE_SMP_ADMIN',
  @caseSensitivityNormalizer.normalizeParticipantId(#serviceGroupId))
")
```

action allowed either for `Group Admin` or `Resource Admin` owing the `serviceGroupId` provided as methods' parameter.

3.7. Quality

SMP quality is supervised by Code Reviews and Continuous Integration processes, which are out of the scope of this document. The quality measurement details presented below focus on technical and source-code point of view.

Unit tests

All utility classes that do not interact with many other classes, which are mostly responsible for conversions, mappings, etc., are unit tested with using Junit and Mockito libraries. Test class name pattern in this case is: `{testedClassName}Test.java`. Tests are run at application build time.

Integration tests

Service classes that combine multiple application modules and in most of the cases require database access are tested in classes with name pattern: `{testedClassName}IntegrationTest.java`.

Tests are executed with JUnit library and configured Spring test context. Also, database instance must be created and defined in maven project files with the following properties:

Property	Description
jdbc.driver	Database Configuration: Driver • MySQL: ◦ com.mysql.jdbc.Driver • Oracle Database: ◦ oracle.jdbc.OracleDriver
jdbc.url	Database Configuration: url • MySQL: jdbc:mysql://dbhost:dbport/smp_database • Oracle Database: jdbc:oracle:thin:@dbhost:dbport:smp_database or jdbc:oracle:thin:@dbhost:dbport/smp_service
jdbc.password	Database User/Password Configuration: User
jdbc.password	Database User/Password Configuration: Password
target-database	Target Database Backend type/Brand: For MySQL, use: MySQL For Oracle Database, use: Oracle
jdbc.read-connections.max	Database Configuration: Max Read Connection

Example:

```
<properties>
 <jdbc.driver>com.mysql.jdbc.Driver</jdbc.driver>
 <jdbc.url>jdbc:mysql://localhost/smp</jdbc.url>
 <jdbc.user>smp</jdbc.user>
```

```
  <jdbc.password>smp</jdbc.password>
  <target-database>MySQL</target-database>
  <jdbc.read-connections.max>10</jdbc.read-connections.max>
</properties>
```

SoapUI integration tests

All functionalities are covered with SoapUI integration tests that run REST requests against the SMP and in some cases access the database directly with SQL statements. The SoapUI project can be found in submodule `smp-soapui-tests\soapui\SMP4.0-Generic-soapui-project.xml` file. These tests are bound to maven build and can be activated at build time with maven profile `-Prun-soapui` switch.

Sonar source code statistics

Maven build is configured to collect standard Sonar code statistics (code test coverage, static code analysis, etc). Apart from that, code test coverage is gathered also when running SoapUI tests. This requires a manual installation of Jacoco Agent in JRE with J2EE container where the SMP is deployed and pointing to this agent when running a build by adding these attributes to maven run:

```
-DjacocoRemotePort=65000 DjacocoRemoteAddress
```

Once build with SoapUI tests is done, statistics from all the sources are gathered by sonar plugin by running `mvn sonar:sonar` goal.

3.8. Technical Requirements

This section describes the minimum and recommended system requirements to operate the SMP component.

Hardware

Type	Minimum	Recommended
Processor	2 CPU core	4 CPU core
Memory (RAM)	2GB	8GB or more
Disk space	5GB	Depends on usage

Recommended stack

- Ubuntu 24.04 LTS 64 bits
- OpenJDK 21
- MySQL 8

Operating Systems and Software

OS

Any operating system that is compliant with the supported JVM.

Java Virtual Machines

- OpenJDK 21

Java Application Servers

- Apache Tomcat 10.1.x

Databases

- MySQL 8
- Oracle Database 19c

Web Browsers

n/a

Chapter 4. Administration Guide

Contents

This guide provides information on how to:

- **Deploy and configure SMP on supported application servers and databases.**
  See Prerequisites and Relevant Resources.
- Perform relevant security configurations (certificates).
- Consume the Soap UI to create, update and delete SMP Service Groups and Metadata and an alternative method to perform creation, update and deletions operations using Swagger UI.

Target Audience

This guide is intended for Administrators who are in charge of installing, managing and troubleshooting an eDelivery SMP.

4.1. Prerequisites and Relevant Resources

Software Requirements

SMP requires:

- one supported Java Runtime Environment (JRE)
- one supported Webserver
- one support Database Management Systems (DBMS)

Supported versions of required software for SMP

Java Runtime Environment	
	Eclipse Temurin JDK 21 : Download it here.
Webservers	
	Tomcat • Apache Tomcat 10.1
Databases	
	• MySQL 8.0.x * tested version, future versions might also work • Oracle 19c * tested version, future versions might also work

NOTE	For more information and installation details for third-party software, refer to their specific documentation.

Binaries Repository

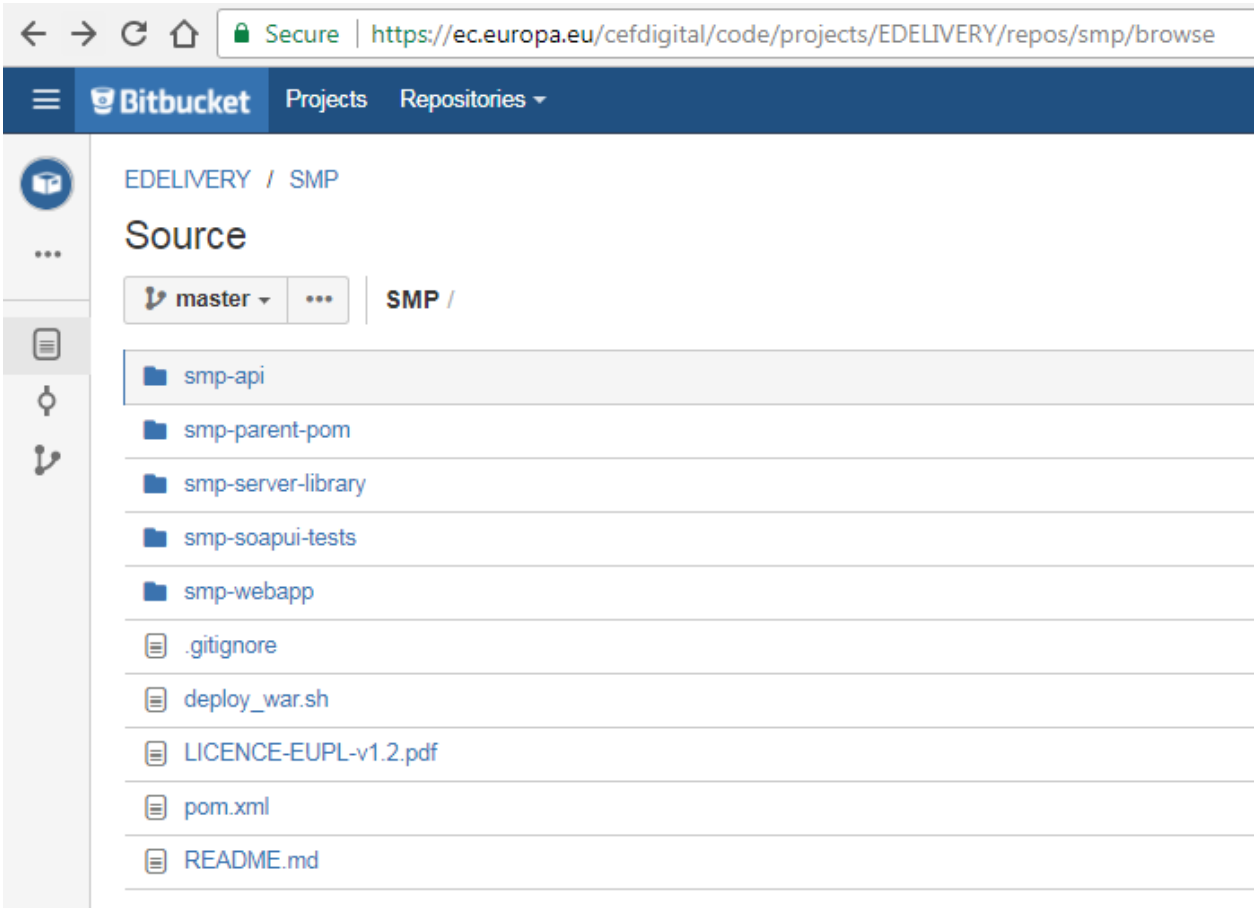
- The DomiSMP **artifacts** can be downloaded from the eDelivery Digital Portal.

Source Code Repository

- The **source code** of eDelivery DomiSMP is available from this public repository.

▼ *For visible URL, click here.*

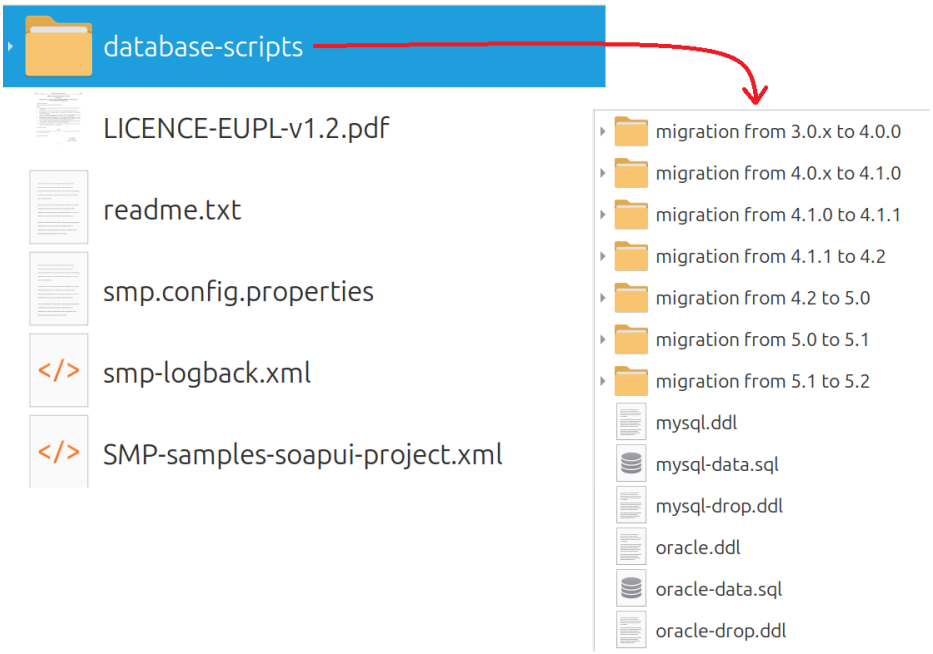
SMP Source Code Repository:
https://ec.europa.eu/digital-building-blocks/code/projects/EDELIVERY/repos/smp/browse



Database Scripts

The scripts for creating or migrating the Oracle and MySQL databases can be found in the DomiSMP Setup Bundle, the `smp-5.2-setup.zip` archive.

SEE ALSO See this SMP's release page to download the DomiSMP Setup Bundle archive.



4.2. Deployment Overview

▼ *SMP Deployment Steps Overview*

As mentioned in the prerequisites, the deployment of the SMP is only supported on Tomcat or WebLogic application servers.

The deployment of the SMP on both platforms is almost identical but minor platform specific changes will be documented in a dedicated section of this manual.

The deployment of the SMP is summarized in the following mandatory steps:

- STEP1 **Database Configuration**
- STEP2 **Application Server Preparation** (Weblogic and Tomcat) for SMP
- STEP3 **SMP Initial Configuration**
- STEP4 **SMP .war file Deployment**

NOTE	The environment variable, `AS_HOME`, refers to the application server home folder where the SMP package is installed.

- For Tomcat, it refers to `CATALINA_HOME`.
- For Oracle WebLogic, it refers to `DOMAIN_HOME`.

NOTE	The environment variable, `SETUP_PATH`, refers to the folder to where the DomiSMP Setup Bundle, the `smp-5.2-setup.zip`, is extracted.

See also

See this SMP's release page to download the DomiSMP Setup Bundle archive.

▼ *Folder structure*

The following subdirectories must be created in the AS_HOME directory. The document describes the default folder settings and can be named or created in a location other than the AS_HOME directory.

AS_HOME/smp
: the folder contains the basic SMP settings, and the folder must be configured as a classpath.

  See also

  - Configuring Extra Class Path in WebLogic
  - Configuring Extra Class Path in Tomcat

AS_HOME/logs
: the purpose of the folder is to contain SMP logs.

AS_HOME/security
: the previous versions of the SMP have security artifacts (truststore, keystore, etc.) under the /smp folder. We recommend creating a separate folder for a more transparent handling of the security artifacts. In case of setting SMP in an application server cluster, this folder must be shared among the cluster nodes.

The location of the folder must be set in the SMP application property: smp.security.folder.

NOTE	before DomiSMP 5.0 version, the application property's name was configuration.dir.

4.3. STEP1 Creating the Database

This section describes the steps necessary to create the database, tables and the DomiSMP database user (dbuser used for database connection purpose).

It also includes the creation of an initial DomiSMP user account that will be used by REST and UI clients to connect to the DomiSMP.

TIP	To find the script used for this step, see Database Scripts.

4.3.1. MySQL

1. Open a command prompt and navigate to the SETUP_PATH/database-scripts folder
2. Execute the following MySQL commands:

```
mysql -h localhost -u root_user -password=<root_password> -e \
"DROP SCHEMA IF EXISTS <smp_schema>;
CREATE SCHEMA <smp_schema>;
ALTER DATABASE <smp_schema> charset=utf8;
CREATE USER smp_dbuser@localhost IDENTIFIED BY 'smp_password';
GRANT ALL ON <smp_schema>.* TO smp_dbuser@localhost;"
```

This creates a <smp_schema> and an <smp_dbuser> with all privileges to the <smp_schema>.

3. Execute the following command to create the required objects (tables, etc.) in the database:

```
mysql -h localhost -u <root_user> --password=<root_password> <smp_schema> <
mysql.ddl
```

4. Execute the following command to fill initial test data:

```
mysql -h localhost -u root_user --password=<root_password> <smp_schema> < mysql-
data.sql
```

4.3.2. Oracle Database

1. Navigate to SETUP_PATH/database-scripts directory
2. Execute the following commands :

```
sqlplus sys as sysdba ①
```

Where:

① Password is the one defined during the Oracle installation.

3. Once logged in Oracle:

```
CREATE USER <smp_dbuser> IDENTIFIED BY <smp_dbpassword>;
GRANT ALL PRIVILEGES TO <smp_dbuser>;
CONNECT <smp_dbuser>;
SHOW USER; ①
@oracle.ddl ②
@oracle-data.ddl ③
exit
```

① Expected return: <smp_dbuser>.

② Run the scripts with the @ sign from the location of the scripts.

③ Fill initial test data.

4.4. STEP2 Configuring the Server

- Tomcat

4.4.1. Configuring Tomcat

To deploy the SMP on Tomcat, the steps below need to be completed.

Configuring Extra CLASSPATH

The purpose of the section is to describe how to set up the `smp` folder as a classpath on the Tomcat server. See also Deployment Overview.

Linux:

Edit the `CATALINA_HOME/bin/setenv.sh` file.

```
#!/bin/sh
# Set CLASSPATH to include $CATALINA_HOME/smp
# where the SMP's "smp.config.properties" file is located
export CLASSPATH=$CATALINA_HOME/smp
```

For Windows:

Edit the `%CATALINA_HOME%/bin/setenv.bat` file.

```
REM Set CLASSPATH to include $CATALINA_HOME/smp
REM where the SMP's "smp.config.properties" file is located
set classpath=%classpath%;%catalina_home%\smp
```

JDBC Driver

The JDBC driver needs to be downloaded from the manufacturer website:

- For Oracle Database:
  https://www.oracle.com/database/technologies/appdev/jdbc-downloads.html
- For Mysql:
  https://www.mysql.com/products/connector/

The JDBC driver (`.jar` file) must be copied to `AS_HOME/lib`.

4.5. STEP3 Configuring SMP

4.5.1. SMP Configuration Resources

The SMP configuration is performed in two different locations, the:

- `smp.config.properties` file
- `smp_configuration` table

▼ **Properties Overview**

DomiSMP 5.2 configuration has two types of properties:

- The system configuration properties: the properties are located in `smp.config.properties` file and define environment settings such as JDBC connection, logging configuration, SMP extension library folder, etc. Before the first eDelivery SMP startup, the mandatory database

connection properties must be set. The complete system property list is described in SMP Configuration Properties.

- The SMP application properties: the property list with default values is stored at initial startup in the database table **SMP_CONFIGURATION**. System administrators can change most properties during the runtime without application restart. The complete application property list is described in section SMP Application Configuration Properties.
- The `smp.config.properties` file must be copied to the CLASSPATH folder configured in
  - Configuring the Extra CLASSPATH **for Tomcat** or
- You can set a different value for a property for SMP's first startup. To do so, update the property in the `smp.config.properties` file.

TIP	Find a sample `smp.config.properties` file in the SMP Source Code Repository.

▼ **Multitenancy and Multidomain Support**

The SMP is able to support multiple certificates in the same SMP. This is very useful in the Acceptance environment where multiple domains like ISA ITB, eHealth and others are hosted.

The SMP has the capability of keeping a relationship between a particular **Service Group** and its related **domain**.

As a result of this feature, the SMP Administration has the option, if need be, to define extra domains for newly created **Service Groups** meaning that the SMP can handle multiple domains environments.

NOTE	In normal circumstances, when any one SMP is used for only one domain, the domain used is then considered as the "domain by default" (or "default domain") for configuration purposes. The domain, in this case, does not need to be specified in the **Service Group** definitions or other configurations of the SMP as in previous versions of SMP.

The SMP configuration is performed in two different locations: in the `smp.config.properties` file as well as in the `smp_configuration` table. The following section describes the details of the parameters that are included in the configuration.

4.5.2. Properties Configuration File

The initial eDelivery SMP configuration is performed via the `smp.config.properties` file. The file contains basic configuration for defining the database connection, logging file configuration and smp folder for deploying the extensions.

At startup, the application looks for `./smp.config.properties` in the classpath. This requires setting up the application server so that it includes the folder with the file in its classpath.

▼ *Sample of* `smp.config.properties` *file*

```
###
# #START_LICENSE#
```

```
# smp-webapp
# %%
# Copyright (C) 2017 - 2024 European Commission | eDelivery | DomiSMP
# %%
# Licensed under the EUPL, Version 1.2 or â as soon they will be approved by the
European Commission - subsequent
# versions of the EUPL (the "Licence");
# You may not use this work except in compliance with the Licence.
# You may obtain a copy of the Licence at:
#
# [PROJECT_HOME]\license\eupl-1.2\license.txt or
https://joinup.ec.europa.eu/collection/eupl/eupl-text-eupl-12
#
# Unless required by applicable law or agreed to in writing, software distributed
under the Licence is
# distributed on an "AS IS" basis, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND,
either express or implied.
# See the Licence for the specific language governing permissions and limitations
under the Licence.
# #END_LICENSE#


# IMPORTANT NOTE: These are sample values: be sure to update them to match your
specific setup.

# ********************************************************************************
# Database connection can be achieved using custom datasource configuration
# or reusing application server datasource.
# ********************************************************************************
## set database hibernate dialect
#smp.jdbc.hibernate.dialect=org.hibernate.dialect.OracleDialect
smp.jdbc.hibernate.dialect=org.hibernate.dialect.MySQLDialect

# *********************************
#  Custom defined datasource
# *********************************
# mysql database example. Please replace the values with your database configuration
and if possible use datasource JNDI configuration
# to avoid exposing the database credentials in the configuration file. Please see
the documentation on how to configure tomcat
# to use JNDI
# weblogic datasource JNDI example
# smp.datasource.jndi=jdbc/eDeliverySmpDs

# tomcat datasource JNDI example
smp.datasource.jndi=java:comp/env/jdbc/eDeliverySmpDs

# *********************************
# db configuration alternative
#smp.jdbc.driver=com.mysql.cj.jdbc.Driver
#smp.jdbc.url=jdbc:mysql://localhost:3306/smp
```

```
#smp.jdbc.user=<username>
#smp.jdbc.password=<password>

# Oracle database example
#smp.jdbc.driver=oracle.jdbc.driver.OracleDriver
#smp.jdbc.url=jdbc:oracle:thin:@localhost:1521/xe
#smp.jdbc.user=<username>
#smp.jdbc.password=<password>

# ********************************
# security folder
# ********************************
# smp.security.folder=./smp/

# ********************************
# Logging properties
# ********************************
# smp log folder
# smp.log.folder=../logs/

# custom logback configuration file
# smp.log.configuration.file=smp-logback.xml

# ********************************
# Extension folder
# ********************************
# path where SMP extensions are located. The Folder is loaded by the SMP classloader
at startup.
# smp.libraries.folder=./ext-lib


# ********************************
# Locale folder
# ********************************
# The locale folder contains the translations for the SMP web application.
# smp.locale.folder=./locales
```

SMP Configuration Properties

The `smp.config.properties` file is used to configure SMP's properties required by SMP at startup. For each release, a sample of this file is provided with the **DomiSMP Setup Bundle**, the `smp-5.2.setup.zip` archive.

SEE ALSO | See this SMP's release page to download the DomiSMP Setup Bundle archive.

▼ **SMP Configuration Properties List**

Configuration Property	Description and Usage	Default
`smp.configuration.file`	*Configuration property file path.*	`smp.config.properties`

Configuration Property	Description and Usage	Default
`smp.init.configuration.file`	*Init configuration property file path.*	`smp.init.properties`
`smp.security.folder`	*Security folder for storing the keystore and the truststore.*	`smp`
`smp.jdbc.driver`	*Database Configuration - Driver* • MySQL: `com.mysql.jdbc.Driver` • Oracle: `oracle.jdbc.OracleDriver`	`com.mysql.jdbc.Driver`
`smp.jdbc.url`	*Database Configuration - URL* • MySQL: `jdbc:mysql://dbhost:dbport/smp_database` • Oracle: ◦ `jdbc:oracle:thin:@dbhost:dbport:smp_database` ◦ `jdbc:oracle:thin:@dbhost:dbport/smp_service`	`jdbc:mysql://localhost:3306/smp`
`smp.jdbc.user`	*Database User/Password Configuration - User*	`smp`
`smp.jdbc.password`	*Database User/password Configuration - Password*	`The_password`
`smp.datasource.jndi`	*If the data source is configured on the application server (*recommended), the property defines the JNDI name of the database connection.*	`jdbc/eDeliverySmpDs`
`smp.database.show-sql`	*Print generated sql queries to logs. The property is effective only when* `smp.mode.development=true`.	`false`
`smp.database.create-ddl`	*Auto create/update database objects. The property is effective only when* `smp.mode.development=true`.	`false`

Configuration Property	Description and Usage	Default
`smp.log.folder`	**IMPORTANT** Do NOT this feature in production, it is only intended for tests, demonstrations and development purposes. *The provided logback.xml configuration defines logging file as* `<file>${log.folder:-logs}/edelivery-smp.log</file>` *With the property we can define the folder for the logging files.*	`/var/logs/smp`
`smp.log.configuration.file`	*Custom logback configuration file (filepath can be absolute or relative to smp configuration.dir).*	`/opt/logging/smp-logback.xml`
`smp.libraries.folder`	*Path where SMP extensions are located. The folder is loaded by the SMP classloader at startup.*	`/opt/smp/extension-libs`
`smp.smp.mode.development`	*The development mode uses semi-random generators for password and key generation. Setting the property value to 'true' makes the first startup and access token generation faster. To ensure high security, this option MUST NOT be enabled in production.*	`false`

SMP Application Configuration

eDelivery SMP Application configuration values are stored in the database table `SMP_CONFIGURATION`. If the table is empty (usually at first SMP startup), edelivery SMP populates the table at startup with all properties and default values.

When updating properties via the user interface, the property values are taken into account immediately if the server starts in non-cluster mode (property: `smp.cluster.enabled = false`).

Otherwise, each node refreshes the properties on all cluster nodes at the same time in accordance with the property refreshes defined in the CRON expression via the `smp.property.refresh.cronJobExpression` property.

▼ **SMP Application Configuration Properties List**

Configuration Property	Description and Usage	Default
`smp.instance.name`	*The name of the DomiSMP instance is used in email notifications and alerts to specify which SMP instance generated the notifications.* **Usage**: Requires restart: No Value type: STRING	`Test DomiSMP Instance`
`contextPath.output`	*This property controls pattern of URLs produced by SMP in GET ServiceGroup responses.* **Usage**: Requires restart: Yes Value type: BOOLEAN	`true`
`encodedSlashesAllowedInUrl`	*Allow encoded slashes in context path. Set to true if slashes are part of identifiers.* **Usage**: Requires restart: Yes Value type: BOOLEAN	`true`
`smp.http.forwarded.headers.enabled`	*Use (value true) or remove (value false) forwarded headers. There are security considerations for forwarded headers since an application cannot know if the headers were added by a proxy, as intended, or by a malicious client.* **Usage**: Requires restart: No Value type: BOOLEAN	`false`
`smp.http.httpStrictTransportSecurity.maxAge`	*How long (in seconds) should HSTS last in the browser cache (default one year).* **Usage**: Requires restart: Yes Value type: INTEGER	`31536000`

Configuration Property	Description and Usage	Default

`smp.http.header.security.policy`	*Content Security Policy (CSP)*
	
	`default-src 'self';`
	`script-src 'self';`
	`connect-src 'self';`
	`img-src 'self';`
	`style-src 'self'`
	`'unsafe-inline';`
	`frame-ancestors 'self';`
	`form-action 'self';`
	
	Usage:
	Requires restart: Yes
	Value type: STRING
		-

Configuration Property	Description and Usage	Default
`smp.proxy.host`	*The http proxy host.* **Usage**: Requires restart: No Value type: STRING	-
`smp.noproxy.hosts`	*List of nor proxy hosts.* **Usage**: Requires restart: No Value type: STRING **Default**: `localhost|127.0.0.1`	See Description
`smp.proxy.password`	*Base64 encrypted password for Proxy.* **Usage**: Requires restart: No Value type: STRING	-
`smp.proxy.port`	*The http proxy port.* **Usage**: Requires restart: No Value type: INTEGER	80
`smp.proxy.user`	*The proxy user.* **Usage**: Requires restart: No Value type: STRING	-

Configuration Property	Description and Usage	Default

Configuration Property	Description and Usage	Default
identifiersBehaviour.ParticipantIdentifierScheme.validationRegex	*Participant Identifier Schema of each PUT ServiceGroup request is validated against this schema.* **Usage**: Requires restart: No Value type: REGEXP **Default**: `^|(?!^.{26})(-[a-z0-9]-)$|^urn:oasis:names:tc:ebcore:partyid-type:(iso6523|unregistered)(:.)?$`	See Description
identifiersBehaviour.ParticipantIdentifierScheme.validationRegexMessage	*Error message for UI.* **Usage**: Requires restart: No Value type: STRING **Value Format**: Participant scheme must start with `urn:oasis:names:tc:ebcore:partyid-type:(iso6523:|unregistered:)` *OR* • must be up to 25 characters long with form `[domain]-[identifierArea]-[identifierType]` • and may only contain the following characters: [a-z0-9]. **Example**: `busdox-actorid-upis`	-
identifiersBehaviour.scheme.mandatory	*Scheme for participant identifier is mandatory.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
Configuration Property	**Description and Usage**	**Default**
smp.ui.session.idle_timeout.admin	*Specifies the time, in seconds, between client requests before the SMP will invalidate session for ADMIN users (System).* **Usage**: Requires restart: No Value type: INTEGER	`300`

Configuration Property	Description and Usage	Default
smp.ui.session.idle_timeout.user	*Specifies the time, in seconds, between client requests before the SMP will invalidate session for users (Service group, SMP Admin).* **Usage**: Requires restart: No Value type: INTEGER	1800
smp.cluster.enabled	*Define if application is set in cluster. In not cluster environment, properties are updated on set Property.* **Usage**: Requires restart: No Value type: BOOLEAN	false
Configuration Property	**Description and Usage**	**Default**
smp.credentials.reset_request.url	*The URL address for resetting the password in DomiSMP. This URL can be used when DomiSMP is behind a reverse proxy. If the property value is not set, the reset URL is created using reverse proxy headers such as Host and X-Forwarded-Host.* **Usage**: Requires restart: No Value type: URL	-
smp.credentials.reset_request.url.validMinutes	*The number of minutes for reset token to be valid.* **Usage**: Requires restart: No Value type: INTEGER	90
smp.passwordPolicy.validationRegex	*Password minimum complexity rules.* **Usage**: Requires restart: No Value type: REGEXP **Default**: `^(?=.*[0-9])(?=.*[a-z])(?=.*[A-Z])(?=.*[~`!@#$%^&+=\-_<>.,?:;*/()\|\[\]\{}'"\\]).\{16,32}$`	See Description

Configuration Property	Description and Usage	Default
`smp.passwordPolicy.validationMessage`	*The error message shown to the user in case the password does not follow the regex put in the* `smp.passwordPolicy.pattern property`. **Usage**: Requires restart: No Value type: STRING Must have: • Minimum length: 16 characters. • Maximum length: 32 characters. • At least one letter in lowercase; • At least one letter in uppercase; • At least one digit; • At least one special character.	-
`smp.passwordPolicy.validDays`	*Number of days password is valid.* **Usage**: Requires restart: No Value type: INTEGER	`90`
`smp.passwordPolicy.warning.beforeExpiration`	*How many days before expiration should the UI warn users at login.* **Usage**: Requires restart: No Value type: INTEGER	`15`
`smp.passwordPolicy.expired.forceChange`	*Force change password at UI login if expired.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
Configuration Property	**Description and Usage**	**Default**
`smp.user.login.fail.delay`	*Delay response in ms on invalid username or password.* **Usage**: Requires restart: No Value type: INTEGER	1000

Configuration Property	Description and Usage	Default
smp.user.login.maximum.attempt	*Number of console login attempt before the user is deactivated.* **Usage**: Requires restart: No Value type: INTEGER	5
smp.user.login.suspension.time	*Time in seconds for a suspended user to be reactivated.* **Usage**: Requires restart: No Value type: INTEGER • If set to 0, the user will not be reactivated.	3600

Configuration Property	Description and Usage	Default
smp.accessToken.validDays	*Number of days access token is valid.* **Usage**: Requires restart: No Value type: INTEGER	60
smp.accessToken.login.maximum.attempt	*Number of accessToken login attempt before the accessToken is deactivated.* **Usage**: Requires restart: No Value type: INTEGER	10
smp.accessToken.login.suspension.time	*Time in seconds for a suspended accessToken to be reactivated.* **Usage**: Requires restart: No Value type: INTEGER • If set to 0, the user will not be reactivated.	3600
smp.accessToken.login.fail.delay	*Delay in ms on invalid token id or token.* **Usage**: Requires restart: No Value type: INTEGER	1000
smp.ui.authentication.types	*Set list of '|' separated authentication types: PASSWORD|SSO.* **Usage**: Requires restart: No Value type: LIST_STRING	PASSWORD

Configuration Property	Description and Usage	Default
smp.automation.authentication.types	*Set list of "|" separated application-automation authentication types (Web-Service integration).* **Usage**: Requires restart: No Value type: LIST_STRING Supported Values: TOKEN, CERTIFICATE. **Default**: `TOKEN|CERTIFICATE`	See Description
smp.automation.authentication.external.tls.clientCert.enabled	*Authentication with external module as: reverse proxy. Authenticated data are sent to application using 'Client-Cert' HTTP header. Do not enable this feature without a properly configured reverse-proxy.* **Usage**: Requires restart: No Value type: BOOLEAN	false
smp.automation.authentication.external.tls.SSLClientCert.enabled	*Authentication with external module as: reverse proxy. Authenticated certificate is sent to application using SSLClientCert HTTP header. Do not enable this feature without properly a configured reverse-proxy.* **Usage**: Requires restart: No Value type: BOOLEAN	false

Configuration Property	Description and Usage	Default
smp.sso.cas.ui.label	*The SSO service provider label.* **Usage**: Requires restart: Yes Value type: STRING	EU Login

Configuration Property	Description and Usage	Default
smp.sso.cas.url	*The SSO CAS URL endpoint.* **Usage**: Requires restart: Yes Value type: URL **Default**: `http://localhost:8080/cas/`	See Description
smp.sso.cas.urlPath.login	*The CAS URL path for login.* *The complete URL is composed by parameters:* `${smp.sso.cas.url}/${smp.sso.cas.urlpath.login}.` **Usage**: Requires restart: Yes Value type: STRING	login
smp.sso.cas.callback.url	*The URL is the callback URL belonging to the local SMP Security System. If using RP,make sure it target SMP path* /ui/public/rest/security/cas. **Usage**: Requires restart: Yes Value type: URL **Default**: http://localhost:8080/smp/ui/public/rest/security/cas	See Description
smp.sso.cas.smp.urlPath	*SMP relative path which triggers CAS authentication.* **Usage**: Requires restart: Yes Value type: STRING **Default**: /smp/ui/public/rest/security/cas	See Description

Configuration Property	Description and Usage	Default

`smp.sso.cas.smp.user.data.urlPaths`	*Relative path for CAS user data. Complete URL is composed by parameters:*
	
	`${smp.sso.cas.url}/${smp.sso.cas.smp.user.data.urlpath}`
	
	Usage:
	Requires restart: Yes
	Value type: STRING
	
	Default:
	`userdata/myAccount.cgi`
		See Description

`smp.sso.cas.token.validation.urlPath`	The CAS URL path for login. Complete URL is composed of parameters: `${smp.sso.cas.url}/${smp.sso.cas.token.validation.urlpath}` **Usage**: Requires restart: Yes Value type: STRING **Default**: `laxValidate`	See Description
`smp.sso.cas.token.validation.params`	*The CAS token validation key:value properties separated with a pipe (`|`).* **Usage**: Requires restart: Yes Value type: MAP_STRING **Default**: `acceptStrengths:BASIC,CLIENT_CERT|assuranceLevel:TOP`	See Description

Configuration Property	Description and Usage	Default
smp.sso.cas.token.validation.groups	*Pipe-separated (|) CAS groups user must belong to.* **Usage**: Requires restart: Yes Value type: LIST_STRING **Default**: `DIGIT_SMP|DIGIT_ADMIN`	See Description
smp.sso.cas.registration.enabled	*If the value is set to true, the user is automatically registered to DomiSMP the first time they use the external CAS. The CAS server provides the necessary user data, which is then mapped to the DomiSMP user entity according to the smp.sso.cas.registration.mapping.* **Usage**: Requires restart: No Value type: BOOLEAN	true
smp.sso.cas.registration.confirmation.mandatory	*The value determines whether the CAS-automatically created user is activated immediately or if the System admin must activate the user before they can log in to DomiSMP.* **Usage**: Requires restart: No Value type: BOOLEAN	false
smp.sso.cas.registration.mapping	*Pipe-separated (|) key:value list of mapping defining how CAS user data is mapped to DomiSMP user entity. Currently supported values are: EMAIL and FULL_NAME. The username of the newly created user is the CAS principal name/identifier* **Usage**: Requires restart: No Value type: MAP_STRING **Default**: `EMAIL:${email}\|FULL_NAME:${firstName} ${lastName}`	See Description

Configuration Property	Description and Usage	Default
Configuration Property	**Description and Usage**	**Default**
`mail.smtp.host`	Email configuration: *Email server.* **Usage**: Requires restart: No Value type: STRING	-
`mail.smtp.port`	Email configuration: *SMTP mail port.* **Usage**: Requires restart: No Value type: INTEGER	`25`
`mail.smtp.protocol`	Email configuration: *SMTP mail protocol.* **Usage**: Requires restart: No Value type: STRING	smtp
`mail.smtp.username`	Email configuration: *SMTP mail protocol; mail sender's username.* **Usage**: Requires restart: No Value type: STRING	-
`mail.smtp.password`	Email configuration: _SMTP mail protocol; mail sender's encrypted password. **Usage**: Requires restart: No Value type: STRING	-
`mail.smtp.properties`	*Pipe-separated (`|`) key:value properties list.* **Usage**: Requires restart: No Value type: MAP_STRING **Example**: `mail.smtp.auth:true|mail.smtp.starttls.enable:true|mail.smtp.quitwait:false`	-
Configuration Property	**Description and Usage**	**Default**

Configuration Property	Description and Usage	Default
`smp.alert.user.created.enabled`	*Enable or disable notifications for user creation events.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.user.created.level`	*User creation event notification alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`HIGH`
`smp.alert.user.updated.enabled`	*Enable or disable notifications when user data is changed.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.user.updated.level`	*User update data event notification alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`HIGH`
`smp.alert.user.login_failure.enabled`	*Enable/disable the login failure alert of the authentication module.* **Usage**: Requires restart: No Value type: BOOLEAN	`false`
`smp.alert.user.login_failure.level`	*Login failure alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`LOW`
`smp.alert.user.suspended.enabled`	*Enable/disable the login suspended alert of the authentication module.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`

Configuration Property	Description and Usage	Default
`smp.alert.user.suspended.level`	*Suspended login alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`HIGH`
`smp.alert.user.suspended.mail.moment`	*When should the account disabled alert be triggered.* **Usage**: Requires restart: No Value type: STRING Possible values: • `AT_LOGON`: if set, an alert is triggered each time a user tries to log into a disabled account. • `WHEN_BLOCKED`: if set, an alert is triggered when the account is suspended.	`WHEN_BLOCKED`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.password.imminent_expiration.enabled`	*Enable/disable the "Password about to expire" alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.password.imminent_expiration.delay_days`	*Number of days before password expiration the system is to send alerts.* **Usage**: Requires restart: No Value type: INTEGER	`15`
`smp.alert.password.imminent_expiration.frequency_days`	*Frequency in days for (re)sending the "Password about to expire" alert.* **Usage**: Requires restart: No Value type: INTEGER	`5`
`smp.alert.password.imminent_expiration.level`	*"Password about to expire" alert's level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`LOW`

Configuration Property	Description and Usage	Default
Configuration Property	**Description and Usage**	**Default**
`smp.alert.password.expired.enabled`	*Enable/disable the "Password expired" alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.password.expired.delay_days`	*Period in days after password expiration the system is to send "password expiration" alerts.* **Usage**: Requires restart: No Value type: INTEGER	`30`
`smp.alert.password.expired.frequency_days`	*Frequency in days between "Password expired" alerts.* **Usage**: Requires restart: No Value type: INTEGER	`5`
`smp.alert.password.expired.level`	*"Password expired" alert's level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH.	`LOW`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.accessToken.imminent_expiration.enabled`	*Enable/disable the "accessToken about to expire" alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.accessToken.imminent_expiration.delay_days`	*Number of days before password expiration the system is to send alerts.* **Usage**: Requires restart: No Value type: INTEGER	15
`smp.alert.accessToken.imminent_expiration.frequency_days`	*Frequency in days between alerts.* **Usage**: Requires restart: No Value type: INTEGER	`5`

Configuration Property	Description and Usage	Default
`smp.alert.accessToken.imminent_expiration.level`	*AccessToken imminent expiration alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH.	`LOW`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.accessToken.expired.enabled`	*Enable/disable the accessToken expiration alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.accessToken.expired.delay_days`	*Number of days after expiration as for how long the system should send alerts.* **Usage**: Requires restart: No Value type: INTEGER	`30`
`smp.alert.accessToken.expired.frequency_days`	*Frequency in days between alerts.* **Usage**: Requires restart: No Value type: INTEGER	`30`
`smp.alert.accessToken.expired.level`	*Access Token expiration alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH.	`LOW`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.certificate.imminent_expiration.enabled`	*Enable/disable the imminent certificate expiration alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.certificate.imminent_expiration.delay_days`	*Number of days before expiration as for how long before expiration the system should send alerts.* **Usage**: Requires restart: No Value type: INTEGER	`15`

Configuration Property	Description and Usage	Default
`smp.alert.certificate.imminent_expiration.frequency_days`	*Frequency in days between alerts.* **Usage**: Requires restart: No Value type: INTEGER	`5`
`smp.alert.certificate.imminent_expiration.level`	*Certificate imminent expiration alert level. Values: \{LOW, MEDIUM, HIGH}* **Usage**: Requires restart: No Value type: STRING	`LOW`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.certificate.expired.enabled`	*Enable/disable the certificate expiration alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.certificate.expired.delay_days`	*Number of days after expiration as for how long the system should send alerts.* **Usage**: Requires restart: No Value type: INTEGER	`30`
`smp.alert.certificate.expired.frequency_days`	*Frequency in days between alerts.* **Usage**: Requires restart: No Value type: INTEGER	`5`
`smp.alert.certificate.expired.level`	*Certificate expiration alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`LOW`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.system.certificate.imminent_expiration.enabled`	*Enable/disable the imminent system certificate expiration alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`

Configuration Property	Description and Usage	Default
`smp.alert.system.certificate.imminent_expiration.delay_days`	*Number of days the system is to send alerts before expiration occurs.* **Usage**: Requires restart: No Value type: INTEGER	`15`
`smp.alert.system.certificate.imminent_expiration.frequency_days`	*Period in days between alerts.* **Usage**: Requires restart: No Value type: INTEGER	`5`
`smp.alert.system.certificate.imminent_expiration.level`	*System certificate imminent expiration alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`LOW`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.system.certificate.expired.enabled`	*Enable/disable the system certificate expiration alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.system.certificate.expired.delay_days`	*Number of days the system is to send alerts after expiration occurs. _* **Usage**: Requires restart: No Value type: INTEGER	`30`
`smp.alert.system.certificate.expired.frequency_days`	*Frequency in days between alerts.* **Usage**: Requires restart: No Value type: INTEGER	`5`
`smp.alert.system.certificate.expired.level`	*System certificate expiration alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`LOW`
Configuration Property	**Description and Usage**	**Default**

Configuration Property	Description and Usage	Default
`smp.alert.credentials.cronJobExpression`	*CRON expression specifying schedule for triggering alert messages about credentials.* **Usage**: Requires restart: No Value type: CRON_EXPRESSION **Default**: `0 52 4 */1 * *`	See Description
`smp.alert.system.certificates.cronJobExpression`	*CRON expression specifying schedule for triggering alert messages about system certificates.* **Usage**: Requires restart: No Value type: CRON_EXPRESSION **Default**: `0 42 4 */1 * *`	See Description
`smp.alert.credentials.serverInstance`	*Which instance (hostname) to generates a report when* `smp.cluster.enabled` *is set to true.* **Usage**: Requires restart: No Value type: STRING	`localhost`
`smp.alert.credentials.batch.size`	Max alerts generated in a batch for the type. **Usage**: Requires restart: No Value type: INTEGER	`200`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.mail.from`	*Alert send mail.* **Usage**: Requires restart: No Value type: EMAIL **Default**: `test@alert-send-mail.eu`	See Description

Configuration Property	Description and Usage	Default
smp.domain.default	*Default domain code.* **Usage**: Requires restart: No Value type: STRING • If the domain cannot be determined from the request, the default domain is used.	-
Configuration Property	**Description and Usage**	**Default**
smp.certificate.validation.allowed.certificate.types	*Allowed user certificate types.* **Usage**: Requires restart: No Value type: LIST_STRING **Example**: `RSA|EC|Ed25519|Ed448` • If empty no restrictions are imposed. • For other values see the java KeyFactory Algorithms.	-
authentication.blueCoat.enabled	**NOTE** Property was replaced by property: smp.automation.authentication.external.tls.clientCert.enabled **Usage**: Requires restart: No Value type: BOOLEAN	false
smp.domain.default	*Default domain code. If the domain cannot be determined from the request, the default domain is used.* **Usage**: Requires restart: No Value type: STRING	-

Configuration Property	Description and Usage	Default
`smp.certificate.validation.allowed.certificate.types`	*Allowed user certificate types.* **Usage**: Requires restart: No Value type: LIST_STRING **Example**: `RSA|EC|Ed25519|Ed448` • If empty no restrictions are imposed. • For other values see the java KeyFactory Algorithms.	-
`smp.authorization.jwt.issuer`	*JWT issuer used for validating the token. The "issuer" in a JWT is the principal which issued the token. It's represented by the registered claim name "iss" and is used by token consumers to verify who created the token.* **Usage**: Requires restart: Yes Value type: STRING **Example**: `https://auth.example.com` • If empty no restrictions are imposed.	-
`smp.authorization.jwt.audience`	*JWT audience used for validating the token. The JWT "audience" (claim name "aud") identifies the intended recipient(s) of the token, i.e., who the token is meant for.* **Usage**: Requires restart: Yes Value type: STRING **Example**: `my-api-v1` • If empty no restrictions are imposed.	-

Configuration Property	Description and Usage	Default
smp.authorization.jwt.key	*JWT public key used to verify the signature of the JWT token.* **Usage**: Requires restart: Yes Value type (Base 64 encoded key): STRING **Example**: `my-api-v1`	-
smp.authorization.jwt.algorithm	*Expected JWT algorithm used in a signature of the JWT token. Default is PS256* Supported algorithms: **RS256** — RSASSA-PKCS1-v1_5 using SHA-256 **RS384** — RSASSA-PKCS1-v1_5 using SHA-384 **RS512** — RSASSA-PKCS1-v1_5 using SHA-512 **PS256** — RSASSA-PSS using SHA-256 **PS384** — RSASSA-PSS using SHA-384 **PS512** — RSASSA-PSS using SHA-512 **ES256** — ECDSA using curve P-256 and SHA-256 **ES384** — ECDSA using curve P-384 and SHA-384 **ES512** — ECDSA using curve P-521 and SHA-512 **EdDSA** — EDDSA using curve Ed25519 [1] [1] ED448 is not supported. **Usage**: Requires restart: Yes Value type: STRING **Example**: `PS256`	-

Configuration Property	Description and Usage	Default
`vault.enabled`	*Enable/Disable Vault integration for sensitive properties* **Usage**: Requires restart: No Value type: BOOLEAN **Example**: `true`	-
`smp.vault.implementation.classname`	*Full class name of the Vault implementation-* **Usage**: Requires restart: No Value type: STRING **Example**: `eu.europa.ec.edelivery.vault.MyVault`	-
`smp.vault.configuration`	*List of vault properties separated by `;`.* **Usage**: Requires restart: No Value type: STRING **Example**: `hashicorp-vault.url:http://vault-service:8200/;hashicorp-vault.token:domisml-valut-test-token`	-

Configuration Property	Description and Usage	Default
`smp.vault.configuration`	*List of vault properties, separated by semicolons (;). Key names vary depending on the vault instance type. For a complete list of supported properties, refer to the specific vault's implementation.* **Usage**: Requires restart: No Value type: STRING **Example**: `hashicorp-vault.url:http://vault-service:8200/;hashicorp-vault.token:domisml-valut-test-token`	-
`smp.vault.authentication.type`	*Specifies the authentication type, such as token or username. The accepted values depend on the vault implementation.* **Usage**: Requires restart: No Value type: STRING **Example**: `token`	-
`smp.vault.authentication.value`	*Authentication value and format depends on the vault implementation and authentication type.* **Usage**: Requires restart: No Value type: STRING **Example**: `my-valult-token`	-

4.5.3. Configuration Table

The `smp_domain` table is used to support the multitenancy feature of the SMP.
Its parameters/fields are:

▼ `smp_domain` **table fields**

SML_SMP_ID

This is the SMP ID that must match the SMP ID registered within the SML.

SML_CLIENT_CERT_HEADER

The SMP's certificate - needed only when accessing BDMSL directly through HTTP. The configured "Client-Cert" HTTP header will be added to each BDMSL request (bypassing SSL certificate verification made normally by SSL terminator).

SML_CLIENT_KEY_ALIAS

This is the Domain scoped alias of the keystore private key used for authentication with the SML. The password is the same as `xmldsig.keystore.password` defined in the SMP configuration file.

SIGNATURE_KEY_ALIAS

This field points to the **Domain scoped** alias of the Keystore private key certificate, used by the SMP to sign GET Signed Service Metadata responses.

SML_SUBDOMAIN

This is the informative identifier of SML domain code (eHealth, Peppol, etc). Since SML subdomain is part of DNS domain it must be a valid DNS domain part.

DOMAIN_CODE

The unique domain code that is used as HTTP domain parameter when adding participants true REST service API to particular a domain. Domain code can be alphanumeric and up to 63 characters long.

Example

Update the default single domain smp_domain table record:

```
UPDATE smp_domain SET SML_SMP_ID='SMP-MCB-ID14',
SML_CLIENT_KEY_ALIAS= 'smp_mock';
```

or

```
UPDATE smp_domain SET SML_SMP_ID='SMP-MCB-ID14',
SML_CLIENT_CERT_HEADER=
'serial=000000000000000000009A195D2DD88C&subject=CN=SMP_1000000000,O=DG-
DIGIT,C=BE&validFrom=Oct 21 02:00:00 2014 CEST&validTo=Oct 21 01:59:59 2016
CEST&issuer=CN=Issuer
Common Name,OU=Issuer Organization Unit,O=Issuer Organization,C=BE'
WHERE domainId='default';
```

4.5.4. Database configuration

The eDelivery SMP database back-end configuration is performed within the eDelivery SMP configuration file (**smp.config.properties** file).

Depending on the selected database back-end, modify the **smp.config.properties** files as indicated below. SMP database's connection can be configured in the properties file, or it can use application server datasource configuration by JNDI.

▼ **Oracle**

- Datasource configured from property file:

```
../
## Sample for Oracle

jdbc.driver=oracle.jdbc.driver.OracleDriver
jdbc.url=jdbc:oracle:thin:@localhost:1521/xe
jdbc.user=smp
jdbc.password=secret123
hibernate.dialect=org.hibernate.dialect.Oracle10gDialect

/..
```

- Datasource (connection pool) configured on the application server using the JNDI (recommended):

```
../
hibernate.dialect=org.hibernate.dialect.Oracle10gDialect

# Tomcat datasource JNDI example
datasource.jndi=java:comp/env/jdbc/edeliverySmpDS
```

▼ **MySQL**

- Datasource configured from property file:

```
../
## Database access

# For mysql connector v8
#jdbc.driver=com.mysql.cj.jdbc.Driver

# For mysql connector v5
jdbc.driver=com.mysql.jdbc.Driver jdbc.url=jdbc:mysql://localhost:3306/smp
jdbc.user=smp
jdbc.password=secret123
hibernate.dialect =org.hibernate.dialect.MySQLDialect
/..
```

- Datasource (connection pool) configured on the application server using the JNDI (recommended):

```
../
hibernate.dialect=org.hibernate.dialect.Oracle10gDialect

# Tomcat datasource JNDI example
datasource.jndi=java:comp/env/jdbc/edeliverySmpDS
/..
```

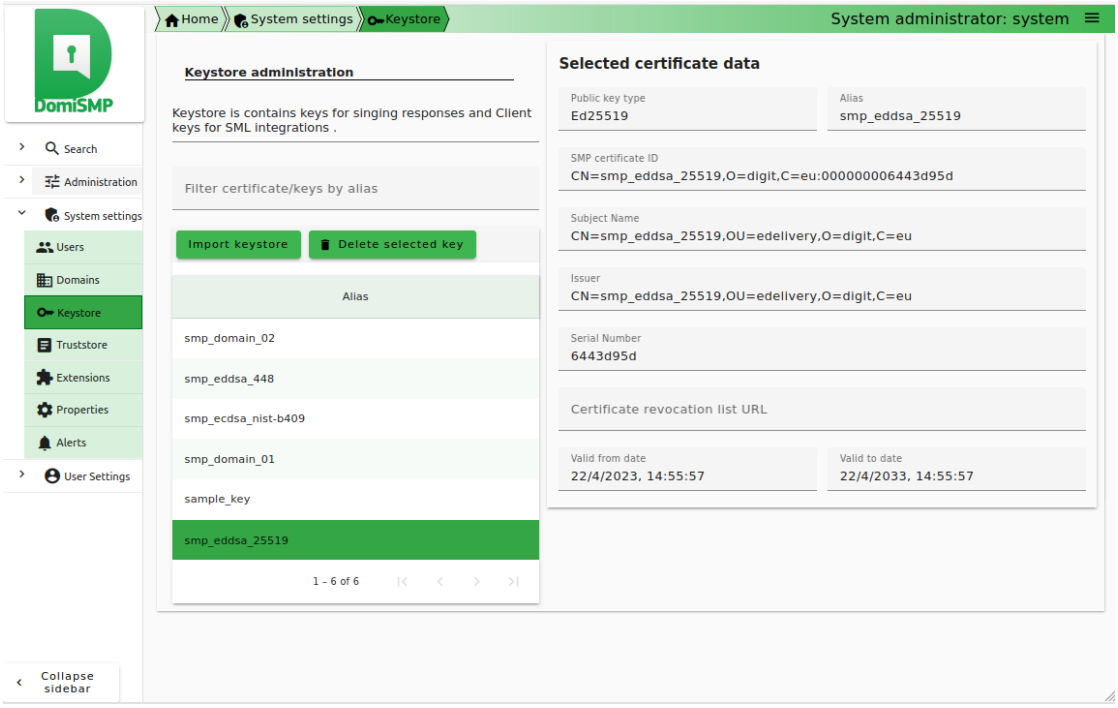
4.5.5. SMP Keystore

eDelivery SMP uses keystore for storing keys for the two different purposes:

- One **mandatory** key is used for signing the responses to **GET** requests (XMLDSIG response signing).
- One **optional** key is used to authenticate SMP using 2-way-SSL when it is calling SML via HTTPS.

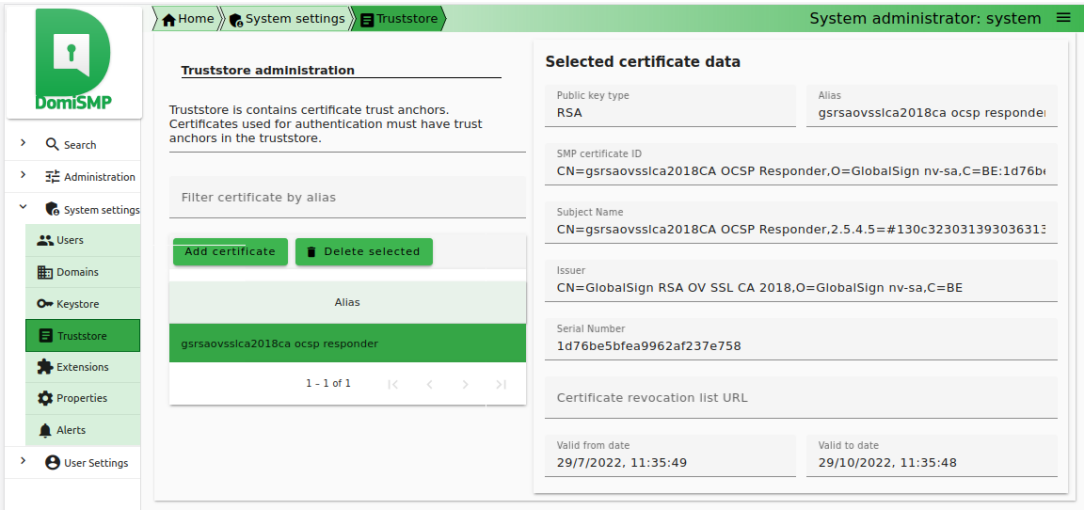
If the Keystore does not exist when the SMP is started for the first time, it is automatically created with a sample key/certificate 'sample_key'.

The user with a system administrator role can update/manage the Keystore entries using the user interface on the
System settings/Keystore page:



4.5.6. SMP Truststore

eDelivery SMP uses truststore for storing trusted X509Certificates for the WebService 2-way-SSL authentication and for storing the SML server certificate. The truststore is automatically created at the initial SMP start-up. The truststore can be managed with a System admin account using the UI tools under the page System settings / Truststore.



4.5.7. Custom Keystore and Truststore

On some systems, generating new passwords and keys can take a long time. To speed up the initial startup, consider the following option:

- Install faster system random generators.
- In case of development or local testing, set property: **smp.mode.development=true** to **smp.config.properties.** To ensure high security, this option MUST NOT be enabled in production.
- Use custom/preprepared keystore and truststore as described below.

Users can configure eDelivery SMP to use prepared keystores at initial startup. To achieve this, the Keystore must be generated manually and saved in the SMP security folder. If the Keystore already contains the keys/certificates, they must have the same Key password as it was set for accessing the Keystore.

The following properties must be set in `smp.config.properties`:

- `smp.security.folder`: SMP's security folder, where the keystore must be located.
- `smp.keystore.filename`: Keystore's filename.
- `smp.keystore.password`: Password for accessing the keystore and the keys.
- `smp.truststore.filename`: Truststore's filename.
- `smp.truststore.password`: Password for accessing the Truststore.

NOTE	Decrypted passwords must be wrapped in `{DEC}\{[PASSWORD]}`;. *Example:* `{DEC}{testPASSkeystore1234}`.

```
../
smp.security.folder=/opt/tomcat/security/

smp.keystore.filename=smp-keystore.p12
smp.keystore.password={DEC}{testPASSkeystore1234}
smp.keystore.type=PKCS12
```

```
smp.truststore.filename=smp-keystore.p12
smp.truststore.password={DEC}{testPASStruststore1234}
smp.truststore.type=PKCS12
/..
```

- After initial startup, the properties are stored (and the password encrypted) inside the `SMP_CONFIGURATION` table, and they should be removed from the `smp.config.properties` file.

4.6. STEP4 Deploying SMP Application

SMP .war file Deployment

The eDelivery SMP is deployed using the steps described in the next sections.

4.6.1. Tomcat

Download and copy `smp-X.war` file in the Tomcat `/webapps` directory (`AS_HOME/webapps/smp.war`).

NOTE

The application's context path is the same as the name of the file. For example:

- `smp.war`, the SMP application is available from the URL: `http://localhost:8080/smp/`.
- `smp-X.war`, the SMP application is available from the URL: `http://localhost:8080/smp-X/`.

NOTE

When TLS is configured in the application server, the URL scheme should be set to **https**. For example:

- `smp.war`, the SMP application is available from the URL: `https://localhost:8080/smp/`.

Installation Verification

Verify the installation by navigating in your browser: `http://<hostname>:<port>/smp`.

If the deployment was successful, the following page is displayed:

4.7. Configuring SMP/BDMSL integration

Configuring the eDelivery SMP for use with an BDMSL

The eDelivery SMP can establish an BDMSL integration using two identification mechanisms:

- **Using HTTP and plain text with metadata embedded** into the HTTP header Client-Cert of the REST request Using HTTP and plain text with metadata embedded into the HTTP header Client-Cert of the REST request.

IMPORTANT	This approach should **be used only for testing purposes** and only if both BDMSL and eDelivery SMP are located in the same network where the BDMSL web services are **not** exposed to the internet.

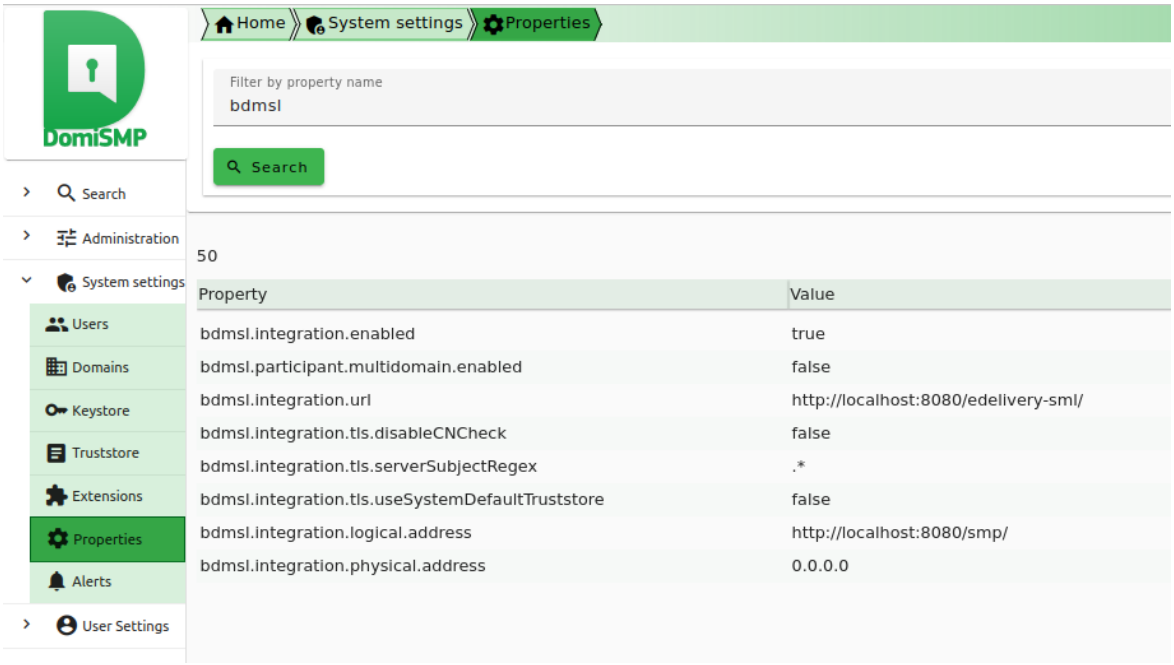
- **Using 2-way HTTPS/TLS** (*Recommended*).

The BDMSL integration configuration has two parts:

- Configuration of the BDMSL integration data as: BDMLS URL, SMPs URL, etc.
- Configuration of the SMP domain credentials/X509Certificate and unique SMP identifier.

4.7.1. Configuring BDSML

The BDMSL integration data can be set using the UI Property tool:



▼ **BDMSL Configuration Properties**

To configure BDMSL, set the following properties:

- `bdmsl.integration.enabled`: set value to true to enable BDMSL (SML) integration.
- `bdmsl.integration.url`: set the URL where BDMSL is located.
  Example: https://acc.edelivery.tech.ec.europa.eu/edelivery-sml/
- `bdmsl.integration.logical.address`: set the public SMP URL address. The URL is used by the BDMSL when generating DNS records for the SMP. Do not change this property once the SMP domain is registered to BDMSL. Example: `https://smp.domain.eu/smp`.
- `bdmsl.integration.physical.address`: IP4 address of the SMP server. The value is informative and can be `0.0.0.0`.

▼ **2-Way TLS Authentication Configuration Properties**

If using the 2-Way TLS authentication, configure:

- `bdmsl.integration.tls.disableCNCheck`: if set to `true`, the BDMSL server domain and Certificate CN value must match with the BDMSL certificate to be trusted.
- `bdmsl.integration.tls.useSystemDefaultTruststore`: if set to `true`, the system default truststore is used to verify the BDMSL truststore. The system default truststore usually points to the `$JAVA_HOME/lib/security/cacerts` truststore, or is configured on the application server using the `javax.net.ssl.trustStore` system parameter. If the property is set to false, the SMP truststore is used to verify the BDMSL server certificate trust.
- `bdmsl.integration.tls.serverSubjectRegex`: regular expression for BDMSL server TLS certificate subject verification.
  Example: CertEx. .`CN=acc.edelivery.tech.ec.europa.eu.`.

4.7.2. Configuring SMP domain credentials

Once BDMSL integration data is configured, the next step is to configure the SMP client certificate and ID for the BDMSL authentication. Because SMP 4.2 can handle multiple domains, each domain

can have its X509Certificate to log into the correct BDMSL DNS domain.

To configure the SMP domain credentials:

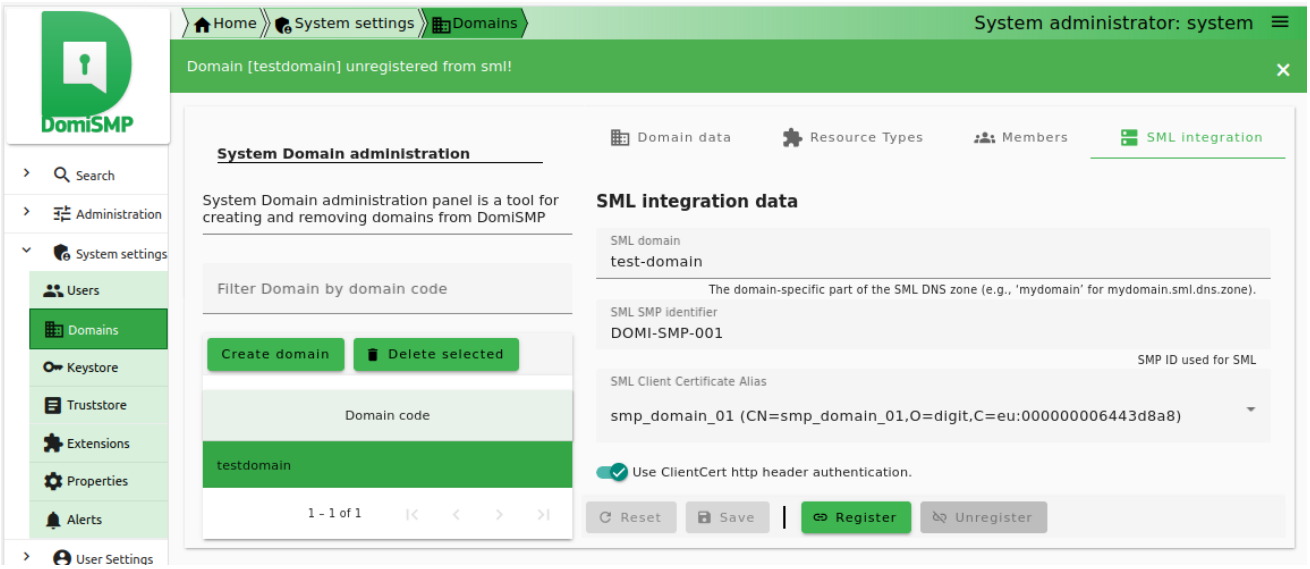
1. Register BDMSL client key/certificate to the SMP Keystore.
2. Create or edit Domain in the UI/Domain tool, enter the SMP ID and choose the client certificate.
3. Choose the authentication type. SML supports two ways of authentication.

ClientCert

HTTP Client-Cert certificate header. This must be used only behind a reverse proxy. The BDMSL should NOT allow this type of authentication from the internet. In practice, the HTTP Client-Cert should be generated only by the reverse proxy.

HTTPS/TLS

Standard mutual TLS authentication (*Recommended*).



4.8. SMP User Management

The DomiSMP has two user application roles:

- System Admin: the role allows user to modify DomiSMP system management and settings such as: Domain management, User management, Truststore management, Key management, DomiSMP configuration, etc.
- User: this role allows the user to log into the DomiSMP.

The user can have additional permissions on editing DomiSMP entities when they are assigned to the Resource, Groups, and Domains. Please read the following chapter for more details.

4.8.1. Domain, Group and Resources

The DomiSMP supports 3-layer security realms.

Resource

the most basic unit.

The Resource is identified by the unique ID, which is part of the URL of the resource as example:

`http://localhost/smp/resource-identifier`

A Resource example is the *Service Group* document from the Oasis SMP specification. The user can be a Resource member with **Admin** or **Viewer** membership roles. If the user has an Admin membership role, it can modify resource document(s) and manage the resource memberships. If the user has role Viewer, it can view/read the Resource if the Resource has visibility set to: `Private`.

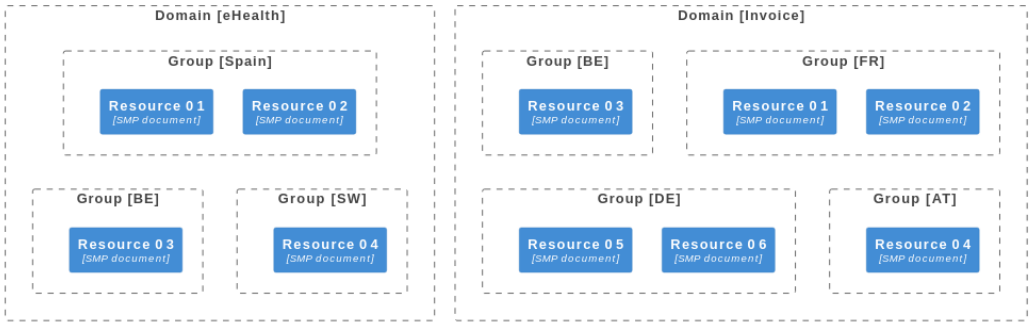
Group

a cluster of resources managed by the dedicated group administrators.
The group admin(s) can create and delete the resource, but *only* the resource admins can modify data/documents for the resource. The user can be a Group member with **Admin** or **Viewer** membership roles. With Admin group membership, the user can create and delete group resources. If the user has group role Viewer, it can view/read the Resources if the Group has visibility set to: “Private”.

Domain

the top layer.
It indicates the business purpose of the network of participants, such as invoice exchange, Health Records message exchanges, etc. The Domain usually has a domain owner who handles participant interoperability, defining message types, network authentication, and authorization methods such as Certificate PKI, Identity Service providers, etc.

In DomiSMP 5.0, the user with a Domain Admin role can create domain groups and assign users to them.



The provided database script creates the following users:

Default Users created by DB Script

User Name	Role	Default Password
`system`	`SYSTEM_ADMIN`	`123456`
`user`	`USER`	`123456`

IMPORTANT	Change these default passwords immediately for security reasons.

4.8.2. User Roles

The following DomiSMP users can be of three types, as briefly described below:

UC	Description	Operation	Data
Actor: Group Admin			
Create or Update resource: Service Group	Create a new ServiceGroup for a new receiver participant. This service stores the Service Group and links it to the specified pair: (participantIdentifier + participantIndentifierScheme) the resource identifier. Information is stored into Resource table. This same service is used to create and update a ServiceGroup.	PUT	ServiceGroup
Actor: Group SMP			
Erase Service Group	Erases the resource (service group definition) AND the list of sub-resources such as service metadata for the specified receiver participant.	DELETE	ServiceGroup
Actor: Resource Admin			
Create or Update Resource such as Service Group Document and sub-resources: Service Metadata	Publish detailed information about one specific document service (multiple processes and endpoints). This same service is used to create and update ServiceMetaData.	PUT	ServiceMetadata
Erase Service Metadata	Remove all information about one specific service (i.e. all related processes and endpoints definitions).	DELETE	ServiceMetadata
Actor: Anonymous User			
Retrieve Service Group	Obtain the list of public services provided by a specific receiver participant (collection of references to the ServiceMetaData's). This service provides the information related to the Service Group according to the input pair: (participantIdentifier + participantIndentifierScheme).	GET	ServiceGroup

UC	Description	Operation	Data
Retrieve Service Metadata	Obtain detailed definition about one specific service of a specific participant for all supported transports. This service retrieves the SignedServiceMetadata according to the input tuple: (`participantIdentifier` + `participantIndentifierScheme` + `documentIdentifier` + `documentIdentifierScheme`).	`GET`	`SignedServiceMetadata`
Actor: `System Admin`			
	Create, modify, and delete users and domains. System admin can be only used in the DomiSMP UI.		

NOTE	For a complete description of the SMP user management, please consult the SMP Interface Control Document (ICD) document available at https://ec.europa.eu/digital-building-blocks/wikis/display/DIGITAL/SMP.

Users can be added, modified and deleted using the SMP Admin console or directly by executing sql commands. Below are instructions on how to modify users in the dat abase.

4.8.3. BCRYPT Password Generation

To manage DomiSMP users, you can use the DomiSMP UI console.

Use the procedure below to create the first system admin user.
An alternative is to use the provided SQL init scripts and replace passwords by first login.

The DomiSMP uses the BCRYPT algorithm to hash users' passwords. A BCRYPT-hashing tool is bundled with the SMP WAR file.

To get the hashing code:

1. Place a copy of the `smp-X.war` file into a temporary directory.
2. Extract the war file using the `jar` command:

```
jar -xvf smp-X.war
```

3. Obtain one or multiple hashes at once, using the following command:

```
java -cp "WEB-INF/lib/*" eu.europa.ec.edelivery.smp.utils.BCryptPasswordHash
<password_to_be_hashed>
```

   The result is a BCRYPT hash of the specified password.

In the example below, <password_to_be_hashed> stands for 123456.

Example

```
java -cp "WEB-INF/lib/" eu.europa.ec.edelivery.smp.utils.BCryptPasswordHash 123456
```

Result

```
2a$10$6nYTSUSh2BQfbOLIyCXn8eUViBcnn.WcjUrWOtJlMNDOdAtI85zMa
```

Multiple Password Hashing

The next command shows the hashing of several passwords at once, separated by a space in the command.

```
java -cp "WEB-INF/lib/" eu.europa.ec.edelivery.smp.BCryptPasswordHash
<password_to_be_hashed_1> <password_to_be_hashed_2>
2a$10$6nYTSUSh2BQfbOLIyCXn8eUViBcnn.WcjUrWOtJlMNDOdAtI85zMa
2a$107zNzSeZpxiHeqY2BRKkHE.HknfIe3aiu6XzU.qHHnnPbUHKtfcmDG
```

4.8.4. SMP Database User Creation

Adding an SMP user is done by adding a new entry in the SMP database SMP_USER table either directly or via the Administration console.

The User role is set in the SMP_USER table APPLICATION_ROLE column as follows:

User Role	Role Value
System Administrator	SYSTEM_ADMIN
DomiSMP User	USER
Anonymous User	N/A *Not defined in the SMP user database*

In the following examples, a **System Admin** user is created.

SYSTEM_ADMIN SMP User Creation

NOTE

To log in the Administration Console (for the first time), it is necessary to, create a user with SYSTEM_ADMIN privileges by entering the details directly into the SMP_USER table.
This initial user's password is generated using the BCRYPT utility described previously.
If PASSWORD_CHANGED is not set, the user will be asked to change the password at first logon.

Example of a SYSTEM_ADMIN user creation:

> **Username**:
> `smp_admin`
> **Password** (*hashed*):
> `$2a$10$6nYTSUSh2BQfbOLIyCXn8eUViBcnn.WcjUrWOtJlMNDOdAtI85zMa`
> **Role**: `SYSTEM_ADMIN`

Execute the following database command using the database user/password created in the Database Configuration section of this guide.

MySQL example:

```
INSERT INTO SMP_USER (USERNAME, ACTIVE, APPLICATION_ROLE, EMAIL, CREATED_ON,
LAST_UPDATED_ON) VALUES (1,'smp_admin', 1, 'SYSTEM_ADMIN', 'system@mail-
example.local', NOW(), NOW());

INSERT INTO SMP_CREDENTIAL (FK_USER_ID, CREDENTIAL_ACTIVE, CREDENTIAL_NAME,
CREDENTIAL_VALUE, CREDENTIAL_TYPE, CREDENTIAL_TARGET, CREATED_ON, LAST_UPDATED_ON)
VALUES ((SELECT id FROM SMP_USER WHERE USERNAME='smp_admin'), 1, 'smp_admin',
'$2a$10$olcGeWKGEoRia2DPuFqRNeca0IEdRSmOrljLz57BAjf1jlC9SohrS', 'USERNAME_PASSWORD',
'UI', sysdate, sysdate);
```

Oracle example:

```
INSERT INTO SMP_USER (ID, USERNAME, ACTIVE, APPLICATION_ROLE, EMAIL, CREATED_ON,
LAST_UPDATED_ON) VALUES (SMP_USER_SEQ.NEXTVAL,'smp_admin', 1, 'SYSTEM_ADMIN',
'system@mail-example.local', sysdate, sysdate);

INSERT INTO SMP_CREDENTIAL (FK_USER_ID, CREDENTIAL_ACTIVE, CREDENTIAL_NAME,
CREDENTIAL_VALUE, CREDENTIAL_TYPE, CREDENTIAL_TARGET, CREATED_ON, LAST_UPDATED_ON)
VALUES ((SELECT id FROM SMP_USER WHERE USERNAME='smp_admin'),1,'smp_admin',
'$2a$10$olcGeWKGEoRia2DPuFqRNeca0IEdRSmOrljLz57BAjf1jlC9SohrS',
'USERNAME_PASSWORD','UI', sysdate, sysdate);
```

NOTE The username/password credential is stored in the `SMP_CREDENTIAL` table.

The record must have the following values set to:

- `CREDENTIAL_VALUE`: the BCrypted password
- `CREDENTIAL_TYPE`: value must be set to: 'USERNAME_PASSWORD'
- `CREDENTIAL_TARGET`: value must be set to: 'UI'
- `FK_USER_ID`: value must be set to user id.

4.9. Logging Configuration

4.9.1. Logging properties

The SMP logging properties are defined in the ./WEB-INF/classes/logback.xml file embedded in the SMP `.war` file.

It is possible to modify the configuration of the logs by editing the embedded `logback.xml` or by defining new logback file in **smp.config.properties** file as example:

```
log.configuration.file=/opt/apache-tomcat-8.5.30/smp/logback.xml
```

Sample `logback.xml` file

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <!-- pattern definition -->
    <property name="encoderPattern" value="%d{ISO8601} [%X{smp_user}]
[%X{smp_session_id}] [%X{smp_request_id}] [%thread] 5p %c{1}: %L - %m%n"
scope="global"/>
    <property name="consolePattern" value="%d{ISO8601} [%X{smp_user}]
[%X{smp_session_id}] [%X{smp_request_id}] [%thread] %5p %c{1}: %L - %m%n"
scope="global"/>

        <appender name="file" class="ch.qos.logback.core.rolling.
RollingFileAppender">
            <file>${log.folder: -logs}/edelivery-smp.log</file>
            <filter class="ch.qos.logback.core.filter.EvaluatorFilter">
                <evaluator class="ch.qos.logback.classic.boolex. OnMarkerEvaluator">
                    <marker>SECURITY</marker>
                    <marker>BUSINESS</marker>
                </evaluator>
                <onMismatch>NEUTRAL</onMismatch>
                <onMatch>DENY</onMatch>
            </filter>
            <rollingPolicy class="ch.qos.logback.core.rolling.SizeAndTimeBasedRolling
Policy">
                <!-- rollover daily -->
                <fileNamePattern>
                    ${log.folder: -logs}/edelivery-smp-%d{yyyy-MM-dd}. %i.log
                </fileNamePattern>
                <!-- each file should be at most 30MB, keep 60 days worth of history,
                but at most 20GB -->
                <maxFileSize>30MB</maxFileSize>
                <maxHistory>60</maxHistory>
                <totalSizeCap>20GB</totalSizeCap>
            </rollingPolicy>
            <encoder>
                <pattern>${encoderPattern}</pattern>
            </encoder>
        </appender>
        <appender name="stdout" class="ch.qos.logback.core.ConsoleAppender">
            <Target>System.out</Target>
```

```
            <encoder>
                <pattern>${consolePattern}</pattern>
            </encoder>
        </appender>

        <logger name="eu.europa.ec.edelivery.smp" level="INFO" />
        <logger name="org.springframework.security.cas" level="DEBUG" />
        <root level="DEBUG">
            <appender-ref ref="file"/>
            <appender-ref ref="stdout"/>
        </root>
    </configuration>
```

More details on how to configure logback can be found at:
https://logback.qos.ch/documentation.html

4.10. Capability Documents

SMP's primary function is to store and provide access to participant capability documents. Once stored in DomiSMP, these documents can be accessed via the SMP REST API during the dynamic discovery process. Examples of capability documents are:

- Oasis SMP 1.0 Service Groups Document
- Oasis SMP 1.0 ServiceMetadata document
- Oasis CPPA3-CPP document

The Oasis SMP 1.0 ServiceMetadata document example is shown below:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<ServiceMetadata
    xmlns="http://docs.oasis-open.org/bdxr/ns/SMP/2016/05"
    xmlns:ns2="http://www.w3.org/2000/09/xmldsig#">
    <ServiceInformation>
        <ParticipantIdentifier scheme="iso6523-scheme-type">
0088:123456</ParticipantIdentifier>
        <DocumentIdentifier
            scheme="my-service-document">document-to-exchange</DocumentIdentifier>
        <ProcessList>
            <Process>
                <ProcessIdentifier scheme="as4-conformance-scheme">
                    conformance-service
                </ProcessIdentifier>
                <ServiceEndpointList>
                    <Endpoint transportProfile="bdxr-transport-ebms3-as4-v1p0">
                        <EndpointURI>http://access-point.eu/msh</EndpointURI>
                        <Certificate>Q2VydGlmaWNhdGUgZGF0YSA=</Certificate>
                        <ServiceDescription>
                            Service description for partners
```

```
                        </ServiceDescription>
                        <TechnicalContactUrl>www.contact-data-
page.eu</TechnicalContactUrl>
                    </Endpoint>
                </ServiceEndpointList>
            </Process>
        </ProcessList>
    </ServiceInformation>
</ServiceMetadata>
```

DomiSMP categorizes these capability documents into two main levels:

Resources

Resources are the main documents of the participant. An example of a resource document is the Oasis SMP 1.0 Service Groups Document.

Sub-resources

The Sub-resources are the documents that provide additional information about a particular participant's service. An example of a Sub-resource document is the Oasis SMP 1.0 Service Metadata document. .

4.10.1. Referencing Document Properties

In DomiSMP capability documents have a set of default properties which users can extend by creating custom properties.

DomiSMP also provides the possibility of referencing the values of these user-defined properties by using the following notation:

`${custom_prop_name}`

This allows users to reference these property values dynamically use them in the content of the document's XML. When properties' values are modified, their references are updated transparently. DomiSMP replaces all these references before returning the resource via an API call.

Below is an example where `ParticipantIdentifier`, `DocumentIdentifier`, `EndpointURI`, and the `Certificate` endpoint are referenced.

Example Referencing Document Capabilities Properties

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServiceMetadata xmlns="http://docs.oasis-open.org/bdxr/ns/SMP/2016/05"
xmlns:ns2="http://www.w3.org/2000/09/xmldsig#">
    <ServiceInformation>
        <ParticipantIdentifier scheme="${resource.identifier.scheme}">
            ${resource.identifier.value}
        </ParticipantIdentifier> ①
        <DocumentIdentifier scheme="${subresource.identifier.scheme}">
            ${subresource.identifier.value}
        </DocumentIdentifier> ②
```

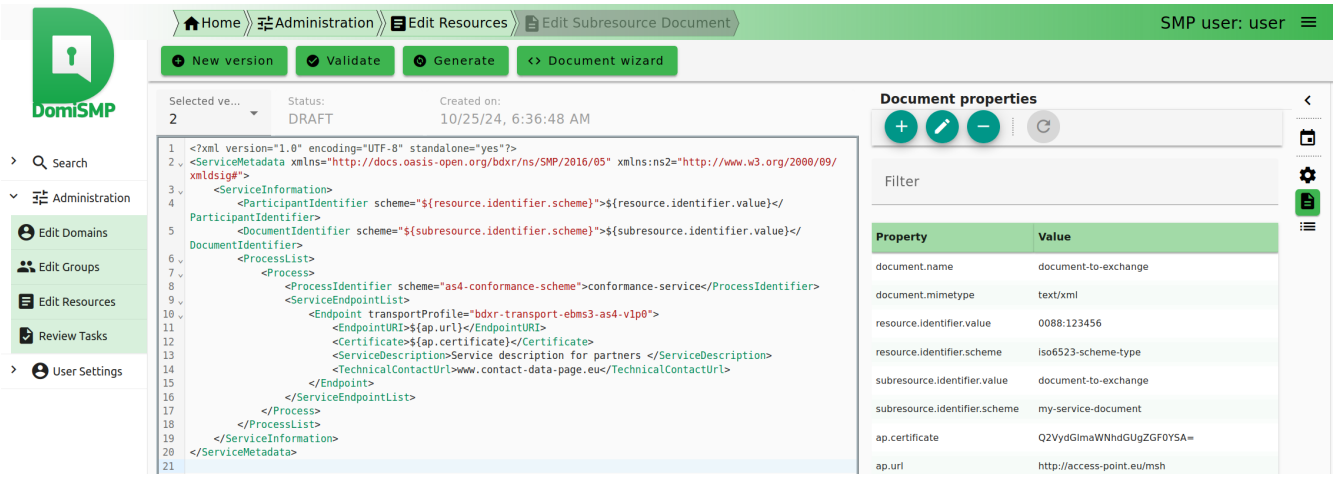
```
        <ProcessList>
            <Process>
                <ProcessIdentifier scheme="as4-conformance-scheme">
                    conformance-service
                </ProcessIdentifier>
                <ServiceEndpointList>
                    <Endpoint transportProfile="bdxr-transport-ebms3-as4-v1p0">
                        <EndpointURI>${ap.url}</EndpointURI>
                        <Certificate>${ap.certificate}</Certificate>
                        <ServiceDescription>
                            Service description for partners
                        </ServiceDescription>
                        <TechnicalContactUrl>www.contact-data-
page.eu</TechnicalContactUrl>
                    </Endpoint>
                </ServiceEndpointList>
            </Process>
        </ProcessList>
    </ServiceInformation>
</ServiceMetadata>
```

Where:

① Is resolved as: `<ParticipantIdentifier scheme="iso6523-scheme-type">0088:123456</ParticipantIdentifier>`.

② Is resolved as: `<DocumentIdentifier scheme="my-service-document">document-to-exchang</DocumentIdentifier>`.

Document properties are managed in the console's document editor under the **Document editor/Document properties** page.



Document's default properties are:

- Resources and sub-resources properties:
  - `document.name` - The name of the document.
  - `document.mimetype` - The version of the document.

- `resource.identifier.scheme` - the scheme of the resource identifier.
- `resource.identifier.value` - the value of the resource identifier.

- Sub-resource documents properties:
  - `subresource.identifier.scheme` - the scheme of the subresource identifier.
  - `subresource.identifier.value` - the value of the subresource identifier.

Other document properties can be defined and managed by the user within the **Document properties** tool.

4.10.2. Referencing Documents

The document referencing feature enables users to reuse the content of a single document across multiple resources. Prior to DomiSMP 5.1, each resource had its own document where users published their message exchange capabilities.

When multiple participants shared the same Access Point service provider, they all published within their service capabilities the same Access Point data: such as Access Point URL and Certificates. This resulted in multiple documents with the same content. To avoid this redundancy, DomiSMP 5.1 introduced the document referencing feature so that users can reference the same document across multiple resources.

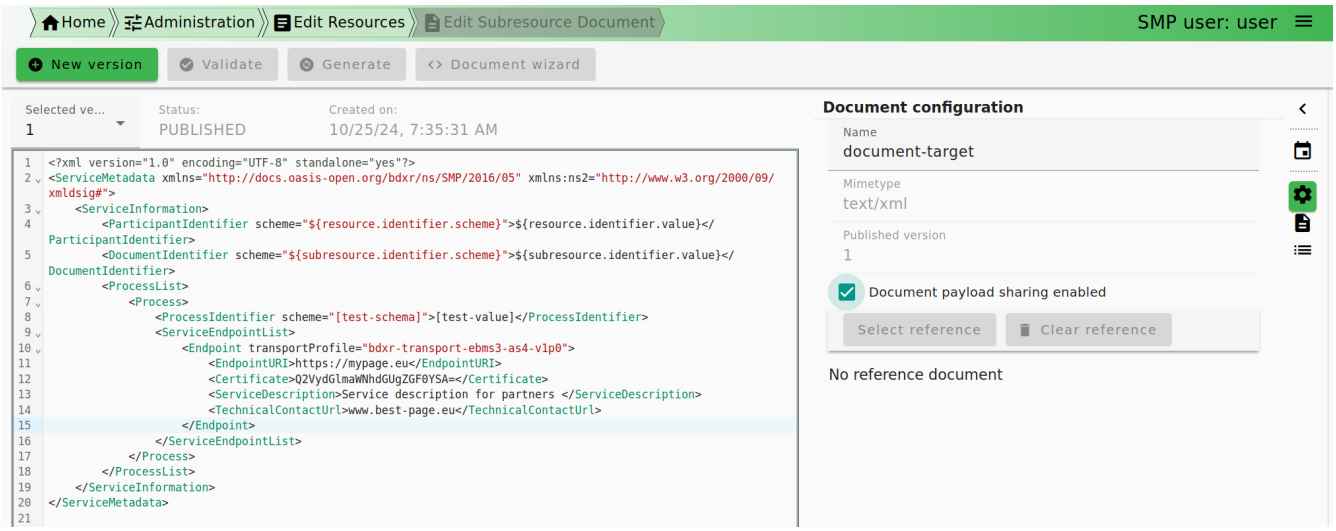
This feature simplifies the maintenance of user's documents by reusing the same document template data multiple times.

Document referencing combined with placeholders allows users to override specific values with their own (e. g. participant identifier).

When placeholder values are not defined in the final document, the values from the referenced document are used.

Making Document Available for Referencing

To allow documents to be referenced, they must have the **Document payload sharing enabled** option set in the **Document configuration** tool. Once the checkbox is selected, the documents can be referenced from other documents.

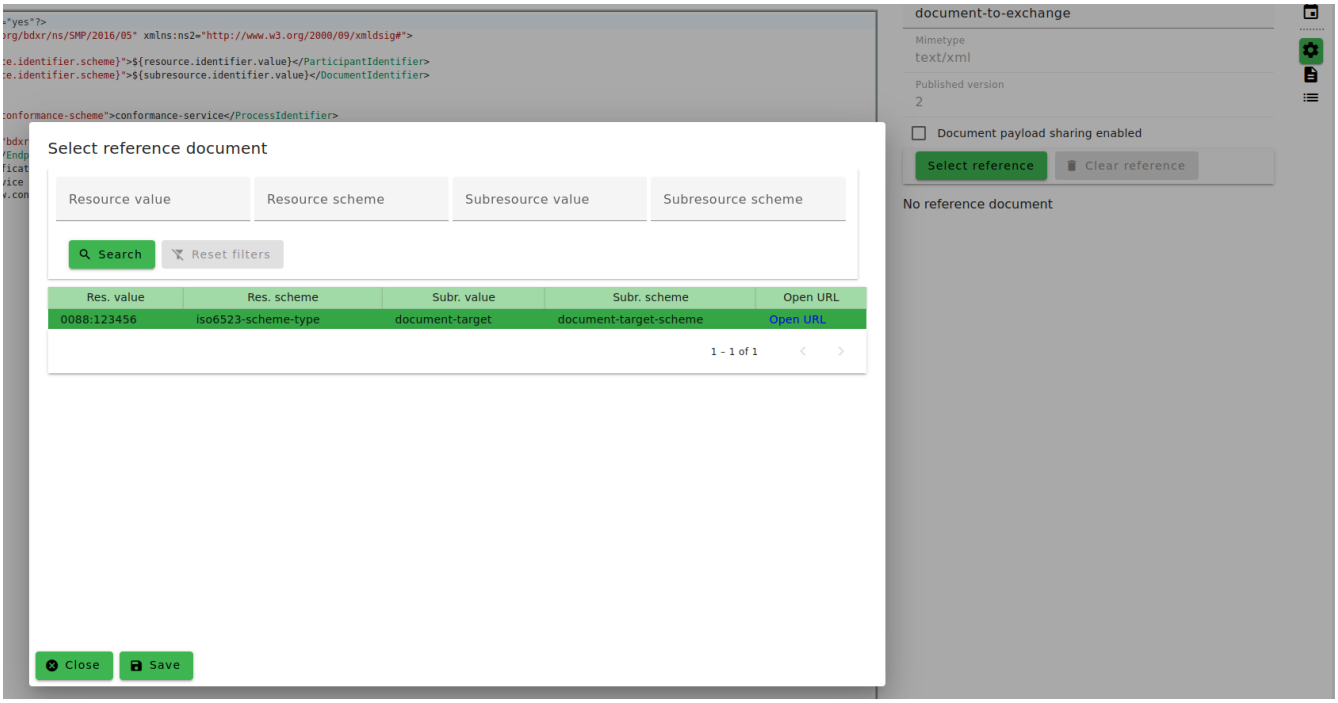


How to Reference a Document

To reference a document, follow these steps in the **Document Configuration** tool:

1. Open the document where the reference will be used and choose the **Document Configuration** tab on the left side of the document editor.
2. Click the **Document References** button to open the **Document Reference** dialog.
3. From the Document Reference Table, select the document to be referenced. If the target document is not listed, use the filter fields to narrow the results.
4. Select the document and click the **Save** button to add the reference.
5. Save the changes to the document.

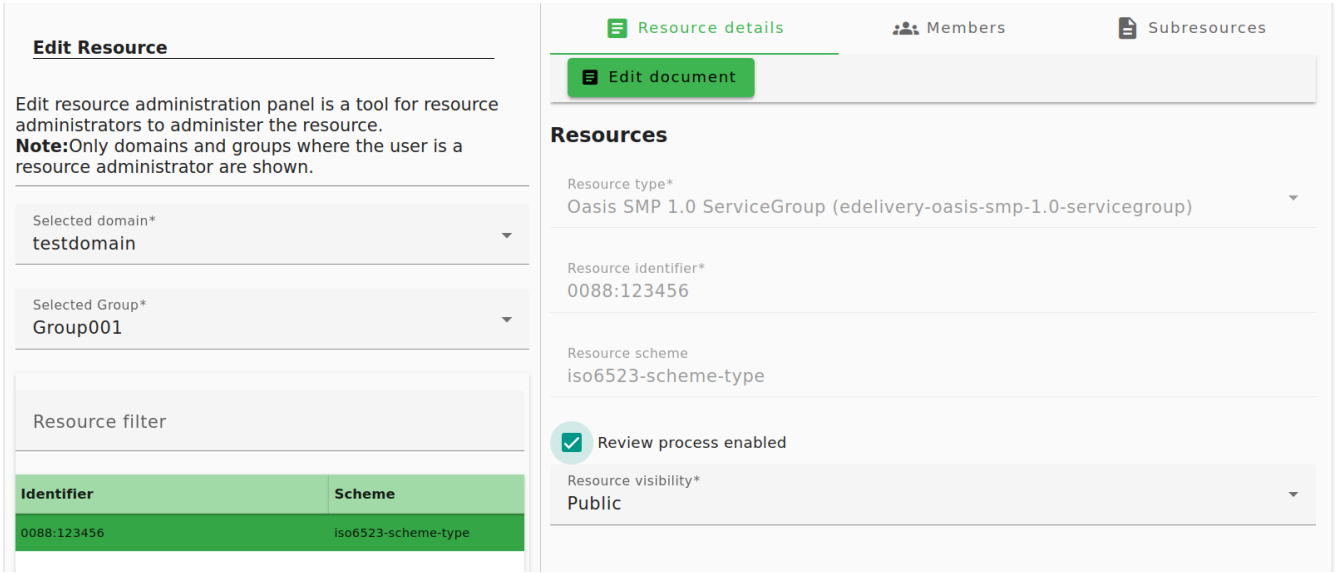
When retrieving the document via REST API, the referenced document is automatically included in the response.



IMPORTANT	If the document is already referencing another document, it cannot be set as "Shared enabled" as "Shared enabled" documents cannot reference other documents. == Document Review

DomiSMP offers a basic document review and approval feature, allowing users to review and approve documents before they are published. Disabled by default, this feature can be enabled for each Resource and Subresource via the DomiSMP Console, in the **Resource Details** page.

When the review process is enabled for a Resource, this option is applied to all related Sub-resources. The same is true for when disabling the review process.



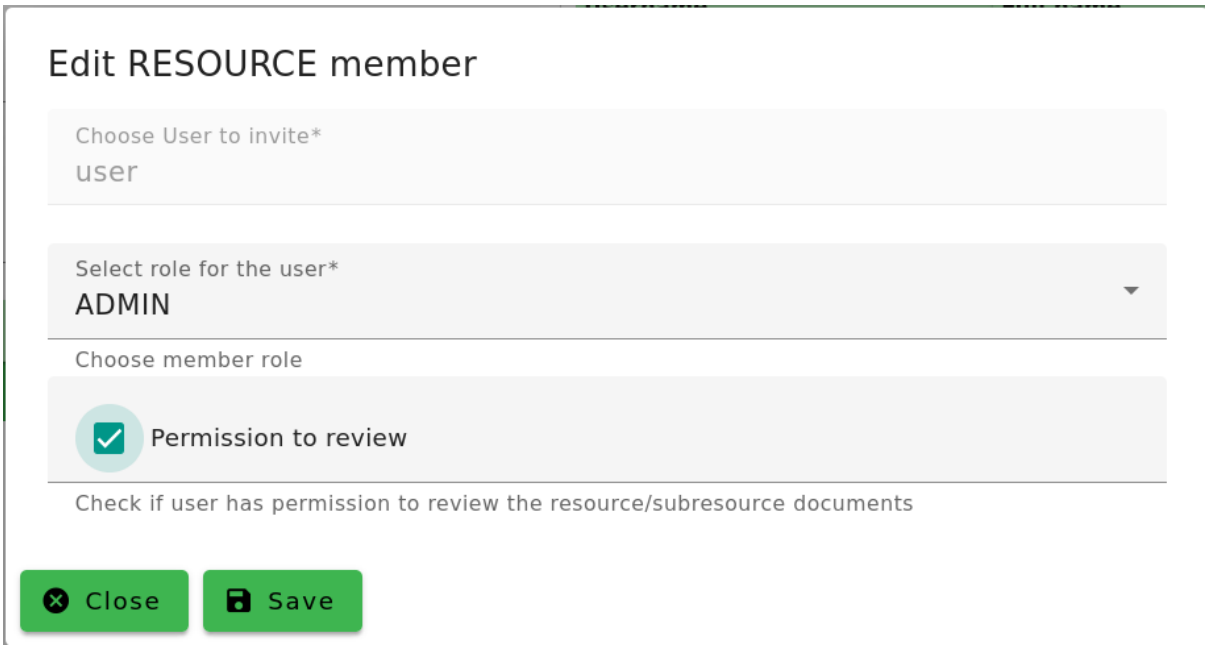
When the Review process is enabled, the following actions become available to the user in the DomiSMP Document editor page:

- `Submit for review` - allows the user to submit the document for review.
- `Approve` - allows the user to approve the document. Visible only for the users with review

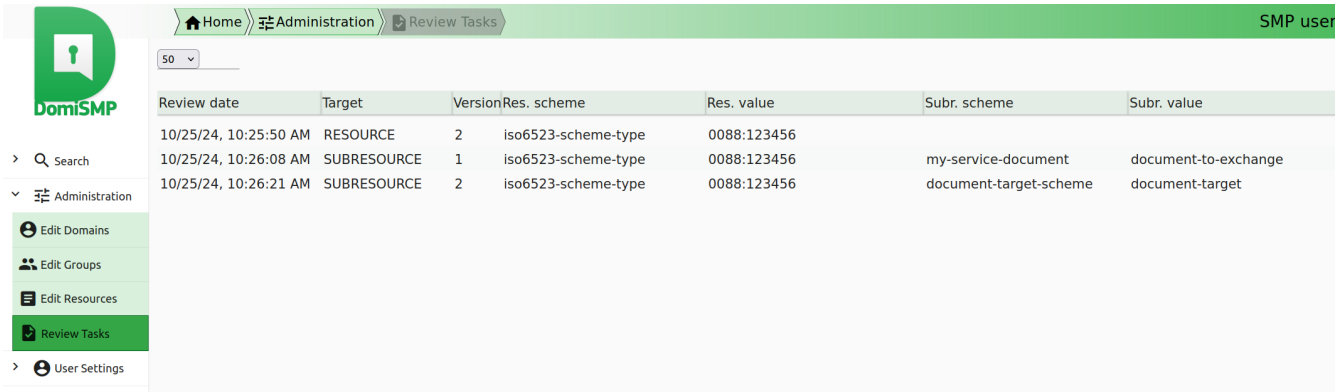
permissions.

- `Reject` - allows the user to reject the document. Resource administrators can always reject documents under review even if when they do not have review permissions for the specific document in review. This ensures that the administrator can remove documents from the review queue.

The documents can be reviewed/approved only by the resource administrators that have the review permission enabled. The permissions can be set in the DomiSMP Console in the **Edit Resource** page. Once the document is approved, it can be published.



The reviewers can find documents under review in the DomiSMP Console in the **Review page**.



by double-clicking on the document, the reviewer can see the document details, and approve or reject it. == Localization

Currently, DomiSMP provides language support and date/time format per locale.

4.10.3. Translations

DomiSMP UI Language Support

By default, DomiSMP UI ships in English, but it provides support for multiple languages. This feature allows users to add custom translations.

4.10.4. Setting Custom Translations

DomiSMP loads user provided translations from a pre-configured folder in its installation location. The location of this folder is set via the `smp.locale.folder=locales` configuration property.

This property takes as value either a relative or an absolute path to a directory accessible from the machine where DomiSMP is running on.

Relative Paths to Translation Files

If relative, the provided path describes the file's location relative to the working directory of the server into which DomiSMP is deployed. If the directory does not yet exist, DomiSMP creates it along with all its parents when at startup.

Multiple Translation Files

Multiple custom translations files can be deployed at the configured location.
Translation files are `.json` files with names that are:

- prefixed with `ui_`;
- followed by the ISO 639 language code (`en`, `fr`, etc.).

DomiSMP automatically loads new valid translation files placed at the configured location.

Creating a New Translation File

See below a part of the default `ui_en.json` file's:

`ui_en.json`

```
{
  "column.selection.link.all": "All",
  "column.selection.link.none": "None",
  "column.selection.link.show": "Show columns",
  "column.selection.link.hide": "Hide columns",

  "cancel.dialog.text": "Do you want to cancel all unsaved operations?",
  "cancel.dialog.title": "Unsaved data",
}
```

When creating a new translation of the UI, copy the default `ui_en.json` file in order to reuse the labels referencing UI items, such as `column.selection.link.show` and replace the language values with its equivalents in the new language.

This new file should be named using the correct language code. See below a partial example for a French translation file.
See also **DomiSMP Supported Locales**.

`ui_fr.json`

```
{
"column.selection.link.all": "Tout",
```

```
"column.selection.link.none": "Aucun",
"column.selection.link.show": "Afficher les colonnes",
"column.selection.link.hide": "Masquer les colonnes",

"cancel.dialog.text": "Voulez-vous annuler toutes les opérations non enregistrées ?",
"cancel.dialog.title": "Données non enregistrées",
}
```

▼ **DomiSMP Supported Locales**

File	Locale
ui_bg.json	Bulgarian
ui_cs.json	`Czech
ui_da.json	Danish
ui_de.json	German
ui_el.json	Greek
ui_en.json	English
ui_es.json	Spanish
ui_et.json	Estonian
ui_fi.json	Finnish
ui_fr.json	French
ui_hr.json	Croatian
ui_hu.json	Hungarian
ui_it.json	Italian
ui_lt.json	Lithuanian
ui_lv.json	Latvian
ui_mt.json	Maltese
ui_nl.json	Dutch
ui_pl.json	Polish
ui_pt.json	Portuguese
ui_ro.json	Romanian
ui_sk.json	Slovak
ui_sl.json	Slovenian
ui_sv.json	Swedish

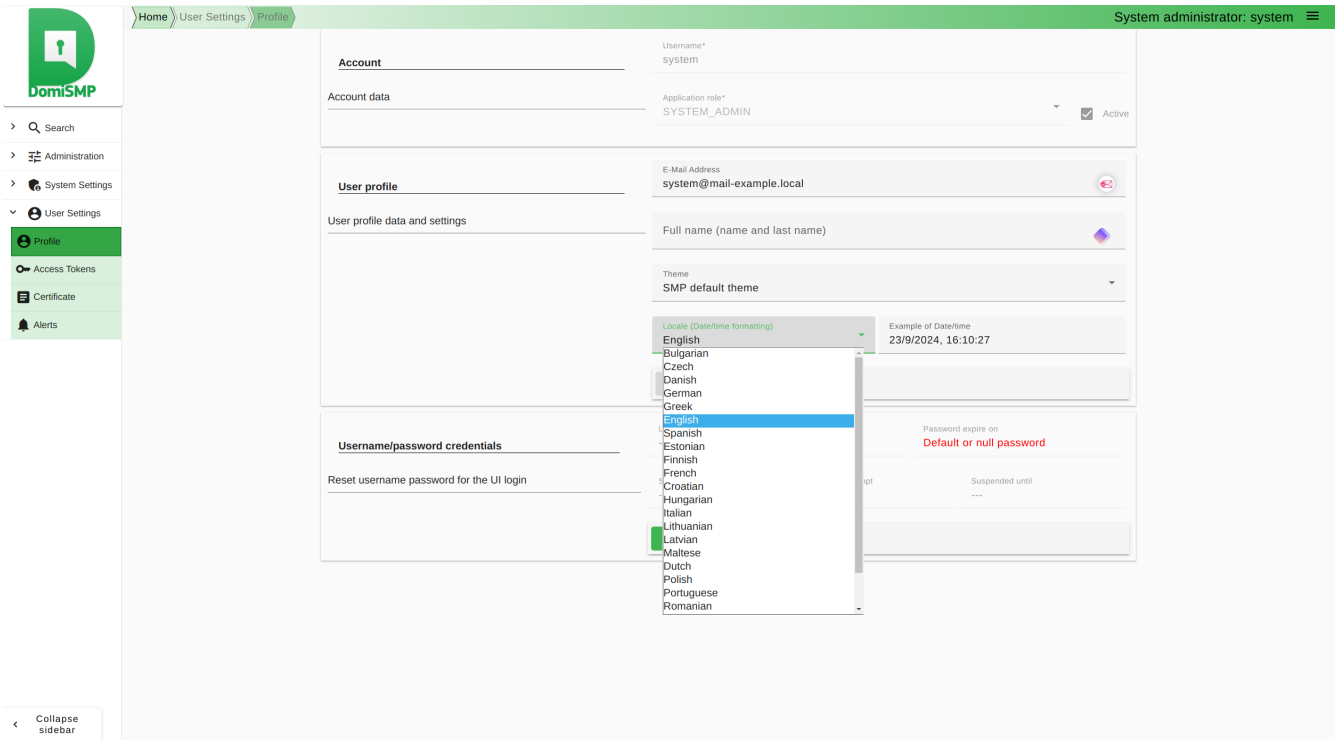
4.10.5. Setting User's Language Preference

Once a translation file is created in the locales folder (see Setting Custom Translations) and loaded in DomiSMP, users can then set their preferred language by navigating to the **User Profile** page in

the DomiSMP UI and selecting the corresponding value from the dropdown.

NOTE	The DomiSMP Console's Language preference option in **User Profile** is currently listing all European countries official languages. If you would like to see other languages added on the dropdown, please contact eDelivery Support.

If a user selects a language for which a translation file cannot be found, DomiSMP then loads English, the default translation file.
This user preference is remembered between visits.



4.11. SMP SOAP UI

SOAP UI Testing

The SOAP UI can be used to create, update and delete Service Groups and Metadata.

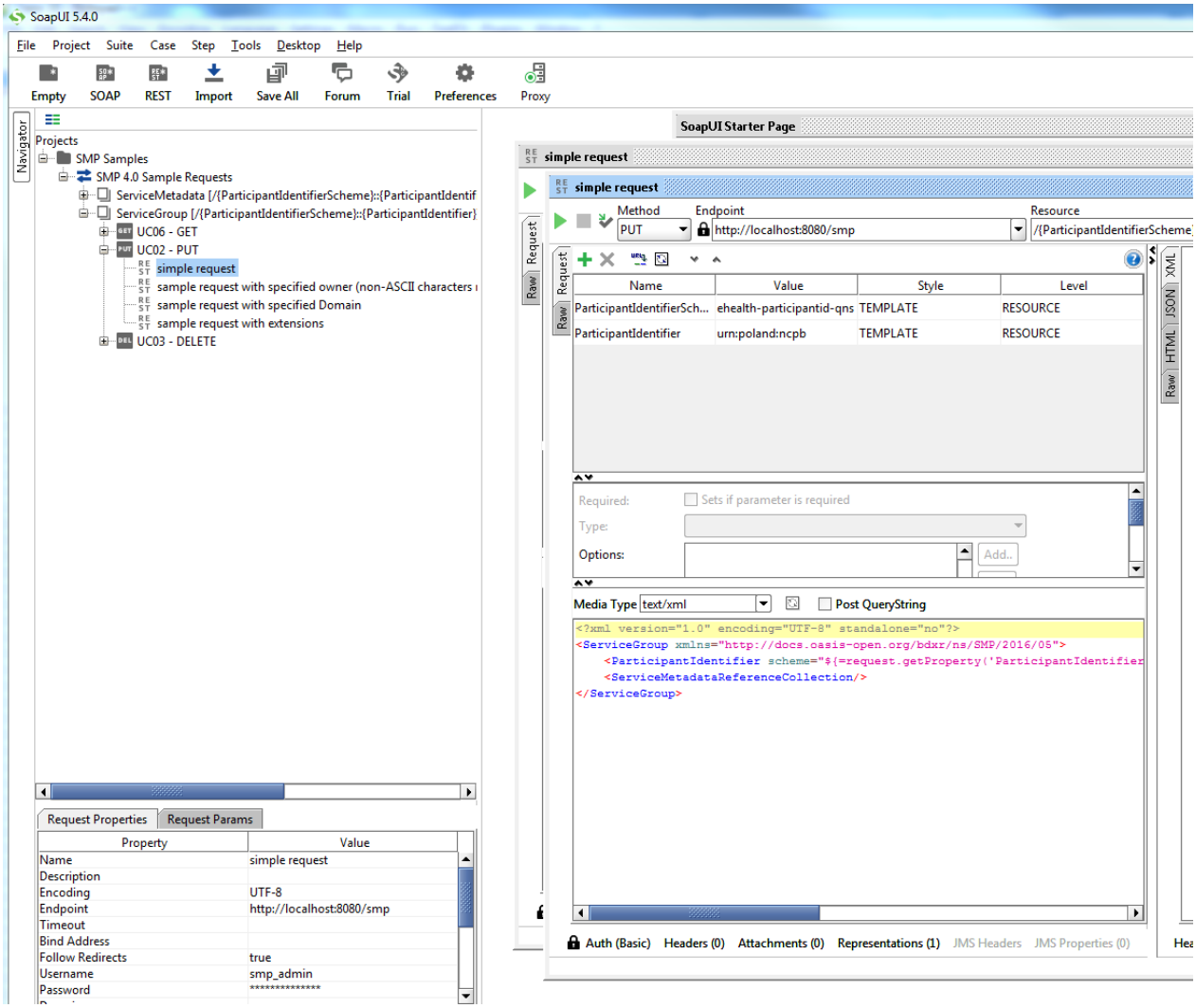
An SMP SoapUI project contains sample requests and is included in the zip file already downloaded.

The procedure to create, update or delete a Service Group is described in the next steps.

4.11.1. Service Groups CRUD Operations

▼ *CREATE Service Group*

In the left navigation pane of the SoapUI interface, browse to the REST PUT method as shown below:

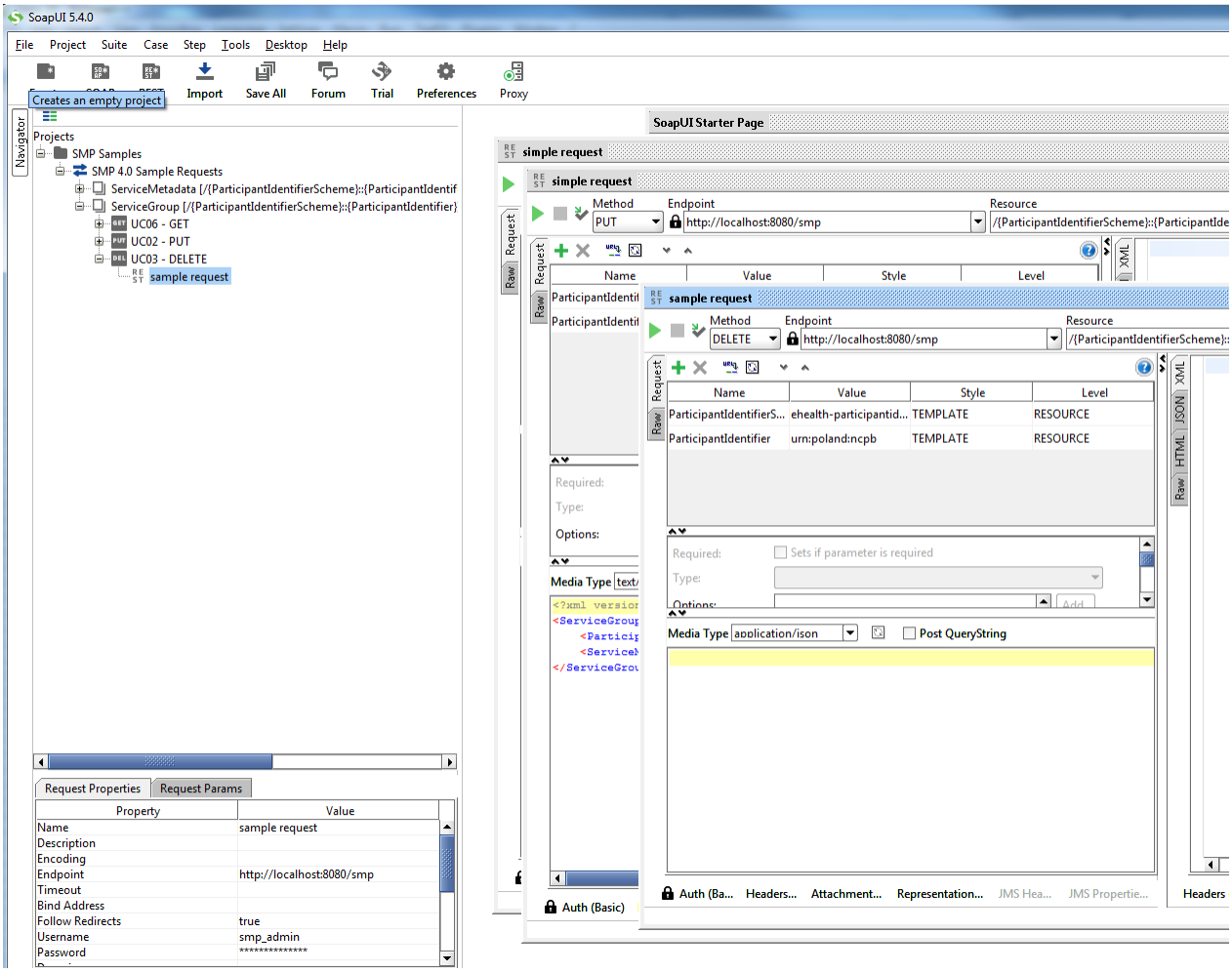


▼ *UPDATE Service Group*

The REST method to update the **ServiceGroup** is the same as the one used for creating **ServiceGroup** described in the previous section.

▼ *DELETE ServiceGroup*

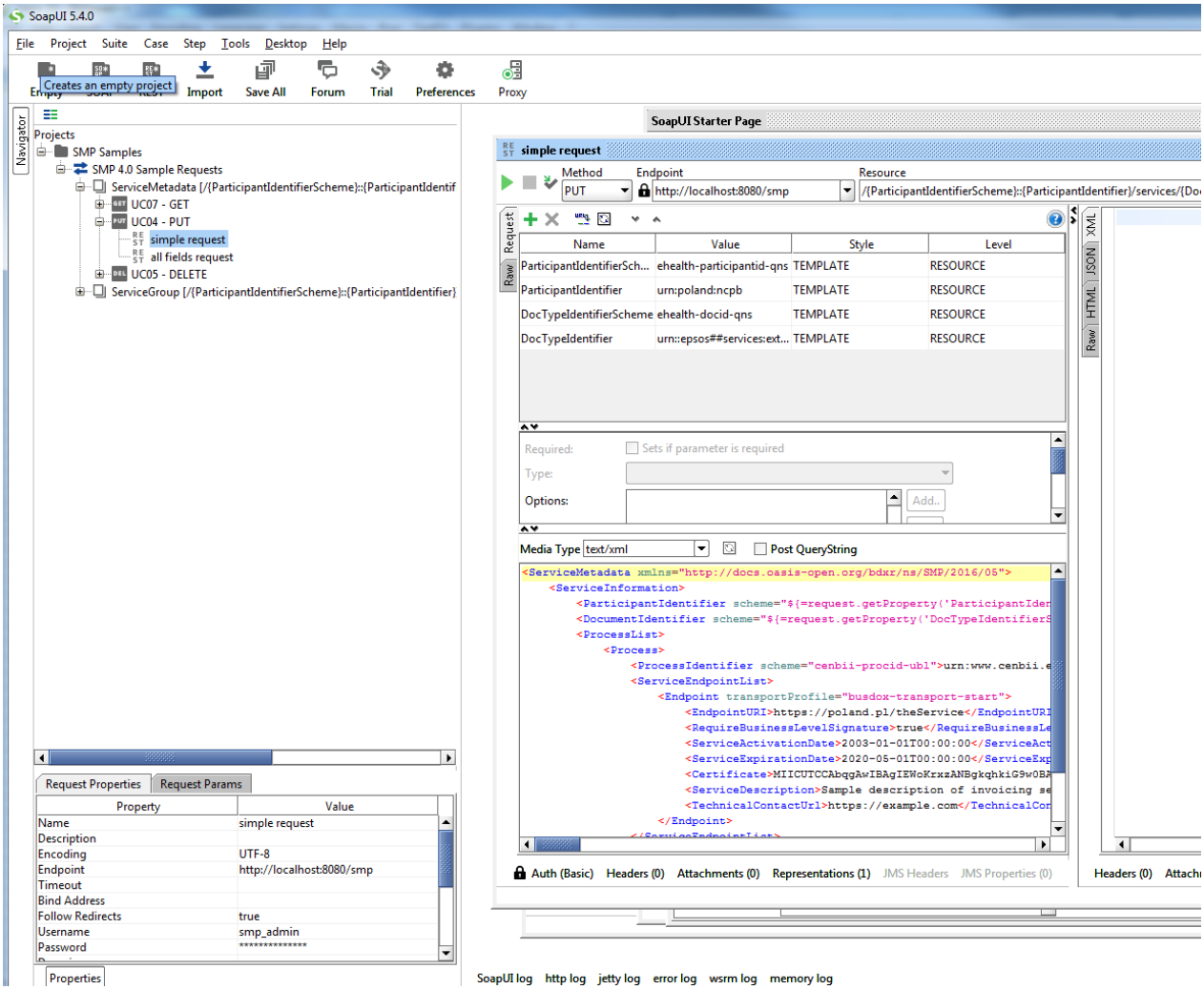
On the SoapUI interface on the left navigation panel, browse to the REST DELETE method as indicated below:



4.11.2. Service Metadata CRUD Operations

▼ *CREATE Service Metadata*

In the left navigation pane of the SoapUI interface, browse to the REST PUT method as shown below:

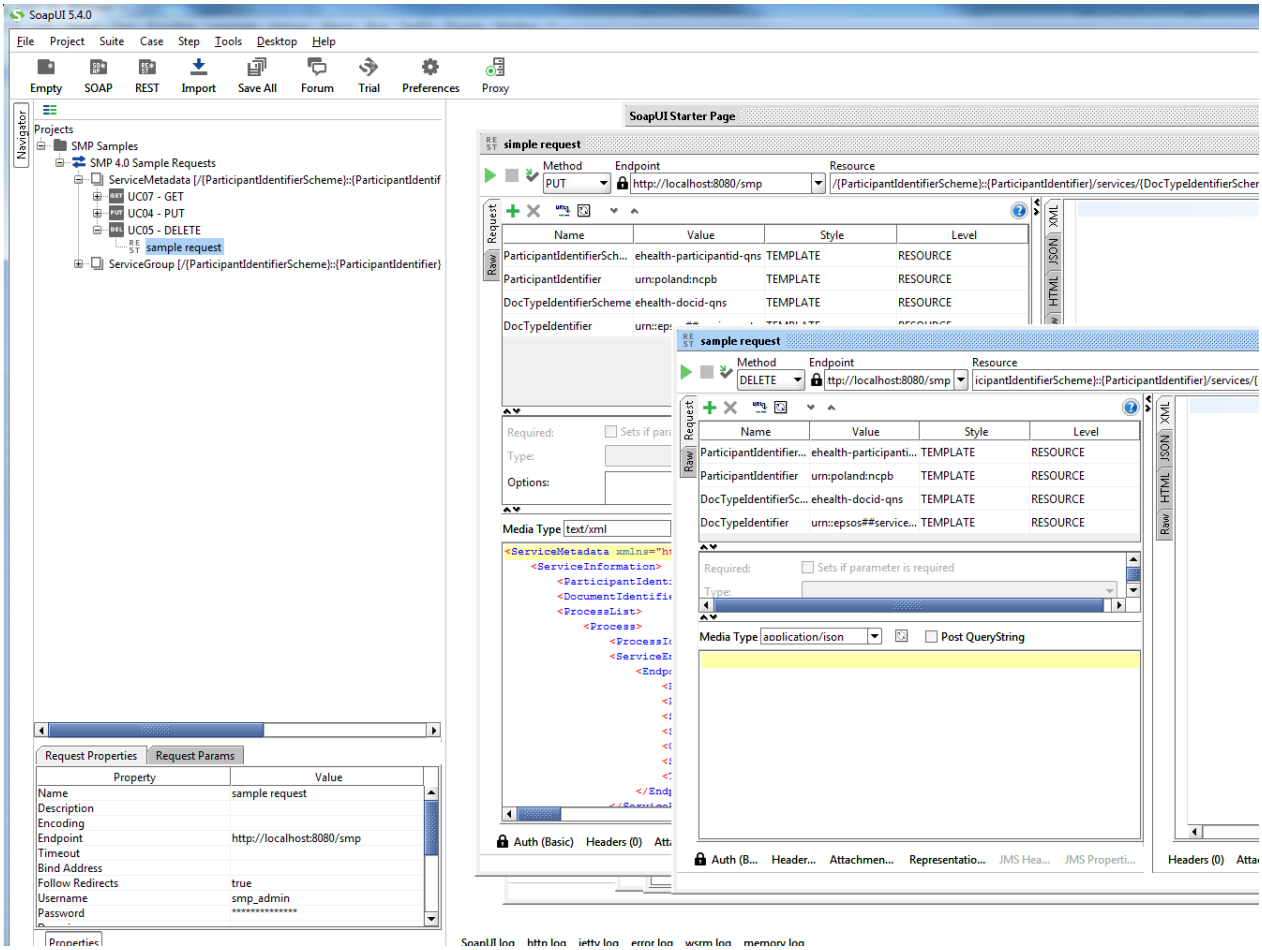


▼ *UPDATE Service Metadata*

The REST method to update `ServiceMetadata` is the same as the one use for creating `ServiceMetadata` as described in the previous section.

▼ *DELETE Service Metadata*

In the left navigation pane of the SoapUI interface, browse to the `REST DELETE` method as indicated below:



4.12. Compiling SMP

4.12.1. Compilation Prerequisites

Supported Operating System Platform

The eDelivery SMP can be built on the following OS platforms:

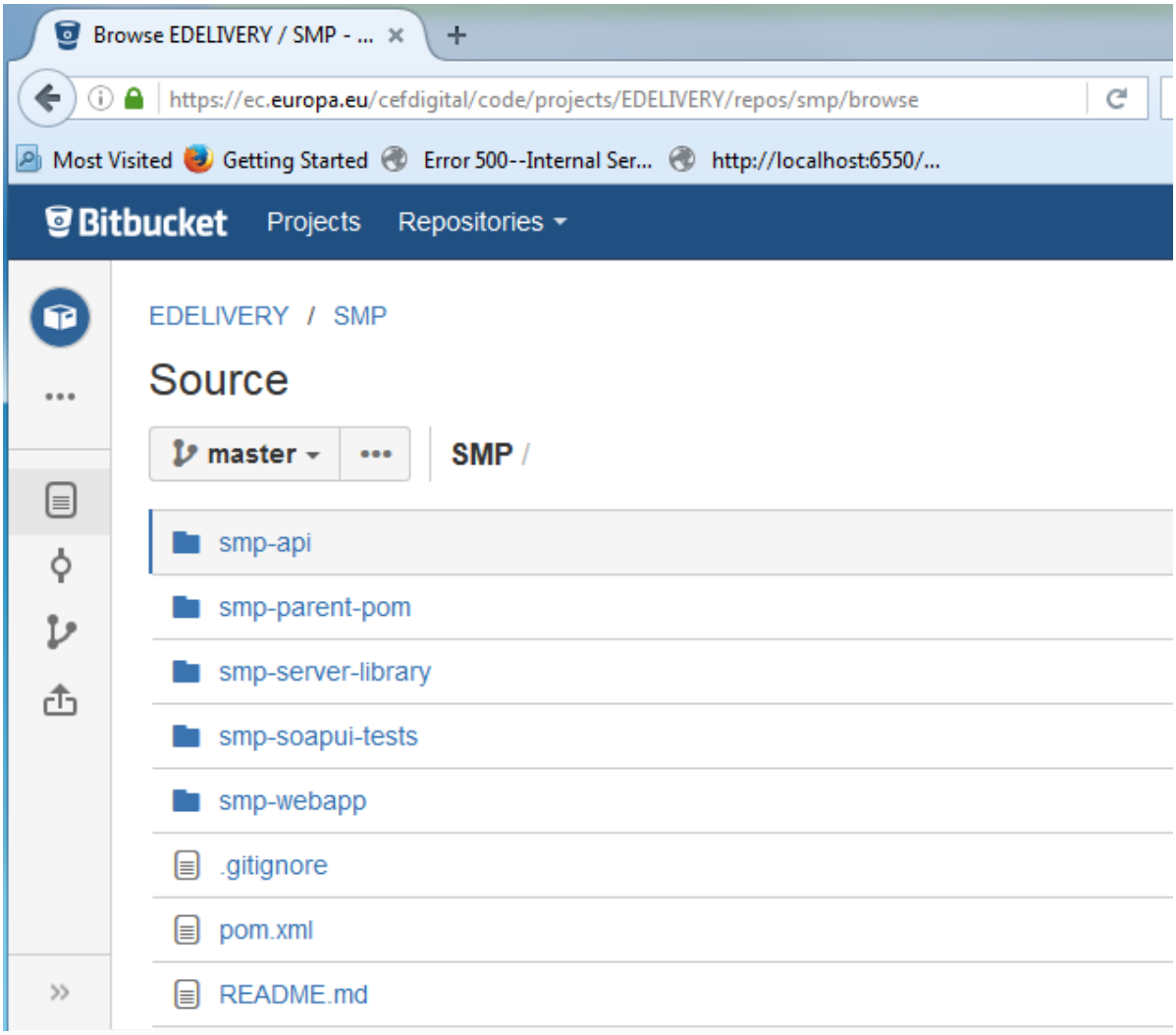
- Windows Workstation & Server
- Linux platform

Software Requirements

The following software components are required on the target system:

- Java Development Kit environment (JDK), version 8: http://www.oracle.com/technetwork/java/javase/downloads/index.html
- Maven 3.6 and above (https://maven.apache.org/download.cgi)
- GIT (optional: Git is only used to download the project sources but these sources can be downloaded from any system having Git installed and then just copied manually on the compilation platform).

4.12.2. Downloading the Source Code



The source code of SMP is freely available and can be downloaded from the SMP Source Code Repository.

▼ *URL*

SMP Source Code Repository:
https://ec.europa.eu/digital-building-blocks/code/projects/EDELIVERY/repos/smp/browse

4.12.3. Compiling SMP Source Code

1. Create a new `/comp_dir` temporary directory.
2. From `/comp_dir`, execute:

```
git clone https://ec.europa.eu/digital-building-blocks/code/scm/edelivery/smp.git
```

3. Go to the newly created `/smp` directory.

   Using a ls command in the above-mentioned directory renders the following contents:

```
pom.xml README.md smp-api smp-parent-pom smp-server-library smp-soapui-tests smp-
webapp
```

4. Start the compilation by executing the following command:

```
mvn clean install -DskipTests
```

5. A successful compilation will result with the following:

```
mvn clean install -DskipTests
```

   Expected Console Output

```
[INFO] Scanning for projects…
/..
../

[INFO] Installing /home/smpcomp/smp/smp/pom.xml to
/home/smpcomp/.m2/repository/eu/europa/ec/smp/3.X/smp-3.X.pom

[INFO]
------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] smp-angular ....................................... SUCCESS [132.375 s]
[INFO] smp-api ........................................... SUCCESS [32.375 s]
[INFO] smp-server-library ................................ SUCCESS [02:01 min]
[INFO] smp-webapp ........................................ SUCCESS [23.314 s]
[INFO] SMP Builder POM ................................... SUCCESS [2.222 s]
[INFO]
------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO]
```

```
------------------------------------------------------------------------
[INFO] Total time: 03:00 min
[INFO] Finished at: 2017-06-08T11:35:27+02:00
[INFO] Final Memory: 61M/726M
[INFO]
------------------------------------------------------------------------
```

As a result the web application, `smp.war` , is created in the `/smp-webapp/target/` directory.

Using a `ls` command in the directory mentioned above renders the following contents:

```
smp-X smp.war classes generated-sources generated-test-sources maven-status test-
classes webapp-classes
```

4.13. SMP Admin Console

The SMP Admin console enables:

- Anonymous users to search and explore published data in the SMP.
  Anonymous users can search for participants by participant ID, schema, or domain.
- Service Group administrators to manage owned Service groups; SMP administrators to manage Service groups registered on SMP, and System Administrators to manage users and domains

Admin Console URL

The Admin console dashboard is reachable via the following URLs:

```
http://<host>:<port>/smp%5b-version%5d/iu/[http://<host>:<port>/smp[-version]/iu/]
```

If the deployment package (war file) filename changed in order to simply upgrade the old SMP version as for example `smp-4.0.0.war` to `cipa-smp-full-webapp.war`, then the application root context might change as well.

Example:

```
http://wlal0079a.cc.cec.eu.int:1043/cipa-smp-full-webapp/ui/[http://
<host>:<port>/cipa-smp-full-webapp/ui/].
```

Roles

Two types of application roles are defined in the SMP Admin Console:

- **System Administrator**: this is a *super admin* who can manage SMP users and domains.
- **User**: a regular user of the DomiSMP: the user can administer Domains, Groups and Resources according to membership roles described in SMP User Management.

When users are logged, their role is displayed in read-only mode (as a label). Only the System

Administrator can change the role of another user.

Chapter 5. Interface Description

eDelivery building blocks helps public administrations exchange electronic data and documents with other public administrations, businesses and citizens, in an interoperable, secure, reliable and trusted way. By using this building block, every participant becomes a node in the network using standard transport protocols and security policies. eDelivery is based on a distributed model, allowing direct communication between participants without the need to set up bilateral channels.

SEE ALSO

- Overview of eDelivery
- SMP Architecture

▼ Guide's Purpose, Scope and Target Audience

Purpose

This document defines the participant's interface to the SMP component of the eDelivery building block as it will extend and evolve in sight of its usage in the framework of eHealth and its additional requirements.

This use case/interface control document will be used as reference for mutual understanding of eHealth requirements on the one hand and the future service delivered by the Digital Europe Programme on the other hand.

Scope

This document is a high-level functional definition of the services provided by SML. This document will be later extended with additional document that further detail the services with technical information intended for the development of eHealth client solutions implementation.

Audience

This document is intended for:

- architects and development teams of the Digital Europe Programme for committing on future service delivery of SMP
- architects and functional analysts of the eHealth team for validating the intended service against their requirements.

▼ Useful References

Concepts

All the **concepts** used throughout this document have been defined in the following documents:

- OpenPEPPOL AISBLPolicy for use of Identifiers
- Service Metadata Publishing (SMP) Version 1.0
- PEPPOL Transport Infrastructure Service Metadata Publishing (SMP)

Other Useful References

Document	Description
HTTP Methods for RESTful Services	Short description of HTTP Methods for RESTful Services.
PEPPOL AISBL Policy for use of Identifiers	-
Service Metadata Publishing (SMP) Version 1.0	This document describes a protocol for publishing service metadata within a 4-corner network.
eSens Building Blocks - ABB - Capability Lookup - 1.6.0	Capability Lookup is a technical service to accommodate a dynamic and flexible interoperability community. A capability lookup can provide metadata about the communication partner's interoperability capabilities on all levels defined in the European Interoperability Framework.
eSens Building Blocks - PR - SMP	e-SENS will use SML (Simple Metadata Publisher) specification originally developed by PEPPOL and generalised and standardised by OASIS.TheSMPspecification usually complements the Location LookUp ABB.
PEPPOL Transport Infrastructure Service Metadata Publishing (SMP)	This document describes the REST (Representational State Transfer) interface for Service Metadata Publication within the Business Document Exchange Network (BUSDOX).
DomiSML Software Architecture Document	This document is the Software Architecture document of the CIPA eDelivery Business Document Metadata Service Location application (BDMSL) sample implementation. It intends to provide detailed information about the project: 1) An overview of the solution 2) The different layers 3) The principles governing its software architecture.
PEPPOL Transport Infrastructure Service Metadata Locator (SML)	This document defines the profiles for the discovery and management interfaces for the Business Document Exchange Network (BUSDOX) Service Metadata Locator service.
OASIS - Service Metadata Publishing (SMP) Version 2.0	This document describes the version 2.0 of the Oasis SMP standard.

IMPORTANT	documents **listed in *bold italic red*** in the above list are to be considered for the detailed design and the implementation of the SMP as this one must be fully compliant to those specifications.

SMP Role

The role of SML in the Four Corner Model is to allow:

- **servers** (*receivers*) to publish the definition of the services they provide, i.e., the documents they are able to receive and the means through which they can receive them,
- **clients** (*senders*) to find out the definitions of those services.

To that end, SMP provides services for:

- **receivers** to register services definitions (such as “put metadata”);
- **senders** to consult those definitions (“retrieve metadata”).

SMP/SML Interactions

To promote the consistency of the whole process SMP sends location information to SML of:

- **SMP**'s own location to allow senders to discover SMP;
- **all Access Points** providing access to declared `ServiceGroups` of the participants SMP is managing.

SML Management Services

SML exposes multiple management services allowing SML to declare new location information or any related changes. They are:

Manage participant identifiers interface
: the interface for Service Metadata publishers for managing the metadata relating to specific participant identifiers that they make available.

Manage service metadata interface
: the interface for Service Metadata publishers for managing the metadata about their services, e.g., binding, interface profile and key information.

NOTE	TManage participant identifiers interface and Manage service metadata interface are not detailed here but the document will refer to these when they are invoked from SML REST services. Refer to the `Execution` sections of the REST Services definitions below for further details on these interactions.

SEE ALSO	The PEPPOL Transport Infrastructure Service Metadata Locator (SML) document.

SML also exposes the,

Service Metadata discovery interface
: This is the lookup interface which enables senders to discover service metadata about specific target participants. As it is out of the scope of this document this service is not further discussed in the present document.

NOTE	This functionality isn’t addressed currently, but is planned for a future release. The use cases envisioned for this functionality are:

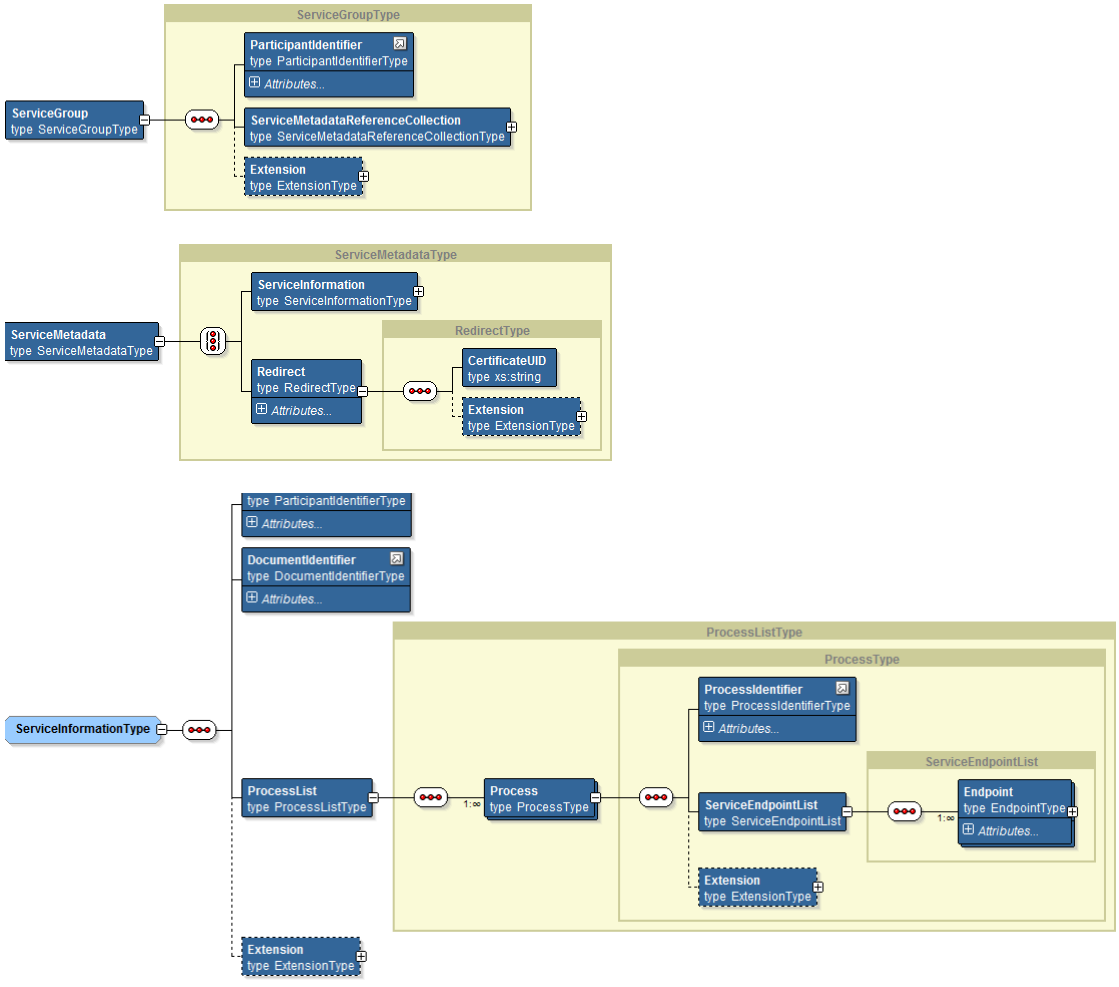
UC08 - Register SMP
UC09 - Change SMP Location
UC10 - Unregister SMP
UC11 - Migrate Metadata SMP

Data model

The SMP interface is built around the data it is intended to manage. Therefore, this document starts by defining the data itself.

Logical data model

The diagram below depicts the major parts of the data model describing the configuration held by SML and managed through the interface described in this document. This model is another view of the XSD definition that can be found in the XSD Files section.



▼ *ServiceGroup*

A service group is defined as structure that represents a set of services associated with a specific **Participant identifier** that is handled by a specific Service Metadata Publisher. The ServiceGroup structure holds a list of references to ServiceMetadata resources in the ServiceList structure

SEE ALSO **Data Model** in PEPPOL Transport Infrastructure Service Metadata

Publishing (SMP)).

SEE ALSO Refer to Identifiers in Oasis SMP 1.0 or **Interfaces and Data Model** in Oasis SMP 1.0 for more details and additional references about identifiers of participants (/businesses), documents and processes.

▼ *ServiceMetadata*

ServiceMetadata is defined as "_a structure that represents Metadata about a specific electronic service. The role of the ServiceMetadata structure is to associate a participant identifier with the ability to receive a specific document type over a specific transport [...]".

SEE ALSO Refer to Identifiers in Oasis SMP 1.0 or **Interfaces and Data Model** in Oasis SMP 1.0 for more details and additional references about identifiers of participants (/businesses), documents and processes.

▼ *Process*

You can find the following definitions for `ServiceMetadata`:

```
"a structure that reresents Metadata about a specific electronic service. The role
of the ServiceMetadata structure is to associate a participant identifier with the
ability to receive a specific document type over a specific transport."
```

```
It also describes which business processes a document can participate [⋯]
```

```
"⋯ and it is the purpose of this intermediate entity (Process) to hold
the process-related information (i.e. its identifier and scheme), and to
allow a participant to use a document type to participate in multiple
business processes (when applicable)."
"It also describes which business processes a document can participate [⋯]"
```

▼ *Endpoint*

The endpoint is the ultimate entity, holding all the necessary information for all services of the ServiceGroup to be accessed by the sender in order to send document(s) to the receiver.

SEE ALSO 2.3.3.3 Description of the individual fields (elements and attributes) and 2.3.4.4 Description of the individual fields (elements and attributes) in Service Metadata Publishing (SMP) Version 1.0 and see PEPPOL Transport Infrastructure Service Metadata Locator (SML).

XSD element	Description
endpointURI Oasis SMP 1.0 Element: /ServiceEndpointList/ Endpoint/EndpointURI Oasis SMP 2.0 Element: /sma:ProcessMetadata/sma:Endpoint/smb:AddressURI	The address of an endpoint, as a URL.
transportProfile Oasis SMP 1.0 Element: ServiceInformation/ ProcessList/../Endpoint/ @transportProfile Oasis SMP 2.0 Element: /ServiceMetadata/sma:ProcessMetadata/sma:Endpoint/smb:TransportProfileID	Indicates the type of transport method that is being used between access points
requireBusinessLevelSignature Oasis SMP 1.0 Element: ServiceInformation/ProcessList/../Endpoint/	Indicates the type of transport method that is being used between access points.

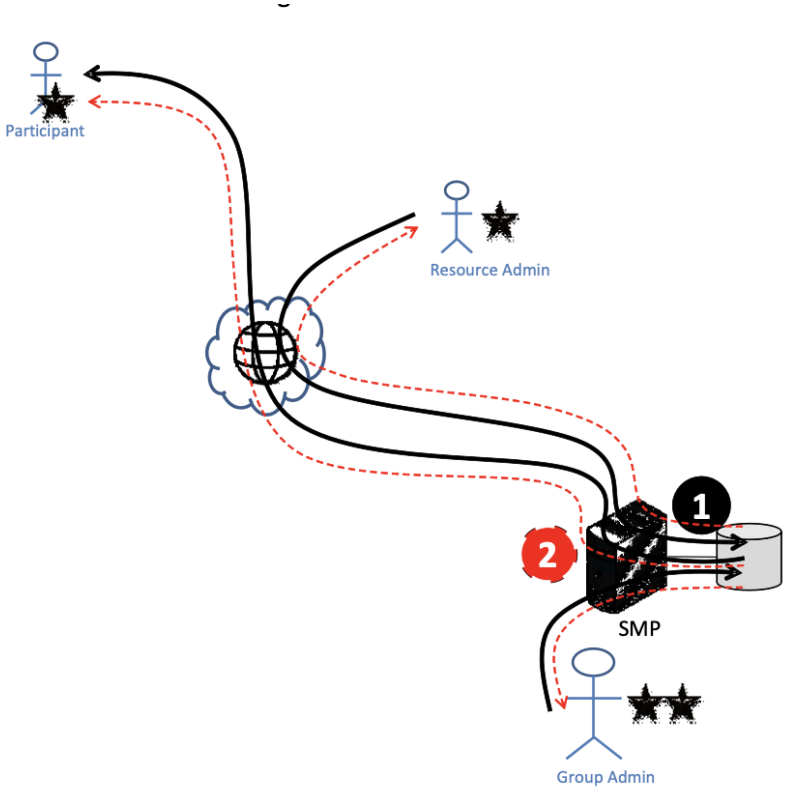
XSD element	Description
requireBusinessLevelSignature Oasis SMP 1.0 Element: ServiceInformation/ ProcessList/../Endpoint/ RequireBusinessLevelSignature Oasis SMP 2.0 Element: /	Set to “true” if the recipient requires business-level signatures for the message, meaning a signature applied to the business message before the message is put on the transport. This is independent of the transport-level signatures that a specific transport profile might mandate. This flag does not indicate which type of business-level signature might be required. Setting or consuming business-level signatures would typically be the responsibility of the final senders and receivers of messages, rather than a set of gateways.
minimumAuthenticationLevel Oasis SMP 1.0 Element: ServiceInformation/ ProcessList/../Endpoint/ MinimumAuthenticationLevel Oasis SMP 2.0 Element: /	Indicates the minimum authentication level that recipient requires. The specific semantics of this field is defined in a specific instance of a 4-corner infrastructure.

XSD element	Description
serviceActivationDate Oasis SMP 1.0 Element: ServiceInformation/ ProcessList/../Endpoint/ ServiceActivationDate Oasis SMP 2.0 Element: sma:ProcessMetadata/sma:Endpoint/smb:ActivationDate	Activation date of the service. Senders should ignore services that are not yet activated. Format of ServiceActivationDate date is xs: dateTime.
serviceExpirationDate Oasis SMP 1.0 Element: /ProcessList/../Endpoint/ ServiceExpirationDate Oasis SMP 2.0 Element: sma:ProcessMetadata/sma:Endpoint/smb:ExpirationDate	Expiration date of the service. Senders should ignore services that are expired. Format of `ServiceExpirationDate` date is `xs:dateTime`.
certificate Oasis SMP 1.0 Element: /ProcessList/../Endpoint/ Certificate Oasis SMP 2.0 Element: /ServiceMetadata/sma:ProcessMetadata/sma:Endpoint/sma:Certificate	Holds the complete **[X509v3]** signing certificate of the recipient gateway, as a PEM base 64 encoded DER formatted value.
serviceDescription Oasis SMP 1.0 Element: /ProcessList/../Endpoint/ ServiceDescription Oasis SMP 2.0 Element: /	A human-readable description of the service

XSD element	Description
technicalContactUrl Oasis SMP 1.0 Element: /ProcessList/../Endpoint/ TechnicalContactUrl Oasis SMP 2.0 Element: /ServiceMetadata/sma:ProcessMetadata/sma:Endpoint/smb:Contact	Represents a link to human-readable contact information. This might also be an email address.
technicalInformationUrl Oasis SMP 1.0 Element: /ProcessList/../Endpoint/ TechnicalInformationUrl Oasis SMP 2.0 Element:	A URL to human-readable documentation of the service format. This could for example be a website containing links to XML Schemas, WSDLs, Schematrons and other relevant resources.
extension Oasis SMP 1.0 Element: /Process/Extension Oasis SMP 2.0 Element: /ServiceMetadata/sma:ProcessMetadata/ext:SMPExtensions	The extension element may contain any XML element. Clients MAY ignore this element. It can be used to add extension metadata to the process metadata block as a whole.
extension Oasis SMP 1.0 Element: /ServiceInformation/ Extension Oasis SMP 2.0 Element: /	The extension element may contain any XML element. Clients MAY ignore this element. It can be used to add extension metadata to the service metadata.

XSD files

Two XSDs are used to support the overall processes as defined in User Administration Process.



1 Original official OASIS SMP XSD

the standard XSD as published by OASIS and which defines the interface for the storage and retrieval of participant's information, see Original official OASIS SMP XSD.

2 Extended SMP XSD

defines the structure of error messages, see Extended SMP XSD.

Use Cases Overview

Actors

System Admin	
	A user granted rights to administer the Domain Admin type of users. This role is symbolised by 4 stars (it has the highest authority). The system admin is application role with permissions to configure SMP systems settings.
Domain Admin	

	A user granted to administer the Domain groups. The domain groups are group of users in the domain/network. This role has the authority to create/delete groups and manage their memberships.
Group Admin	
	A user granted rights to administer the participants for the group. The group admin administers the resources (Service groups) and assigns the memberships to the resources.
Resource Admin	
	A user granted rights to administer the national access points (i.e. one or more ServiceGroups); i.e. to define the access points services metadata. This role is symbolised by 1 single stars (it has the authority to administer (update) the resource (service groups) and subresources (ServiceMetadata), but cannot create or delete the resource).
User	Any participant sending documents to any other receiver participant and consulting SML in that purpose. This role is symbolised by no single star since he has only public read accesses.

`` In addition to the roles described above, **two additional relevant terms are used**:

Sender

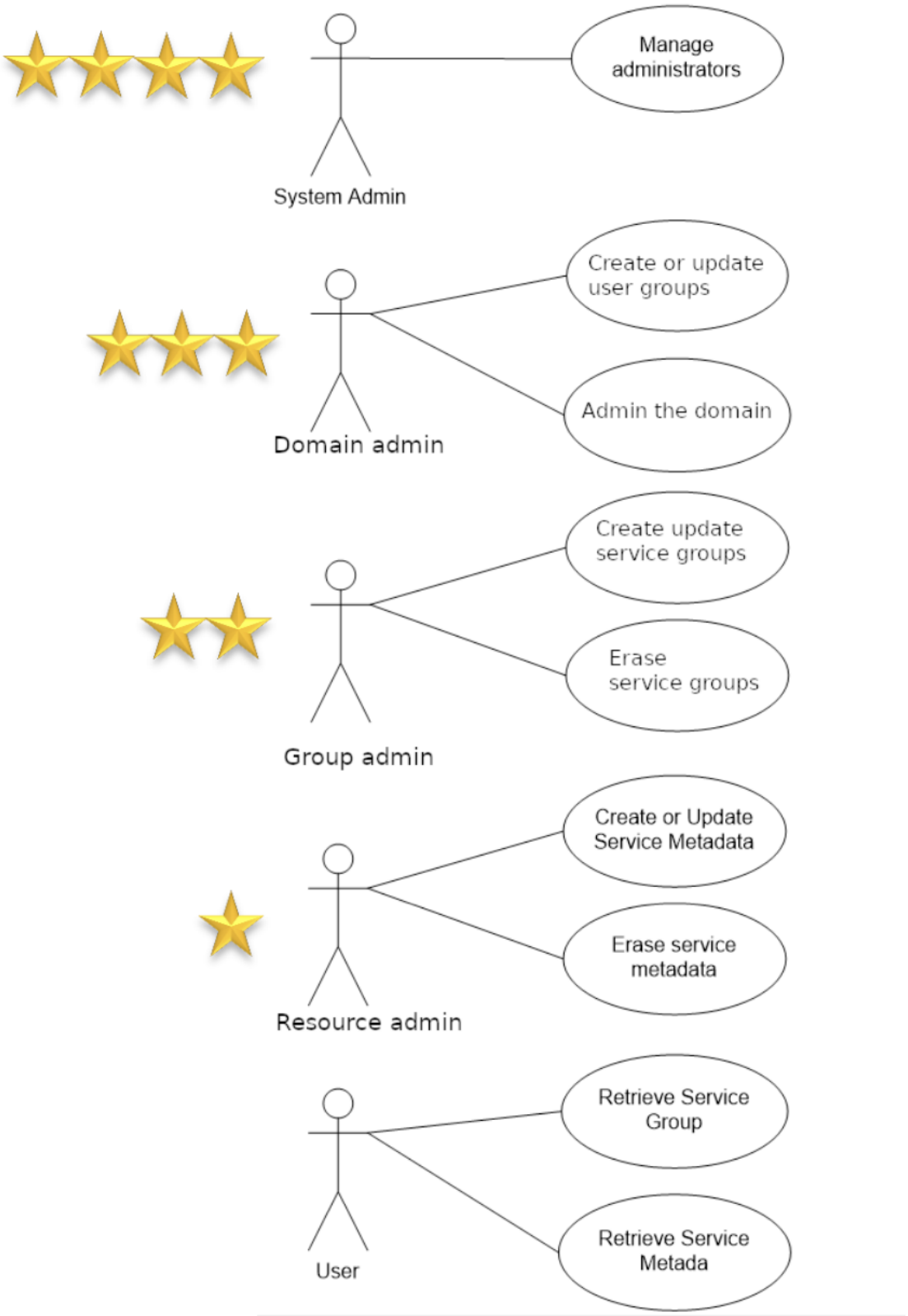
refers to an actor who uses the system (SML) on the left hand side of the *four corner model* introduced in eDelivery in a nutshell. In the current use cases, the *sender* only behaves as a `User` (see the *Actors* list above).

Receiver

refers to an actor who uses the system (SML) on the right hand side of the same model. In the current use cases, the *receiver* behaves as `Resource Admin`.

The `System Admin` being neither on the left nor on the right of that model, but rather on top of it, is never referred to as *sender* nor as *receiver*.

Use Cases Diagrams



Use Cases Diagram

Use Cases List

▼ **Use Cases List**

ID	Actor	UC	Description	Operation	Data
UC01	System Admin	**Manage Administrators**	Create and modify user information in the SMP *Aministrator* table.	N/A	User (table)
UC02	Group Admin	**Create or Update Resources/Service Group**	Create a new ServiceGroup for a new receiver participant. This service stores the *Service Group* and links it to the specified pair `participantIdentifier` + `participantIndentifierScheme`. Information is store into `ServiceGroup` table. This same service is used to create and update a `ServiceGroup`.	`PUT`	`ServiceGroup`
UC03	Group Admin	**Erase Resource/Service Group**	Erases the service group definition AND the list of services for the specified receiver participant.	`DELETE`	`ServiceGroup`
UC04	Resource Admin	**Create or Update Service Metadata**	Publish detailed information about one specific document service (multiple processes and endpoints). This same service is used to create and update ServiceMetaData.	`PUT`	`ServiceMetadata`
UC05	Resource Admin	**Erase Service Metadata**	Remove all information about one specific service (i.e. all related processes and endpoints definitions).	`DELETE`	`ServiceMetadata`
UC06	User	**Retrieve Service Group**	Obtain the list of services provided by a specific receiver participant (collection of references to the ServiceMetaData's) This service provides the information related to the *Service Group* according to the input pair `participantIdentifier` `participantIndentifierScheme`. Returns information from the table `ServiceMetadata` only (references to actual MetaData).	`GET`	`ServiceGroup`

ID	Actor	UC	Description	Operation	Data
UC07	User	**Retrieve Service Metadata**	Obtain detailed definition about one specific service of a specific participant for all supported transport. This service retrieves the `SignedServiceMetadata` according to the input tuple `participantIdentifier`participantIdentifierScheme`documentIdentifier+documentIdentifierScheme`. Returns information from the Endpoint table.	`GET`	`SignedServiceMetadata`

▼ **User Stories**

Below we list a typical use cases sequence thinking of a real-life example.
When the sequence step involves the SML system, the corresponding SML Use Case ID is referenced. Otherwise, if the step only involves SML partially, *UC N/A* is mentioned.

1. **(UC01) As a *System Admin***, I create a new `Group Admin` to allow the creation and the management of a new `ServiceGroup` for a participant.
2. **(UC01) As a *System Admin***, I create a new `Resource Admin` to allow the creation and the management of the metadata of that new ServiceGroup.
3. **(UC02) As a *Group Admin***, I create a new `ServiceGroup` and link it to the administrator `Resource Admin` to allow the management of `ServiceMetadata` for the related participant.
4. **(UC04) As a *Resource Admin***, I define all the `ServiceMetadata` for the participant that I administer.
5. **(UC N/A) As a *User***, I ask the DNS to resolve the address of SML hosting the receiver's metadata.
6. **(UC07) As a *User***, I retrieve the definition of the service (metadata) I need to invoke to send a document to the receiver.
7. **(UC N/A) As a *User***, I send the document to the receiver.

Administration Use Cases

The *Administration* use cases purpose is to help help the different types of administrators defining all services (`ServiceGroup` and `ServiceMetada`).

UC01 Manage Administrators

▼ *UC01 Use Case Description*

This use case introduces the foundation for an administration console: creating an 'Group Admin' user is the task of superuser, and no REST service shall consequently support that functionality. As this is a necessary functionality, this one should be included into the administration console.

Brief description

Create and modify administrator information in SMP table `Administrator`.

NOTE	This is a temporary solution planned to be later replaced by functionality in the administration console.

Actors

- System Admin

Preconditions

The actor (system admin) has all access rights to modify content of SMP configuration tables.

Basic Flow

Step	Description
1	System admin creates a new administrator in table 'Administrator'
2	Use case ends with success

Alternative Flow

Step	Description
1a	**Administrator must be removed**
1a1	System admin removes all ServiceGroup definitions linked to that administrator by calling "DeleteServiceGroup" SMP service for all ServiceGroups this administrator is linked to (as defined by the "ownership" relationship).
1a2	System admin removes the administrator from table 'Administrator'.
1b	**New administrator must take over administration of some participant(s)**
1b1	After creating the new user (step 1), the system admin reassigns specific ServiceGroups to that user by changing the 'username' foreign key in table Ownership.
1b2	Use case ends.
1c	**Administrator already exists and must be modified**
1c1	System admin modifies some data (role, password) of the user in table 'User'.
1c2	Use case ends.

Post Conditions

N/A

Successful Conditions

- Administrator definition has been modified

Failure Conditions N/A

▼ *REST Service: None*

This functionality should be implemented into the administrator's console of SML which is not

further detailed it the present document.

UC02 Create/Update Service Group

▼ *UC02 Use Case Description*

Brief description

Create a new `ServiceGroup` for a new receiver participant. This service stores the Service Group and links it to the specified pair `participantIdentifier` + `participantIndentifierScheme`. Information is stored into `ServiceGroup` table. This same service is used to create and update a `ServiceGroup`.

Actors

- Group Admin

Preconditions

- The authenticated user has the role of `Group Admin`.
- If the ServiceGroup is managed remotely, the *Resource Admin* must have been created before in the `Administrator` table.
- Identifier and scheme of the service group provided in the request must comply to the policy defined in OpenPEPPOL AISBL - Policy for use of Identifiers.
- If SML is serving multiple domains, the header field `Domain` must be populated and refer to one of the domains served by SML.

Basic Flow

Step	Description
1	The receiver declares its service group and the related Administrator (Resource Admin) to SML.
2	The SMP authenticates the user, validates the request, and add or replace the information into its configuration database and passes the information to SML.
3	The receiver receives the confirmation that the definitions were stored properly with HTTP response "201 Created".
4	Use case ends with success.

Alternative Flow

3a	**ServiceGroup already exists**
3a1	The receiver receives the confirmation that the definitions were updated properly with HTTP response `200 OK`.
3a2	Use case ends with success.

Exception Flow

1a	**SMP is not reachable**
1a1	The user receives a network connection error.

1a	**SMP is not reachable**
1a2	Use case ends.
2a	**Authentication / Authorization fails**
2a1	The SMP replies with HTTP error "401 Unauthorized"
2a2	The receiver receives the error message
2a3	Use case ends.
2b	**Request is not well formed (or any other business/technical error)**
2b1	The SMP replies with HTTP error "400 Bad request" or "500 Internal server error" with details on the error allowing to identify the error in the request (see Error Codes Table).
2b2	The receiver receives the error message.
2b3	Use case ends.
2c	**SMP is serving multiple domains and the Domain field is not specified in the header**
2c1	The SMP replies with HTTP error `400 Bad request` (business code `WRONG_FIELD`) with message: "SMP is configured to use multiple domains, but no Domain is specified in request. Please specify Domain in request."
2c2	The receiver receives the error message.
2c3	Use case ends.
2d	**Domain field refers to a domain that is not served by SML**
2d1	The SMP replies with HTTP error "400 Bad request" (business code WRONG_FIELD) with message: Requested domain does not exist " (followed by the domain field value).
2d2	The receiver receives the error message.
2d3	Use case ends.

Post Conditions

N/A

Successful conditions

- ServiceGroup is either created or updated, and the corresponding *Resource Admin* is defined.

Failure conditions

- In case of error, no change occurs into the configuration database and the response gives technical details on the exception condition.

▼ *REST Service:* `PutServiceGroup`

Input

- **In the URL:**

- The participant's identifier and identifier's scheme (`ParticipantIdentifier`).

- **In the header (optional fields):**
  - the Certificate Identifier required for authenticating the remote user as "Resource Admin" for this service group.

    NOTE If the Certificate Identifier is not provided, the "Group Admin "will manage the metadata of that Service Group – the username of the basic authentication is used to identify the "Group Admin " to link him with the Service Group.

  - The "Domain" for which the ServiceGroup must be created.

    NOTE this field is optional and relevant only if SML is serving multiple domains. In that case this field must be provided.

- **In the TEXT:**
  a `ServiceGroup` structure as defined in the standard OASIS XSD (OASIS SMP XSD) containing:
  - The Participant's identifier and scheme that uniquely identifies this service group;These must be identical to the ones provided in the URL.
  - Optionally, the Extension information in the HTTP TEXT.

`ServiceGroup-Owner` *structure details:*

- The following attributes of the certificate are used in this order:
  - CN,
  - O,
  - C, and
  - Serial number

An example is the following certificate,

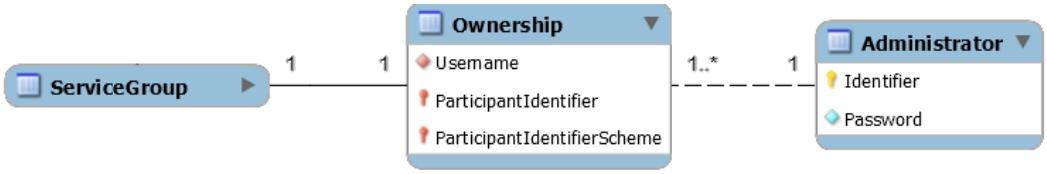
```
sno=0001&subject=EMAILADDRESS=receiver@test.be, CN=SMP_receiverCN,
OU=B4, O=DIGIT, L=Brussels, ST=BE, C=BE:df48f09389f034&validfrom=Jun 1
10:37:53 2015 CEST&validto=Jun 1 10:37:53 2035
CEST&issuer=EMAILADDRESS=root@test.be,CN=rootCN,OU=B4,O=DIGIT,L=Brussels,ST=BE,C=BE
```

would be provided in the HTTP header:

```
ServiceGroup-Owner: CN=SMP_receiverCN, O=DIGIT, C=BE:df48f09389f034
```

Execution

- Start a new transaction.
- Create or update (overwrites) the corresponding rows in the configuration, ownership and ServiceGroup identified by the participant's identifier and identifier's scheme keys:



- *If* attribute `ServiceGroup-Owner` is present in the HTTP Header, *then* use this as information to store as `Identifier`. *Else*, store instead the basic authentication information provided in the HTTP header.

In the case of:

- a newly created `ServiceGroup`, invoke "SML's Create Business Identifier" service.
- an existing `ServiceGroup`, invoke "SML's Delete Business Identifier" and the "Create Business Identifier" services.
- If the service invocation:
  - succeeds, commit the transaction.
  - fails, rollback the transaction.
- If necessary (Delete succeeded), try to reinvoke the "Create Business Identifier" service with the old information to restore SML properly;
  - The Response to this service is "failure".

Output

- Return a response confirming the success (or eventually the failure) of the operation.

Sample Request

HTTP Header (No `AdminServiceGroup` authentication information – Group Admin creates or updates the `ServiceGroup`)

`PUT` http://smp-digit-mock.publisher.ehealth.acc.edelivery.tech.ec.europa.eu/ehealth-actorid-qns::urn:poland:ncpb HTTP/1.1

Host: `smp-digit-mock.publisher.ehealth.acc.edelivery.tech.ec.europa.eu`
Accept: `application/xml`
Content-Type: `application/xml`
Authorization: `Basic dXNlcjpwYXNz`
Content-Length: `278`

HTTP Header (Resource Admin authentication information is certificate)

Host: `smp-digit-mock.publisher.ehealth.acc.edelivery.tech.ec.europa.eu`
Accept: `application/xml`
Content-Type: `application/xml`
ServiceGroup-Owner: `CN=SMP_1000000181,O=DIGIT,C=DK:406b2abf0bd1d46ac4292efee597d414`
Authorization: `Basic dXNlcjpwYXNz`
Content-Length: `278`

Text

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
    <ServiceGroup xmlns="http://docs.oasis-open.org/bdxr/ns/SMP/2016/05">
        <ParticipantIdentifier
            scheme="ehealth-actorid-qns">urn:poland:ncpb</ParticipantIdentifier>
        <ServiceMetadataReferenceCollection/>
</ServiceGroup>
```

Sample Response

HTTP header HTTP/1.1 201 Created

Server: `Apache-Coyote/1.1` Pragma: `No-cache` Expires: `Thu, 01 Jan 1970 01:00:00 CET` Content-Length: `0` Date: `Wed, 27 Jan 2016 10:32:40 GMT` Cache-Control: `no-cache, proxy-revalidate` Connection: `Keep-Alive`

NOTE	If the `ServiceGroup` previously existed, `200 OK` is returned as HTTP response instead of `201 Created` as showed in the above example.

Text

N/A

UC02 Error Codes

HTTP code	HTTP Message	Business code	Meaning
`200`	`OK`	N/A	The request was completed successfully.
`201`	`Created`	N/A	The PUT operation completed successfully.
`400`	`Bad Request`	`XSD_INVALID`	The XML included in the request is not validate against the XSD defining the input structure.
		`MISSING_FIELD`	Some field that is optional in the XSD but mandatory for this invocation is missing (missing field name in description).
		`WRONG_FIELD` aa	Some field is valid against XSD definition, but the more specific content is invalid (erroneous field name in description). + Or + Some header field is either missing or invalid.
		`FORMAT_ERROR`	Some field is expected to have a specific format is not valid (erroneous field name in description).
		`USER_NOT_FOUND`	The referenced " Resource Admin" was not found as Administrator.

HTTP code	HTTP Message	Business code	Meaning
401	Unauthorized	UNAUTHORIZED	The user is not granted the right to issue this request.
404	Resource not found	NOT_FOUND	The requested information was not found.
500	Internal Server Error	TECHNICAL	Some unexpected technical error occurred (detailed information is available in the response).

NOTE The business code and the description are both in the response, and are compliant with `ErrorResponseType`as described in Errors Structure Detail.

Audit Information

Audit information that must be audited per service:

Logged Info	UC1	UC2	UC3	UC4	UC5	UC6	UC7
AdministratorIdentifier	-	x	x	x	x	-	-
AccessTime	-	x	x	x	x	x	x
Operation	-	x	x	x	x	x	x
ParticipantIdentifier	-	x	x	x	x	x	x
ParticipantIdentifierScheme	-	x	x	x	x	x	x
DocumentIdentifier	-	-	-	x	x	-	x
DocumentIdentifierScheme	-	-	-	x	x	-	x
IpAddress	-	x	x	x	x	x	x
RequestHeader	-	x	x	x	x	x	x
RequestText	-	x	-	x	x	?	?
ResponseHeader	-	x	x	x	x	x	x
ResponseText	-	-	-	-	-	x	x
HTTP code	-	x	x	x	x	x	x
Business code	-	x	x	x	x	-	-
ErrorDescription	-	x	x	x	x	-	-

NOTE See also Auditing.

UC03 Erase Service Group

▼ *UC03 Use Case Description*

Brief description

Erases the service group definition *and* the list of services for the specified receiver participant.

Actors

- Group Admin

Preconditions

- The authenticated user has the role of `Group Admin`.
- Referenced service group was previously defined.

Basic Flow

Step	Description
1	The receiver requests its service group to be removed from SML.
2	The SMP authenticates the user, validates the request, and removes all the information on the service group from its configuration and from SML.
3	The receiver receives the confirmation that the definitions were removed properly with HTTP response `200 OK`.
4	Use case ends with success.

Exception flows

1a	**SMP is not reachable**
1a1	The user receives a network connection error.
1a2	Use case ends.
2a	**Authentication / authorization fails**
2a1	The SMP replies with HTTP error `401 Unauthorized`.
2a2	The receiver receives the error message.
2a3	Use case ends.
2b	**Request is not well formed (or any other business/technical error)**
2b1	The SMP replies with HTTP error `400 Bad request` or `500 Internal server error` with details on the error allowing to identify the error in the request. see the Error codes table below.
2b2	The receiver receives the error message.
2b3	Use case ends.
2c	**ServiceGroup is not defined**
2c1	The SMP replies with HTTP error "404 Resource not found".
2c2	The receiver receives the error message.
2c3	Use case ends.

Post Conditions

N/A

Successful conditions

- The specified service group is removed with all its related information.

Failure conditions

- In case of error, no change occurs into the configuration database and the response gives technical details on the exception condition.

▼ *REST Service:* `DeleteServiceGroup`

Input

ServiceGroup identifier: `ParticipantIdentifier`, `ParticipantIdentifierScheme` in the HTTP header.

Execution

- The username or the certificate from the HTTP header is verified to be the owner of the specified Service Group. If not, the operation is rejected.
- Start a new transaction.
- Delete ALL information related to that service group in tables: Endpoint, Process, ServiceMetadata and finally the ServiceGroup itself where the `ParticipantIdentifiers` match the specified `ServiceGoup` identifier.
- Invoke SML service “Delete Business Identifier”.
- If SML service invocation:
  - succeeds, commit the transaction;
  - fails, rollback the transaction.
    - Response to this service is “failure”.

Output

- HTTP `200` if done,
- HTTP `404` if the specified service group does not exist
- HTTP `500` if any error occurred.

Sample Request

HTTP Header

`DELETE` http://smp-digit-mock.publisher.ehealth.acc.edelivery.tech.ec.europa.eu/ehealth-actorid-qns::urn:poland:ncpb HTTP/1.1

Host: `smp-digit-mock.publisher.ehealth.acc.edelivery.tech.ec.europa.eu`
User-Agent: `Mozilla/5.0 (Windows NT 6.1; WOW64; rv:45.0) Gecko/20100101 Firefox/45.0`
Accept: `application/xml`
Accept-Language: `en-GB,en;q=0.8,de;q=0.5,fr;q=0.3`
Accept-Encoding: `gzip, deflate`
DNT: `1`
Referer: `http://130.206.118.4/smp-swagger-ui/`
Origin: `http://130.206.118.4`
Proxy-Authorization: `Basic dXNlcm5hbWU6cGFzc3dvcmQ=`

Connection: `keep-alive`

Text

N/A

Sample Response

```
HTTP header
HTTP/1.1 200 OK
Connection: Keep-Alive
Content-Encoding: gzip
Content-Length: 20
Content-Type: application/xml
Date: Thu, 22 Dec 2016 10:47:56 GMT
Server: Jetty(6.1.26)
Set-Cookie: BCIDSLB=PS1LUX-56; domain=europa.eu; path=/; HttpOnly
access-control-allow-origin:*
```

Text

N/A

UC05 Error codes

HTTP Code	HTTP Message	Business Code	Description
`200`	OK	N/A	The request was completed successfully.
`400`	`Bad Request`	`FORMAT_ERROR`	Some field is expected to have a specific format is not valid (erroneous field name in description).
`401`	`Unauthorized`	`UNAUTHORIZED`	The user is not granted the right to issue this request.
`404`	`Resource not found`	`NOT_FOUND`	The requested information was not found.
`500`	`Internal Server Error`	`TECHNICAL`	Some unexpected technical error occurred (detailed information is available in the response).

Audit Information

Audit information that must be audited per service:

Logged Info	UC1	UC2	UC3	UC4	UC5	UC6	UC7
`AdministratorIdentifier`	-	x	x	x	x	-	-
`AccessTime`	-	x	x	x	x	x	x
`Operation`	-	x	x	x	x	x	x
`ParticipantIdentifier`	-	x	x	x	x	x	x

Logged Info	UC1	UC2	UC3	UC4	UC5	UC6	UC7
ParticipantIdentifierScheme	-	x	x	x	x	x	x
DocumentIdentifier	-	-	-	x	x	-	x
DocumentIdentifierScheme	-	-	-	x	x	-	x
IpAddress	-	x	x	x	x	x	x
RequestHeader	-	x	x	x	x	x	x
RequestText	-	x	-	x	x	?	?
ResponseHeader	-	x	x	x	x	x	x
ResponseText	-	-	-	-	-	x	x
HTTP code	-	x	x	x	x	x	x
Business code	-	x	x	x	x	-	-
ErrorDescription	-	x	x	x	x	-	-

NOTE See also Auditing.

UC04 Create/Update Service Metadata

▼ *UC04 Use Case Description*

Brief description

Publish detailed information about one specific document service (multiple processes and endpoints).
This same service is used to create and update ServiceMetadata.

SEE ALSO **2.1 Discovery Flow** in PEPPOL Transport Infrastructure Service Metadata Publishing (SMP).

A sender (ed. user) may want to discover what document types can be handled by a specific participant identifier. Such discovery is relevant for applications supporting several equivalent business processes. Knowing the capabilities of the recipient is valuable information to a sender application and ultimately to an end user. For example, the end user may be presented with a choice between a simple and a rich business process.
This is enabled by a pattern where the sender first retrieves the ServiceGroup entity, which holds a list of references to the ServiceMetadata resources associated with it. The ServiceMetadata in turn holds the metadata information that describes the capabilities associated with the recipient participant identifier.

Actors

- Resource Admin

Preconditions

- The authenticated user has the `Resource Admin` role.
- `Resource Admin` user initiating the request is linked to the specified `ServiceGroup`.

- The `Resource Admin's certificate:
  - is valid;
  - was previously stored in the configuration.
- Identifier and scheme of the service group and documents provided in the request comply to the policy defined in OpenPEPPOL AISBL - Policy for use of Identifiers.

Basic Flow

Step	Description
1	The receiver requests its service metadata to be put into SML.
2	The SMP verifies the certificate of the " Resource Admin" against its information in the database, validates the request, and either create or update all the information into its configuration database.
3	The receiver receives the confirmation that the definitions were created properly with HTTP response "201 Created".
4	Use case ends.

Alternative Flow

3a	**ServiceMetadata already exists**
3a1	The receiver receives the confirmation that the definitions were updated properly with HTTP response `200 OK`.
3a2	Use case ends with success.

Exception Flow

1a	**SMP is not reachable**
1a1	The user receives a network connection error.
1a2	Use case ends with success.
2a	**Authentication / authorization fails**
2a1	The SMP replies with HTTP error "401 Unauthorized".
2a2	The receiver receives the error message.
2a3	Use case ends.
2b	**Request is not well formed (or any other business/technical error)**
2b1	The SMP replies with HTTP error "400 Bad request" or "500 Internal server error" with details on the error allowing to identify the error in the request (see the Error Codes Table).
2b2	The receiver receives the error message.
2b3	Use case ends.
2c	**ServiceGroup is not defined**

1a	SMP is not reachable
2c1	The SMP replies with HTTP error "404 Resource not found".
2c2	The receiver receives the error message.
2c3	Use case ends.

Post Conditions

N/A

Successful conditions

- ServiceMetadata is defined

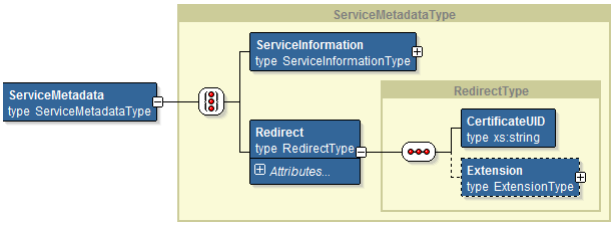
Failure conditions

- In case of error, no change occurs into the configuration database and the response gives technical details on the exception condition.

▼ *REST Service:* `PutServiceMetadata`

Input

- `ServiceGroup` and Document's identifiers in the URL and
- `ServiceMetadata` in the text.



This input structure, from the `ServiceInformation` node down to the Process leaves, will fully define the content of the referenced service metadata as defined by the four identifiers of the participant AND related specific document.

This means that the configuration of a Service must be done with a single call (for all `Processes`) to this service, and it can be considered that all previously existing information in `ServiceInformation`, Process and Endpoint tables are discarded (if they exist) and completely replaced by the newly provided information.

Execution

- Start a new transaction.
- Insert or replace the all the `ServiceInformation` for that `ServiceGroup` Document.
- In case of error:
  - Rollback the transaction.
  - Response to this service is "failure".
- If no error occurred:
  - Commit the transaction

- Response to this service is “success”.

Authorization

The operation will be allowed if and only if the authenticated user matches the `Resource Admin` user linked to the `ServiceGroup`.

For this user to be the eligible `Resource Admin` it must have been referenced as such in the ServiceGroup definition (see `PutServiceGroup`) by an `Group Admin` user via service `PutServiceGroup` (by the `Group Admin` who was previously defined by the `Domain Admin`).

All the provided information can be either;

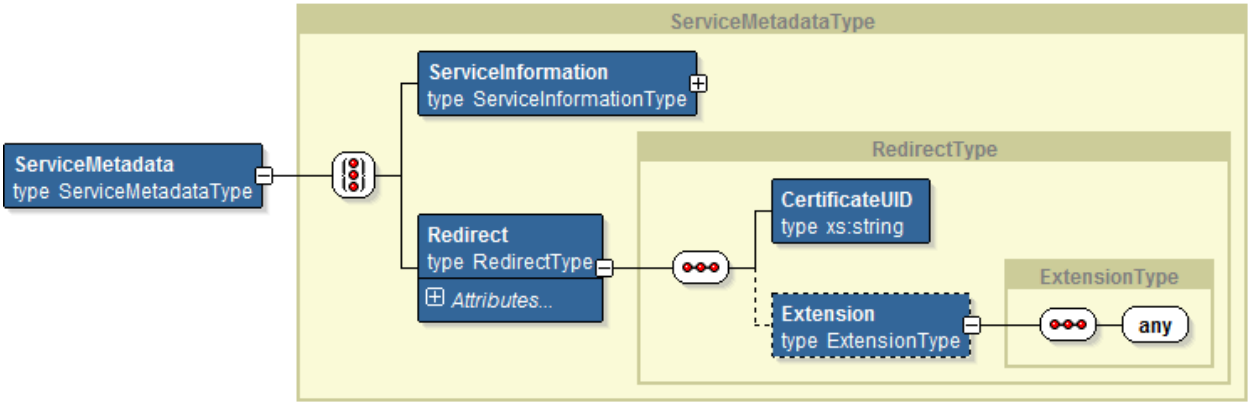
- *created* in the configuration (PUT = *create*) or
- *overwritten* (PUT =*update*);

this means, `PUT` does both.

Redirection

As explained above, in some cases ServiceMetadata information can be stored in ‘another SMP’; i.e., another SMP than the one that is queried by the user. In such case, ‘redirect’ information is provided to the user to allow him to query the appropriate SMP for obtaining the ServiceMetadata information from the relevant SMP.

For that to be possible, the receiver must eventually be able to store that redirect information. That is why this service provides this possibility, by allowing provision of “Redirect” information instead of the “ServiceInformation” itself:



The fields are in used as follows:

- `CertificateUID`: holds the Subject Unique Identifier of the certificate of the destination SMP. A client SHOULD validate that the Subject Unique Identifier of the certificate used to sign the resource at the destination SMP matches the Subject Unique Identifier published in the redirecting SMP.
- `href` attribute of the Redirect element contains the full address of the destination SMP record that the client is redirected to.
- `Extension`: not defined and optional.

NOTE **About Cascaded redirections**

> In the case where a client encounters such a redirection element, the client MUST follow the first redirect reference to the alternative SMP. If the SignedServiceMetadata resource at the alternative SMP also contains a redirection element, the client SHOULD NOT follow that redirect. It is the responsibility of the client to enforce this constraint.

Output

HTTP response code 200 if ok, 401 if not allowed and 400 if any other error occurred. Details are available in the response text.

Sample Request 1

This example sends actual information of the service and uses a certificate in the header.

HTTP Header (with certificate)

PUT http://smp-digit-mock.publisher.ehealth.acc.edelivery.tech.ec.europa.eu/ehealth-actorid-qns::urn:poland:ncpb/services/ehealth-resid-qns::urn::epsos##services:extended:epsos::107

HTTP/1.1

Accept-Encoding: gzip,deflate
Content-Type: text/xml;charset=UTF-8
Client-Cert: sno=0001&subject=EMAILADDRESS=receiver@test.be, CN=SMP_receiverCN, OU=B4, O=DIGIT, L=Brussels, ST=BE, C=BE&validfrom=Jun 1 10:37:53 2015 CEST&validto=Jun 1 10:37:53 2035 CEST&issuer=EMAILADDRESS=root@test.be,CN=rootCN,OU=B4,O=DIGIT,L=Brussels,ST=BE,C=BE
Host: smp-digit-mock.publisher.ehealth.acc.edelivery.tech.ec.europa.eu
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:45.0) Gecko/20100101 Firefox/45.0

Accept: application/xml
Accept-Language: en-GB,en;q=0.8,de;q=0.5,fr;q=0.3
Accept-Encoding: gzip, deflate
DNT: 1
Content-Type: application/xml
Referer: http://130.206.118.4/smp-swagger-ui/
Content-Length: 4741
Origin: http://130.206.118.4
Proxy-Authorization: Basic dXNlcm5hbWU6cGFzc3dvcmQ=
Connection: keep-alive

> **NOTE** The `Client-Cert` value in the HTTP header above is only an example that is specific to production and acceptance environments at DIGIT and should not be considered as constraining.

Text (Information)

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<ServiceMetadata xmlns="http://docs.oasis-open.org/bdxr/ns/SMP/2016/05">
<ServiceInformation>
<ParticipantIdentifier scheme="ehealth-actorid-qns">
```

```
urn:poland:ncpb</ParticipantIdentifier>
<DocumentIdentifier
    scheme="ehealth-resid-qns">urn::epsos##services:extended:epsos::107
</DocumentIdentifier>
<ProcessList>
    <Process>
            <ProcessIdentifier scheme="ehealth-procid-qns">
                urn:epsosPatientService::List</ProcessIdentifier>
            <ServiceEndpointList>
            <Endpoint transportProfile="urn:ihe:iti:2013:xcpd">
            <EndpointURI>http://poland.pl/ncp/patient/list</EndpointURI>
            <RequireBusinessLevelSignature>false</RequireBusinessLevelSignature>
            <MinimumAuthenticationLevel>urn:epSOS:loa:1</MinimumAuthenticationLevel>
            <ServiceActivationDate>2016-06-
06T11:06:02.000+02:00</ServiceActivationDate>
            <ServiceExpirationDate>2026-06-06T11:06:02+02:00</ServiceExpirationDate>

<Certificate>MIID7jCCA1egAwIBAgICA+YwDQYJKoZIhvcNAQENBQAwOjELMAkGA1UEBhMCRlIxEzARBgN
VBAoMCklIRSBFdXJvcGUxFjAUBgNVBAMMDUlIRSBFdXJvcGUgQ0EwHhcNMTYwNjAxMTQzNTUzWhcNMjYwNjA
xMTQzNTUzWjCBgzELMAkGA1UEBhMCUFQxDDAKBgNVBAoMA01vSDENMAsGA1UECwwEU1BNUzENMAsGA1UEKgw
ESm9hbzEOMAwGA1UEBRMFQ3VuaGExHTAbBgNVBAMMFHFhZXBzb3MubWluLXNhdWRlLnB0MRkwFwYDVQQMDBB
TZXJ2aWNlIFByb3ZpZGVyMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA1eN4qPSSRZqjVFG9Tlc
Plxf2WiSimQK9L1nf9Z/s0ezeGQjCukDeDq/Wzqd9fpHhaMMq+XSSOtyEtIr5K/As4kFrViONUUkG12J6Ull
SWogp0NYFwA4wIqKSFiTnQS5/nRTs05oONCCGILCyJNNeO53JzPlaq3/QbPLssuSAr6XucPE8wBBGM8b/TsB
2G/zjG8yuSTgGbhaZekq/Vnf9ftj1fr/vJDDAQgH6Yvzd88Z0DACJPHfW1p4F/OWLI386Bq7g/bo1DUPAyEw
lf+CkLgJWRKki3yJlOCIZ9enMA5O7rfeG3rXdgYGmWS7tNEgKXxgC+heiYvi7ZWd7M+/SUwIDAQABo4IBMzC
CAS8wPgYDVR0fBDcwNTAzoDGgL4YtaHR0cHM6Ly9nYXplbGxlLmloZS5uZXQvcGtpL2NybC82NDMvY2Fjcmw
uY3JsMDwGCWCGSAGG+EIBBAQvFi1odHRwczovL2dhemVsbGUuaWhlLm5ldC9wa2kvY3JsLzY0My9jYWNybC5
jcmwwPAYJYIZIAYb4QgEDBC8WLWh0dHBzOi8vZ2F6ZWxsZS5paGUubmV0L3BraS9jcmwvNjQzL2NhY3JsLmN
ybDAfBgNVHSMEGDAWgBTsMw4TyCJeouFrr0N7el3Sd3MdfjAdBgNVHQ4EFgQU1GQ/K1ykIwWFgiONzWJLQzu
fF/8wDAYDVR0TAQH/BAIwADAOBgNVHQ8BAf8EBAMCBSAwEwYDVR0lBAwwCgYIKwYBBQUHAwEwDQYJKoZIhvc
NAQENBQADgYEAZ7t1Qkr9wz3q6+WcF6p/YX7Jr0CzVe7w58FvJFk2AsHeYkSlOyO5hxNpQbs1L1v6JrcqziN
Frh2QKGT2v6iPdWtdCT8HBLjmuvVWxxnfzYjdQ0J+kdKMAEV6EtWU78OqL60CCtUZKXE/NKJUq7TTUCFP2fw
iARy/t1dTD2NZo8c=</Certificate>
            <ServiceDescription>
                This is the epSOS Patient Service List for the Polish NCP
            </ServiceDescription>
            <TechnicalContactUrl>http://poland.pl/contact</TechnicalContactUrl>
            <TechnicalInformationUrl>
http://poland.pl/contact</TechnicalInformationUrl>
            </Endpoint>
            </ServiceEndpointList>
        <Process>
    </ProcessList>
<Extension>
    <Signature
        xmlns=
"http://www.w3.org/2000/09/xmldsig#"><SignedInfo><CanonicalizationMethod
        Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
20010315"/><SignatureMethod
        Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/><Reference
```

```
        URI=""><Transforms><Transform
        Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature"/></Transforms><DigestMethod
        Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
    <DigestValue>
        CJeDJ72nQkwsZ2XWc8eput8pcBzfHSwO6uHr77/xbQo=
    </DigestValue>
    </Reference>
    </SignedInfo>
    <SignatureValue>
        WlCUwlHJy9sehansEjFXSPkAobodbeM8OxXfLjQVYs7Vh085dESYaAbcDoDZ6t8IaHbsRtkiCgZG
        yVRvOwB42EVRkhyWu0zVnlowfieBgvMqtZdYMbx6Z7Npwvo0UDcYI/HnHnzsyHhklKKNGPymXJXH
        waEt4QJw+ne2n7Tb0Qg=
    </SignatureValue>
    <KeyInfo>
    <X509Data>
    <X509SubjectName>
        CN=Sample National Infrastructure,OU=Sante,C=PT
    </X509SubjectName>
    <X509Certificate>
        MIICAzCCAWygAwIBAgIEWCRzHjANBgkqhkiG9w0BAQsFADBGMQswCQYDVQQGEwJQVDEOMAwGA1UE
        CwwFU2FudGUxJzAlBgNVBAMMHlNhbXBsZSBOYXRpb25hbCBJbmZyYXN0cnVjdHVyZTAeFw0xNjEx
        MTAxMzE2NTBaFw0yNjExMTAxMzE2NTBaMEYxCzAJBgNVBAYTAlBUMQ4wDAYDVQQLDAVTYW50ZTEn
        MCUGA1UEAwweU2FtcGxlIE5hdGlvbmFsIEluZnJhc3RydWN0dXJlMIGfMA0GCSqGSIb3DQEBAQUA
        A4GNADCBiQKBgQCywt50WXEWIiWytRGcMqzeMM/EyxruNthPdiUEUTbs9un7lzGGjpfFMTgd83wJ
        haB6FgpaVd8V2w/JBdkim5Ltuhu2vA0d6hHOsa58neIfe4z1ZhswwNmB0+mDTjwnd/gg8IJyQhhY
        c5G4x7m0ZGdDKZDizjtDTEPTsl8D4FzBFwIDAQABMA0GCSqGSIb3DQEBCwUAA4GBACKxUpAx0PYm
        ZZi4DfAzBkQ0+CvQw/l6Yo8wonVdpcQXO3khpWIcXhgYhTLHwm8IwJLEyFatmMyCKklSA3CLebJU
        L4XH1GcdCg6oPKPUc+ovbgN7/iR265Elp4qHfpVteBijBTyZReH4oAK9hRhK1gLwtjI7vpjVaPXv
        vkV1fbrz
    </X509Certificate>
    </X509Data>
    </KeyInfo>
    </Signature>
</Extension>
</ServiceInformation>
</ServiceMetadata>
```

Sample Response (applicable for both examples request above)

```
HTTP header
HTTP/1.1 201 Created
Server: Apache-Coyote/1.1
Pragma: No-cache
Expires: Thu, 01 Jan 1970 01:00:00 CET
Content-Length: 0
Date: Fri, 22 Jan 2016 09:46:10 GMT
Cache-Control: no-cache, proxy-revalidate
Connection: Keep-Alive
```

NOTE	if the `ServiceMetadata` previously existed, `200 OK` will be returned as HTTP response instead of `201 Created` as show in the above example.

Text N/A

UC06 Error codes

HTTP code	HTTP Message	Business code	Meaning
200	OK	N/A	The request was completed successfully.
201	Created	N/A	The PUT operation completed successfully.
400	Bad Request	XSD_INVALID	The XML included in the request is not validate against the XSD defining the input structure.
		MISSING_FIELD	Some field that is optional in the XSD but mandatory for this invocation is missing (missing field name in description).
		WRONG_FIELD	Some field is valid against XSD definition, but the more specific content is invalid (erroneous field name in description).
		OUT_OF_RANGE	Some numeric (or date field) is out of the valid range (erroneous field name in description).
		UNAUTHOR_FIELD	Some field that is optional in the XSD but forbidden for this invocation is present (unauthorized field name in description).
		FORMAT_ERROR	Some field is expected to have a specific format is not valid (erroneous field name in description).
		OTHER_ERROR	Some other specific error was encountered processing the request (more information in the ErrorDescription field).
401	Unauthorized	UNAUTHORIZED	The user is not granted the right to issue this request.
404	Resource not found	NOT_FOUND	The requested information was not found.
500	Internal Server Error	TECHNICAL	Some unexpected technical error occurred (detailed information is available in the response).

Audit Information

Audit information that must be audited per service:

Logged Info	UC1	UC2	UC3	UC4	UC5	UC6	UC7
AdministratorIdentifier	-	x	x	x	x	-	-
AccessTime	-	x	x	x	x	x	x
Operation	-	x	x	x	x	x	x
ParticipantIdentifier	-	x	x	x	x	x	x
ParticipantIdentifierScheme	-	x	x	x	x	x	x
DocumentIdentifier	-	-	-	x	x	-	x
DocumentIdentifierScheme	-	-	-	x	x	-	x
IpAddress	-	x	x	x	x	x	x
RequestHeader	-	x	x	x	x	x	x
RequestText	-	x	-	x	x	?	?
ResponseHeader	-	x	x	x	x	x	x
ResponseText	-	-	-	-	-	x	x
HTTP code	-	x	x	x	x	x	x
Business code	-	x	x	x	x	-	-
ErrorDescription	-	x	x	x	x	-	-

NOTE See also Auditing.

UC05 Erase Service Metadata

▼ *UC05 Use Case Description*

Brief description

- Remove all information about one specific service (i.e., all related processes and endpoints definitions).

Actors

- Resource Admin (or Group Admin).

Preconditions

- The user knows the address of SML.
- Resource Admin initiating the request is linked to the specified `ServiceGroup`.
- The authenticated user has the role of `Resource Admin`.
- The referenced ServiceMetadata exists.

Basic Flow

Step	Description
1	The receiver requests its service metadata to be removed from the SMP.

Step	Description
2	The SMP authenticates the user, validates the request, and delete any information from the referenced ServiceMetadata from its configuration database (from table ServiceMetadata and all its tables).
3	The receiver receives the confirmation that the definitions were removed properly with HTTP response `200 OK`.
4	Use case ends with success.

Exception Flow

1a	**SMP is not reachable**
1a1	The user receives a network connection error.
1a2	Use case ends.
2a	**Authentication / authorization fails**
2a1	The SMP replies with HTTP error "401 Unauthorized".
2a2	The receiver receives the error message.
2a3	Use case ends.
2b	**Request is not well formed (or any other business/technical error)**
2b1	The SMP replies with HTTP error "400 Bad request" or "500 Internal server error" with details on the error allowing to identify the error in the request (see the Error Codes Table).
2b2	The receiver receives the error message.
2b3	Use case ends.
2c	**ServiceGroup or ServiceMetadata is not defined**
2c1	The SMP replies with HTTP error "404 Resource not found".
2c2	The receiver receives the error message.
2c3	Use case ends.

Post Conditions

N/A

Successful conditions

- `ServiceMetadata` are absent.

Failure conditions

- In case of error, no change occurs into the configuration database and the response gives technical details on the exception condition

▼ *REST Service:* `DeleteServiceMetadata`

Execution

Authorization The operation will be allowed if and only the authenticated user matches the "Resource Admin" user linked to the Service Group.

For this user to be the eligible "Resource Admin" it must have been referenced as such in the ServiceGroup definition (see PutServiceGroup) by an `Group Admin` user via service "PutServiceGroup".

Start a new transaction.

NOTE	If no more ServiceMetadata information is available on the related ServiceGroup, the limited information on the ServiceGroup is nevertheless kept to allow keeping track of the previously defined administrator and the service group. Should it be deleted, it is the responsibility of the `Group Admin` user to issue the required operation (DeleteServiceGroup) if necessary.

Delete in one single transaction any information related to that service where participant and documents identifiers match the provided ServiceMetadata identifier.

In case of operation abort, the deletion to undo what was previously done:

- Rollback the transaction
- Response to this service is "failure".

If no error occurred:

- Commit the transaction
- Response to this service is "success".

Output

- HTTP `200` if done
- HTTP `404` if the service metadata or the service group does not exist
- HTTP `500` if any error occurred.

Sample Request

HTTP Header

`DELETE` http://130.206.118.4:8080/cipa-smp-full-webapp/iso6523-actorid-upis::0088:5798000000112/services/busdox-docid-qns::urn:oasis:names:specification:ubl:schema:xsd:Invoice-12::Invoice%23%23urn:www.cenbii.eu:transaction:biicoretrdm010:ver1.0:%23urn:www.peppol.eu:bis:peppol4a:ver1.0::2.0

HTTP/1.1

Accept-Encoding: `gzip,deflate`
Authorization: `Basic dGVzdGVyOnRlc3Q=`
Host: `130.206.118.4:8080`
Connection: `Keep-Alive`
User-Agent: `Apache-HttpClient/4.1.1 (java 1.5)`

Text

N/A

Sample Response

HTTP header

HTTP/1.1 `200 OK`

Server: `Apache-Coyote/1.1`
Pragma: `No-cache`
Expires: `Thu, 01 Jan 1970 01:00:00 CET`
Content-Length: `0`
Date: `Fri, 22 Jan 2016 09:47:52 GMT`
Cache-Control: `no-cache, proxy-revalidate`
Connection: `Keep-Alive`

Text

N/A

UC05 Error Codes Table

HTTP code	HTTP Message	Business code	Description
200	OK	N/A	*The request was completed successfully.*
400	Bad Request	OTHER_ERROR	*Some other specific error was encountered processing the request (more information in the `ErrorDescription` field).*
401	Unauthorized	UNAUTHORIZED	*The user is not granted the right to issue this request.*
404	Resource not found	NOT_FOUND	*The requested information was not found.*
500	Internal Server Error	TECHNICAL	*Some unexpected technical error occurred (detailed information is available in the response).*

Audit Information

Audit information that must be audited per service:

Logged Info	UC1	UC2	UC3	UC4	UC5	UC6	UC7
`AdministratorIdentifier`	-	x	x	x	x	-	-
`AccessTime`	-	x	x	x	x	x	x
`Operation`	-	x	x	x	x	x	x
`ParticipantIdentifier`	-	x	x	x	x	x	x
`ParticipantIdentifierScheme`	-	x	x	x	x	x	x
`DocumentIdentifier`	-	-	-	x	x	-	x

Logged Info	UC1	UC2	UC3	UC4	UC5	UC6	UC7
DocumentIdentifierScheme	-	-	-	x	x	-	x
IpAddress	-	x	x	x	x	x	x
RequestHeader	-	x	x	x	x	x	x
RequestText	-	x	-	x	x	?	?
ResponseHeader	-	x	x	x	x	x	x
ResponseText	-	-	-	-	-	x	x
HTTP code	-	x	x	x	x	x	x
Business code	-	x	x	x	x	-	-
ErrorDescription	-	x	x	x	x	-	-

NOTE See also Auditing.

5..1. Information Retrieval Use Cases

These use cases, mainly aimed at *sender* participants and have the purpose of collecting information on the target receivers.

They are based on the OASIS XSD standard. See OASIS SMP XSD

UC06 Retrieve Service Group

▼ *UC06 Use Case Description*

Brief description

Obtain the list of services provided by a specific receiver participant (collection of references to the ServiceMetaData's).
This service provides the information related to the Service Group according to the input duplet `participantIdentifier`+`participantIndentifierScheme`.
Returns information from the ServiceMetadata table only (references to actual MetaData)

SEE ALSO **2.1 Discovery Flow** in PEPPOL Transport Infrastructure Service Metadata Publishing (SMP).

A sender (ed. *user*) may want to discover what document types can be handled by a specific participant identifier.
Such discovery is relevant for applications supporting several equivalent business processes.
This is enabled by a pattern where the sender first retrieves the `ServiceGroup` entity, which holds a list of references to the `ServiceMetadata` resources associated with it.
The `ServiceMetadata` in turn holds the metadata information that describes the capabilities associated with the recipient participant identifier.

Actors

- User

Preconditions

- The requester application has previously resolved the address of the SMP from the DNS.
- Referenced service group was previously defined by the receiver.

Basic Flow

Step	Description
1	The user request one service group references to SML.
2	The SMP validates the request and retrieves the information from its configuration database (into table ServiceGroup and Service Metadata tables).
3	The user receives the participant's service group information.
4	Use case ends with success.

Exception Flow

1a	**SMP is not reachable**
1a1	The user receives a network connection error.
1a2	Use case ends.
2a	**Request is not well formed (or any other business/technical error)**
2a1	The SMP replies with HTTP error "400 Bad request" or "500 Internal server error" with details on the error allowing to identify the error in the request (see the Error Codes Table).
2a2	The receiver receives the error message.
2a3	Use case ends.
2b	**ServiceGroup is not defined**
2b1	The SMP replies with HTTP error "404 Resource not found".
2b2	The receiver receives the error message.
2b3	Use case ends.

Post Conditions

N/A

Successful Conditions

- The user receives ServiceGroup information for the requested receiver participant.

Failure Conditions

The user received no ServiceGroup information about the requested receiver participant.

▼ *REST Service:* `GetServiceGroup`

Input `ParticipantIdentifier`

Represents the business level endpoint key and key type, for example, a DUNS or GLN number that is associated with a group of services.

SEE ALSO	The `ParticipantIdentifier` section of the Common Definitions document [BDEN-CDEF] for more information on this data type.

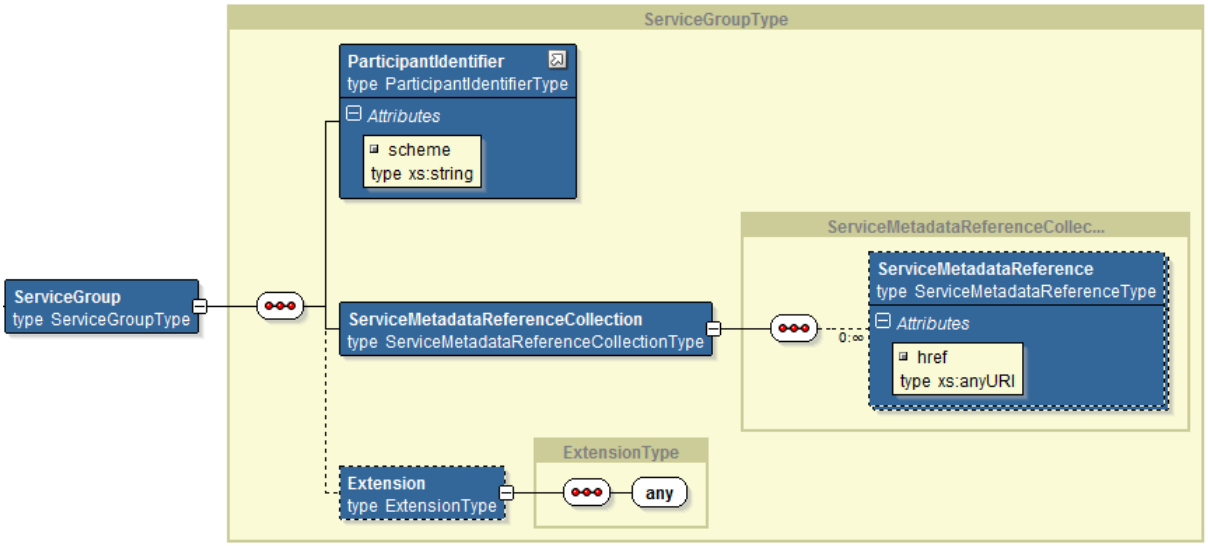
Execution

- Selects all service Metadata related to the `ServiceGroup` specified by the provided `ParticipantIdentifier` and build the corresponding URI from it.

NOTE	There is no interaction with SML (from SML).

Output `ServiceGroup`

This SMP service will return the reference URI for the user that will enable him to retrieve all information about the services that a participant (receiver) participates in, i.e., all service metadata of the specified participant. To obtain the details on those services, the ServiceMetadata can be obtained from SML using the references provided.



Sample Request

HTTP Header

`GET` http://130.206.118.4:8080/cipa-smp-full-webapp/iso6523-actorid-upis::0088:5798000000112

HTTP/1.1

Accept-Encoding: `gzip,deflate`
Host: `130.206.118.4:8080`
Connection: `Keep-Alive`
User-Agent: `Apache-HttpClient/4.1.1 (java 1.5)`

Text

N/A

Sample Response

HTTP header

HTTP/1.1 `200 OK`

Server: Apache-Coyote/1.1
Content-Type: text/xml
Content-Length: 959
Date: Thu, 21 Jan 2016 08:38:33 GMT
Cache-Control: proxy-revalidate
Connection: Keep-Alive

Text

```
<ServiceGroup xmlns="http://docs.oasis-open.org/bdxr/ns/SMP/2016/05">
    <ParticipantIdentifier scheme="busdox-actorid-upis">
        0010:5798000000001
    </ParticipantIdentifier>
    <ServiceMetadataReferenceCollection>
        <ServiceMetadataReference
        href="http://serviceMetadata.eu/busdox-actorid-
upis%3A%3A0010%3A5798000000001/services/bdx-docid-
qns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AInvoice-
        2%3A%3AInvoice%23%23UBL-2.0" />
    </ServiceMetadataReferenceCollection>
    <Extension>
        <ex:Test xmlns:ex="http://test.eu">Test</ex:Test>
    </Extension>
</ServiceGroup>
```

UC06 Error codes

HTTP code	HTTP Message	Business code	Meaning
200	OK	N/A	*The request was completed successfully*
400	Bad Request	OTHER_ERROR	*Some other specific error was encountered processing the request (more information in the ErrorDescription field)*
404	Resource not found	NOT_FOUND	*The requested information was not found*
500	Internal Server Error	TECHNICAL	*Some unexpected technical error occurred (detailed information is available in the response)*

Audit Information

Audit information that must be audited per service:

Logged Info	UC1	UC2	UC3	UC4	UC5	UC6	UC7
AdministratorIdentifier	-	x	x	x	x	-	-
AccessTime	-	x	x	x	x	x	x
Operation	-	x	x	x	x	x	x
ParticipantIdentifier	-	x	x	x	x	x	x

Logged Info	UC1	UC2	UC3	UC4	UC5	UC6	UC7
ParticipantIdentifierScheme	-	x	x	x	x	x	x
DocumentIdentifier	-	-	-	x	x	-	x
DocumentIdentifierScheme	-	-	-	x	x	-	x
IpAddress	-	x	x	x	x	x	x
RequestHeader	-	x	x	x	x	x	x
RequestText	-	x	-	x	x	?	?
ResponseHeader	-	x	x	x	x	x	x
ResponseText	-	-	-	-	-	x	x
HTTP code	-	x	x	x	x	x	x
Business code	-	x	x	x	x	-	-
ErrorDescription	-	x	x	x	x	-	-

NOTE See also Auditing.

UC07 Retrieve Service Metadata

▼ *UC07 Use Case Description*

Brief description

Obtain detailed definition about one specific service of a specific participant for all supported transport.
This service retrieves the SignedServiceMetadata according to the input quadruplet `participantIdentifier` `participantIndentifierScheme` `documentIdentifier` + `documentIdentifierScheme`.
Returns information from the Endpoint table.

Actors

- User

Preconditions

- The user application has previously resolved the address of SML from the DNS.
- Referenced service group and required Service Meta data were previously defined by the receiver.

Basic Flow

Step	Description
1	The user requests the detailed information of a receiver's service to SML.
2	The SMP validates the request, retrieves the information from its configuration database and sends it as response to the user.
3	The user receives the participant's service detailed information.

Step	Description
4	Use case ends with success.

Alternative Flow

Step	Description
3a	**Redirect**
3a1	The configuration refers to another SMP. The SMP returns the redirection information to the user.
3a2	The user reinitiates the same request to that other SMP: restart use case at step 1.
3a3	Use case ends.

Exception Flow

Step	Description
1a	**SMP is not reachable**
1a1	The user receives a network connection error.
1a2	Use case ends.
2a	**Request is not well formed (or any other business/technical error)**
2a1	The SMP replies with HTTP error "400 Bad request" or "500 Internal server error" with details on the error allowing to identify the error in the request (see the Error Codes Table).
2a2	The receiver receives the error message.
2a3	Use case ends.
2b	**ServiceGroup or ServiceMetadata is not defined**
2b1	The SMP replies with HTTP error "404 Resource not found".
2b2	The receiver receives the error message.
2b3	Use case ends.
2a2a	**Multiple redirect**
2a2a1	The client receives redirect information for the 2nd time (and must ignore it).
2a2a2	Use case ends.

Post Conditions

N/A

Successful conditions

- The user receives ServiceMetaData information for the requested receiver participant.

Failure conditions

- The user received no Metadata information about the requested receiver participant.

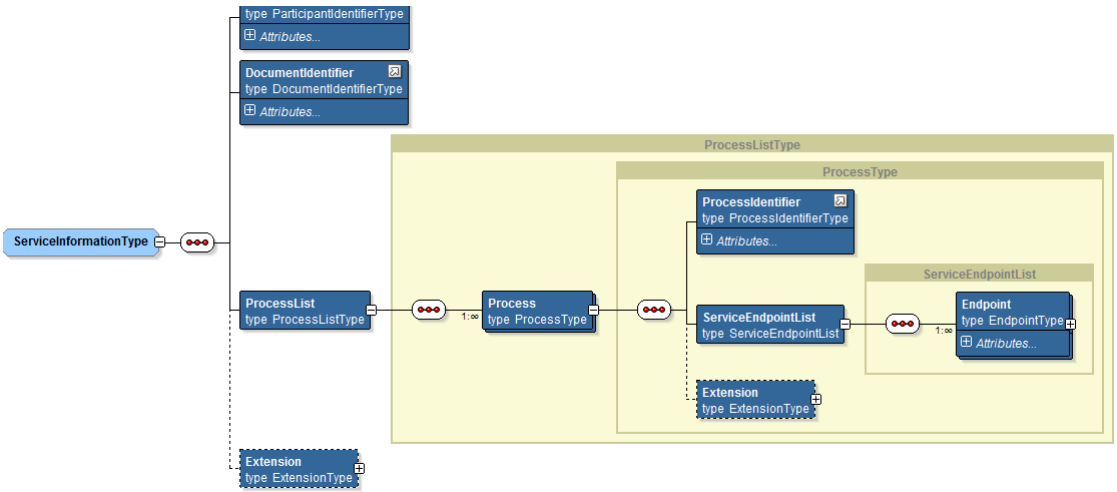
▼ *REST Service:* `GetSignedServiceMetadata`

Input `ServiceMetadataReference` *i.e., the PK made of*

4 fields that uniquely identify the ServiceMetadata entry in SML configuration.

Execution

This service will return necessary information for the user to send documents to the receiver, this information is held in the `ServiceInformation` structure, i.e., the information stored in tables Process and Endpoint (related to the requested service metadata and highlighted into red squares below):



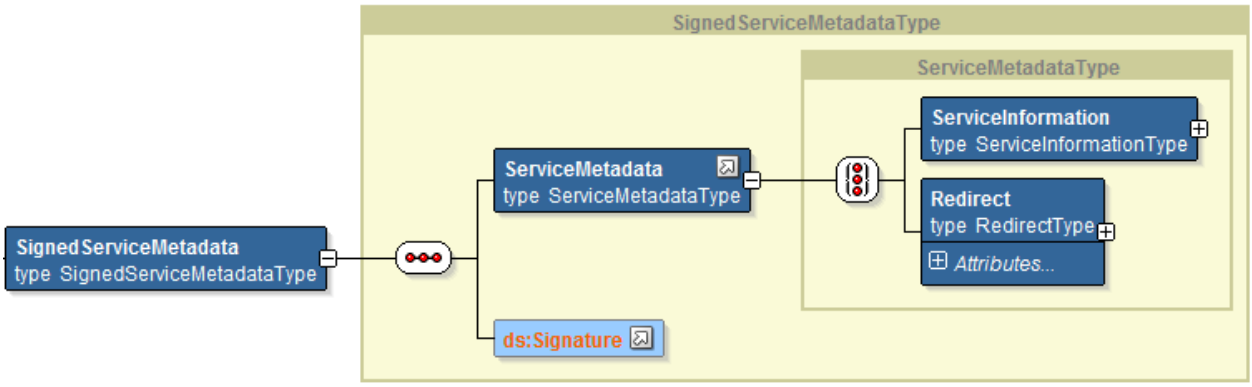
NOTE	There is no interaction with SML.

Output `SignedServiceMetadata`

See 4.3 ServiceMetadata section of the PEPPOL Transport Infrastructure Service Metadata Publishing document; this data structure represents metadata about a specific electronic service.

The role of the ServiceMetadata structure is to associate a participant identifier with the ability to receive a specific document type over a specific transport. It also describes which business processes a document can participate in, and various operational data such as service activation and expiration times. The ServiceMetadata resource contains all the metadata about a service that a user Access Point needs to know in order to send a message to that service.

The `SignedServiceMetadata` structure holds both a `ServiceMetadata` structure and the corresponding signature by SML to allow the user (or any other user) verifying the authenticity of the information provided by SML by using the public key of SML before sending any document to the receiver.



Output (alternative) Redirection

Supports the alternative flow 'a' in the use case.

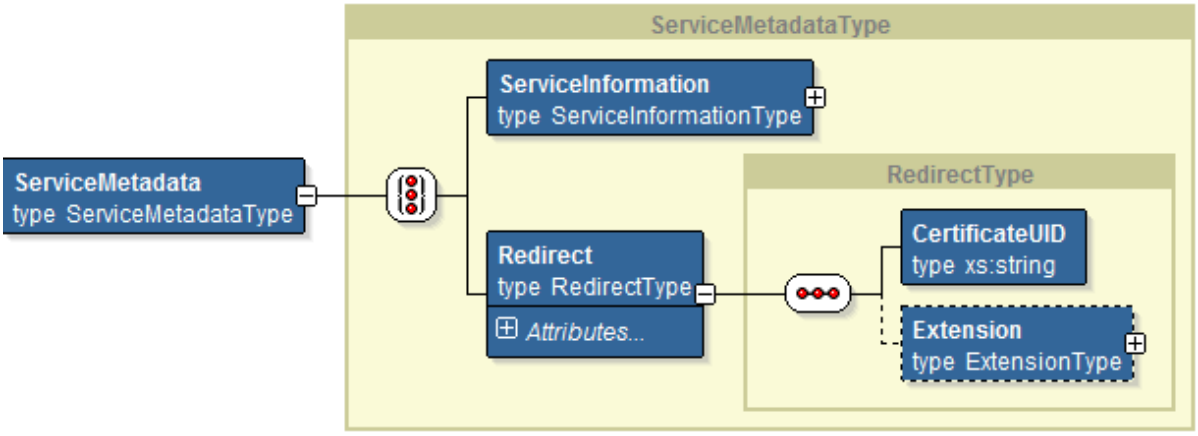
Eventually, this service will return *redirect* information instead of the `ServiceInformation` information itself, when it is held by another SMP.

SEE ALSO

Redirection is exhaustively explained in the 4.3 ServiceMetadata section of the PEPPOL Transport Infrastructure Service Metadata Publishing document. and in the 2.1.3 Service Metadata Publisher Redirection section of the Service Metadata Publishing (SMP) Version 1.0 document.

In such a case, the information returned is the reference to SML that holds the corresponding `ServiceMetadata`; i.e., in the Redirect structure containing the target URI.

The queried SMP has in fact no information about the participant services (there is no related Process entry for that participant), instead, he has the target URI of the other SMP in the `Redirect` column of the `ServiceMetadata` row for that receiver.



Sample Request

HTTP Header

`GET` http://130.206.118.4:8080/cipa-smp-full-webapp/iso6523-actorid-upis::0088:5798000000112/services/busdox-docid-qns::urn:oasis:names:specification:ubl:schema:xsd:Invoice-12::Invoice%23%23urn:www.cenbii.eu:transaction:biicoretrdm010:ver1.0:%23urn:www.peppol.eu:bis:peppol4a:ver1.0::2.0* HTTP/1.1

Accept-Encoding: `gzip,deflate`+ Host: `130.206.118.4:8080`

Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)

Text

N/A

Sample Response

HTTP header

HTTP/1.1 200 OK

Server: Apache-Coyote/1.1
Content-Type: text/xml
Transfer-Encoding: chunked
Date: Thu, 21 Jan 2016 10:22:38 GMT
Cache-Control: proxy-revalidate
Connection: Keep-Alive

Text

```
<SignedServiceMetadata
    xmlns="[.mark]#http://docs.oasis-open.org/bdxr/ns/SMP/2016/05# ">
    <ServiceMetadata
        xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd">
        <ServiceInformation>
            <ParticipantIdentifier scheme="busdox-actorid-upis">
                0010:5798000000001
            </ParticipantIdentifier>
        <DocumentIdentifier scheme="bdx-docid-qns">
                urn:oasis:names:specification:ubl:schema:xsd:Invoice-
2::Invoice##UBL-2.02
        </DocumentIdentifier>
            <ProcessList>
                <Process>
                    <ProcessIdentifier scheme="cenbii-procid-ubl">
                        BII04
                    </ProcessIdentifier>
                    <ServiceEndpointList>
                        <Endpoint transportProfile="busdox-transport-start">
                            <EndpointURI>
http://busdox.org/sampleService/</EndpointURI>
                            <RequireBusinessLevelSignature>
                                false
                            </RequireBusinessLevelSignature>
                            <MinimumAuthenticationLevel>
2</MinimumAuthenticationLevel>
                            <ServiceActivationDate>
                                2009-05-01T09:00:00
                            </ServiceActivationDate>
                            <ServiceExpirationDate>
```

```
                            2016-05-01T09:00:00
                        </ServiceExpirationDate>
                        <Certificate>
                            AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
                        </Certificate>
                        <ServiceDescription>invoice service</ServiceDescription>
                        <TechnicalContactUrl>
                            https://example.com
                        </TechnicalContactUrl>
                        <TechnicalInformationUrl>
                            http://example.com/info
                        </TechnicalInformationUrl>
                    </Endpoint>
                </ServiceEndpointList>
            </Process>
            <Process>
                <ProcessIdentifier scheme="cenbii-procid-ubl">
                    BII07
                </ProcessIdentifier>
                <ServiceEndpointList>
                    <Endpoint transportProfile="busdox-transport-start">
                        <EndpointURI>
http://busdox.org/sampleService/</EndpointURI>
                        <RequireBusinessLevelSignature>
                            true
                        </RequireBusinessLevelSignature>
                        <MinimumAuthenticationLevel>
1</MinimumAuthenticationLevel>
                        <ServiceActivationDate>
                            2009-05-01T09:00:00
                        </ServiceActivationDate>
                        <ServiceExpirationDate>
                            2016-05-01T09:00:00
                        </ServiceExpirationDate>
                        <Certificate>
                            AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
                        </Certificate>
                        <ServiceDescription>invoice service</ServiceDescription>
                        <TechnicalContactUrl>
                            https://example.com
                        </TechnicalContactUrl>
                        <TechnicalInformationUrl>
                            http://example.com/info
                        </TechnicalInformationUrl>
                        <Extension>
                            <ex:Test
                                xmlns:ex="http://test.eu">Test
                            </ex:Test>
                        </Extension>
                    </Endpoint>
                </ServiceEndpointList>
```

```
                    <Extension>
                        <ex:Test
                            xmlns:ex="http://test.eu">Test
                        </ex:Test>
                    </Extension>
                </Process>
            </ProcessList>
            <Extension>
                <ex:Test
                    xmlns:ex="http://test.eu">Test
                </ex:Test>
            </Extension>
        </ServiceInformation>
    </ServiceMetadata>
    <Signature
        xmlns="http://www.w3.org/2000/09/xmldsig#">
        <SignedInfo>
            <CanonicalizationMethod
Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
            <SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
            <Reference URI="">
                <Transforms>
                    <Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
                </Transforms>
                <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                <DigestValue>6r3W426Gx5foBPtasSdIEj6JvAY=</DigestValue>
            </Reference>
        </SignedInfo>

<SignatureValue>2NJB0Pv3ORL+EpPYLCl/InXI+mDbUsV8CrWzRVJvEJMnnyuI2bPMe6k4MJwp9A4bTkzj
vkMPARYAhyVNm6MNNlJRAFL4qddsRrWa4Jgf/QF0zQgpJ7ZUPdVQ8L8A54FiPZWltOIgZCfO7sDbEcB00V4g
KmzVPBsVu6BIBOws/UY=</SignatureValue>
        <KeyInfo>
            <X509Data>

<X509SubjectName>1.2.840.113549.1.9.1=#160e73656e64657240746573742e6265,CN=senderCN,
OU=B4,O=DIGIT,L=Brussels,ST=BE,C=BE</X509SubjectName>

<X509Certificate>MIICpTCCAg6gAwIBAgIBATANBgkqhkiG9w0BAQUFADB4MQswCQYDVQQGEwJCRTELMAk
GA1UECAwCQkUxETAPBgNVBAcMCEJydXNzZWxzMQ4wDAYDVQQKDAVESUdJVDELMAkGA1UECwwCQjQxDzANBgN
VBAMMBnJvb3RDTjEbMBkGCSqGSIb3DQEJARYMcm9vdEB0ZXN0LmJlMB4XDTE1MDMxNzE2MTkwN1oXDTI1MDM
xNDE2MTkwN1owfDELMAkGA1UEBhMCQkUxCzAJBgNVBAgMAkJFMREwDwYDVQQHDAhCcnVzc2VsczEOMAwGA1U
ECgwFRElHSVQxCzAJBgNVBAsMAkI0MREwDwYDVQQDDAhzZW5kZXJDTjEdMBsGCSqGSIb3DQEJARYOc2VuZGV
yQHRlc3QuYmUwgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBANxLUPjIn7R0CsHf86kIwNzCu+6AdmWM8fB
LUHL+VXT6ayr1kwgGbFMb/vUUX6a46jRCiZBM+9IK1Hpjg9QX/QIQiWtvD+yDr6jUxahZ/w13kqFG/K81IVu
9DwLBoiNwDvQ6l6UbvMvV+1nWy3gjRcKlFs/C+E2uybgJxSM/sMkbAgMBAAGjOzA5MB8GA1UdIwQYMBaAFHC
VSh4WnWR8MGBGedr+bJH96tc4MAkGA1UdEwQCMAAwCwYDVR0PBAQDAgTwMA0GCSqGSIb3DQEBBQUAA4GBAK6
idNRxyeBmqPoSKxq7Ck3ej6R2QPyWbwZ+6/S7iCRt8PfgOu++Yu5YEjlUX1hlkbQKF/JuKTLqxNnKIE6Ef65
+JP2ZaI9O2wdzpRclAhAd00XbNKpyipr4jMdWmu2U8vyBBwn/utG1ZrLhAUiqnPvmaQrResiGHM2xzCmVwts
```

```
e</X509Certificate>
            </X509Data>
        </KeyInfo>
    </Signature>
</SignedServiceMetadata>
```

Sample Response (redirect alternative)

HTTP header

HTTP/1.1 `200 OK`

Server: `Apache-Coyote/1.1`
Content-Type: `text/xml`
Transfer-Encoding: `chunked`
Date: `Thu, 21 Jan 2016 10:22:38 GMT`
Cache-Control: `proxy-revalidate`
Connection: `Keep-Alive`

Text

```
<?xml version="1.0" encoding="utf-8" ?>
<SignedServiceMetadata
    xmlns="[.mark]#http://docs.oasis-open.org/bdxr/ns/SMP/2016/05# ">
    <ServiceMetadata>
        <Redirect

href="http://serviceMetadata2.eu/busdox-actorid-
upis%3A%3A0010%3A5798000000001/services/bdx-docid-
qns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AInvoice-
2%3A%3AInvoice%23%23UBL-2.0">
            <CertificateUID>PID:9208-2001-3-279815395</CertificateUID>
            <Extension>
                <ex:Test
                    xmlns:ex="http://test.eu">Test
                </ex:Test>
            </Extension>
        </Redirect>
    </ServiceMetadata>
    <Signature
        xmlns="http://www.w3.org/2000/09/xmldsig#">
        <SignedInfo>
            <CanonicalizationMethod
Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
            <SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
            <Reference URI="">
                <Transforms>
                    <Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
                </Transforms>
```

```
                <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                <DigestValue>6r3W426Gx5foBPtasSdIEj6JvAY=</DigestValue>
            </Reference>
        </SignedInfo>

<SignatureValue>2NJB0Pv3ORL+EpPYLCl/InXI+mDbUsV8CrWzRVJvEJMnnyuI2bPMe6k4MJwp9A4bTkzj
vkMPARYAhyVNm6MNNlJRAFL4qddsRrWa4Jgf/QF0zQgpJ7ZUPdVQ8L8A54FiPZWltOIgZCfO7sDbEcB00V4g
KmzVPBsVu6BIBOws/UY=</SignatureValue>
        <KeyInfo>
            <X509Data>

<X509SubjectName>1.2.840.113549.1.9.1=#160e73656e64657240746573742e6265,CN=senderCN,
OU=B4,O=DIGIT,L=Brussels,ST=BE,C=BE</X509SubjectName>

<X509Certificate>MIICpTCCAg6gAwIBAgIBATANBgkqhkiG9w0BAQUFADB4MQswCQYDVQQGEwJCRTELMAk
GA1UECAwCQkUxETAPBgNVBAcMCEJydXNzZWxzMQ4wDAYDVQQKDAVESUdJVDELMAkGA1UECwwCQjQxDzANBgN
VBAMMBnJvb3RDTjEbMBkGCSqGSIb3DQEJARYMcm9vdEB0ZXN0LmJlMB4XDTE1MDMxNzE2MTkwN1oXDTI1MDM
xNDE2MTkwN1owfDELMAkGA1UEBhMCQkUxCzAJBgNVBAgMAkJFMREwDwYDVQQHDAhCcnVzc2VsczEOMAwGA1U
ECgwFRElHSVQxCzAJBgNVBAsMAkI0MREwDwYDVQQDDAhzZW5kZXJDTjEdMBsGCSqGSIb3DQEJARYOc2VuZGV
yQHRlc3QuYmUwgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBANxLUPjIn7R0CsHf86kIwNzCu+6AdmWM8fB
LUHL+VXT6ayr1kwgGbFMb/vUUX6a46jRCiZBM+9IK1Hpjg9QX/QIQiWtvD+yDr6jUxahZ/w13kqFG/K81IVu
9DwLBoiNwDvQ6l6UbvMvV+1nWy3gjRcKlFs/C+E2uybgJxSM/sMkbAgMBAAGjOzA5MB8GA1UdIwQYMBaAFHC
VSh4WnWR8MGBGedr+bJH96tc4MAkGA1UdEwQCMAAwCwYDVR0PBAQDAgTwMA0GCSqGSIb3DQEBBQUAA4GBAK6
idNRxyeBmqPoSKxq7Ck3ej6R2QPyWbwZ+6/S7iCRt8PfgOu++Yu5YEjlUX1hlkbQKF/JuKTLqxNnKIE6Ef65
+JP2ZaI9O2wdzpRclAhAd00XbNKpyipr4jMdWmu2U8vyBBwn/utG1ZrLhAUiqnPvmaQrResiGHM2xzCmVwts
e</X509Certificate>
            </X509Data>
        </KeyInfo>
    </Signature>
</SignedServiceMetadata>
```

UC07 Error Codes

HTTP code	HTTP Message	Business code	Description
200	OK	N/A	*The request was completed successfully*
400	Bad Request	OTHER_ERROR	*Some other specific error was encountered processing the request (more information in the ErrorDescription field).*
404	Resource not found	NOT_FOUND	*The requested information was not found.*
500	Internal Server Error	TECHNICAL	*Some unexpected technical error occurred (detailed information is available in the response).*

Audit Information

Audit information that must be audited per service:

Logged Info	UC1	UC2	UC3	UC4	UC5	UC6	UC7
AdministratorIdentifier	-	x	x	x	x	-	-
AccessTime	-	x	x	x	x	x	x
Operation	-	x	x	x	x	x	x
ParticipantIdentifier	-	x	x	x	x	x	x
ParticipantIdentifierScheme	-	x	x	x	x	x	x
DocumentIdentifier	-	-	-	x	x	-	x
DocumentIdentifierScheme	-	-	-	x	x	-	x
IpAddress	-	x	x	x	x	x	x
RequestHeader	-	x	x	x	x	x	x
RequestText	-	x	-	x	x	?	?
ResponseHeader	-	x	x	x	x	x	x
ResponseText	-	-	-	-	-	x	x
HTTP code	-	x	x	x	x	x	x
Business code	-	x	x	x	x	-	-
ErrorDescription	-	x	x	x	x	-	-

NOTE See also Auditing.

5.1. Security

5.1.1. User Management

Administration Process

As described in Actors, there are three types of users accessing SML. Among them, only Resource Admin and Group Admin types of users are registered into the configuration of SML.

This paragraph summarizes the process for defining the users who are responsible for managing the overall configuration of SMPs.

1. **Creation of a Group Admin**

   ▼ *Details*

   The "Domain Admin" sets existing users as an `Group Admin`.

   In the picture below, "System Admin b" creates one user "Group Admin " that will manage the service groups on this SMP's.

2. **Creation of a remote ServiceGroup administrator (for one Participant)**

   ▼ *Details*

This step is necessary for remote administration of ServiceGroups (if administration is local it is done by the `Group Admin` himself).

The "System Admin”:

- deploys the certificates that will be used to access SML for a new participant’s administration (if certificates are used);
- creates manually the " Resource Admin" entry in the "Administrator" table

3. **Creation of the ServiceGroup (for one Participant)**

▼ *Details*

The `Group Admin` accesses SML via http with basic authentication with the previously assigned username and password by the "System Admin".

He uses UC02 - Create or Update Service Group to define new service groups.

When doing so, the `Group Admin` provides either:

- A `ServiceGroup-Owner` in the HTTP header, i.e. some pieces of the Participant’s certificates that will be used to identify the " Resource Admin" user accessing SML for configuration purposes (mostly for distributed SMP model)
- Nothing: in which case, the basic authentication information of the `Group Admin` (in the HTTP header) will be stored as identifier and will be himself the administrator of this `ServiceGroup` (see step 1 of Create/Update Service Group).

Later,that Service Group can be removed via the same access method, using Erase Service Group.

In the picture below, "Group Admin b" creates one user "Resource Admin D, E, F" that will manage parties D,E and F.

4. **Creation of ServiceMetadata**

▼ *Details*

- Resource Admin accesses SML using its certificate.
- The Admin can defines new services using UC04 - Create or Update Service Metadata.
- If needed, deprecated services can be removed using UC05 - Erase Service Metadata.

In the picture below, Resource Admin D, E, F defines some of the services for one or several parties among D, E and F.

5. **Discovering a participant’s services capabilities**

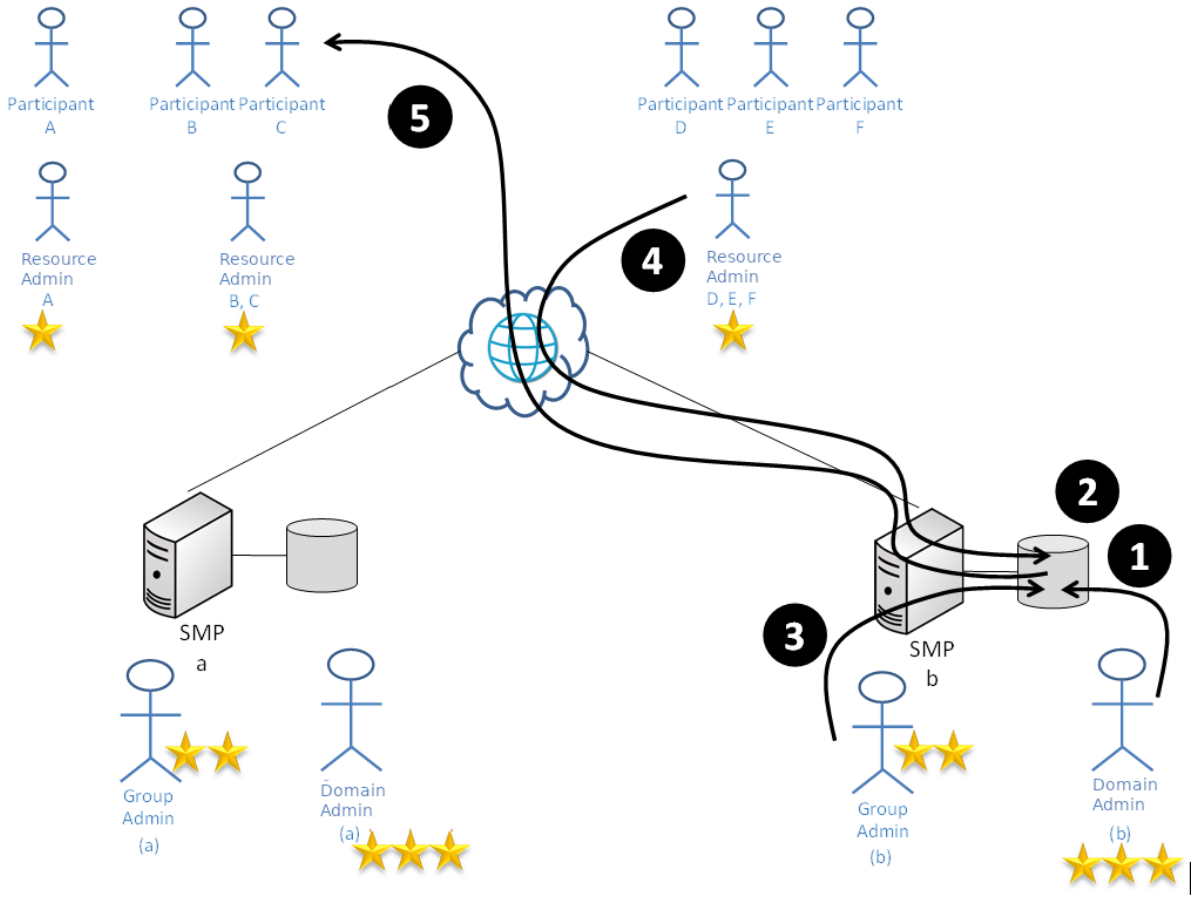
▼ *Details*

- Participant access SML with no authentication.
- To assist participants exchanging messages with each other, they can collect eDelivery information using:

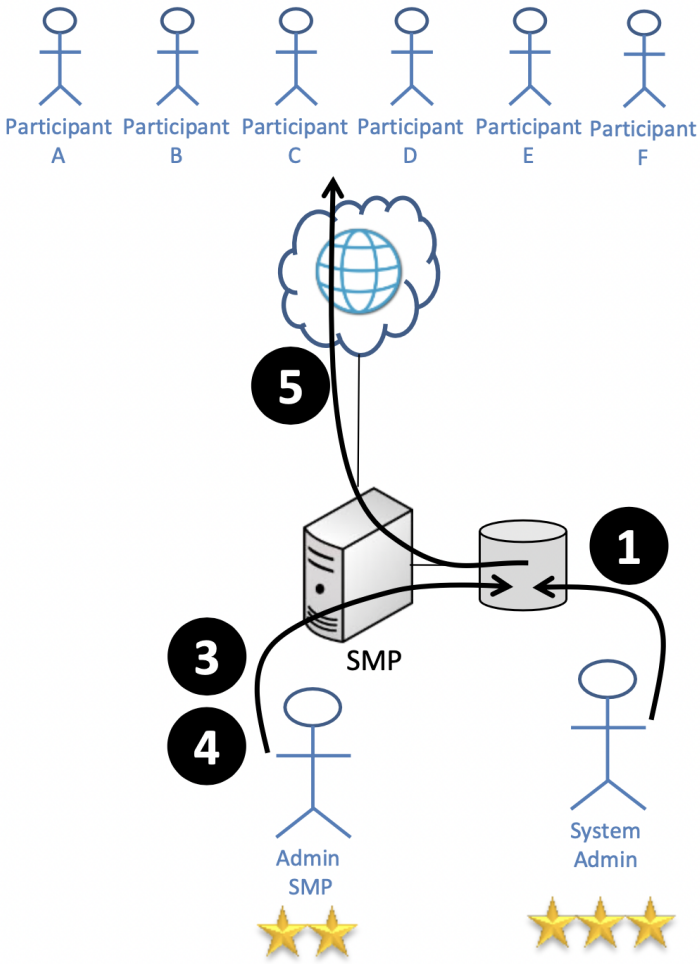
- Retrieve Service Group,
- Retrieve Service Metadata.

In the picture below, "Participant C" collects metadata from one (and only one) participant among D, E and F.

The following diagram illustrates distributed (remote) Resource Admin's:



The following diagram illustrates centralised ServiceGroup management (by the `Group Admin`):



The specifications allow the coexistence of both models: some domain may decide to manage some ServiceGroups centrally (by the `Group Admin`), others in a distributed manner (by multiple remote "Resource Admin's").

▼ *Simple User*

The regular users (Actor "User") are any user accessing the system public services. As these users do not need to be authenticated, they do not have to be known in advance by the System and are therefore not preregistered in any way on SML.

▼ *System Admin*

The `System Admin` actor is, as the name suggests, a system user having special accesses to the system. In the purpose of user administration for SML, this system user is able to modify the content of SML configuration database, i.e. he must have full read/write data access on this configuration database, in particular the *Administrator* table described in Security tables.

This actor is responsible for creating and maintaining the definition of all `Group Admin` and `Resource Admin` administrators (as described by use case UC01).

▼ *Administrator*

This table identifies the administrators of SML; this means `Group Admin` and Resource `Admin` actors introduced above.

There are two possible means to obtain access to SML non-public services:

- through **basic authentication**; i.e. with a simple **username/password** authentication method:
  - **Identifier** column contains then the username used to identify the administrator at logon
  - **Password** column contains then the hash of the password used to authenticate the user at logon
- through **two-way SSL** using PKI infrastructure (i.e., X.509 certificates):
  - the **Identifier** column contains pieces of the client certificate that are forwarded by the reverse proxy in the http header to the server (see HTTP Authentication).
  - **Password** column is unused for 2-way-ssl since the certificate is not validated by the application layer itself; the prerequisite being that the user's certificate is already present in the truststore of the reverse proxy server.

In all cases, it is the responsibility of SML to hash the password (and apply the same algorithm for authentication). The participant will send the password in 'clear' in the HTTP header.

Service Group Ownership

1-N relationship materialization between the service groups and the `Resource Admin` type of users of SML. More details are available under Resource Resource Admin.

This relationship allows the system to identify which user (singular) is allowed to modify(/delete) all the information related to all the ServiceMetadata of one given `ServiceGroup`.

▼ *Group Admin*

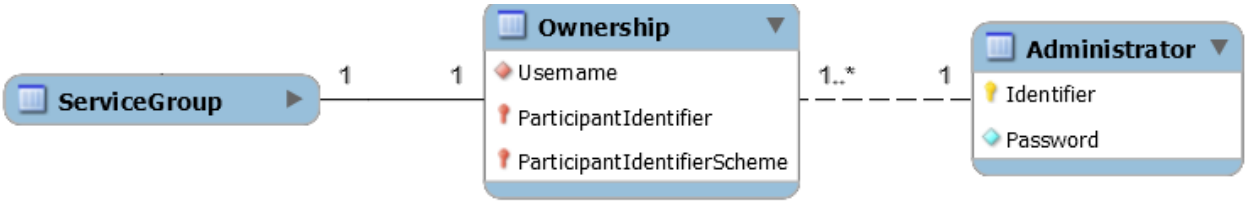
The Group Admin user is created by the system administrator (see Actors and UC01 Manage Administrators).

Some information in the system (not detailed here) allows the system to identify this specificity of such users.

▼ *Resource Admin*

The `Resource Admin` user of one specific participant will be allowed to use all the services that modify the definition of the ServiceGroups, i.e. to create, modify or delete SignedServiceMetadata belonging to/referenced by a ServiceGroup.

To allow the access right verification, the configuration holds a link between the `Resource Admin` and the related ServiceGroup via an "ownership relationship" materialized as shown here in the configuration:



The ServiceGroup can be managed by:

- The related "Resource Admin" (if any); and,
- The Group Admin (who may administer all service groups).

This **link** is established when the `ServiceGroup` is created (or updated).

5.1.2. Access rights

The following matrix clarifies the access rights of each actor to all use cases and the type of authentication method that are supported for each user role:

The following matrix clarifies the access rights

Access Rights Matrix

Use Case	System Admin	Group Admin	Resource Admin	User
UC01 Manage Administrators	x	-	-	-
UC02 Create or Update Service Group	-	x	-	-
UC03 Erase Service Group	-	x	-	-
UC04 Create or Update Service Metadata	-	x	x	-
UC05 Erase Service Metadata	-	x	x	-
UC06 Retrieve Service Group	x	x	x	x
UC07 Retrieve Service Metadata	x	x	x	x

Authentication Method (Acceptance and Production at EC)

Method	System Admin	Group Admin	Resource Admin	User
System & Database Authentication	x	-	-	-
HTTP Basic authentication	-	x	N/A	-
HTTP 2-way-ssl	-	-	x	-
None	-	-	-	x

Authentication method (Test at EC)

Method	System Admin	Group Admin	Resource Admin	User
System & database authentication	**x**	-	-	-
HTTP Basic authentication	-	**x**	**x**	-
HTTP 2-way-ssl	-	-	**N/A**	-
None	-	-	-	**x**

SEE ALSO `Group Admin` user may act on behalf of all the "Resource Admin" defined in SML.

5.1.3. HTTP Authentication

SSL will be used at all times (i.e., for any exchange of message between a SMP and any participant, acting as a sender or as a receiver.) to guarantee the validity of the information provided by SML to the sender and receiver.

Two authentication methods are supported and vary with services and/or user's roles:

1. Basic HTTP authentication (username/password) – for `Group Admin` users and optionally for `Resource Admin` users (see the Authentication method (Test at EC) table).
2. HTTP 2-way SSL for remote `Resource Admin` users (only) when and if this method is preferred for those to basic authentication (see the Authentication Method tables in Access Rights. This authentication method might be used at EC in production environment).

If HTTP basic authentication is available for both types of users, 2-way SSL will also be usable for authenticating `Resource Admin` users. In order to achieve this, all the PUT and DELETE services on ServiceMetadata data type (see UC04 and UC05) will be able to use that type of authentication.

In order to provide this possibility, the certificates of the authorized administrators (`Resource Admin` users) will be deployed on the necessary SMPs on dedicated keystores. This will allow the transport layers to establish necessary trust without any addition to the existing message structure.

Also, the fields in *Administrator* table will be used as follows differently in the different possible cases (by user roles and authentication methods):

	Authentication Method	Identifier	Password
Group Admin	Basic Authentication	Basic User Name	Password Hash
Resource Admin	2 way-ssl	HTTP Client Cert	*N/A*
	Basic Authentication	Basic User Name	Password Hash

NOTE Only basic authentication is allowed for `Group Admin` users since they are intended to be intranet users rather than internet ones.

The password field, when applicable, holds a hash value of the password.

5.1.4. Reverse proxy

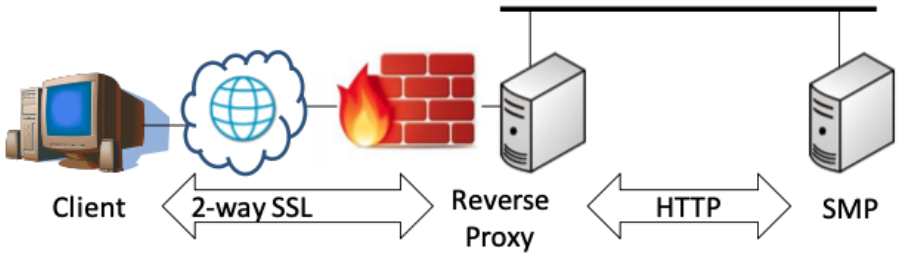
Here's a description of the European Commission's SML deployment in Production. In this environment,

- a BDMSL server is already hosted behind a *Reverse Proxy*.

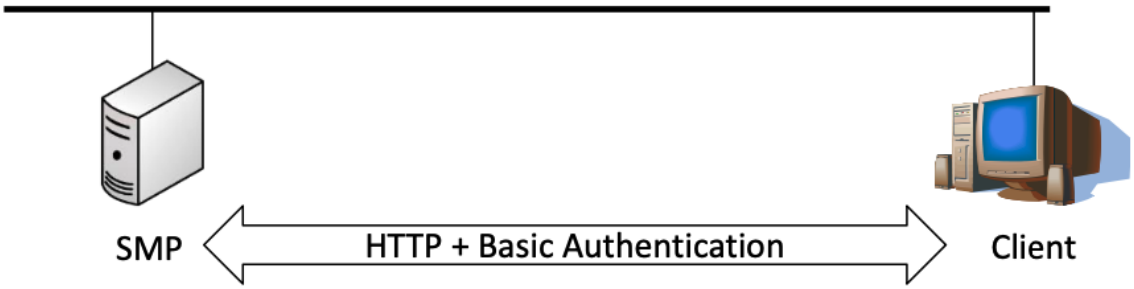
SEE ALSO	See **Reverse proxy with SSL** (p. 70), in DomiSML Software Architecture Document.

- 2-way SSL is set up on the reverse proxy and the server hosting the application can use the HTTP protocol.

A similar configuration could be used at the European Commission for SMP's where 2-way SSL must be used.



As stated above, this type of access will be provided for remote `Resource Admin` type of users only and is optional. Basic authentication will be used instead when there is no remote " Resource Admin"; i.e., when the `Group Admin` administers the ServiceGroup himself.



Therefore, the authentication mechanism for services modifying Service Metadata will behave as follow:

- Search HTTP header for "Client Certificate" data (conversion performed by the reverse proxy). If present, use these to authenticate user against the "username" present in table "Administrator".
  The "Client Certificate" values will be inserted in the HTTP header to SML by the Reverse Proxy out of the X.509 Certificate.

The X.509 attributes to be used will be defined in the detailed design.

The value stored in the "Administrator" table column "username" should contain necessary information to validate that the provided value match.

- If no “Client certificate” information is available (meaning there is no reverse proxy between the client and SML), use Basic HTTP authentication: check provided username and password (clear value) to identify and authenticate the requesting user and authorize access.

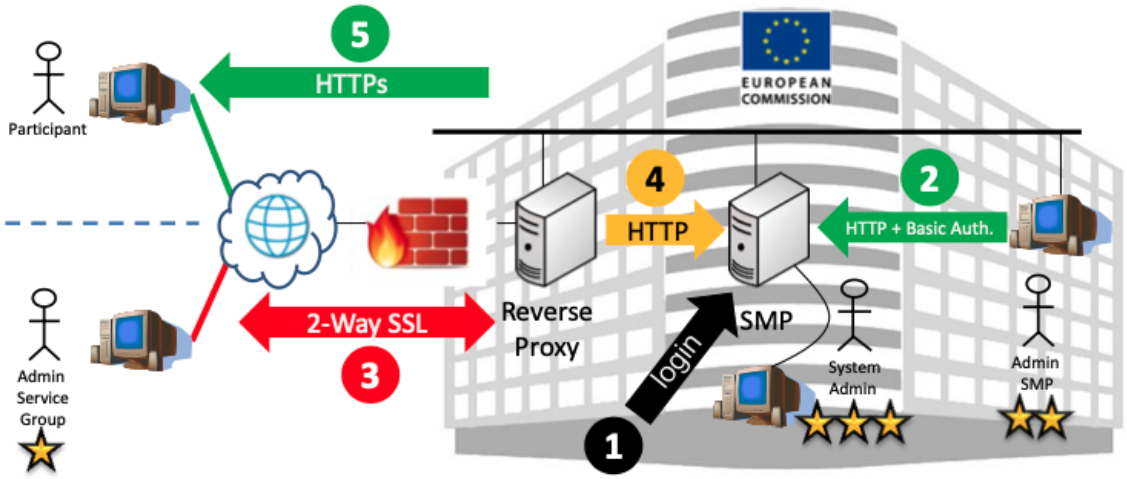


Figure 2. SML Deployment in European Commission

The SML deployed at the European Commission has the following accesses:

1. Direct system and database logins are used by the System Admin.
2. Basic authentication over HTTP is used for the *Group Admin* and *Resource’s* which share the same network location with SML.
   SMP authenticates the local Group Admin’s based on the hash of the password stored by the System Admin.
3. Certificates of remote *Admin Service Group’s* are authenticated by the Reverse Proxy.
4. Information of the client’s certificate is provided to SML for authorization (`Client-Cert` attribute), while the password is blank.
5. Parties do not have to authenticate themselves but may use SML’s certificate to authenticate it.

5.1.5. Auditing

Authentication Types Usage

All SMP services will log relevant information regarding access as specified in the table below.

▼ **Service Information Logging** *table*

Service’s Information Logging

	Manage Admins	Create or Update Service Group	Delete Service Group	Create or Update Service Metadata	Delete Service Metadata	Retrieve Service Group	Retrieve Service Metadata
Column/ *Description*	**UC01**	**UC02**	**UC03**	**UC04**	**UC05**	**UC06**	**UC07**
AdministratorIdentifier *Identified of the agent issuing request*	N/A	x	x	x	x	-	-
AccessTime *Time when access was executed*	N/A	x	x	x	x	x	x
Operation *Executed operation* (servicename)	N/A	x	x	x	x	x	x
ParticipantIdentifier *Identifier of the participant*	N/A	x	x	x	x	x	x
ParticipantIdentifierScheme *Scheme of the participant's identifier*	N/A	x	x	x	x	x	x
DocumentIdentifier *Identifier of the document*	N/A	-	-	x	x	-	x
DocumentIdentifierScheme *Scheme of the document's identifier*	N/A	-	-	x	x	-	x
IpAddress *Source IP address issuing the request*	N/A	x	x	x	x	x	x
RequestHeader *HTTP Header of the request*	N/A	x	x	x	x	x	x
RequestText *Text of the request (XML)*	N/A	x	-	x	-	-	-
ResponseHeader *HTTP Header of the response*	N/A	x	x	x	x	x	x
ResponseText *Rext of the response (XML)*	N/A	-	-	-	-	x	x
HTTP code *HTTP response code*	N/A	x	x	x	x	x	x
Business code *Application-level error code for HTTP error 40x*	N/A	x	x	x	x	-	-
ErrorDescription *Description of the error (free text)*	N/A	x	x	x	x	-	-

Whether to persist the logged information in a database table, log files or any other type of

persistence solution is a design decision, provided that the information is persisted and searchable.

NOTE	Audited information must be kept accessible (online or offline) for a minimum of three months.

No hard link (with foreign keys) is established between this table and the User or the Participant Identifier's to allow keeping:

- the logs relating to one user or one participant that is later removed from the database (if ever applicable),
- track of unauthorized calls from unidentified users or erroneous participant identifications.

5.2. Special requirements

- SMP should be available 99%.
- Response time should be less than:
  - 5s for `GET` services calls for 90% of the requests.
  - 10s for `PUT`/`DELETE` service calls for 90% of the requests.

Chapter 6. How-To Guides

About the DomiSMP How-To Guides

DomiSMP 5.0 How-to guides' with videos guide you through DomiSMP's features and its use. The guides also delve into the feature's related capabilities to help you in assessing their fitness for purpose in a particular scenario.

NOTE	The information provided in the videos, and used examples, are tied to the DomiSMP 5.0 version. For updated information regarding later releases, please see the specific's release documentation.

Guides Overview

Guide	Related Video
• Introducing the Dynamic Delegation Discovery System Infrastructure **Description:** Discover the components of a Dynamic Delegation Discovery System (DDDS). Learn about the Service Metadata Publisher (SMP), Service Metadata Locator (SML), Dynamic Discovery Client (DDC) and the Access Point (AP), and how they work together. Gain insights into creating a more efficient and secure message exchange network.	► https://www.youtube.com/watch?v=KSkWZ8PWlj4 *(YouTube video)* *DDDS Infrastructure*
• eDelivery Dynamic Discover Client (DDC) **Description:** Dive into the functionalities and advantages of the Dynamic Discovery Client. Understand how to implement and test DDDS infrastructure using the eDelivery DDC sample. Equip yourself with practical knowledge to enhance your digital communication infrastructure.	► https://www.youtube.com/watch?v=cnab-sYbt4Y *(YouTube video)* *eDelivery's DDC*
• Spring Boot deployment of DomiSMP **Description:** Learn how to launch DomiSMP as a Spring Boot application, including database setup and initial configuration, for demonstration and testing purposes. Master these steps to effectively deploy and test DomiSMP in your environment.	► https://www.youtube.com/watch?v=LjyXqnFhb90 *(YouTube video)* *Spring Boot deployment of DomiSMP*
• DomiSMP user interface overview and user settings **Description:** Get a comprehensive overview of the DomiSMP web application. This guide covers user profile management and the tools for managing user access tokens and certificates. Enhance your proficiency in DomiSMP.	► https://www.youtube.com/watch?v=I81U-9aRhcg *(YouTube video)* *User Interface Overview*

Guide	Related Video
• DomiSMP resource location and permissions **Description:** Explore the core concepts of DomiSMP, focusing on the resource locator for message capability documents and the system of realms, user roles and permissions. Understand the foundations of user administration in DomiSMP for better management and security.	► https://www.youtube.com/watch?v=I81U-9aRhcg *(YouTube video)* *DomiSMP resource location and permissions*

6.1. Guide: DDDS Infrastructure

This guide intends to provide you with an understanding of the **Dynamic Delegation Discovery System (DDDS)**, it:

- explains the essential components required to set up a DDDS infrastructure and their specific functions.
- highlights the benefits of integrating a DDDS into your message exchange networks.

TIP	We recommend following this guide sequentially, as each section builds upon the last to offer a complete overview.

6.1.1. DDS Infrastructure

This section elaborates on the *eDelivery's Dynamic Discovery Infrastructure*, including its primary components:

- Service Metadata Publisher (**SMP**)
- Service Metadata Locator (**SML**)
- Access Point (**AP**)

In this guide, we use a story-based approach to show how two enterprises can connect their information systems for exchanging business messages, and how to extend it to other business partners.
This approach aims to illustrate the purpose, reasons and benefits of using various components in the Dynamic Discovery infrastructure.

Business Case

Meet Alice from ConnectGlobe and Bob from PerfectService. Both companies seek to enhance their communication by transitioning from email and paper mail to electronic invoicing and order placements through their information systems.

Alice wants to order goods from Bob and receive electronic invoices from him through her information system. Similarly, Bob wants to get orders from Alice and send her electronic invoices.

To achieve this, Alice and Bob agree to implement integration modules on top of their information

systems. As a result, they get secure message exchanges over using HTTPS and TLS certificates. This not only streamlines communication but also lays the foundation for expanding their network.

This shift promises significant time and cost savings in processing business exchanges.

Reusing Integration Modules

The integration works very well for Alice and Bob. Alice wants to expand it to other business cases and message types.

Reusing a component greatly reduces the development and maintenance costs. Since the components are now loosely coupled, it makes it *easier to maintain* and *develop new integration module features*.
It also allows different teams to be in charge of the integration or messaging service on the backend system.

In this situation, the backend team can focus on the content and processing of documents, while the integration team can focus on reliable and secure message exchange.

Decoupling of components also simplifies system maintenance and feature development, allowing for easier team management and *potential outsourcing of the messaging service* to a third-party provider.
As a result, Alice and Bob can replace the messaging service with new technologies and standards with minimal or no interruptions on the backend side.

6.1.2. Service Metadata Publisher

Alice and Bob were happy with their integration and decided to *invite their colleagues and other business partners to join* them. At first, everything worked well, but as the number of partners increased, it became *hard to maintain the configuration for each of them.*

To manage the growing network and simplify configuration, Alice and Bob proposed a DDDS infrastructure layer to their messaging service, enabling automatic address and certificate retrieval for message exchanges. Behind that idea was *an address book*, similar to what Alice and Bob used internally for emails, looking up the address based on the receiver name or ID.

All the participants of the message exchange network would publish their addresses, certificates and other supported service capability metadata. Alice and Bob *upgraded their Access Point with a Dynamic Discovery Client (DDC)* enhancement that would look up the messages' recipient data and retrieve the missing information such as the receiver’s URL address and certificates.

This way Alice and Bob would not have to worry about collecting and maintaining the necessary transport data for the message exchange. They would just add the identifier of the recipients to the message, and the Access Point would find all the missing transport data from the address book and send the message to the right Access Point.

Because the *address book* contained the messaging service’s capabilities metadata, they renamed the component to *Service Metadata Publisher (SMP)*.

6.1.3. Service Metadata Locator

Some of the partners already had their own SMP components, some were using them as a service offered by SMP providers. And since these partners were satisfied with their existing setup, they did not want to move over to the shared SMP component as this would bring more maintenance and operational costs.

Alice and Bob faced the challenge: how a DDC could find the right SMP component for the final recipient? They extended their DDDS infrastructure with the *Service Metadata Locator* (SML), a component designed to easily locate the appropriate SMP server for message recipients, using the *Domain Name System* (DNS).

The goal was also to make that SML component user-friendly and its job as simple as retrieving an HTML page in the web browser can be. When you enter the URL address in you browser, the HTTP request targets the right server though you don't know the exact location of that server. Alice and Bob used the very same principle: DNS became the main technology of their SML service.

With this last component, everyone on the network could use their preferred SMP service instance or service provider. The network was ready to grow with new business partners.

6.1.4. Business Domain Owner

Every message exchange network has a community of participants that set up the network governance body, also known as the **business domain owner**.

The business domain owner, in our story represented by Alice and Bob, ensure the network is secure and efficient by taking care of the following:

- Network purpose and objectives: What problems will it solve, what benefits will it bring to the participants? Who are targeted participants and what are their roles in the network?
- Format and type of the participant identifiers: Who are the identifier providers and how to ensure the uniqueness of the identifiers?
- Visibility of the network: Will it be open to the public or restricted to a specific community?
- Types and structure of the documents they would exchange over the network, which is crucial for the backends to validate and process messages automatically.
- Trust model of the network, to ensure authenticity, integrity and confidentiality of the messages. This is a combination of technical solutions and network policies.
- How many SMP and SML providers there would be, who could operate them and how to ensure resources, sustainability and reliability of the services.

We will not go into more details on the business domain owner, but it's important to know that this role is the key to the success of a network.

6.1.5. Delegated Dynamic Discovery Process

This chapter provides an overview of how the Delegated Dynamic Discovery process works using all the described components.

Registering Participants

Before a new participant in the network (called "party") can start sending and receiving messages, it needs to do the following:

1. **Obtain its own unique identifier** in the network. This identifier will be used to address the party and to locate its services. The party can then **choose the AP and SMP service providers**. The party can reuse existing AP/SMP service providers or develop/set up its own instances and register them in the network.
2. **Register a unique identifier with the SMP service**. The SMP adds the entry to the SML to bind the participant identifier to the SMP instance. This way anyone looking for the party's data will be redirected to the dedicated SMP instance.
3. **Start publishing on the selected SMP services/message types it can accept**, along with the Access Point URL and transport certificates.

Submitting Messages

Here is a step-by-step explanation on how a network uses the delegated dynamic discovery process to submit a message from the sender to the recipient:

1. **Create the message**: The sender creates the message, assigning the recipient's unique identifier as the receiver.
2. **Send to Access Point (AP)**: The message is sent to the sender's AP. At this stage, the sender's AP lacks the recipient's delivery details.
3. **Query with DDC**: To find the recipient's details, the sender's AP uses the DDC to query the SML using the recipient's identifier.
4. **Retrieve SMP address data**: The SML responds with the address of the recipient's SMP, which holds their AP URL and certificate.
5. **Verify data integrity and authenticity**: Before proceeding, the sender's AP checks the integrity and authenticity of the retrieved data.
6. **Message delivery to recipient's AP**: Equipped with the necessary details, the sender's AP delivers the message to the recipient's AP.
7. **Recipient's AP validation**: Upon receipt, the recipient's AP validates the sender's message for integrity and authenticity. If validated, the message is accepted.
8. **Message retrieval by recipient**: Finally, the recipient fetches the message from their AP, completing the exchange process.

6.2. Guide: Dynamic Discovery Client

The **Dynamic Discovery Client (DDC)** serves as a pivotal tool within the **Dynamic Discovery** process. It allows for the identification and retrieval of **message exchange capabilities** from counterparties.

▼ *About this Guide*

Purpose of this guide

This document serves as a guide for utilising the DDC, detailing both its operation as a standalone command-line interface (CLI) component and its Access Point integration as a library.

This guide shows you how to integrate the **Dynamic Discovery Client** (DDC) as an independent component within your project. It aims to familiarise technical decision-makers, developers and testers with the eDelivery Dynamic Discovery infrastructure, providing insights into using the DDC for evaluating the SMP API and SML Dynamic Discovery processes.

What you will achieve

By the conclusion of this guide, you will have learnt how to use eDelivery's DDC sample command-line interface (CLI) to:

- Locate and retrieve SMP documents from both EU SEND SMP instances and GitLab repositories.
- Conduct document searches using custom NAPTR services (for example, "Meta:CPPA").
- Configure DDC's TLS Client Authentication for SMP document retrieval.
- Verify Service Metadata signatures.
- Directly fetch SMP Service Metadata, bypassing DNS lookup.

What you will need

- Approximately 30 minutes.
- A basic to medium understanding of Dynamic Discovery processes.
- Java 8 or higher installed.
- BIND 9 dig tool for DNS queries.
- The DDC .jar file.
- Access to Bash or an equivalent shell.

6.2.1. How DDC Works

The DDC performs the following to discover those capabilities:

- First, it locates the **Service Metadata Publisher (SMP)** where the counterparty has published its Service Metadata. It does this by sending a DNS query to the **Service Metadata Locator (SML)** DNS.
- Secondly, it sends an HTTP GET request to the SMP and obtains the counterpart's service capabilities metadata document.

```
@startuml
'style
autonumber "<b>(##)"
skinparam minClassWidth 120
skinparam sequenceArrowThickness 2
skinparam roundcorner 20
skinparam participantPadding 10
skinparam roundcorner 20
```

```
skinparam padding 5
skinparam sequenceMessageAlign center
skinparam sequenceLifeLineBorderColor #090909
skinparam sequenceLifeLineBackgroundColor #aaaaaa


'entities
participant "Dynamic Discovery\nClient" as DDC
participant "DNS (SML)\n" as SML
participant "SMP \n" as SMP
'Diagram

group Discovery target SMP
    activate DDC
    activate SML
    DDC ->  SML: DNS query
    SML --> DDC: SMP location Data
    deactivate SML
end
group Fetch target message exchange capabilities document
    activate SMP
    DDC -> SMP: HTTP GET request for metadata
    SMP --> DDC: Return metadata XML
deactivate SMP
end
@enduml
```

6.2.2. Step-by-step Guide

▼ *How to Complete This Guide*

Familiarise yourself with the Dynamic Discovery service and DDC features through brief feature descriptions and step-by-step instructions provided below.

Below find a step-by-step guide to the process of using the DDC to discover and fetch the counterpart's message exchange capabilities metadata document.

▼ **Step 1** *Download eDelivery's Dynamic Discovery Client*

Download the DDC sample. You can do it in the following ways:

- Use the wget command to get it from the eDelivery's Nexus Repository.

```
wget -O ddc.jar https://ec.europa.eu/digital-building-
blocks/artifact/repository/eDelivery/eu/europa/ec/dynamic-discovery/dynamic-
discovery-cli/2.1.1/dynamic-discovery-cli-2.1.1.jar
```

- Use your web browser (click here). Then, rename the downloaded file to **ddc.jar** to simplify the next steps.

▼ **Step 2** *Execute DDC client commands*

This step introduces you to basic DDC operations, such as finding SMP locations and fetching metadata.

Our DDC client is a Java application: ensure Java 8 or higher is installed on your computer.

Launch the DDC client via the terminal to display the available commands and their descriptions:

```
java -jar ddc.jar
```

```
Commands:

 -dns        Command discovers SMP location
 -get        Command discovers and fetch metadata from SMP
 -h,--help   Prints help

Use "java -jar ddc.jar -command_name --help" for usage of command_name.
```

The `-dns` command discovers the SMP location for the given participant identifier with the following options:

```
usage: java -jar ddc.jar -dns [-options]

'dns' options:

 -d,--domain <arg>                Network DNS domain: eq.:
                                  acc.edelivery.tech.ec.europa.eu
 -dns                             Command discovers SMP location
 -h,--help                        Prints help
 -o,--output <arg>                Output filename. If file already exists
                                  it is overwritten. If not provided,
                                  output is printed to console
 -ri,--resource-identifier <arg>  party identifier: ex: 0088:98765digit
 -rs,--resource-scheme <arg>      party identifier: iso6523-actorid-upis
 -s,--services <arg>              Comma separated NAPTR service value as:
                                  Meta:SMP,meta:cppa
 -t,--dns-type <arg>              List of DNS record types, CNAME,NAPTR

Use "java -jar ddc.jar -dns --help" for usage of command_name.
```

The `-get` command discovers and fetches metadata from the SMP.

```
usage: java -jar ddc.jar -get [-options]

'get' options:

 -d,--domain <arg>                   Network DNS domain: eq.:
                                     acc.edelivery.tech.ec.europa.eu
```

```
 -get                                  Command discovers and fetch metadata
                                       from SMP
 -h,--help                             Prints help
 -kf,--keystore-filepath <arg>         Client TLS keystore file path
 -kp,--keystore-password <arg>         Client TLS keystore password
 -kt,--keystore-type <arg>             Client TLS keystore type: Default
                                       PKCS12
 -o,--output <arg>                     Output filename. If file already
                                       exists it is overwritten. If not
                                       provided, output is printed to
                                       console
 -ri,--resource-identifier <arg>       party identifier: ex:
                                       0088:98765digit
 -rs,--resource-scheme <arg>           party identifier:
                                       iso6523-actorid-upis
 -s,--services <arg>                   Comma separated NAPTR service value
                                       as: Meta:SMP,meta:cppa
 -si,--subresource-identifier <arg>    Subresource identifier: ex: Invoice
 -ss,--subresource-scheme <arg>        Subresource identifier:org:xml
 -t,--dns-type <arg>                   List of DNS record types,
                                       CNAME,NAPTR
 -tf,--truststore-filepath <arg>       TLS truststore file path
 -tp,--truststore-password <arg>       TLS truststore password
 -tt,--truststore-type <arg>           TLS truststore type: Default PKCS12

Use "java -jar ddc.jar -get --help" for usage of command_name.
```

▼ **Step 3** *Discover SMP service for participant*

Now that we have successfully executed our first command, we can discover and fetch a sample document. For this, we will use the DDC to identify the message exchange capabilities metadata for the following Participant Identifier: , `urn:oasis:names:tc:ebcore:partyid-type:unregistered:smp-guide:party-id-001`.

The participant published its data on the eDelivery EU SEND SMP service with the following SML DNS domain: `test.acc.edelivery.tech.ec.europa.eu`.

The following command discovers and fetches the OASIS SMP 1.0 ServiceGroup document from the EU SEND SMP:

```
java -jar ddc.jar -dns -ri urn:oasis:names:tc:ebcore:partyid-type:unregistered:smp-
guide:party-id-001  -d test.acc.edelivery.tech.ec.europa.eu -t CNAME,NAPTR
```

Note the following parameters:

- `-ri urn:oasis:names:tc:ebcore:partyid-type:unregistered:smp-guide:party-id-001`: Participant Identifier.
- `-d test.acc.edelivery.tech.ec.europa.eu`: DNS domain name of the SML service.
- `-t CNAME,NAPTR`: List of DNS record types to query. If list is not given, only NAPTR records are

queried.

The output to the previous command is the following:

```
Resolving DNS for participant: [ParticipantIdentifier{identifier
='urn:oasis:names:tc:ebcore:partyid-type:unregistered:smp-guide:party-id-001',
scheme='null'}] and domain: [test.acc.edelivery.tech.ec.europa.eu]

CNAME query: B-70c708123575f9c8e308f5abbdaf979a.test.acc.edelivery.tech.ec.europa.eu
B-70c708123575f9c8e308f5abbdaf979a.test.acc.edelivery.tech.ec.europa.eu.    60  IN
CNAME   SMP-SHS-ACC-TEST.publisher.test.acc.edelivery.tech.ec.europa.eu.
SMP-SHS-ACC-TEST.publisher.test.acc.edelivery.tech.ec.europa.eu.    60  IN  CNAME
smp-ext-acc.eusfx.ec.europa.eu.
smp-ext-acc.eusfx.ec.europa.eu. 2706    IN  CNAME   ip-star-eusfx.ec.europa.eu.


NAPTR query:
B4FBLI2PMR6YHV4TBK27AB6LFYTG4GQAITDM4O74IUE7CPEINVGQ.test.acc.edelivery.tech.ec.euro
pa.eu
B4FBLI2PMR6YHV4TBK27AB6LFYTG4GQAITDM4O74IUE7CPEINVGQ.test.acc.edelivery.tech.ec.euro
pa.eu.  60  IN  NAPTR   100 10 "U" "Meta:SMP" "!.*!https://smp-ext-
acc.eusfx.ec.europa.eu/!" .
```

▼ **Step 4** *Fetch OASIS SMP 1.0 ServiceGroup document*

Fetch OASIS SMP 1.0 ServiceGroup document from eDelivery EU SEND SMP service

In this step we will discover and retrieve an OASIS SMP 1.0 ServiceGroup document for the following Participant Identifier: `urn:oasis:names:tc:ebcore:partyid-type:unregistered:smp-guide:party-id-001`.

The document is part of the OASIS SMP 1.0 specification, which contains a list of documents for which Service Metadata capabilities are published by the participant.

To discover and retrieve the document, use the `-get` command:

```
java -jar ddc.jar -get  -ri urn:oasis:names:tc:ebcore:partyid-type:unregistered:smp-
guide:party-id-001 -d test.acc.edelivery.tech.ec.europa.eu -o service-group.xml
```

Note the following parameters:

- `-ri urn:oasis:names:tc:ebcore:partyid-type:unregistered:smp-guide:party-id-001`: Participant Identifier.
- `-d test.acc.edelivery.tech.ec.europa.eu`: DNS domain name of the SML service.
- `-o service-group.xml`: File name for the retrieved document. If the file already exists, it is overwritten. If not provided, the output is printed to the console.

Our document has been saved as `service-group.xml`. Here is its content:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServiceGroup xmlns="http://docs.oasis-open.org/bdxr/ns/SMP/2016/05">
    <ParticipantIdentifier scheme="urn:oasis:names:tc:ebcore:partyid-
type:unregistered:smp-guide">party-id-001
    </ParticipantIdentifier>
    <ServiceMetadataReferenceCollection>
        <ServiceMetadataReference
                href="https://smp-ext-
acc.eusfx.ec.europa.eu/urn%3Aoasis%3Anames%3Atc%3Aebcore%3Apartyid-
type%3Aunregistered%3Asmp-guide%3Aparty-id-001/services/message-
exchange%3A%3Aconnectivity-document-01"/>
    </ServiceMetadataReferenceCollection>
</ServiceGroup>
```

The file was retrieved from the eDelivery EU SEND SMP service. The SMP instance's address (https://smp-ext-acc.eusfx.ec.europa.eu) was discovered using the DNS query as described in the previous chapter. To retrieve the document, the DDC client appended the Participant Identifier to the SMP URL address as in the example below:

```
https://smp-ext-acc.eusfx.ec.europa.eu/urn:oasis:names:tc:ebcore:partyid-
type:unregistered:smp-guide:party-id-001
```

▼ **Step 5** *Fetch sub-resource document*

Fetch sub-resource document: OASIS SMP 1.0 Service Metadata from eDelivery EU SEND SMP service.

The OASIS SMP 1.0 ServiceMetadata document is part of the OASIS SMP 1.0 specification, which contains the endpoint information for a specific document type and process identifier that an entity can receive in the network. A client can request a ServiceMetadata document from an SMP service to discover the transport methods, certificates, and other details that are required to exchange business documents with the participant.

To retrieve the ServiceMetadata document with the `message-exchange::connectivity-document-01` ID for the participant `urn:oasis:names:tc:ebcore:partyid-type:unregistered:smp-guide:party-id-001`, use the following command:

```
java -jar ddc.jar -get -ri urn:oasis:names:tc:ebcore:partyid-type:unregistered:smp-
guide:party-id-001 \
      -si connectivity-document-01 \
      -ss message-exchange  \
      -d test.acc.edelivery.tech.ec.europa.eu  \
      -o service-metadata.xml
```

Note the following parameters:

- `-ri urn:oasis:names:tc:ebcore:partyid-type:unregistered:smp-guide:party-id-001`: Participant Identifier.
- `-d test.acc.edelivery.tech.ec.europa.eu`: DNS domain name of the SML service.

- `-si connectivity-document-01`: Sub-resource identifier.
- `-ss message-exchange`: Sub-resource scheme.
- `-o service-metadata.xml`: File name for the retrieved document. If the file already exists, it is overwritten. If not provided, the output is printed to the console.

Our document has been saved as **service-metadata.xml**. Here is its content:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<SignedServiceMetadata xmlns="http://docs.oasis-open.org/bdxr/ns/SMP/2016/05">
    <ServiceMetadata>
        <ServiceInformation>
            <ParticipantIdentifier scheme="urn:oasis:names:tc:ebcore:partyid-
type:unregistered:smp-guide">
                party-id-001
            </ParticipantIdentifier>
            <DocumentIdentifier scheme="message-exchange">connectivity-document-
01</DocumentIdentifier>
            <ProcessList>
                <Process>
                    <ProcessIdentifier scheme="mmessage-service-type">submit-
message</ProcessIdentifier>
                    <ServiceEndpointList>
                        <Endpoint transportProfile="bdxr-transport-ebms3-as4-v1p0">
                            <EndpointURI>https://my-ap-address.local</EndpointURI>
                            <Certificate>Q2VydGlmaWNhdGUgZGF0YSA=</Certificate>
                            <ServiceDescription>Service for submitting
messages</ServiceDescription>
                            <TechnicalContactUrl>www.my-
company.eu</TechnicalContactUrl>
                        </Endpoint>
                    </ServiceEndpointList>
                </Process>
            </ProcessList>
        </ServiceInformation>
    </ServiceMetadata>
    <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
        <SignedInfo>
            <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-
c14n-20010315"/>
            <SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-
sha256"/>
            <Reference URI="">
                <Transforms>
                    <Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
                </Transforms>
                <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
                <DigestValue>
e5riWHrertUkilvKGRmNY8clcrOk3YToAreumZyq7XY=</DigestValue>
```

```
            </Reference>
        </SignedInfo>
        <SignatureValue>im9n8cRJ...GA27YzA==
        </SignatureValue>
        <KeyInfo>
            <X509Data>

<X509SubjectName>CN=EUSEND_SMP_ACC,OU=EUSEND,OU=SMP_ACC,OU=CEF_eDelivery.europa.eu,O
=eDelivery
                    Support,C=BE
                </X509SubjectName>
                <X509Certificate>MIIFyDCC...fTI=</X509Certificate>
            </X509Data>
        </KeyInfo>
    </Signature>
</SignedServiceMetadata>
```

▼ **Step 6** *Fetch document from git repository*

Because the retrieval of the document uses a basic HTTP GET request, the DDC can be used to fetch documents from any HTTP server beyond the dedicated SMP services. In this example we will fetch a sample document from a git repository.

The document is published at Gitlab: https://code.europa.eu/edelivery/cppa3/-/tree/main/cpp/.

The participant has the following Participant Identifier: `iso6523-actorid-upis::0088:test:cppa3`.

For that participant, the following NAPTR record was added to the SML service:

```
DLORA46AHSNQ56YTTUBALSYNTZYLHQ7JROBAUJJMR6D2TUF6CFJA.iso6523-actorid-
upis.test.acc.edelivery.tech.ec.europa.eu 100 10 "U" "meta:cppa3"
"!.*!https://code.europa.eu/edelivery/cppa3/-/raw/main/cpp/!" .
```

The DNS NAPTR record can be validated with the following DDC command:

```
java -jar ddc.jar -dns -ri 0088:test:cppa3 -rs iso6523-actorid-upis -s meta:cppa3 -d
acc.edelivery.tech.ec.europa.eu
```

Note the following parameters:

- `-ri 0088:test:cppa3`: Participant identifier, the Id part of the participant identifier.
- `-rs iso6523-actorid-upis`: Participant identifier scheme.
- `-d test.acc.edelivery.tech.ec.europa.eu`: DNS domain name of the SML service.
- `-s Meta:CPPA3`: NAPTR service value to search for. By default, the DDC client searches for the Meta:SMP service value, but it can be configured to search for other service values as well such as CPPA/CPP documents in this case.

The same DNS query can be executed using the Bind9 dig command, but when using dig, we

have to create a query string/domain DLORA46AHSNQ56YTTUBALSYNTZYLHQ7JROBAUJJMR6D2TUF6CFJA.iso6523-actorid-upis.test.acc.edelivery.tech.ec.europa.eu manually. The following command can be used to execute the DNS query:

```
dig DLORA46AHSNQ56YTTUBALSYNTZYLHQ7JROBAUJJMR6D2TUF6CFJA.iso6523-actorid-
upis.acc.edelivery.tech.ec.europa.eu NAPTR
```

To retrieve the document the participant/resource identifier, use the following command -get:

```
java -jar ddc.jar -pi 0088:test:cppa3 -ps iso6523-actorid-upis -s meta:cppa3 -d
acc.edelivery.tech.ec.europa.eu -o data.xml
```

Our document has been saved as `data.xml`.

▼ **Step** 7 *Direct document fetch*

In some cases the user may want to fetch the document directly from the known SMP instance without the DNS lookup. The DDC can be configured to fetch the document directly from the known SMP instance. The following example shows how to fetch the document directly from the known SMP instance.

```
java -jar ddc.jar -get  -ri urn:oasis:names:tc:ebcore:partyid-type:unregistered:smp-
guide:party-id-001 \
    -smp https://smp-ext-acc.eusfx.ec.europa.eu/ \
    -o direct-fetch.xml
```

Note the following parameters:

- `-smp https://smp-ext-acc.eusfx.ec.europa.eu/test/smp-1/`: SMP instance address to fetch the document from. When the SMP instance address is provided, the DDC client will fetch the document directly from the SMP instance without the DNS lookup.

Further reading

For a summary of the knowledge discussed until now and recommendations for next steps, see the next guide, Spring Boot deployment of DomiSMP.

6.3. Guide: DomiSMP as a Spring Boot Application

▼ *About this Guide*

Purpose of the guide

This guide provides instructions for starting the DomiSMP application using Spring Boot, aiming to simplify the process for demonstrations and testing. It is intended for project's technical decision-makers, developers and testers interested in using DomiSMP for Access-Point development.

What you will achieve

By following this guide, you will learn how to set up and start the DomiSMP as a Spring Boot application.

What you will need

- About 20 minutes of your time.
- A basic understanding of the Dynamic Discovery process.
- Java 8 or higher installed.
- MySQL 8.x+ database server or Docker with the pre-downloaded MySQL 8.3.0 image.
- Access to a Bash shell.

6.3.1. Step-by-step Guide

How to complete this guide

This guide will walk you through the steps to start DomiSMP with Spring Boot.

Step 1: Download DomiSMP application & deployment bundle

Create a working folder referred to as **${SMP_HOME}**. Then, navigate to it. This folder will store the DomiSMP application and deployment bundle.

Download DomiSMP from one of the following sources:

- eDelivery Nexus repository. Make sure the file is called **smp.jar**.
- Use the following wget command:

```
wget -O smp.jar https://ec.europa.eu/digital-building-
blocks/artifact/repository/eDelivery/eu/europa/ec/edelivery/smp-springboot/5.0.1/smp-
springboot-5.0.1-exec.jar
```

Download the DomiSMP deployment bundle from one of the following sources:

- eDelivery Nexus repository.
- Use the following wget command:

```
wget -O smp-5.0.1-setup.zip https://ec.europa.eu/digital-building-
blocks/artifact/repository/eDelivery/eu/europa/ec/edelivery/smp/5.0.1/smp-5.0.1-
setup.zip
```

Step 2: Start the Docker MySQL database

Skip this step if using a local MySQL database.

If using Docker, pull and start the MySQL 8.3.0 image with the following commands:

```
docker pull mysql:8.0.23
docker run --name domismp-mysql -e MYSQL_ROOT_PASSWORD=root -p 3307:3306 -d
mysql:8.3.0
```

To connect to the MySQL database:

```
docker exec -it domismp-mysql mysql -uroot -proot
```

The command starts the MySQL database with the following parameters:

- `--name domismp-mysql`: Name of the container.
- `-e MYSQL_ROOT_PASSWORD=root`: Root password of the MySQL database.
- `-p 3307:3306`: Port mapping of the MySQL database. Port **3307** of the host is mapped to port 3306 of the container.
- `-d`: Starts the container in the background.

Step 3: Initialise the database

In this step, we will initialise the database with the DomiSMP database schema and initial data. We will execute the database commands manually. In the next chapter, we will provide Bash scripts that can be used to initialise the database.

To initialize the database, we will use script files that are part of the DomiSMP deployment bundle.

Unzip `smp-5.2-setup.zip` to `${SMP_HOME}`. Navigate to the `${SMP_HOME}/smp-5.2/database-scripts/` folder, which contains the `mysql.ddl` file with the DomiSMP database schema and the `mysql-data.sql` file with the initial data for the DomiSMP database.

To create DomiSMP, we will use the following variables:

- Database schema: **smpdb**
- Database user: **smp**
- Database user password: **smp**
- Database root user: **root**
- Database root user password: **root**

Step 3.1: Initialise the database schema

If using a local MySQL database, connect to it and execute the following command:

```
mysql -h localhost -u root --password=root -e "drop schema if exists smpdb;DROP USER
IF EXISTS smp;create schema smpdb;alter database smpdb charset=utf8;create user smp
identified by 'smp';grant all on smpdb.* to smp"
```

If using a Docker MySQL database, the same can be done with the following command:

```
docker exec -it domismp-mysql mysql -h localhost -u root --password=root -e "drop
schema if exists smpdb;DROP USER IF EXISTS smp;create schema smpdb;alter database
smpdb charset=utf8;create user smp identified by 'smp';grant all on smpdb.* to smp"
```

When using a Docker MySQL database, the command is prepended by `docker exec -it domismp-mysql` which executes the `mysql` command inside the Docker container. Moving forward, we will provide here only the `mysql` command.

Step 3.2 Create the DomiSMP database schema objects

Execute the following command (replace `${SMP_HOME}` with the actual path to the `smp-<x>-setup` parent folder):

```
mysql -u smp --password=smp smpdb < ${SMP_HOME}/smp-<x>/database-scripts/mysql.ddl ①
```

Where:

① `<x>`, stands for the current release, 5.2, so refers to the path: `/smp-5.2/database-scripts/mysql.ddl`.

When script is executed, the DomiSMP database schema is created and we can validate the schema with the following:

```
mysql -u smp --password=smp smpdb -e "show tables"
```

The command should return the list of the tables in the **smpdb** database.

Step 3.3: Insert the initial data

Execute the following:

```
mysql -u smp --password=smp smpdb < ${SMP_HOME}/smp-<x>/database-scripts/mysql-
data.sql
```

Where:

① `<x>`, stands for the current release, 5.2, so refers to the path: `/smp-5.2/database-scripts/mysql.ddl`.

The main purpose of the script is to insert demo users:

- **system/123456** with the SYSTEM_ADMIN role
- **user/123456** with the USER role

To check if the data is inserted, execute the following:

```
mysql -u smp --password=smp smpdb -e "select * from SMP_USER"
```

Step 4: Initialise the database (scripts)

For the testing and demo purposes, we would like to have a clean database each time we start the DomiSMP application. In this step we will provide the Bash script that can be used to initialise the database. The script is provided as an example and should be adjusted to meet the local environment requirements.

To run the script, ensure the following:

- Correct command-line emulator (example uses **/bin/sh**).
- Locally installed MySQL database.
- Clone DomiSMP repository and check out the right branch (for example, check out the development branch for development). The repository contains the database DDL scripts.

Before executing the sample script, set the following variables:

- `PROJECT_HOME`: The DomiSMP code/project home (for example, **/code/smp**).
- `DATABASE`: smp database schema.
- `DB_ADMIN`: MySQL database root username.
- `DB_ADMIN_PASSWORD`: MySQL database root password.
- `DB_USERNAME`: DomiSMP MySQL database username.
- `DB_PASSWORD`: DomiSMP MySQL database username.

Explanation of the script: The script connects to MySQL database using **mysql** (CLI tool) and deletes database/schema and user defined in the variables `DATABASE` and `DB_USERNAME`. The DomiSMP schema is generated from the following script:

```
${SMP_HOME}/smp-<x>/database-scripts/mysql.ddl ①
```

and the initial data is inserted from the following script:

```
${SMP_HOME}/smp-<x>/database-scripts/mysql-data.sql ①
```

Where:

① `<x>`, stands for the current release, 5.2, so refers to the path: `/smp-5.2/database-scripts/mysql.ddl`.

Linux OS:

```
#!/bin/sh

PROJECT_HOME=/cef/code/smp

DATABASE=smpdb
DB_ADMIN=root
DB_ADMIN_PASSWORD=root
DB_USERNAME=smp;
DB_PASSWORD=smp;
DB_CREATE_SCRIP=${PROJECT_HOME}/smp-<x>/database-scripts/mysql.ddl ①
DB_INIT_SCRIPT=${PROJECT_HOME}/smp-<x>>/database-scripts/mysql-data.sql ①

# recreate database
echo "clean the database $DATABASE if exists "
mysql -h localhost -u $DB_ADMIN --password=$DB_ADMIN_PASSWORD -e "drop schema if
exists $DATABASE;DROP USER IF EXISTS $DB_USERNAME;  create schema $DATABASE;alter
database $DATABASE charset=utf8; create user $DB_USERNAME identified by '
$DB_PASSWORD';grant all on $DATABASE.* to $DB_USERNAME;"

# create new database
echo "create database"
mysql -h localhost -u $DB_ADMIN --password=$DB_ADMIN_PASSWORD $DATABASE <
"$DB_CREATE_SCRIP"
echo "init database for soapui tests"
mysql -h localhost -u $DB_ADMIN --password=$DB_ADMIN_PASSWORD $DATABASE <
"$DB_INIT_SCRIPT"
```

Where:

① <x>, stands for the current release, 5.2, so refers to the path: /smp-5.2/database-scripts/mysql.ddl.

Windows OS:

```
@echo off


set PROJECT_HOME=C:\\cef\\code\\smp
set DATABASE=smpdb
set DB_ADMIN=root
set DB_ADMIN_PASSWORD=
set DB_USERNAME=smp
set DB_PASSWORD=smp
set DB_CREATE_SCRIP=${PROJECT_HOME}\\smp-<x>\\database-scripts\\mysql.ddl ①
set DB_INIT_SCRIPT=${PROJECT_HOME}\\smp-<x>\\database-scripts\\mysql-data.sql ①

REM recreate database
echo "clean the database %DATABASE% if exists "
mysql -h localhost -u %DB_ADMIN% --password=%DB_ADMIN_PASSWORD% -e "drop schema if
```

```
exists %DATABASE%;DROP USER IF EXISTS %DB_USERNAME%;create schema %DATABASE%;alter
database %DATABASE% charset=utf8;create user %DB_USERNAME% identified by
'%DB_PASSWORD%';grant all on %DATABASE%.* to %DB_USERNAME%;"


REM create new database
echo "create database"
mysql -h localhost -u %DB_ADMIN% --password=%DB_ADMIN_PASSWORD% %DATABASE% <
"%DB_CREATE_SCRIP%"
echo "init database for soapui tests"
mysql -h localhost -u %DB_ADMIN% --password=%DB_ADMIN_PASSWORD% %DATABASE% <
"%DB_INIT_SCRIPT%"
```

Where:

① `<x>`, stands for the current release, 5.2, so refers to the path: `/smp-5.2/database-scripts/mysql.ddl`.

Step 5: Prepare the application.properties

For the DomiSMP database configuration, set the following properties:

- `smp.jdbc.hibernate.dialect`: Database hibernate dialect name.
- `jdbc.driver`: jdbc driver. The MySQL driver is embedded by default. In case of other drivers, add them to **pom.xml** and rebuild the Spring Boot application.
  - `smp.jdbc.url`: URL of the database.
  - `smp.jdbc.user`: Database username
  - `smp.jdbc.password`: Database password.

To set/change other Spring Boot parameters (for example, `server.port`), refer to the Spring Boot documentation.

The configuration properties must be set in the **application.properties** file and placed in the working directory of the DomiSMP Spring Boot application.

For alternatives on how to set Spring Boot properties, refer to the Spring Boot documentation.

For DomiSMP startup properties, refer to DomiSMP Administration Guide.

Only the "java property type" of the Spring Boot properties format is supported (JSON or YAML types are not supported!).

Example of Spring Boot configuration in the **application.properties** file (adapt the properties to your local MySQL installation configuration):

```
# the tomcat server port
server.port=8084
```

```
# Database configuration
smp.jdbc.hibernate.dialect=org.hibernate.dialect.MySQLDialect

# *********************************
#  Custom defined datasource
# *********************************
# mysql database example
smp.jdbc.driver=com.mysql.jdbc.Driver
# in case of the docker mysql database the url is jdbc:mysql://localhost:3307/smpdb
smp.jdbc.url=jdbc:mysql://localhost:3306/smpdb
smp.jdbc.user=smp
smp.jdbc.password=smp
```

Step 6: Start the application

Start the application with the following command in the working directory of the DomiSMP Spring Boot application:

```
java -jar smp.jar
```

If configured to use port 8084, the application will be accessible at http://localhost:8084/smp/.

Further reading

Review the concepts covered in this guide and explore further documentation for advanced features and next steps.

6.4. Guide: Resource Locator and Permissions

The purpose of the eDelivery Building Block is to enable the electronic exchange of digital data and documents in an interoperable, secure, reliable and trusted way for businesses and public administrations. eDelivery provides sample component implementations such as Domibus, DomiSML and DomiSMP to promote and facilitate the electronic message exchange for various participant messaging networks (also called business domains).

This document explains explains two key concepts of the DomiSMP application: The resources locator and the DomiSMP realms.

▼ *Purpose of this guide*

This guide will help you understand DomiSMP realms such as domain, groups and resources and the access permissions. In the second part of the guide, you will learn how to locate specific resources in DomiSMP with the HTTP-GET query path.

We recommend reading from start to end for a comprehensive understanding, though you may skip to relevant sections if familiar with some concepts.

What you will learn

- How to locate specific resources in DomiSMP.

- About DomiSMP realms and how to authorise access to resources.

What you will need

- Approximately 10 minutes of your time.

6.4.1. DomiSMP Application

DomiSMP supports various network requirements, allowing for custom extensions and supporting multiple domains where each domain can have its own metadata resource types and users/network administrators.

The application is document-agnostic, allowing to publish document metadata business logic in extensions, which network users can develop independently from the core application.

Users and resources can be organised into groups, each having an SMP administrator. Groups come in handy in case of large networks spanning multiple countries, where each country can have its own SMP administrator and its own set of resources and users.

DomiSMP is shipped with a default extension for OASIS SMP 1.0/2.0 and as an experimantal feature also the OASIS CPPA3 CPP documents.

In the following sections we will explain how specific resources are located in DomiSMP and then how access to these resources is controlled.

Realms and Permissions

Within DomiSMP, resources, users, and permissions are organised into a structured framework known as realms. This section delves into the three-layer hierarchical realms of the domain, group, and resource.

▼ *Realms*

Domain realm

The highest level, indicating the network's business aim, like invoice exchange or health records exchange. Domain admins manage the overall structure, including authentication and authorisation methods (like PKI certificates), Participant Identifier types, and messages formats. In DomiSMP, domain admins are responsible for creating/deleting groups and assigning group admins.

Group realm

This mid-level layer consists of clusters of resources and users. It allows for the organisation of resources and users into manageable segments, particularly useful in large networks. Managed by group administrators, who can create and delete resources, but only the resource admins can edit resource documents. A group admin invites participants to the exchange network and assigns them resources where they can publish their connectivity data.

Resource realm

The most basic layer. It consists of resources and its subresources, both of which are documents describing participant connectivity capabilities. Resources are identified by unique IDs (usually matching the owner identifier) and can include formats like OASIS SMP

1.0, OASIS SMP 2.0 and CPPA3 CPP document. The resource type support is done by the DomiSMP extensions, which will be explained in one of the following user guides.

▼ *Permissions*

Each realm has its own set of permissions, determined by the user's role within that realm. Here's how permissions work across different realms:

Resource realm permissions

Control over specific documents. While resource admins can modify documents, group admins can invite participants but cannot alter documents directly.

Group realm permissions

Group admins manage resources within their group, including adding or removing resources and inviting new participants.

Domain realm permissions

Domain admins have the authority to create or delete groups within their domain, assign group admins, and set the framework for the network's operational guidelines.

In essence, the DomiSMP's structured approach to realms and permissions facilitates a clear and manageable way to organise, access, and administer resources within the network, ensuring secure and efficient data exchange.

In the diagram below, see the DomiSMP realms and the relations between them. DomiSMP is the host to the domains **eHealth** and **Invoice**. The groups in each domain are organised by EU Member States: Spain, Belgium, and Sweden organise the eHealth domain, while Belgium, France, Germany, and Austria organise the Invoice domain.

```
@startuml
!include https://raw.githubusercontent.com/plantuml-stdlib/C4-
PlantUML/master/C4_Container.puml

HIDE_STEREOTYPE()
Boundary(Domain2, "Domain [eHealth]"){

    Boundary(D2G1, "Group [Spain]"){
        Container(D2R1G1, "Resource H02", "SMP document")
        Container(D2R2G1, "Resource H01", "SMP document")
    }
    Boundary(D2G2, "Group [Belgium]"){
        Container(D2R1G21, "Resource H03", "SMP document")
    }

    Boundary(D2G3, "Group [Sweden]"){
            Container(D2R1G4, "Resource H04", "SMP document")
        }
}

Boundary(Domain, "Domain [Invoice]"){
```

```
    Boundary(G2, "Group [Belgium]"){
        Container(R1G2, "Resource I03", "SMP document")
    }
    Boundary(G1, "Group [France]"){
        Container(R1G1, "Resource I02", "SMP document")
        Container(R2G1, "Resource I01", "SMP document")
    }

    Boundary(G3, "Group [Germany]"){
        Container(R1G3, "Resource I06", "SMP document")
        Container(R2G3, "Resource I05", "SMP document")
    }

    Boundary(G4, "Group [Austria]"){
            Container(R1G4, "Resource I04", "SMP document")
        }
}

Lay_D(G1, G4)
Lay_D(G2, G3)

Lay_D(D2G1, D2G2)
Lay_D(D2G1, D2G3)


@enduml
```

Users

In DomiSMP, users are individuals or entities with the capability to interact with and manage resources and subresources. Users can be either added manually by a system administrator or automatically through CAS, such EU Login. The permissions assigned to users depend on two factors: their role within DomiSMP and their membership within specific realms (domains, groups, and resources).

User Application Roles

The DomiSMP has two application roles which will be described below. The application roles are used to define the user's access and control levels in the DomiSMP application.

User

The default role assigned to all new users. Initially, it offers the same permissions as an unauthenticated user. Additional privileges can be granted by assigning further roles relevant to specific realms.

System Admin

This role has comprehensive control over the DomiSMP system. A system admin can configure the system settings, register new document types and extensions, and set up authentication methods for different domains.

Memberships

Memberships define a user's access level within the various realms of DomiSMP. There are two types of memberships:

Direct membership: Users directly invited to a realm have direct access to it.

Indirect membership: Users gain access through membership in a sub-realm, extending their permissions to the parent realm.

Administrators manage memberships within their respective realms. They can invite new users and assign roles, determining what the user is allowed to do within that realm. The roles and their permissions are specific to the activities within the realms of domains, groups, and resources.

6.4.2. Membership Roles

DomiSMP organises users and their access through a structured hierarchy of realms, each with distinct roles and responsibilities. This section outlines the various roles within these realms and what they allow a user to do.

Roles list

- **Anonymous**: This is the default role for all unauthenticated users, providing the most basic access level.
- **Resource Viewer**: Users with this role can view resources within the realms they belong to but cannot make any changes.
- **Resource Admin**: Users in this role manage resources directly, capable also of editing the resource membership. Before, this role was called ServiceGroup Owner.
- **Group Viewer**: Group viewers have visibility over all resources within a group but cannot modify these resources nor add/delete any.
- **Group Admin**: These users manage group resources, including adding or removing resources and editing group and resource memberships. Before, this role was called SMP Admin.
- **Domain Viewer**: With this role, a user can see all resources within a domain but not modify them.
- **Domain Admin**: Domain admins have the highest level of control, able to create or delete domain groups, and modify domain and group memberships.

Roles Overview

A user can have only one role assigned.

Member	**Resource**		**Group**		**Domain**		**System**
Action	**Viewer**	**Admin**	**Viewer**	**Admin**	**Viewer**	**Admin**	**Admin**
Domain: - Create/Delete							✔
Domain: - Modify						✔	

Member	Resource		Group		Domain		System
Domain: - View all internal domain resources					✔	✔	
Domain Group: - Create/Delete						✔	
Domain Group: - Modify				✔			
Domain : - View all internal domain group resources			✔	✔			
Resource: - Create/Delete				✔			
Resource: - Modify		✔					
Resource: - View	✔	✔	✔	✔	✔	✔	✔
Members: - Invite/Remove	x	✔ (to resource)	✔ (to resource)	✔	✔	x	x
Members: - Modify role	x	x	x	✔	✔	x	x

Membership Organisation

The membership model in DomiSMP is hierarchical. At the top, System Admins have the authority to add new members across DomiSMP domains. Each domain, managed by Domain Admins, can contain multiple groups. Domain Admins are responsible for creating groups and appointing Group Admins. Group Admins, in turn, can invite members to their groups and assign specific resources. This structure ensures a clear and organised way to manage DomiSMP users and their permissions.

Below, we detail this structure in a diagram, which presents the hierarchical roles that define the user's access and control levels in DomiSMP.

```
@startuml
hide empty description
skinparam rectangleBorderThickness 1
skinparam defaultTextAlignment center
skinparam lifelineStrategy solid
skinparam monochrome true
skinparam style strictuml
hide empty members
skinparam Linetype ortho
skinparam nodesep 80
skinparam ranksep 80
```

```
skinparam SameClassWidth true
skinparam class {
  BackgroundColor lightgray
  BorderColor black
  BorderThickness 1
  FontColor black
  FontStyle normal
  FontSize 18
}

    SYSTEM_ADMIN ||--o{ DOMAIN_ADMIN : "Invite/Authorize"
    DOMAIN_ADMIN ||--o{ DOMAIN_ADMIN :  "Invite/Authorize"
    DOMAIN_ADMIN ||--o{ GROUP_ADMIN :  "Invite/Authorize"
    GROUP_ADMIN ||--o{ GROUP_ADMIN :  "Invite/Authorize "
    GROUP_ADMIN ||--o{ RESOURCE_ADMIN: "Authorize"
@enduml
```

Resources and Sub-resources

In DomiSMP, resources refer to documents that contain information about the services a participant offers. These are often in XML format but can also be in JSON, YAML, or other document types. The DomiSMP administrator assigns a resource to an end-user (the resource owner), who then updates and publishes their connectivity capabilities within this resource. Resources may contain subresources or documents to further organise connectivity capabilities.

Resource Locator Coordinates

In DomiSMP, finding and identifying specific resources or subresources is done through a set of unique identifiers known as resource locator coordinates. This method is inspired by Maven's locator, where identifiers like `groupId`, `artifactId`, `version`, and `packaging` help locate a specific resource, such as library, plugin or project. Similarly, DomiSMP uses these coordinates to pinpoint resources within its system.

The DomiSMP resource locator coordinates include the following:

- **DomainCode** (Optional): Identifier for the domain within DomiSMP. If not specified, DomiSMP uses the default domain.
- **ResourceType** (Optional): Identifies the kind of resource. If not defined, the default resource type is applied.
- **ResourceIdentifier** (Mandatory): Identifier of the resource. In most networks, it matches the Participant Identifier; for example `urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088:test1234`.
- **SubResourceType** (Mandatory if **SubResourceIdentifier** is defined): Subtype of the resource for more granular identification.
- **SubResourceIdentifier** (Optional): Identifier of the subresource. Further specifies the subresource, used in conjunction with **SubResourceType**.

To access a document within DomiSMP, an HTTP GET request is formed using a URL pattern that

incorporates these coordinates. The structure for this request is:

```
<DomiSMP-
URL>/[{DomainCode}/][{ResourceType}/]{ResourceIdentifier}[/{SubResourceType}/{SubResou
rceIdentifier}]
```

For scenarios where HTTP headers are preferable for specifying the domain and resource type, DomiSMP accommodates this through the `Domain` and `Resource-Type` headers. When used, these headers take precedence over URL path parameters, which in this case must not be used!

Example 1: Retrieving the resource

In this example, we demonstrate how to access a specific document from DomiSMP using the wget command. The document in question is an OASIS SMP 1.0 ServiceGroup document for a counterparty identified by **urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088:test1234** within the Invoice domain.

To access this document from DomiSMP, whose instance is located at the URL http://localhost:8080/, and assuming `invoice-domain` is the domain code for the Invoice domain and `smp-1` is the code for the OASIS SMP 1.0 ServiceGroup document type, use the following wget command:

```
wget http://localhost:8080/invoice-domain/smp-1/urn:oasis:names:tc:ebcore:partyid-
type:iso6523:0088:test1234
```

However, if `invoice-domain` is the default domain for the DomiSMP instance and OASIS SMP 1.0 is the default document type within this domain, you can simplify the command by omitting the domain and document type codes:

```
wget http://localhost:8080/urn:oasis:names:tc:ebcore:partyid-
type:iso6523:0088:test1234
```

This simplified command fetches the desired document based on the assumption that the default settings in DomiSMP match the required domain and document type for this retrieval.

Example 2: Retrieving the sub-resource

Let's say Party A wants to obtain a specific OASIS SMP 1.0 ServiceMetadata document for Party B. The document's identifier is `busdox-docid-qns::invoice-v01`, and Party B's identifier is `urn:oasis:names:tc:ebcore:partyid-type:iso6523:0088:test1234`. This operation occurs within the Invoice domain of the DomiSMP, accessible at the URL http://localhost:8080/. For this domain, the SMP instance code is `invoice-domain`, and the document type code for OASIS SMP 1.0 ServiceGroup documents is `smp-1`. The subresource code is `services`.

To retrieve the document, use the following command:

```
wget http://localhost:8080/invoice-domain/smp-1/urn:oasis:names:tc:ebcore:partyid-
type:iso6523:0088:test1234/services/busdox-docid-qns::invoice-v01
```

However, if `invoice-domain` and OASIS SMP 1.0 are the default domain and document type within your DomiSMP instance, you can shorten your request. The shorter version does not require specifying the domain and document type codes:

```
wget http://localhost:8080/urn:oasis:names:tc:ebcore:partyid-
type:iso6523:0088:test1234/services/busdox-docid-qns::invoice-v01
```

This simplified command retrieves the same OASIS SMP 1.0 ServiceMetadata document for the specified resource identifier.

Resolving location vector coordinates

To accurately find a resource or subresource, DomiSMP uses location vector coordinates. Here's how the process works:

DomiSMP begins by breaking down the URL into its path components. If the URL has more than five parameters, DomiSMP returns an error. Parameters are resolved one by one, to determine domain, resource type, etc. If any parameter cannot be resolved, the process halts and an error is thrown.

The process in presented in the diagram below:

```
@startuml

start
    :Split URI to path parameters;
if (More than\n 5 parts?) then (yes)
    label lbl_error
    #pink:error;
    stop
else (no)
    :Set first parameter;
    repeat :Resolve parameter; <<procedure>>

        if (Parameter resolved) then (yes)
        else (no)
            #pink:error;
            stop
        endif;
        backward:Set next parameter;
    repeat while (Has next parameter?) is (yes);
endif
:Path resolved,
return resource;
stop

@enduml
```

Resolving the DomainCode

Initially, DomiSMP attempts to determine the domain. If `Domain` is specified through an HTTP header, that value is used directly. If no domain code matches or is provided, DomiSMP checks the first URL parameter against registered domains. Failing a match, it defaults to a pre-configured domain in the DomiSMP settings. This process ensures that the resource lookup aligns with the correct domain context, whether specified explicitly or inferred by the system defaults.

Resolving the ResourceType

Identifying the resource type occurs after determining the domain. If the domain is specified in a path parameter, DomiSMP attempts to identify the resource type based on the next path parameter.

This is the order in which DomiSMP resolves the resource type:

1. Default setting: If the domain has only one registered resource type, this type is automatically assigned (legacy).
2. HTTP header: DomiSMP looks for a `Resource-Type` HTTP header. If the header specifies an invalid type, an error is generated.
3. URL path parameter: In this case, the path must contain at least two parameters.
4. Default resource type for domain: If no specific type is found, DomiSMP uses the default type configured for the domain.
5. First registered type: In the absence of a specified default, the first type registered for the domain is used.

Note: To accommodate simultaneous use of URL path parameters and HTTP headers, if the current path parameter matches the resource type and more than two path parameters remain, DomiSMP skips the current parameter and proceeds to resolve the next.

Resolving the ResourceIdentifier

The DomiSMP carefully analyses the current path parameter to accurately identify the resource linked with the domain and document type. Should the resource identifier not be located, DomiSMP will return an error. Conversely, if the identifier is successfully found and it represents the last parameter in the path, DomiSMP will directly return the resource document. However, if there are additional parameters, DomiSMP proceeds to determine the subresource by evaluating the last two parameters.

Resolving the subresource

When locating a subresource, DomiSMP examines the final two path parameters provided in the request URL. These parameters correspond to the type and identifier of the subresource.

If both the subresource type and the subresource identifier cannot be located, DomiSMP returns an error. Otherwise, if they are found, the process yields the corresponding subresource document.

Path options

DomiSMP provides flexible ways to locate resources and subresources through different HTTP path configurations. Depending on the information available and specific needs, you can use one of the following path options to access the desired document.

The path options accommodate various combinations of domain codes, resource types, resource

identifiers and subresource identifiers.

Please note that for options 2, 3 and 4, two variants of the path parameters are possible: if the first path variant does not give results, DomiSMP tries with the second path variant.

Here's a guide to using these path options.

Variants -SubResourceType mandatory

- **1 path parameter**: Use only the resource identifier in the URL path. If the domain or resource type isn't specified, DomiSMP will use default settings or those provided via HTTP headers.

```
{ResourceIdentifier}
```

- **2 path parameters**: Include either the domain code or resource type along with the resource identifier. Missing values default to HTTP headers or DomiSMP defaults.

```
/{DomainCode}/{ResourceIdentifier}
/{ResourceType}/{ResourceIdentifier}
```

- **3 path parameters**: Specify the domain code, resource type, and resource identifier. This configuration can also represent a resource identifier followed by subresource type and identifier if the domain and resource type are set by default or HTTP headers.

```
/{DomainCode}/{ResourceType}/{ResourceIdentifier}
/{ResourceIdentifier}/{SubResourceType}/{SubResourceIdentifier}
```

- **4 path parameters**: This option covers cases where one value (domain or resource type) is set by default or HTTP headers, and you're including resource and subresource identifiers.

```
/{DomainCode}/{ResourceIdentifier}/{SubResourceType}/{SubResourceIdentifier}
/{ResourceType}/{ResourceIdentifier}/{SubResourceType}/{SubResourceIdentifier}
```

- **5 path parameters**: Use this when specifying all identifiers explicitly, including domain code, resource type, resource and subresource identifiers.

```
/{DomainCode}/{ResourceType}/{ResourceIdentifier}/{SubResourceType}/{SubResourceIdenti
fier}
```

Path examples

This section illustrates how to construct URLs for accessing resources in DomiSMP, using OASIS SMP 1.0 and OASIS SMP 2.0 documents as examples.

Note that your DomiSMP setup should have a default domain and resource type that match your intended document type (either OASIS SMP 1.0 or OASIS SMP 2.0) for these examples to apply directly.

▼ *OASIS SMP 1.0 documents*

The path is built in the following way:

```
    {ResourceIdentifier}[/services/{SubResourceIdentifier}]
```

You can retrieve ServiceGroup documents by specifying the resource identifier directly in the URL. If the domain and resource type match the default settings, you don't need to specify them in your request.

Example for a ServiceGroup document:

```
    /iso6523-actorid-upis::0088:test1234
```

And if you're specifying a domain and resource type explicitly:

```
    /invoice-domain/smp-1/iso6523-actorid-upis::0088:test1234
```

To access a ServiceMetadata document, include the subresource type and identifier in the URL. **busdox-docid-qns::invoice-v01** is our resource identifier. Like with ServiceGroup documents, you can omit domain and resource type if they are the default.

Example for a ServiceMetadata document:

```
    /iso6523-actorid-upis::0088:test1234/services/busdox-docid-qns::invoice-v01
```

With explicit domain and resource type:

```
    /invoice-domain/smp-1/iso6523-actorid-upis::0088:test1234/services/busdox-docid-
qns::invoice-v01
```

▼ *OASIS SMP 2.0*

The path is similar:

```
    oasis-bdxr-smp-2/{ResourceIdentifier}/{services}/{SubResourceIdentifier}
```

Example for accessing an OASIS SMP 2.0 ServiceGroup document with the resource identifier **iso6523-actorid-upis::0088:test1234**:

```
/oasis-bdxr-smp-2/iso6523-actorid-upis::0088:test1234
```

And with a specified domain and resource type:

```
/invoice-domain/oasis-bdxr-smp-2/iso6523-actorid-upis::0088:test1234
```

Example for accessing an OASIS SMP 2.0 ServiceMetadata document with **busdox-docid-qns::invoice-v01** as the subresource identifier:

```
/oasis-bdxr-smp-2/iso6523-actorid-upis::0088:test1234/services/busdox-docid-
qns::invoice-v01
```

The previous example included already the resource type. Here is an example with a defined domain:

```
/invoice-domain/oasis-bdxr-smp-2/iso6523-actorid-
upis::0088:test1234/services/busdox-docid-qns::invoice-v01
```

Let's break this path down:

- `invoice-domain` is the domain.
- `oasis-bdxr-smp-2` is the resource type.
- `iso6523-actorid-upis::0088:test1234` is the resource identifier.
- `services` is the subresource type.
- `busdox-docid-qns::invoice-v01` is the subresource identifier.

▼ *OASIS CPPA3 CPP*

For OASIS CPPA3 CPP documents, specify the custom domain and document type as part of your query path.

```
/custom-domain/cpp/iso6523-actorid-upis::0088:test1234
```

6.5. Guide: User Interface Overview

DomiSMP combines a document repository with a straightforward text editor. The UI is divided into a tool sidebar on the left and a main workspace on the right, designed for simplicity and ease of use.

▼ *About this guide*

Purpose of this guide

This document is designed for new users of DomiSMP, providing a clear overview of how to

navigate the user interface (UI) and manage settings. We recommend reading through this guide from start to finish to gain a comprehensive understanding. If you're already familiar with some parts, feel free to jump to the sections that interest you.

What you will learn

- Navigate the DomiSMP user interface.
- Manage your user settings.

What you will need

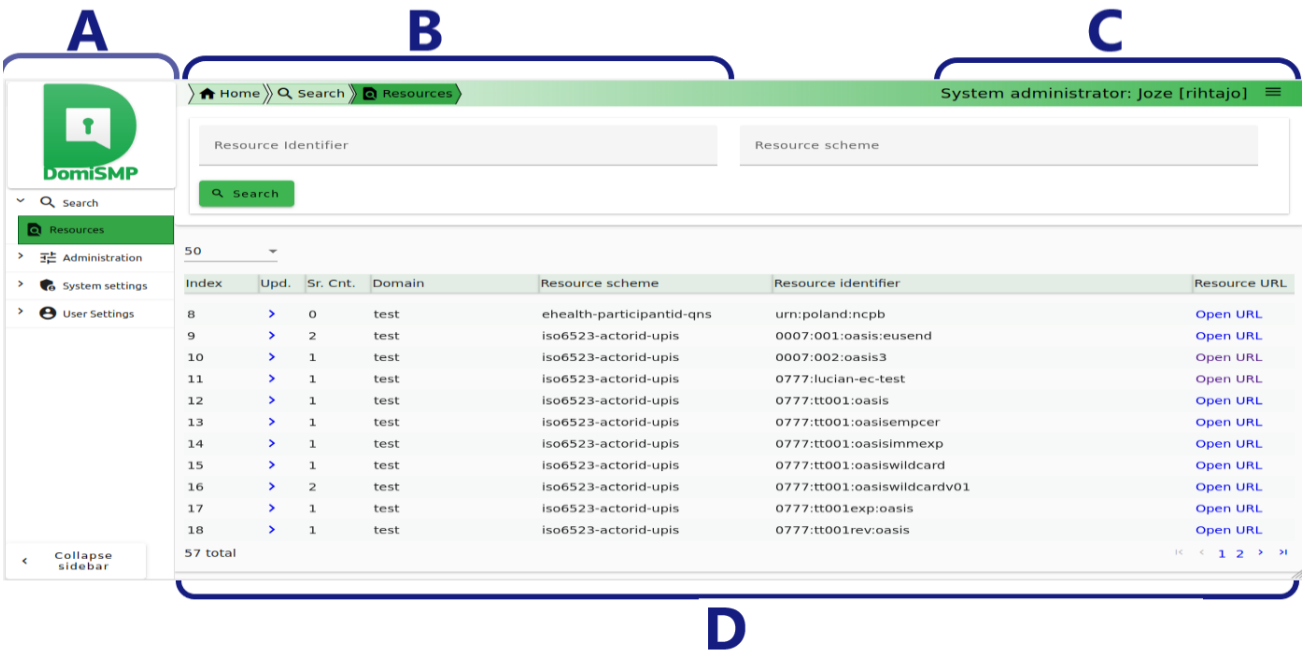
- Approximately 5-10 minutes of your time.

UI Overview

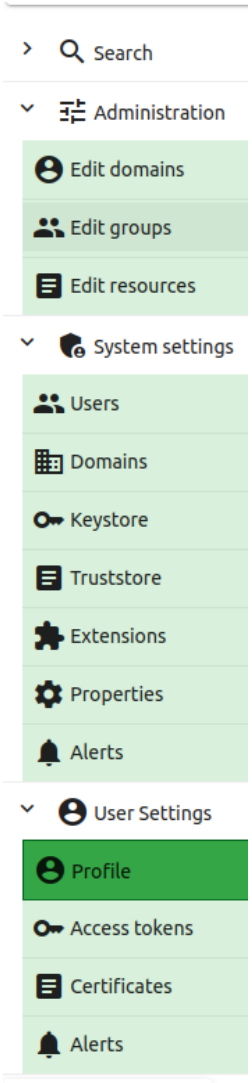
▼ Layout

DomiSMP features a simple layout:

- **Collapsible sidebar (A)** that contains tools and views.
- **Breadcrumb bar (B)** that displays your current location in the application.
- **Top bar (C)** with login status and menu.
- **Main workspace (D)**, where you edit documents (resources/sub-resources) and configure entities.



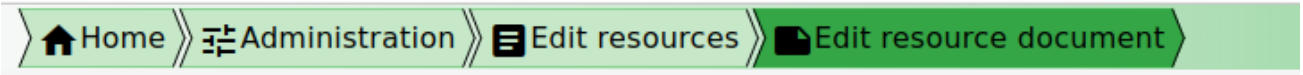
▼ Sidebar



The sidebar is your primary navigation tool, offering:

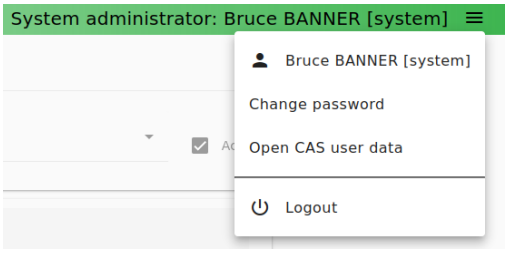
- **Search**: Find resources within DomiSMP. The tool is visible to all users.
- **Administration**: Manage domains, groups, and resources (for logged-in users).
- **System Settings**: Accessible by system admins for DomiSMP configuration.
- **User Settings**: Personalise the look and feel, as well as modify your UI preferences and credentials. It's visible only to logged-in users.

▼ Breadcrumb



The breadcrumb at the top helps you navigate your current location, making it easier to move between resources.

▼ Top bar



This bar displays your login status and includes a menu for accessing user settings, changing your password, and logging out.

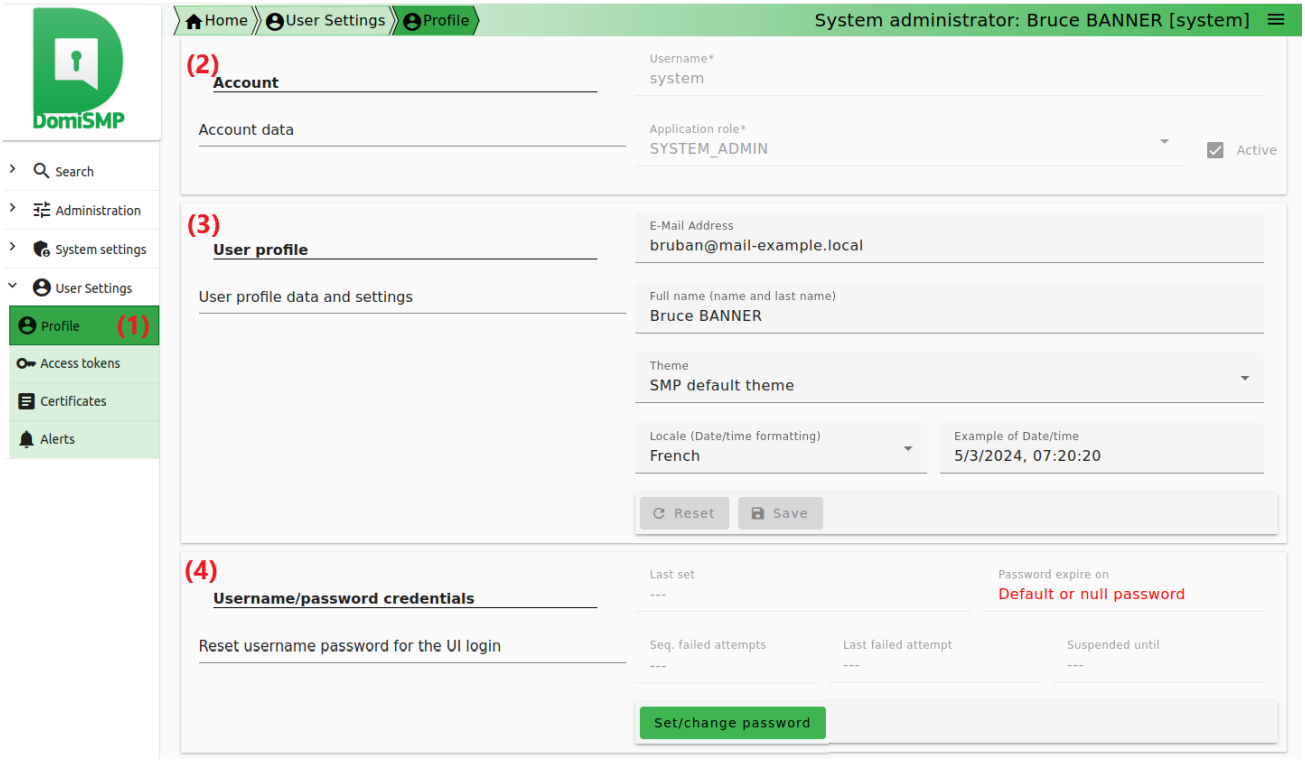
▼ Main Workspace

The workspace changes based on the selected tool. In this guide, we will focus on the user settings displayed in the main workspace.

▼ User Settings

User settings allow you to personalise your DomiSMP experience, including your name, email, theme, and credentials.

For user preferences, click the hamburger menu on the top bar and select **User**, or click **User Settings** on the left sidebar (1 in the image below).

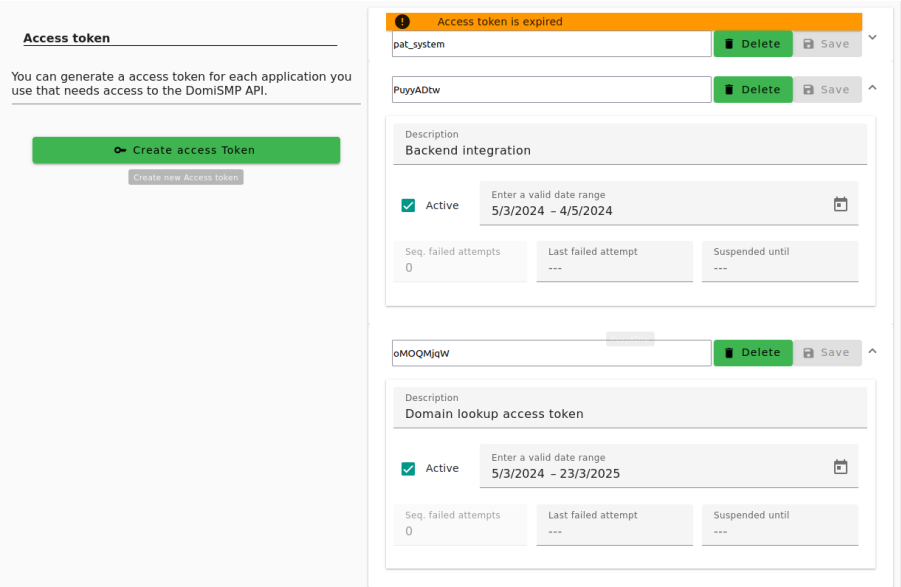


▼ Profile

In the **Profile** section, you can:

- View **Account** (2) info: See your username and role. Only System Admins can modify this information.
- Customise your **User profile** (3): Change your name, email, theme (choose between the light themes: SMP Default, Blue, Indigo & Pink theme, and dark themes: Pink & Blue grey and Purple & Green theme), and locale. Confirm changes by clicking the the **Save** button.
- Change your **Username/password credentials** (4): This section allows you to change the password.

▼ Access Tokens



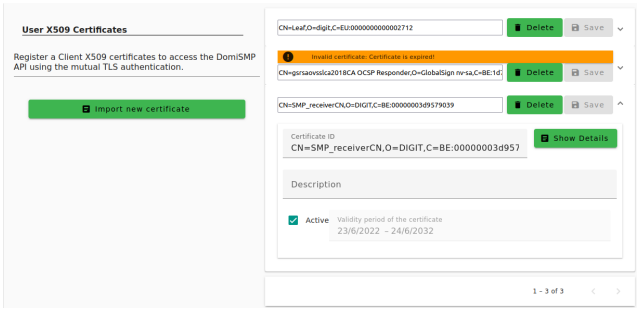
Use access tokens for REST API access. Create, edit, or delete tokens.

To generate an access token, click **Create access Token**. Note that the full access token is only shown once and it's not possible to retrieve it again.

The description and validity period are editable; to confirm changes, click the **Save** button. If an access token is expired, it can not be used for accessing the REST API and you see a warning bar over its name.

To delete an access token, click the **Delete** button.

▼ Certificates



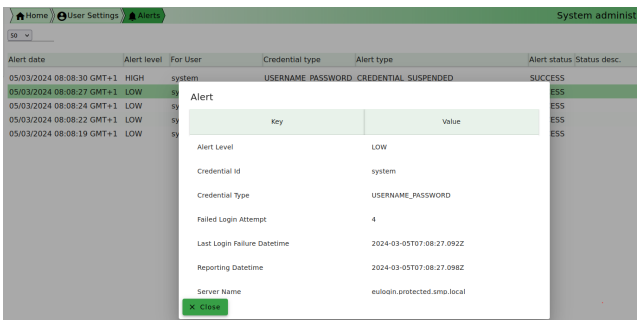
X.509 certificates are used for TLS authentication and authorisation for REST API access.

1. To upload a certificate, click **Import new certificate**.
   Update description if you wish.
2. Click the **Save** button to persist changes.

To see certificate details, expand the certificate tab and click the **Show Details** button.
If a certificate is expired, it can not be used for accessing the REST API and you see warning bar over its name.

To delete a certificate, click the **Delete** button.

▼ Alerts



- The alert table lets you view alerts that may not have reached you via email, providing another way to stay informed.
- Double-click on an alert row to see details.

Chapter 7. Reference Guides

SMP Properties Reference

Mandatory Properties

▼ **SMP Configuration Properties**

Configuration Property	Description and Usage	Default
`smp.configuration.file`	*Configuration property file path.*	`smp.config.properties`
`smp.init.configuration.file`	*Init configuration property file path.*	`smp.init.properties`
`smp.security.folder`	*Security folder for storing the keystore and the truststore.*	`smp`
`smp.jdbc.driver`	*Database Configuration - Driver* • MySQL: `com.mysql.jdbc.Driver` • Oracle: `oracle.jdbc.OracleDriver`	`com.mysql.jdbc.Driver`
`smp.jdbc.url`	*Database Configuration - URL* • MySQL: `jdbc:mysql://dbhost:dbport/smp_database` • Oracle: ◦ `jdbc:oracle:thin:@dbhost:dbport:smp_database` ◦ `jdbc:oracle:thin:@dbhost:dbport/smp_service`	`jdbc:mysql://localhost:3306/smp`
`smp.jdbc.user`	*Database User/Password Configuration - User*	`smp`
`smp.jdbc.password`	*Database User/password Configuration - Password*	`The_password`
`smp.datasource.jndi`	*If the data source is configured on the application server (*recommended), the property defines the JNDI name of the database connection.*	`jdbc/eDeliverySmpDs`
`smp.database.show-sql`	*Print generated sql queries to logs. The property is effective only when* `smp.mode.development=true`.	`false`

Configuration Property	Description and Usage	Default
`smp.database.create-ddl`	*Auto create/update database objects. The property is effective only when* `smp.mode.development=true`.	`false`
`smp.log.folder`	**IMPORTANT** Do NOT this feature in production, it is only intended for tests, demonstrations and development purposes. *The provided logback.xml configuration defines logging file as* `<file>${log.folder:-logs}/edelivery-smp.log</file>` *With the property we can define the folder for the logging files.*	`/var/logs/smp`
`smp.log.configuration.file`	*Custom logback configuration file (filepath can be absolute or relative to smp configuration.dir).*	`/opt/logging/smp-logback.xml`
`smp.libraries.folder`	*Path where SMP extensions are located. The folder is loaded by the SMP classloader at startup.*	`/opt/smp/extension-libs`
`smp.smp.mode.development`	*The development mode uses semi-random generators for password and key generation. Setting the property value to 'true' makes the first startup and access token generation faster. To ensure high security, this option MUST NOT be enabled in production.*	`false`

Application Properties

▼ **SMP Application Configuration Properties**

Configuration Property	Description and Usage	Default
`smp.instance.name`	*The name of the DomiSMP instance is used in email notifications and alerts to specify which SMP instance generated the notifications.* **Usage**: Requires restart: No Value type: STRING	`Test DomiSMP Instance`
`contextPath.output`	*This property controls pattern of URLs produced by SMP in GET ServiceGroup responses.* **Usage**: Requires restart: Yes Value type: BOOLEAN	`true`
`encodedSlashesAllowedInUrl`	*Allow encoded slashes in context path. Set to true if slashes are part of identifiers.* **Usage**: Requires restart: Yes Value type: BOOLEAN	`true`
`smp.http.forwarded.headers.enabled`	*Use (value true) or remove (value false) forwarded headers. There are security considerations for forwarded headers since an application cannot know if the headers were added by a proxy, as intended, or by a malicious client.* **Usage**: Requires restart: No Value type: BOOLEAN	`false`
`smp.http.httpStrictTransportSecurity.maxAge`	*How long (in seconds) should HSTS last in the browser cache (default one year).* **Usage**: Requires restart: Yes Value type: INTEGER	`31536000`

Configuration Property	Description and Usage	Default

smp.http.header.security.policy	*Content Security Policy (CSP)*
	
	`default-src 'self';`
	`script-src 'self';`
	`connect-src 'self';`
	`img-src 'self';`
	`style-src 'self'`
	`'unsafe-inline';`
	`frame-ancestors 'self';`
	`form-action 'self';`
	
	Usage:
	Requires restart: Yes
	Value type: STRING
		-

Configuration Property	Description and Usage	Default
smp.proxy.host	*The http proxy host.* **Usage**: Requires restart: No Value type: STRING	-
smp.noproxy.hosts	*List of nor proxy hosts.* **Usage**: Requires restart: No Value type: STRING **Default**: `localhost|127.0.0.1`	See Description
smp.proxy.password	*Base64 encrypted password for Proxy.* **Usage**: Requires restart: No Value type: STRING	-
smp.proxy.port	*The http proxy port.* **Usage**: Requires restart: No Value type: INTEGER	80
smp.proxy.user	*The proxy user.* **Usage**: Requires restart: No Value type: STRING	-

Configuration Property	Description and Usage	Default

Configuration Property	Description and Usage	Default
identifiersBehaviour.ParticipantIdentifierScheme.validationRegex	*Participant Identifier Schema of each PUT ServiceGroup request is validated against this schema.* **Usage**: Requires restart: No Value type: REGEXP **Default**: `$\|(?!^.{26})(-[a-z0-9]-)$|^urn:oasis:names:tc:ebcore:partyid-type:(iso6523|unregistered)(:.)?$`	See Description
identifiersBehaviour.ParticipantIdentifierScheme.validationRegexMessage	*Error message for UI.* **Usage**: Requires restart: No Value type: STRING **Value Format**: Participant scheme must start with `urn:oasis:names:tc:ebcore:partyid-type:(iso6523:|unregistered:)` *OR* • must be up to 25 characters long with form `[domain]-[identifierArea]-[identifierType]` • and may only contain the following characters: [a-z0-9]. **Example**: `busdox-actorid-upis`	-
identifiersBehaviour.scheme.mandatory	*Scheme for participant identifier is mandatory.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
Configuration Property	**Description and Usage**	**Default**
smp.ui.session.idle_timeout.admin	*Specifies the time, in seconds, between client requests before the SMP will invalidate session for ADMIN users (System).* **Usage**: Requires restart: No Value type: INTEGER	`300`

Configuration Property	Description and Usage	Default
smp.ui.session.idle_timeout.user	*Specifies the time, in seconds, between client requests before the SMP will invalidate session for users (Service group, SMP Admin).* **Usage**: Requires restart: No Value type: INTEGER	1800
smp.cluster.enabled	*Define if application is set in cluster. In not cluster environment, properties are updated on set Property.* **Usage**: Requires restart: No Value type: BOOLEAN	false
Configuration Property	**Description and Usage**	**Default**
smp.credentials.reset_request.url	*The URL address for resetting the password in DomiSMP. This URL can be used when DomiSMP is behind a reverse proxy. If the property value is not set, the reset URL is created using reverse proxy headers such as Host and X-Forwarded-Host.* **Usage**: Requires restart: No Value type: URL	-
smp.credentials.reset_request.url.validMinutes	*The number of minutes for reset token to be valid.* **Usage**: Requires restart: No Value type: INTEGER	90
smp.passwordPolicy.validationRegex	*Password minimum complexity rules.* **Usage**: Requires restart: No Value type: REGEXP **Default**: `^(?=.*[0-9])(?=.*[a-z])(?=.*[A-Z])(?=.*[~`!@#$%^&+=\-_<>.,?:;*/()\|\[\]\{}'"\\]).\{16,32}$`	See Description

Configuration Property	Description and Usage	Default
`smp.passwordPolicy.validationMessage`	*The error message shown to the user in case the password does not follow the regex put in the* `smp.passwordPolicy.pattern property`. **Usage**: Requires restart: No Value type: STRING Must have: • Minimum length: 16 characters. • Maximum length: 32 characters. • At least one letter in lowercase; • At least one letter in uppercase; • At least one digit; • At least one special character.	-
`smp.passwordPolicy.validDays`	*Number of days password is valid.* **Usage**: Requires restart: No Value type: INTEGER	`90`
`smp.passwordPolicy.warning.beforeExpiration`	*How many days before expiration should the UI warn users at login.* **Usage**: Requires restart: No Value type: INTEGER	`15`
`smp.passwordPolicy.expired.forceChange`	*Force change password at UI login if expired.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
Configuration Property	**Description and Usage**	**Default**
`smp.user.login.fail.delay`	*Delay response in ms on invalid username or password.* **Usage**: Requires restart: No Value type: INTEGER	1000

Configuration Property	Description and Usage	Default
smp.user.login.maximum.attempt	*Number of console login attempt before the user is deactivated.* **Usage**: Requires restart: No Value type: INTEGER	5
smp.user.login.suspension.time	*Time in seconds for a suspended user to be reactivated.* **Usage**: Requires restart: No Value type: INTEGER • If set to 0, the user will not be reactivated.	3600

Configuration Property	Description and Usage	Default
smp.accessToken.validDays	*Number of days access token is valid.* **Usage**: Requires restart: No Value type: INTEGER	60
smp.accessToken.login.maximum.attempt	*Number of accessToken login attempt before the accessToken is deactivated.* **Usage**: Requires restart: No Value type: INTEGER	10
smp.accessToken.login.suspension.time	*Time in seconds for a suspended accessToken to be reactivated.* **Usage**: Requires restart: No Value type: INTEGER • If set to 0, the user will not be reactivated.	3600
smp.accessToken.login.fail.delay	*Delay in ms on invalid token id or token.* **Usage**: Requires restart: No Value type: INTEGER	1000
smp.ui.authentication.types	*Set list of '|' separated authentication types: PASSWORD|SSO.* **Usage**: Requires restart: No Value type: LIST_STRING	PASSWORD

Configuration Property	Description and Usage	Default
smp.automation.authentication.types	*Set list of "|" separated application-automation authentication types (Web-Service integration).* **Usage**: Requires restart: No Value type: LIST_STRING Supported Values: TOKEN, CERTIFICATE. **Default**: `TOKEN|CERTIFICATE`	See Description
smp.automation.authentication.external.tls.clientCert.enabled	*Authentication with external module as: reverse proxy. Authenticated data are sent to application using 'Client-Cert' HTTP header. Do not enable this feature without a properly configured reverse-proxy.* **Usage**: Requires restart: No Value type: BOOLEAN	false
smp.automation.authentication.external.tls.SSLClientCert.enabled	*Authentication with external module as: reverse proxy. Authenticated certificate is sent to application using SSLClientCert HTTP header. Do not enable this feature without properly a configured reverse-proxy.* **Usage**: Requires restart: No Value type: BOOLEAN	false

Configuration Property	Description and Usage	Default
smp.sso.cas.ui.label	*The SSO service provider label.* **Usage**: Requires restart: Yes Value type: STRING	EU Login

Configuration Property	Description and Usage	Default
`smp.sso.cas.url`	*The SSO CAS URL endpoint.* **Usage**: Requires restart: Yes Value type: URL **Default**: `http://localhost:8080/cas/`	See Description
`smp.sso.cas.urlPath.login`	*The CAS URL path for login.* *The complete URL is composed by parameters:* `${smp.sso.cas.url}/${smp.sso.cas.urlpath.login}.` **Usage**: Requires restart: Yes Value type: STRING	`login`
`smp.sso.cas.callback.url`	*The URL is the callback URL belonging to the local SMP Security System. If using RP,make sure it target SMP path* `/ui/public/rest/security/cas`. **Usage**: Requires restart: Yes Value type: URL **Default**: `http://localhost:8080/smp/ui/public/rest/security/cas`	See Description
`smp.sso.cas.smp.urlPath`	*SMP relative path which triggers CAS authentication.* **Usage**: Requires restart: Yes Value type: STRING **Default**: `/smp/ui/public/rest/security/cas`	See Description

Configuration Property	Description and Usage	Default

`smp.sso.cas.smp.user.data.urlPaths`	*Relative path for CAS user data. Complete URL is composed by parameters:*
	
	`${smp.sso.cas.url}/${smp.sso.cas.smp.user.data.urlpath}`
	
	Usage:
	Requires restart: Yes
	Value type: STRING
	
	Default:
	`userdata/myAccount.cgi`
		See Description

`smp.sso.cas.token.validation.urlPath`	The CAS URL path for login. Complete URL is composed of parameters: `${smp.sso.cas.url}/${smp.sso.cas.token.validation.urlpath}` **Usage**: Requires restart: Yes Value type: STRING **Default**: `laxValidate`	See Description
`smp.sso.cas.token.validation.params`	*The CAS token validation key:value properties separated with a pipe (`|`).* **Usage**: Requires restart: Yes Value type: MAP_STRING **Default**: `acceptStrengths:BASIC,CLIENT_CERT|assuranceLevel:TOP`	See Description

Configuration Property	Description and Usage	Default
smp.sso.cas.token.validation.groups	*Pipe-separated (|) CAS groups user must belong to.* **Usage**: Requires restart: Yes Value type: LIST_STRING **Default**: `DIGIT_SMP|DIGIT_ADMIN`	See Description
smp.sso.cas.registration.enabled	*If the value is set to true, the user is automatically registered to DomiSMP the first time they use the external CAS. The CAS server provides the necessary user data, which is then mapped to the DomiSMP user entity according to the smp.sso.cas.registration.mapping.* **Usage**: Requires restart: No Value type: BOOLEAN	true
smp.sso.cas.registration.confirmation.mandatory	*The value determines whether the CAS-automatically created user is activated immediately or if the System admin must activate the user before they can log in to DomiSMP.* **Usage**: Requires restart: No Value type: BOOLEAN	false
smp.sso.cas.registration.mapping	*Pipe-separated (|) key:value list of mapping defining how CAS user data is mapped to DomiSMP user entity. Currently supported values are: EMAIL and FULL_NAME. The username of the newly created user is the CAS principal name/identifier* **Usage**: Requires restart: No Value type: MAP_STRING **Default**: `EMAIL:${email}\|FULL_NAME:${firstName} ${lastName}`	See Description

Configuration Property	Description and Usage	Default
Configuration Property	Description and Usage	Default
`mail.smtp.host`	Email configuration: *Email server.* **Usage**: Requires restart: No Value type: STRING	-
`mail.smtp.port`	Email configuration: *SMTP mail port.* **Usage**: Requires restart: No Value type: INTEGER	`25`
`mail.smtp.protocol`	Email configuration: *SMTP mail protocol.* **Usage**: Requires restart: No Value type: STRING	smtp
`mail.smtp.username`	Email configuration: *SMTP mail protocol; mail sender's username.* **Usage**: Requires restart: No Value type: STRING	-
`mail.smtp.password`	Email configuration: _SMTP mail protocol; mail sender's encrypted password. **Usage**: Requires restart: No Value type: STRING	-
`mail.smtp.properties`	*Pipe-separated (`|`) key:value properties list.* **Usage**: Requires restart: No Value type: MAP_STRING **Example**: `mail.smtp.auth:true|mail.smtp.starttls.enable:true|mail.smtp.quitwait:false`	-
Configuration Property	Description and Usage	Default

Configuration Property	Description and Usage	Default
`smp.alert.user.created.enabled`	*Enable or disable notifications for user creation events.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.user.created.level`	*User creation event notification alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`HIGH`
`smp.alert.user.updated.enabled`	*Enable or disable notifications when user data is changed.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.user.updated.level`	*User update data event notification alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`HIGH`
`smp.alert.user.login_failure.enabled`	*Enable/disable the login failure alert of the authentication module.* **Usage**: Requires restart: No Value type: BOOLEAN	`false`
`smp.alert.user.login_failure.level`	*Login failure alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`LOW`
`smp.alert.user.suspended.enabled`	*Enable/disable the login suspended alert of the authentication module.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`

Configuration Property	Description and Usage	Default
`smp.alert.user.suspended.level`	*Suspended login alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`HIGH`
`smp.alert.user.suspended.mail.moment`	*When should the account disabled alert be triggered.* **Usage**: Requires restart: No Value type: STRING Possible values: • `AT_LOGON`: if set, an alert is triggered each time a user tries to log into a disabled account. • `WHEN_BLOCKED`: if set, an alert is triggered when the account is suspended.	`WHEN_BLOCKED`

Configuration Property	Description and Usage	Default
`smp.alert.password.imminent_expiration.enabled`	*Enable/disable the "Password about to expire" alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.password.imminent_expiration.delay_days`	*Number of days before password expiration the system is to send alerts.* **Usage**: Requires restart: No Value type: INTEGER	`15`
`smp.alert.password.imminent_expiration.frequency_days`	*Frequency in days for (re)sending the "Password about to expire" alert.* **Usage**: Requires restart: No Value type: INTEGER	`5`
`smp.alert.password.imminent_expiration.level`	*"Password about to expire" alert's level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`LOW`

Configuration Property	Description and Usage	Default
Configuration Property	**Description and Usage**	**Default**
`smp.alert.password.expired.enabled`	*Enable/disable the "Password expired" alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.password.expired.delay_days`	*Period in days after password expiration the system is to send "password expiration" alerts.* **Usage**: Requires restart: No Value type: INTEGER	`30`
`smp.alert.password.expired.frequency_days`	*Frequency in days between "Password expired" alerts.* **Usage**: Requires restart: No Value type: INTEGER	`5`
`smp.alert.password.expired.level`	*"Password expired" alert's level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH.	`LOW`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.accessToken.imminent_expiration.enabled`	*Enable/disable the "accessToken about to expire" alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.accessToken.imminent_expiration.delay_days`	*Number of days before password expiration the system is to send alerts.* **Usage**: Requires restart: No Value type: INTEGER	15
`smp.alert.accessToken.imminent_expiration.frequency_days`	*Frequency in days between alerts.* **Usage**: Requires restart: No Value type: INTEGER	`5`

Configuration Property	Description and Usage	Default
`smp.alert.accessToken.imminent_expiration.level`	*AccessToken imminent expiration alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH.	`LOW`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.accessToken.expired.enabled`	*Enable/disable the accessToken expiration alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.accessToken.expired.delay_days`	*Number of days after expiration as for how long the system should send alerts.* **Usage**: Requires restart: No Value type: INTEGER	`30`
`smp.alert.accessToken.expired.frequency_days`	*Frequency in days between alerts.* **Usage**: Requires restart: No Value type: INTEGER	`30`
`smp.alert.accessToken.expired.level`	*Access Token expiration alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH.	`LOW`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.certificate.imminent_expiration.enabled`	*Enable/disable the imminent certificate expiration alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.certificate.imminent_expiration.delay_days`	*Number of days before expiration as for how long before expiration the system should send alerts.* **Usage**: Requires restart: No Value type: INTEGER	`15`

Configuration Property	Description and Usage	Default
`smp.alert.certificate.imminent_expiration.frequency_days`	*Frequency in days between alerts.* **Usage**: Requires restart: No Value type: INTEGER	`5`
`smp.alert.certificate.imminent_expiration.level`	*Certificate imminent expiration alert level. Values: \{LOW, MEDIUM, HIGH}* **Usage**: Requires restart: No Value type: STRING	`LOW`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.certificate.expired.enabled`	*Enable/disable the certificate expiration alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.certificate.expired.delay_days`	*Number of days after expiration as for how long the system should send alerts.* **Usage**: Requires restart: No Value type: INTEGER	`30`
`smp.alert.certificate.expired.frequency_days`	*Frequency in days between alerts.* **Usage**: Requires restart: No Value type: INTEGER	`5`
`smp.alert.certificate.expired.level`	*Certificate expiration alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`LOW`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.system.certificate.imminent_expiration.enabled`	*Enable/disable the imminent system certificate expiration alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`

Configuration Property	Description and Usage	Default
`smp.alert.system.certificate.imminent_expiration.delay_days`	*Number of days the system is to send alerts before expiration occurs.* **Usage**: Requires restart: No Value type: INTEGER	`15`
`smp.alert.system.certificate.imminent_expiration.frequency_days`	*Period in days between alerts.* **Usage**: Requires restart: No Value type: INTEGER	`5`
`smp.alert.system.certificate.imminent_expiration.level`	*System certificate imminent expiration alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`LOW`
Configuration Property	**Description and Usage**	**Default**
`smp.alert.system.certificate.expired.enabled`	*Enable/disable the system certificate expiration alert.* **Usage**: Requires restart: No Value type: BOOLEAN	`true`
`smp.alert.system.certificate.expired.delay_days`	*Number of days the system is to send alerts after expiration occurs. _* **Usage**: Requires restart: No Value type: INTEGER	`30`
`smp.alert.system.certificate.expired.frequency_days`	*Frequency in days between alerts.* **Usage**: Requires restart: No Value type: INTEGER	`5`
`smp.alert.system.certificate.expired.level`	*System certificate expiration alert level.* **Usage**: Requires restart: No Value type: STRING Possible values: LOW, MEDIUM, HIGH	`LOW`
Configuration Property	**Description and Usage**	**Default**

Configuration Property	Description and Usage	Default
`smp.alert.credentials.cronJobExpression`	*CRON expression specifying schedule for triggering alert messages about credentials.* **Usage**: Requires restart: No Value type: CRON_EXPRESSION **Default**: `0 52 4 */1 * *`	See Description
`smp.alert.system.certificates.cronJobExpression`	*CRON expression specifying schedule for triggering alert messages about system certificates.* **Usage**: Requires restart: No Value type: CRON_EXPRESSION **Default**: `0 42 4 */1 * *`	See Description
`smp.alert.credentials.serverInstance`	*Which instance (hostname) to generates a report when* `smp.cluster.enabled` *is set to true.* **Usage**: Requires restart: No Value type: STRING	`localhost`
`smp.alert.credentials.batch.size`	Max alerts generated in a batch for the type. **Usage**: Requires restart: No Value type: INTEGER	`200`

Configuration Property	Description and Usage	Default
`smp.alert.mail.from`	*Alert send mail.* **Usage**: Requires restart: No Value type: EMAIL **Default**: `test@alert-send-mail.eu`	See Description

Configuration Property	Description and Usage	Default
smp.domain.default	*Default domain code.* **Usage**: Requires restart: No Value type: STRING • If the domain cannot be determined from the request, the default domain is used.	-

Configuration Property	Description and Usage	Default
smp.certificate.validation.allowed.certificate.types	*Allowed user certificate types.* **Usage**: Requires restart: No Value type: LIST_STRING **Example**: `RSA|EC|Ed25519|Ed448` • If empty no restrictions are imposed. • For other values see the java KeyFactory Algorithms.	-
authentication.blueCoat.enabled	**NOTE** Property was replaced by property: smp.automation.authentication.external.tls.clientCert.enabled **Usage**: Requires restart: No Value type: BOOLEAN	false
smp.domain.default	*Default domain code. If the domain cannot be determined from the request, the default domain is used.* **Usage**: Requires restart: No Value type: STRING	-

Configuration Property	Description and Usage	Default
`smp.certificate.validation.allowed.certificate.types`	*Allowed user certificate types.* **Usage**: Requires restart: No Value type: LIST_STRING **Example**: `RSA|EC|Ed25519|Ed448` • If empty no restrictions are imposed. • For other values see the java KeyFactory Algorithms.	-
`smp.authorization.jwt.issuer`	*JWT issuer used for validating the token. The "issuer" in a JWT is the principal which issued the token. It's represented by the registered claim name "iss" and is used by token consumers to verify who created the token.* **Usage**: Requires restart: Yes Value type: STRING **Example**: `https://auth.example.com` • If empty no restrictions are imposed.	-
`smp.authorization.jwt.audience`	*JWT audience used for validating the token. The JWT "audience" (claim name "aud") identifies the intended recipient(s) of the token, i.e., who the token is meant for.* **Usage**: Requires restart: Yes Value type: STRING **Example**: `my-api-v1` • If empty no restrictions are imposed.	-

Configuration Property	Description and Usage	Default
smp.authorization.jwt.key	*JWT public key used to verify the signature of the JWT token.* **Usage**: Requires restart: Yes Value type (Base 64 encoded key): STRING **Example**: `my-api-v1`	-
smp.authorization.jwt.algorithm	*Expected JWT algorithm used in a signature of the JWT token. Default is PS256* Supported algorithms: **RS256** — RSASSA-PKCS1-v1_5 using SHA-256 **RS384** — RSASSA-PKCS1-v1_5 using SHA-384 **RS512** — RSASSA-PKCS1-v1_5 using SHA-512 **PS256** — RSASSA-PSS using SHA-256 **PS384** — RSASSA-PSS using SHA-384 **PS512** — RSASSA-PSS using SHA-512 **ES256** — ECDSA using curve P-256 and SHA-256 **ES384** — ECDSA using curve P-384 and SHA-384 **ES512** — ECDSA using curve P-521 and SHA-512 **EdDSA** — EDDSA using curve Ed25519 [1] [1] ED448 is not supported. **Usage**: Requires restart: Yes Value type: STRING **Example**: `PS256`	-

Configuration Property	Description and Usage	Default
`vault.enabled`	*Enable/Disable Vault integration for sensitive properties* **Usage**: Requires restart: No Value type: BOOLEAN **Example**: `true`	-
`smp.vault.implementation.classname`	*Full class name of the Vault implementation-* **Usage**: Requires restart: No Value type: STRING **Example**: `eu.europa.ec.edelivery.vault.MyVault`	-
`smp.vault.configuration`	*List of vault properties separated by `;`.* **Usage**: Requires restart: No Value type: STRING **Example**: `hashicorp-vault.url:http://vault-service:8200/;hashicorp-vault.token:domisml-valut-test-token`	-

Configuration Property	Description and Usage	Default
`smp.vault.configuration`	*List of vault properties, separated by semicolons (`;`). Key names vary depending on the vault instance type. For a complete list of supported properties, refer to the specific vault's implementation.* **Usage**: Requires restart: No Value type: STRING **Example**: `hashicorp-vault.url:http://vault-service:8200/;hashicorp-vault.token:domisml-valut-test-token`	-
`smp.vault.authentication.type`	*Specifies the authentication type, such as token or username. The accepted values depend on the vault implementation.* **Usage**: Requires restart: No Value type: STRING **Example**: `token`	-
`smp.vault.authentication.value`	*Authentication value and format depends on the vault implementation and authentication type.* **Usage**: Requires restart: No Value type: STRING **Example**: `my-valult-token`	-

Other References

Errors

Services Error Codes Table

The following table summarizes all possible errors returned by SML services:

▼ **Services Error Codes** *table*

		Applicable UC/ Request Method						
HTTP Code/ HTTP Message	**Business Code**/ *Error Code Description*	**UC01** *N/A*	**UC02** PUT	**UC03** DEL	**UC04** PUT	**UC05** DEL	**UC06** GET	**UC07** GET
200 OK	*N/A* *The request was completed successfully.*	-	x	x	x	x	x	x
201 Created	*N/A* *The PUT operation completed successfully.*	-	-	x	x	-	-	-
400 Bad Request	XSD_INVALID *The XML included in the request is not* validate against the XSD defining the input structure.	-	x	-	x	-	-	-
400 Bad Request	MISSING_FIELD *Some field that is optional in the XSD but mandatory for this invocation is missing* (missing field name in description).	-	x	-	x	-	-	-
400 Bad Request	WRONG_FIELD *Some field is valid against XSD definition, but the more specific content is invalid* (erroneous field name in description) or some header field is either missing or invalid.	-	x	-	x	-	-	-
400 Bad Request	OUT_OF_RANGE *Some numeric (or date field) is out of the valid range (erroneous field name in description).*	-	-	-	x	-	-	-
400 Bad Request	UNAUTHOR_FIELD *Some field that is optional in the XSD but forbidden for this invocation is present (unauthorized field name in description).*	-	-	-	x	-	-	-
400 Bad Request	FORMAT_ERROR *Some field is expected to have a specific format is not valid (erroneous field name in description).*	-	x	x	x	-	-	-
400 Bad Request	USER_NOT_FOUND *The referenced " Resource Admin" was not found as Administrator.*	-	x	-	-	-	-	-
400 Bad Request	OTHER_ERROR *Some other specific error was encountered processing the request* (more information in the *ErrorDescription* field).	-	-	-	(x)	(x)	(x)	(x)

		Applicable UC/ Request Method						
401 Unauthorized	UNAUTHORIZED *The user is not granted the right to issue this request.*	-	x	x	x	x	-	-
404 Resource not found	NOT_FOUND *Requested information was not found.*	-	-	x	-	x	x	x
500 Internal Server Error	TECHNICAL *Some unexpected technical error occurred* (detailed information is available in the response).	-	x	x	x	x	x	x

Table Key:
x - Service returns this error
(x) - Service may return this error. When returning this error it may provide additional unstructured information in the fields: `errorDescription` and `ErrorResponse`.

Errors Structure Detail

In case of error, a response text will be provided, in an `ErrorResponse` type of element (see definition Extended SMP XSD).

The `ErrorResponse` holds the following elements:

`BusinessCode`

This code allows the client application to behave appropriately according to the encountered error. The expected values are summarized in Error Codes Table and their applicability explicitly specified for each service in the corresponding paragraph.

`ErrorDescription`

This description provides some detailed information on the encountered error. Its content is not predefined and should be intended to help the client developer or administrator to investigate the encountered error.

`ErrorUniqueId`

This identifier uniquely identifies the occurrence of the error. This value is intended to facilitate further investigations on a specific error in particular to search into log files.

Example

```
<ErrorResponse xmlns="ec:services:SMP:1.0">
    <BusinessCode>TECHNICAL</BusinessCode>
    <ErrorDescription>Some unexpected technical error occurred. (detailed
    information available here)</ErrorDescription>
    <ErrorUniqueId>5378C627DA4275F698458AB6845C68456845</ErrorUniqueId>
</ErrorResponse>
```

XSD Files

OASIS SMP XSD

▼ bdx-smp-201605.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
     Service Metadata Publishing (SMP) Version 1.0
     Committee Specification 03
     30 June 2016
     Copyright (c) OASIS Open 2016. All Rights Reserved.
     Source: http://docs.oasis-open.org/bdxr/bdx-smp/v1.0/cs03/schemas/
     Latest version of the specification: http://docs.oasis-open.org/bdxr/bdx-
smp/v1.0/bdx-smp-v1.0.html
     TC IPR Statement: https://www.oasis-open.org/committees/bdxr/ipr.php
     -->
<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns="http://docs.oasis-open.org/bdxr/ns/SMP/2016/05"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#" elementFormDefault="qualified"
targetNamespace="http://docs.oasis-open.org/bdxr/ns/SMP/2016/05"
id="ServiceMetadataPublishing">
    <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
schemaLocation="http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd"/>
    <xs:element name="ServiceGroup" type="ServiceGroupType"/>
    <xs:element name="ServiceMetadata" type="ServiceMetadataType"/>
    <xs:element name="SignedServiceMetadata" type="SignedServiceMetadataType"/>
    <xs:complexType name="SignedServiceMetadataType">
        <xs:sequence>
            <xs:element ref="ServiceMetadata"/>
            <xs:element ref="ds:Signature"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ServiceMetadataType">
        <xs:choice>
            <xs:element name="ServiceInformation" type="ServiceInformationType"/>
            <xs:element name="Redirect" type="RedirectType"/>
        </xs:choice>
    </xs:complexType>
    <xs:complexType name="ServiceInformationType">
        <xs:sequence>
            <xs:element ref="ParticipantIdentifier"/>
            <xs:element ref="DocumentIdentifier"/>
            <xs:element name="ProcessList" type="ProcessListType"/>
            <xs:element name="Extension" type="ExtensionType" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ProcessListType">
        <xs:sequence>
```

```
            <xs:element name="Process" type="ProcessType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ProcessType">
        <xs:sequence>
            <xs:element ref="ProcessIdentifier"/>
            <xs:element name="ServiceEndpointList" type="ServiceEndpointList"/>
            <xs:element name="Extension" type="ExtensionType" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ServiceEndpointList">
        <xs:sequence>
            <xs:element name="Endpoint" type="EndpointType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="EndpointType">
        <xs:sequence>
            <xs:element name="EndpointURI" type="xs:anyURI"/>
            <xs:element name="RequireBusinessLevelSignature" type="xs:boolean"
minOccurs="0" default="false"/>
            <xs:element name="MinimumAuthenticationLevel" type="xs:string"
minOccurs="0"/>
            <xs:element name="ServiceActivationDate" type="xs:dateTime"
minOccurs="0"/>
            <xs:element name="ServiceExpirationDate" type="xs:dateTime"
minOccurs="0"/>
            <xs:element name="Certificate" type="xs:base64Binary"/>
            <xs:element name="ServiceDescription" type="xs:string"/>
            <xs:element name="TechnicalContactUrl" type="xs:anyURI"/>
            <xs:element name="TechnicalInformationUrl" type="xs:anyURI"
minOccurs="0"/>
            <xs:element name="Extension" type="ExtensionType" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="transportProfile" type="xs:string" use="required"/>
    </xs:complexType>
    <xs:complexType name="ServiceGroupType">
        <xs:sequence>
            <xs:element ref="ParticipantIdentifier"/>
            <xs:element name="ServiceMetadataReferenceCollection"
type="ServiceMetadataReferenceCollectionType"/>
            <xs:element name="Extension" type="ExtensionType" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ServiceMetadataReferenceCollectionType">
        <xs:sequence>
            <xs:element name="ServiceMetadataReference"
type="ServiceMetadataReferenceType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
```

```
    </xs:complexType>
    <xs:complexType name="ServiceMetadataReferenceType">
        <xs:attribute name="href" type="xs:anyURI"/>
    </xs:complexType>
    <xs:complexType name="RedirectType">
        <xs:sequence>
            <xs:element name="CertificateUID" type="xs:string"/>
            <xs:element name="Extension" type="ExtensionType" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="href" type="xs:anyURI" use="required"/>
    </xs:complexType>
    <xs:element name="ParticipantIdentifier" type="ParticipantIdentifierType"/>
    <xs:element name="DocumentIdentifier" type="DocumentIdentifierType"/>
    <xs:element name="ProcessIdentifier" type="ProcessIdentifierType"/>
    <xs:element name="RecipientIdentifier" type="ParticipantIdentifierType"/>
    <xs:element name="SenderIdentifier" type="ParticipantIdentifierType"/>
    <xs:complexType name="ParticipantIdentifierType">
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute name="scheme" type="xs:string"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
    <xs:complexType name="DocumentIdentifierType">
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute name="scheme" type="xs:string"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
    <xs:complexType name="ProcessIdentifierType">
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute name="scheme" type="xs:string"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
    <xs:complexType name="ExtensionType">
        <xs:annotation>
            <xs:documentation>
                A single extension for private use.
            </xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:element maxOccurs="1" minOccurs="0" name="ExtensionID"
type="xs:token">
                <xs:annotation>
                    <xs:documentation>
                        An identifier for the Extension assigned by the creator of
the extension.
```

```
                    </xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element maxOccurs="1" minOccurs="0" name="ExtensionName"
type="xs:string">
                <xs:annotation>
                    <xs:documentation>
                        A name for the Extension assigned by the creator of the
extension.
                    </xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element maxOccurs="1" minOccurs="0" name="ExtensionAgencyID"
type="xs:string">
                <xs:annotation>
                    <xs:documentation>
                        An agency that maintains one or more Extensions.
                    </xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element maxOccurs="1" minOccurs="0" name="ExtensionAgencyName"
type="xs:string">
                <xs:annotation>
                    <xs:documentation>
                        The name of the agency that maintains the Extension.
                    </xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element maxOccurs="1" minOccurs="0" name="ExtensionAgencyURI"
type="xs:anyURI">
                <xs:annotation>
                    <xs:documentation>
                        A URI for the Agency that maintains the Extension.
                    </xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element maxOccurs="1" minOccurs="0" name="ExtensionVersionID"
type="xs:normalizedString">
                <xs:annotation>
                    <xs:documentation>
                        The version of the Extension.
                    </xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element maxOccurs="1" minOccurs="0" name="ExtensionURI"
type="xs:anyURI">
                <xs:annotation>
                    <xs:documentation>
                        A URI for the Extension.
                    </xs:documentation>
                </xs:annotation>
```

```
            </xs:element>
            <xs:element maxOccurs="1" minOccurs="0" name="ExtensionReasonCode"
 type="xs:token">
                <xs:annotation>
                    <xs:documentation>
                        A code for reason the Extension is being included.
                     </xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element maxOccurs="1" minOccurs="0" name="ExtensionReason"
 type="xs:string">
                <xs:annotation>
                    <xs:documentation>
                        A description of the reason for the Extension.
                    </xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:any namespace="##other" processContents="lax"/>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

- source: http://docs.oasis-open.org/bdxr/bdx-smp/v1.0/cs03/schemas/bdx-smp-201605.xsd

Extended SMP XSD

`ErrorResponse` was defined as a response to return available detailed information on occurring error(s). You can find additional information for the values of elements `BusinessCode` and `ErrorDescription` in the Error Codes Table.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns="ec:services:SMP:1.0" targetNamespace="ec:services:SMP:1.0"
elementFormDefault="qualified" id="ServiceMetadataPublishing">
    <xs:element name="ErrorResponse" type="ErrorResponseType"/>
    <xs:complexType name="ErrorResponseType">
        <xs:sequence>
            <xs:element name="BusinessCode" type="xs:string"/>
            <xs:element name="ErrorDescription" type="xs:string" minOccurs="0"/>
            <xs:element name="ErrorUniqueId" type="xs:string"/>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

Support

eDelivery Support Team maintains and supports the DomiSMP Documentation. For any questions, comments or requests for change, please contact:

- **Email**: ec-edelivery-support@ec.europa.eu
- **Hours**: 8AM to 6PM (Normal EC working days)