Service Design Template for SECOM Service

Contents

[1. Introduction 6](#_Toc158626386)

[1.1. General 6](#_Toc158626387)

[1.2. Intended Readership 6](#_Toc158626388)

[1.3. Inputs from Other Sources 6](#_Toc158626389)

[2. SERVICE DESIGN TEMPLATE IDENTIFICATION 6](#_Toc158626390)

[3. TECHNOLOGY INTRODUCTION 7](#_Toc158626391)

[3.1. General 7](#_Toc158626392)

[3.2. Service technology and service transportation protocol 7](#_Toc158626393)

[3.3. Security 7](#_Toc158626394)

[3.3.1. Communication channel security 7](#_Toc158626395)

[3.3.2. Data protection 8](#_Toc158626396)

[3.3.3. Data Signature 8](#_Toc158626397)

[3.3.4. Data Encryption 8](#_Toc158626398)

[4. SERVICE DESIGN TEMPLATE OVERVIEW 9](#_Toc158626399)

[4.1. General 9](#_Toc158626400)

[4.2. Service Template Interfaces 9](#_Toc158626401)

[4.3. TEMPLATE User Instructions (PERHAPS MOVE SECTION) 10](#_Toc158626402)

[5. PHYSICAL SERVICE TEMPLATE DATA MODEL 11](#_Toc158626403)

[6. SERVICE TEMPLATE INTERFACE DESIGN 11](#_Toc158626404)

[6.1. Service Interface SECOM REST 11](#_Toc158626405)

[6.2. Operation UPLOAD 11](#_Toc158626406)

[6.2.1. Operation Functionality 11](#_Toc158626407)

[6.2.2. Operation Parameters 12](#_Toc158626408)

[6.2.3. Interpretation guidelines 13](#_Toc158626409)

[6.3. Operation GET 14](#_Toc158626410)

[6.3.1. Operation Functionality 14](#_Toc158626411)

[6.3.2. Operation Parameters 14](#_Toc158626412)

[6.3.3. Interpretation guidelines 15](#_Toc158626413)

[6.4. Operation UPLOAD LINK 16](#_Toc158626414)

[6.4.1. Operation Functionality 16](#_Toc158626415)

[6.4.2. Operation Parameters 16](#_Toc158626416)

[6.4.3. Interpretation guidelines 17](#_Toc158626417)

[6.5. Operation GET BY LINK 18](#_Toc158626418)

[6.5.1. Operation Functionality 18](#_Toc158626419)

[6.5.2. operation Parameters 18](#_Toc158626420)

[6.5.3. Interpretation guidelines 19](#_Toc158626421)

[6.6. Operation ACKNOWLEDGEMENT 19](#_Toc158626422)

[6.6.1. Operation Functionality 19](#_Toc158626423)

[6.6.2. operation Parameters 19](#_Toc158626424)

[6.6.3. Interpretation guidelines 20](#_Toc158626425)

[6.7. Operation GET SUMMARY 20](#_Toc158626426)

[6.7.1. Operation Functionality 20](#_Toc158626427)

[6.7.2. Operation Parameters 20](#_Toc158626428)

[6.7.3. Interpretation guidelines 22](#_Toc158626429)

[6.8. Operation ACCESS 22](#_Toc158626430)

[6.8.1. Operation Functionality 22](#_Toc158626431)

[6.8.2. Operation Parameters 22](#_Toc158626432)

[6.8.3. Interpretation guidelines 23](#_Toc158626433)

[6.9. Operation ACCESS NOTIFICATION 23](#_Toc158626434)

[6.9.1. Operation Functionality 23](#_Toc158626435)

[6.9.2. Operation Parameters 23](#_Toc158626436)

[6.9.3. Interpretation guidelines 24](#_Toc158626437)

[6.10. Operation SUBSCRIPTION 25](#_Toc158626438)

[6.10.1. Operation Functionality 25](#_Toc158626439)

[6.10.2. Operation Parameters 25](#_Toc158626440)

[6.10.3. Interpretation guidelines 26](#_Toc158626441)

[6.11. Operation SUBSCRIPTION NOTIFICATION 26](#_Toc158626442)

[6.11.1. Operation Functionality 26](#_Toc158626443)

[6.11.2. Operation Parameters 26](#_Toc158626444)

[6.11.3. Interpretation guidelines 27](#_Toc158626445)

[6.12. Operation REMOVE SUBSCRIPTION 27](#_Toc158626446)

[6.12.1. Operation Functionality 27](#_Toc158626447)

[6.12.2. Operation Parameters 27](#_Toc158626448)

[6.12.3. Interpretation guidelines 28](#_Toc158626449)

[6.13. Operation CAPABILITY 28](#_Toc158626450)

[6.13.1. Operation Functionality 28](#_Toc158626451)

[6.13.2. Operation Parameters 28](#_Toc158626452)

[6.13.3. Interpretation guidelines 29](#_Toc158626453)

[6.14. Operation PING 29](#_Toc158626454)

[6.14.1. Operation Functionality 29](#_Toc158626455)

[6.14.2. Operation Parameters 29](#_Toc158626456)

[6.14.3. Interpretation guidelines 30](#_Toc158626457)

[6.15. Operation ENCRYPTIONKEY 30](#_Toc158626458)

[6.15.1. Operation Functionality 30](#_Toc158626459)

[6.15.2. Operation Parameters 30](#_Toc158626460)

[6.15.3. Interpretation guidelines 31](#_Toc158626461)

[6.16. Operation ENCRYPTIONKEY NOTIFICATION 32](#_Toc158626462)

[6.16.1. Operation Functionality 32](#_Toc158626463)

[6.16.2. Operation Parameters 32](#_Toc158626464)

[6.16.3. Interpretation guidelines 32](#_Toc158626465)

[6.17. Operation GET PUBLICKEY 33](#_Toc158626466)

[6.17.1. Operation Functionality 33](#_Toc158626467)

[6.17.2. Operation Parameters 33](#_Toc158626468)

[6.17.3. Interpretation guidelines 33](#_Toc158626469)

[6.18. Operation UPLOAD PUBLIC KEY 34](#_Toc158626470)

[6.18.1. Operation Functionality 34](#_Toc158626471)

[6.18.2. Operation Parameters 34](#_Toc158626472)

[6.18.3. Interpretation guidelines 34](#_Toc158626473)

[7. SERVICE TEMPLATE DYNAMIC BEHAVIOUR 35](#_Toc158626474)

[7.1. Generic sequence for signatures 35](#_Toc158626475)

[8. DEFINITIONS 35](#_Toc158626476)

[8.1. Terminology 36](#_Toc158626477)

[9. ACRONYMS 36](#_Toc158626478)

[10. REFERENCES 36](#_Toc158626479)

[11. APPENDIX - SERVICE DESIGN TEMPLATE OpenAPI (Swagger) in JSON 38](#_Toc158626480)

Table of figures

**No table of figures entries found.**

List of tables

N/A

# Introduction

The IEC 63173-2 SECOM contains the definition of a REST API that is payload agnostic and security procedure for the communication between the services. This template references IEC 63173-2 SECOM and complements where needed with clarifications and experiences from implementing.

The rationale for this document is to describe the SECOM REST Service API as a service design Template according to G1128. The concept of design templates in G1128 is a proposal at this date but foreseen to be included in future versions of G1128.

## General

The purpose of this document is to define a service design template that other service designs can either copy or refer to. This template is therefore payload agnostic and describes the complete set of SECOM service interfaces. A service design may not implement the optional interfaces.

The intention is that service technical designs can refer to this template and add the specific data exchanged, the specific operational context (by reference to service specification) and the specific service interfaces used in that design.

The template can be used by both ship services and shore services.

## Intended Readership

This service design description document is intended to be read by service architects and designers of technical services.

## Inputs from Other Sources

The main part of this template is from IEC 63173-2 SECOM.

# SERVICE DESIGN TEMPLATE IDENTIFICATION

The purpose of this section is to provide a unique identification of the service design and describe where the service is in terms of the engineering lifecycle.

1. Service Design Identification

|  |  |
| --- | --- |
| Name | SECOM REST Service Design Template |
| ID | urn:mrn:iala:techsvc:design:secom:template |
| Version | 0.1 |
| Technology | REST |
| Service Specification ID | N/A |
| Service Specification Version | N/A |
| Description | REST Service design template according to IEC 63173-2 SECOM v1.0.0 |
| Keywords | REST, SECOM, IEC, Template |
| Architect(s) | IEC 63173-2 SECOM |
| Status | Provisional |

# TECHNOLOGY INTRODUCTION

## General

This design template realizes the SECOM Service defined in IEC 63173-2 SECOM as a service design template in G1128 format.

The chosen technology is REST and HTTPS TLS as communication protection of the service.

## Service technology and service transportation protocol

**Reference: IEC 63173-2 SECOM v1.0.0 Clause 5.3 Service Technology**

The technology (architectural style) chosen is REST (REpresentational State Transfer) upon HTTP/1.1 (RFC 7231).

## Security

### Communication channel security

**Reference: IEC 63173-2 SECOM v1.0.0 Clause 6 SECOM communication channel security**

The channel security between the SECOM REST service and a consumer are

* HTTP/1.1 according to RFC-7231
* HTTPS over TLS according to RFC-2818

Valid versions of TLS for this version of service design template are

* TLS version 1.1 (RFC-4346)
* TLS version 1.2 (RFC-5246)
* TLS version 1.3 (RFC-8446)

X.509 Certificates are used in the TLS according to RFC 5280 and RFC 2459.

Certificates shall be verified with OCSP and/or CRL methods.

### Data protection

**Reference: IEC 63173-2 SECOM v1.0.0 Clause 7 SECOM data protection**

**Reference: IHO S-100 ed5.0.0 Part 15 Data Protection Scheme**

The data is mandatory to be signed by the sender to enable data authentication and integrity check by the receiver.

The data can optionally be encrypted by the sender, and the sender is responsible for exchanging the encryption key with the receiver.

The data (one or more data files) can optionally be packaged and compressed before being signed and sent.

### Data Signature

**Reference: IEC 63173-2 SECOM v1.0.0 Clause 7.3 Data authentication and signing**

**Reference: IHO S-100 ed5.0.0 Part 15-8 Data Authentication**

**Reference: NIST Digital Signature Standard (DSS–FIPS Publication 186)**

The recommended algorithm for signing data is ECDSA but the design supports also other algorithms.

The signature is transported in HEX.

### Data Encryption

**Reference: IEC 63173-2 SECOM v1.0.0 Clause 7.4 Data encryption**

**Reference: IHO S-100 ed5.0.0 Part 15-6 Data Encryption**

The encryption algorithm for encryption is AES and CBC mode.

The symmetric encryption key can be exchanged by different means, including using the SECOM REST API to exchange the encryption key.

# SERVICE DESIGN TEMPLATE OVERVIEW

## General

**Reference: IEC 63173-2 SECOM v1.0.0 Clause 5 SECOM information service interface**

This service design template is based on IEC 63173-2 SECOM Service and can be referred to by service designs for specific data formats and purposes.

## Service Template Interfaces

**Reference: IEC 63173-2 SECOM v1.0.0 Clause 5.7** **SECOM service interface definitions**

The table below shows the interfaces defined in this template.

|  |  |  |
| --- | --- | --- |
| Interface | SECOM Reference | Comment |
| Upload | IEC 63173-2 SECOM v1.0.0 Clause 5.7.2 service interface – Upload | This interface is called when the client uploads (pushes) data to the service. The sender (client) decides the format and protection of the data. |
| Upload Link | IEC 63173-2 SECOM v1.0.0 Clause 5.7.3 service interface – Upload Link | This interface is called when the client uploads (pushes) a reference pointer to large data. The data is downloaded using interface Get By Link. |
| Acknowledgement | IEC 63173-2 SECOM v1.0.0 Clause 5.7.4 service interface – Acknowledgement | This interface is called as response to Acknowledgement request in Upload. |
| Get | IEC 63173-2 SECOM v1.0.0 Clause 5.7.5 service interface – Get | This interface is called when the client gets (pulls) data from the service. |
| Get Summary | IEC 63173-2 SECOM v1.0.0 Clause 5.7.6 service interface – Get Summary | This interface is called when the client gets a summary of available data from the service. The data is retrieved (pulled) using the interface Get. |
| Get By Link | IEC 63173-2 SECOM v1.0.0 Clause 5.7.7 service interface – Get By Link | This interface is called when the client downloads (pulls) large data by reference given from interface Upload Link. |
| Access | IEC 63173-2 SECOM v1.0.0 Clause 5.7.8 service interface – Access | This interface is called when the client asks for access to data from the service. Response is given by callback to Access Notification. |
| Access Notification | IEC 63173-2 SECOM v1.0.0 Clause 5.7.9 service interface – Access Notification | This interface is called as response to interface Access. |
| Subscription | IEC 63173-2 SECOM v1.0.0 Clause 5.7.10 service interface – Subscription | This interface is called when the client or server initiates subscription on data from the service. Response is given with interface Upload and Subscription Notification. |
| Remove Subscription | IEC 63173-2 SECOM v1.0.0 Clause 5.7.11 service interface – Remove Subscription | This interface is called when the client or server removes subscription. Response is given with interface Subscription Notification. |
| Subscription Notification | IEC 63173-2 SECOM v1.0.0 Clause 5.7.12 service interface – Subscription Notification | This interface is called as response from Subscription or Remove Subscription. |
| Capability | IEC 63173-2 SECOM v1.0.0 Clause 5.7.13 service interface – Capability | This interface is called when the client asks for the service capabilities. |
| Ping | IEC 63173-2 SECOM v1.0.0 Clause 5.7.14 service interface – Ping | This interface is called when the client checks the availability of the service. |
| EncryptionKey | IEC 63173-2 SECOM v1.0.0 Clause 5.7.15 service interface – EncryptionKey | This interface is called when sending (pushing) encryption key to a receiver. |
| PublicKey | IEC 63173-2 SECOM v1.0.0 Clause 5.7.16 service interface – PublicKey | This interface is called when the client gets (pulls) the public certificate(s) from the service. |

## TEMPLATE User Instructions (PERHAPS MOVE SECTION)

This document is intended to be used as either reference or as a document template for any new technical design according to G1128. The exact identifier and version of this technical design template shall be referenced.

The operations not used shall be kept as headings, but the content is replaced with “Not implemented” and the HTTP return code shall be HTTP 405 Method not allowed or HTTP 501 Not implemented.

Additional operations may be added but be careful, it may compromise anyone who implements a service based on the IEC Standard alone.

Suggested change of above paragraph: Exercise caution when considering the inclusion of supplementary operations, as their integration may pose a potential risk for entities implementing a service solely based on the IEC Standard.

The section “Interpretation guidelines” may be added with specific guidelines related to the focus in Technical Design.

The baseURL registered in Maritime Service registry may include information regarding the information exchanged, such as baseURL/s124/object/v1, but it’s not a requirement. It may also be baseURL/object/v1 that is agnostic to payload (handles multiple payload types).

# PHYSICAL SERVICE TEMPLATE DATA MODEL

The data model of the payload is not part of the template and need to be specified in service specification and/or the payload dependent service design.

For the service data exchange model, see each interface design.

# SERVICE TEMPLATE INTERFACE DESIGN

**Reference: IEC 63173-2 SECOM v1.0.0 Clause 5.7 SECOM service interface definitions**

This section describes each service interface based on information in the open API file for SECOM. For all details, please see IEC 63173-2 SECOM Standard.

The Service Interface design covers the static design description while the dynamic design (behaviour) is described in the section SERVICE TEMPLATE DYNAMIC BEHAVIOUR.

## Service Interface SECOM REST

## Operation UPLOAD

This interface is called when the client uploads (pushes) data to the service. The sender (client) decides format and protection of the data. If acknowledgement is requested, it will be given by callback to interface Acknowledgement.

The purpose with this interface is to upload (push) information that is not larger than 350kb (Base64 encoded) to an information consumer. Larger data may generate HTTP response code 413 Payload too large.

An information consumer shall implement this interface in order to receive information.

### Operation Functionality

The incoming data is restored and validated against the data schema. If valid, the data is forwarded to end user for processing.

If acknowledgement is requested, the acknowledgement is sent when the data is beginning to be processed by the end user.

### Operation Parameters

**POST baseUrl/v1/object {body} : response**

The data (payload) package with its metadata.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Format | Require |
| envelope  data | string | Base64 | Required |
| envelope  containerType | integer | enum | Required |
| envelope  dataProductType | integer | enum | Required |
| envelope  exchangeMetadata  dataProtection | boolean |  | Required |
| envelope  exchangeMetadata  protectionScheme | string |  | Required |
| envelope  exchangeMetadata  dataSignatureReference | string |  | Required |
| envelope  exchangeMetadata  dataSignatureValue  publicRootCertificateThumbprint | string | SHA1 and HEX | Optional |
| envelope  exchangeMetadata  dataSignatureValue  publicCertificate | string | Minified PEM | Required |
| envelope  exchangeMetadata  compressionFlag | boolean |  | Required |
| envelope  fromSubscription | boolean |  | Optional |
| envelope  ackRequest | boolean |  | Required |
| envelope  transactionIdentifier | string | uuid | Required |
| envelope  envelopeSignatureCertificate | string | Minified PEM | Required |
| envelope  envelopeRootCertificateThumbprint | string | SHA1 and HEX | Optional |
| envelope  envelopeSignatureTime | string | date-time | Optional |
| envelopeSignature | string | HEX | Required |

Table 1 Service body

|  |  |  |  |
| --- | --- | --- | --- |
| Response Code | Object | Type | Description |
| 200 | message | string | OK |
| Secom\_ResponseCodeEnum | integer |  |
| 400 | message | string | Bad request |
| 401 | message | string | Unauthorized |
| 403 | message | string | Forbidden |
| 405 | message | string | Method not allowed |
| 500 | message | string | Internal server error |
| 501 | message | string | Not implemented |

Table 2 Service response (complete list in RFC 9110)

### Interpretation guidelines

Thedata is always provided in a one-line Base64 encoded string.

The datacontent is defined by

* Type of message in dataProductType [enum] as integer  
  e.g. 24 = S-421
* Wrapping according to containerType [enum] as integer  
  e.g. S-100 DataSet
* ZIP according to compressionFlag
* Encrypted according to dataProtection [flag]  
  True means data is encrypted and a encryptionKey is needed

**Data signature** shall be provided in a one-line HEX format using DSA (**dataSignatureReference**).

The name of the **protectionScheme** is not specified by SECOM and need to be agreed upon. The proposal here is to use SECOM but may need to be more specified, e.g. Navelink SECOM.

**Exchange information**

* Transaction identifier in UUID (unique for every upload)
* Standalone or in subscription according to fromSubscription [flag]
* Acknowledgement request according to ackRequest

**Envelope signature**

The envelope signature shall be made on a dot (.) separated “CSV” data structure of the envelope:

data (Base64).containerType (int).dataProductType (int).dataProtection (true/false).protectionScheme (String).publicRootCertificateThumbprint .publicCertificate (Base64 minified PEM).digitalSignature (one line HEX).compressionFlag (true/false).fromSubscription (true/false).ackrequest (int).transactionidentifier (String UUID).envelopeSignatureCertificate (Base64 minified PEM).envelopeRootCertificateThumbprint.envelopeSignatureTime (UNIX seconds)

## Operation GET

This interface is called when the client gets (pulls) data from the service.

The Get interface is used for pulling information from a service provider. The owner of the information (provider) is responsible for the authorization procedure before returning information. The consumer can ask for information by its reference, geometry, time or arbitrary query for e.g. status on the information product. If no filtering parameters are given, all authorized information is to be sent. The information owner decides what information the consumer is authorized to based on the identity in the TLS client certificate i.e. the identity the service instance belongs to.

### Operation Functionality

Authorized data is packaged according to the request and returned in the service call.

Request-Response pattern.

### Operation Parameters

**GET baseUrl/v1/object/pathParams?queryParams : response**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Format | Require |
| dataReference | string | uuid | optional |
| containerType | integer | enum | optional |
| dataProductType | integer | enum | optional |
| productVersion | string |  | optional |
| geometry | string | WKT | optional |
| unlocode | string | [A-Z]{5} | optional |
| validFrom | string | date-time | optional |
| validTo | string | date-time | optional |
| page | integer |  | optional |
| pageSize | integer |  | optional |

Table 3 Service parameter

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 200 | application/json | string | OK |
| array of { |  |  |
| data | string |  |
| exchangeMetadata  dataProtection | boolean | True if data encrypted |
| exchangeMetadata  protectionScheme | string | Reference to protection schema administrator |
| exchangeMetadata  digitalSignatureReference | string |  |
| exchangeMetadata  digitalSignatureValue  publicRootCertificateThumbprint | string |  |
| exchangeMetadata  digitalSignatureValue  publicCertificate | string |  |
| exchangeMetadata  digitalSignatureValue  digitalSignature | string |  |
| exchangeMetadata   compressionFlag | boolean | True if compressed |
| ackRequest | integer |  |
| } |  |  |
| pagination  totalItems | integer |  |
| pagination  maxItemsPerPage | integer |  |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 403 | application/json | string | Forbidden |
| 404 | application/json | string | Information not found |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 4 Service response (complete list in RFC 9110)

### Interpretation guidelines

If no data is returned, the response is HTTP code 404.

The dataReference is used when a known data object is retrieved. The dataReference is provided in the response from Get Summary and is typically an UUID.

The containterType describes how the data is packaged, e.g. S100\_DataSet, S100\_ExchangeSet or no specific container, e.g. RTZ in XML.

The dataProductType describes which dataproduct that is in focus in the operation. The dataProductType can be set to OTHER if it is not in the SECOM enumeration. An S100\_exchangeSet can contain more than one S-product. In that case the dataProductType is set to the most dominant S-product requested.

The productVersion describes the specific version of the data product specification of the data requested.

The geometry describes geographical criteria for the data requested. The geometry is given as a one-line WKT formatted string and can contain? one or more geographical objects.

The unlocode describes with 5 characters the UN code for city or other area, such as a port. The unlocode is a list administered by UN and can be found on internet. Many unlocodes have a position in long/lat attached, but not all. In most cases the unlocode will be used as text only but this may depend on the service developer.

The validFrom/validTo describes the requested validitytime for the requested data in ISO 8601 UTC formatted as yyyy-mm-dd hh:mmZ.

## Operation UPLOAD LINK

The purpose with this interface is to upload (push) a link to information to a consumer. Hence, a consumer shall implement this interface in order to receive a link to the information that can be retrieved. This interface shall be used when large amounts of data shall be exchanged. The provider of information then uploads a link to a consumer, and the consumer then uses the Get by Link interface to pull the data from the provider.

Use this when data is larger than allowed with POST in REST.

### Operation Functionality

The link to data represented by the transactionIdentifier is stored and the called service is identified. A service lookup in MSR may be required to retrieve the URL to the called service. A service call is made to the Get By Link operation and the transactionIdentifier is provided. The sent data is then downloaded.

### Operation Parameters

**POST baseUrl/v1/object {body} : response**

The message link with its metadata

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Format | Require |
| envelope  containerType | integer | enum | required |
| envelope  dataProductType | integer | enum | required |
| envelope  exchangeMetadata  dataProtection | boolean |  | required |
| envelope  exchangeMetadata  protectionScheme | string |  | required |
| envelope  exchangeMetadata  dataSignatureReference | string |  | required |
| envelope  exchangeMetadata  dataSignatureValue  publicRootCertificateThumbprint | string | SHA1 and HEX | optional |
| envelope  exchangeMetadata  dataSignatureValue  publicCertificate | string | Minified PEM | required |
| envelope  exchangeMetadata  compressionFlag | boolean |  | required |
| envelope  fromSubscription | boolean |  | optional |
| envelope  ackRequest | integer | enum | required |
| envelope  transactionIdentifier | string | uuid | required |
| envelope  envelopeSignatureCertificate | string | Minified PEM | required |
| envelope  envelopeRootCertificateThumbprint | string | SHA1 and HEX | optional |
| envelope  size | integer |  | optional |
| envelope  timeToLive | string | date-time | required |
| envelope  envelopeSignatureTime | string | date-time | optional |
| envelopeSignature | string | HEX | required |

Table 5 Service body

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Response Code | Content | Object | Type | Description |
| 200 | application/json | message | string | OK |
|  | Secom\_ResponseCodeEnum | integer |  |
| 400 | application/json | message | string | Bad request |
| 401 | application/json | message | string | Unauthorized |
| 403 | application/json | message | string | Forbidden |
| 405 | application/json | message | string | Method not allowed |
| 500 | application/json | message | string | Internal server error |
| 501 | application/json | message | string | Not implemented |

Table 6 Service response (complete list in RFC 9110)

### Interpretation guidelines

The main data is the transactionIdentifier that shall be used as reference in a GET data By Link. The REST GET method allows any size and type of data to be downloaded.

The envelope contains further information on the data that want to be sent. (that the client wants to send)

If delivery acknowledgement is requested (ackRequest = 1 or 3) an Acknowledgement shall be sent back to the called service when data is retrieved with Get By Link. If opened acknowledgement is requested (ackRequest = 2 or 3) an Acknowledgement shall be sent back to the called service when data is being processed by the end user.

The size shall be given in kBytes of the data that will be returned in the Get By Link call. The rationale is that the receiver may assess when in time it is suitable to actually retrieve the information.

The timeToLive is given as information until what time the data will be available to retrieve with get By Link.

## Operation GET BY LINK

This interface is called when the client downloads (pulls) large data by reference given from interface Upload Link.

The Get By Link interface is used for pulling information from a data storage handled by the information owner. The link to the data storage can be exchanged with Upload Link interface. The owner of the information (provider) is responsible for the relevant authentication and authorization procedure before returning information.

### Operation Functionality

The operation validates the given transactionIdentifier against the caller’s identity and the timeToLive given for the data. The data is then packaged according to agreement in Upload By Link and returned in the call.

Request-Response pattern.

### operation Parameters

**GET baseUrl/v1/object/link/pathParam?queryParam : response**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Param |  | Type | Format | Require |
| transactionIdentifier | query | string | uuid | required |

Table 7 Service parameters

|  |  |  |  |
| --- | --- | --- | --- |
| Response Code | Content | Type | Description |
| 200 | application/octet-stream | string | OK |
| 200 | application/json | string | OK |
| 400 | application/octet-stream |  | Bad request |
| 400 | application/json |  | Bad request |
| 401 | application/octet-stream |  | Unauthorized |
| 401 | application/json |  | Unauthorized |
| 403 | application/octet-stream |  | Forbidden |
| 403 | application/json |  | Forbidden |
| 404 | application/octet-stream |  | Not Found |
| 404 | application/json |  | Not Found |
| 405 | application/octet-stream |  | Method not allowed |
| 405 | application/json |  | Method not allowed |
| 500 | application/octet-stream |  | Internal server error |
| 500 | application/json |  | Internal server error |
| 501 | application/octet-stream |  | Not implemented |
| 501 | application/json |  | Not implemented |

Table 8 Service response (complete list in RFC 9110)

### Interpretation guidelines

The unique transactionIdentifier given by the Upload By Link call.

## Operation ACKNOWLEDGEMENT

This interface is called as response to Acknowledgement request in Upload.

During upload of information, an acknowledgement can be requested which is expected to be asynchronously received when the uploaded message has been delivered to the end system (technical acknowledgement), and an acknowledgement is delivered when the message has been opened and/or processed by the end user (operational acknowledgement). The acknowledgement contains a reference to the object delivered and has no time limit.

### Operation Functionality

The operation validates the given transactionIdentifier and handles the acknowledgement of sent data. The acknowledgement can be either for deliver, opened or both.

### operation Parameters

**POST baseUrl/v1/acknowledgement {body} : response**

Object with reference to information and time when delivered

|  |  |  |  |
| --- | --- | --- | --- |
| Body | Type | Format | Require |
| envelope  createdAt | string | date-time | required |
| Envelope  envelopeCertificate | string | Minified PEM | required |
| Envelope envelopeRootCertificateThumbprint | string | SHA1 | required |
| Envelope  transactionIdentifier | string | uuid | required |
| Envelope  ackType | integer | enum | required |
| Envelope  nackType | integer | enum | optional |
| Envelope  envelopeSignatureTime | string | date-time | required |
| digitalSignature | string | HEX | required |

Table 9 Service body

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 200 | application/json | string | OK |
| message | string |  |
| Secom\_ResponseCodeEnum | integer |  |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 403 | application/json | string | Forbidden |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 10 Service response (complete list in RFC 9110)

### Interpretation guidelines

The transactionIdentifier describes the reference to what sending transaction of data that is acknowledged.

The response can be either positive acknowledgement (ackType = 1 or 2) or negative acknowledgement (ackType = 3 (error). If negative acknowledgement is given, the error message is expected to be provided in nackType (0-4) describing the error.

## Operation GET SUMMARY

This interface is called when the client wants a summary of available data from the service. The actual data is retrieved (pulled) using the interface Get.

A list of information shall be returned from this interface. The summary contains identity, status, size and a short description of each information object. The actual information object shall be retrieved using the Get interface. The consumer can ask for information by geometry, location and time. If no filtering parameters are given, available summary information is to be sent.

### Operation Functionality

The operation provides a list of information references that can be retrieved by the called party.

### Operation Parameters

**GET baseUrl/v1/object/summary/pathParam?queryParam : response**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Param | Param | Type | Format | Require |
| containerType | Query | Integer | Enum | Optional |
| dataProductType | Query | Integer | Enum | Optional |
| productVersion | Query | String |  | Optional |
| geometry | Query | String | WKT | Optional |
| unlocode | Query | String | 5[A-Z] | Optional |
| validFrom | Query | String | date-time | Optional |
| validTo | Query | String | date-time | Optional |
| page | Query | Integer |  | Optional |
| pageSize | Query | Integer |  | Optional |

Table 11 Service parameters

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 200 | application/json | string | OK |
| array of { |  |  |
| informationSummaryObject  dataReference | string |  |
| informationSummaryObject  dataProtection |  |  |
| informationSummaryObject  dataCompression |  |  |
| informationSummaryObject  containerType | integer |  |
| informationSummaryObject  dataProductType | integer |  |
| informationSummaryObject  info\_productVersion | string |  |
| informationSummaryObject  info\_identifier | string |  |
| informationSummaryObject  info\_name | string |  |
| informationSummaryObject  info\_Status | string |  |
| informationSummaryObject  info\_Description | string |  |
| informationSummaryObject  info\_lastModifiedDate | string |  |
| informationSummaryObject  info\_size | string |  |
| } |  |  |
| pagination  totalItems | integer |  |
| pagination  maxItemsPerPage | integer |  |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 12 Service response (complete list in RFC 9110)

### Interpretation guidelines

The data criteria are all optional. If none is given, all available information references shall be returned. If several critera are given, they shall be combined with logical AND, hence only information references are returned that meet all given critera.

The response could contain both authorized data and data that required further authorization.

In the response, the dataReference are given as UUID for each data object listed. The dataReference can then be provided as input to a Get call.

The dataProtection flag indicates whether the data is encrypted or not. If encrypted, an encryptionKey is required to decrypt the data.

The compression flag indicates whether the data is compressed with ZIP or not.

The dataProductType and productVersion indicate type and version of the data specification for the data object, e.g. 24=S-421 and productVersion “1.0.0”.

The info\_identifier, info\_name, info\_Status, info\_Description, and info\_lastModofoedData are data from the specific productType.

The info\_size represents the size of the data attribute when retrieved by Get data. The size does not include the envelope.

## Operation ACCESS

This interface is called when the client asks for access to data from the service. Response is given by callback to interface Access Notification.

Access to information can be requested through the Access interface. The result is sent asynchronously through the Access Notification interface.

### Operation Functionality

The operation checks the availability for the caller to requested information.

Request-Callback pattern. (explain)

### Operation Parameters

**POST baseUrl/v1/access {body} : response**

Description of reason for requesting access to information

|  |  |  |  |
| --- | --- | --- | --- |
| Body | Type | Format | Require |
| reason | String |  | required |
| reasonEnum | Integer | enum | required |
| containerType | Integer | enum | optional |
| dataProductType |  |  | optional |
| dataReference | String |  | optional |
| productVersion | String |  | optional |

Table 13 Service body

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 200 | application/json | string | The operation was performed successfully |
| Secom\_ResponseCodeEnum | integer |  |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 14 Service response (complete list in RFC 9110)

### Interpretation guidelines

The reason given in text is for a human user to read before replying on the request. For automatic processing, the reason is also given in the enumeration.

The dataProductType and productVersion indicates what information is requested access to.

Suggested formulation of above sentence: The dataProductType and productVersion parameters specify the type of information for which access is being requested.

The dataReference indicates request to access a specific data object provided in Get Summary.

The response is provided in an Access Notification object.

## Operation ACCESS NOTIFICATION

This interface is called as callback response to interface Access.

Results from Access Request shall be sent asynchronously through this interface.

### Operation Functionality

The response to an Access request is received in this operation.

### Operation Parameters

**POST baseUrl/v1/access/** **notification {body} : response**

Result from the request for access; True or False

|  |  |  |  |
| --- | --- | --- | --- |
| Body | Type | Format | Require |
| Decision | boolean |  | required |
| decisonReason | String |  | required |
| transactionIdentifier | String | uuid | required |

Table 15 Service body

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 200 | application/json | string | The operation was performed successfully |
| Secom\_ResponseCodeEnum | integer |  |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 16 Service response (complete list in RFC 9110)

### Interpretation guidelines

The decision by the information owner is given as true or false.

A reason for the decision is always given in decisionReason as text for a human user.

If the decision is granted, a dataReference to the data object should be provided. This can be used in Get data call or Subscription call.

## Operation SUBSCRIPTION

This interface is called when the client or server initiates subscription on data from the service. Response is given as callback to interface Upload and Subscription Notification.

The purpose of the interface is to request subscription on information, either specific information according to parameters, or the information accessible upon decision by the information provider. Each subscription request reflects one parameter query set.

### Operation Functionality

The operation checks authorization to requested data and initiates a subscription if granted access. The response contains a subscriptionIdentifier that can be used to remove the subscription. Internally this subscriptionIdentifier is related to the baseURL on called service, where it is assumed that the service calling the subscription operation does also contain the Upload operation where subscribed data shall be pushed.

The data is returned in Upload operation(s) where fromSubscription is set to true.

Request-Response pattern that initiates a Request-Callback (SubscriptioNotofication) and Publish-Subscribe pattern (Upload).

### Operation Parameters

**POST baseUrl/v1/** **subscription {body} : response**

Specific id on the information object subscription is requested for specific status on the information or specific area of interest.

|  |  |  |  |
| --- | --- | --- | --- |
| Body | Type | Format | Require |
| containerType | integer | enum | optional |
| geometry | string |  | optional |
| unlocode | string |  | optional |
| dataProductType | integer | enum | optional |
| productVersion | string |  | optional |
| dataReference | string |  | optional |
| subscriptionPeriodStart | string | date-time | optional |
| subscriptionPeriodEnd | string | date-time | optional |

Table 17 Service body

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 200 | application/json | string | OK |
| message | string |  |
| subscriptionIdentifier | string |  |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 18 Service response (complete list in RFC 9110)

### Interpretation guidelines

All criteria for the subscription are optional. If none are provided, all authorized data from the service will be included in the subscription. If more than one criterion is provided, the criteria will be combined with logical AND, hence all criteria must be met.

The containerType criteria, e.g. S100\_DataSet, means that the user wishes to only receive data from this subscription packaged into S100\_Datasets.

The geometry criteria means that only information intersecting with the geometry given will be sent in the subscription. This requires that data is geotagged somehow. If information is not geotagged, the criteria is ignored.

The unlocode criteria means that only information related to the unlocode will be sent in the subscription. If there is no unlocode related to the information provided by the service, the unlocode is ignored.

The dataProductType and productVersion criteria means that only information of the type and version will be sent in the subscription.

The dataReference criteria means that only new updates of the specific data will be sent in the subscription. If the data object ends, the subscription ends.

The subscriptionPeriodStart and subscriptionPeriodEnd criteria mean that data produced in the given time period will be sent in the subscription. When subscriptionPeriodEnd is reached, the subscription shall be removed automatically. The time period is recommended to be formatted as ISO 8601 yyyy-mm-dd hh:mm:ssZ in UTC.

## Operation SUBSCRIPTION NOTIFICATION

This interface is called as callback response from interface Subscription or Remove Subscription.

The interface receives notifications when the subscription is created or removed by the information producer.

### Operation Functionality

When the subscription changes (i.e. is created or removed), the event will be received in this operation as information. The change (create or remove) can be initiated both by the client and by the server.

### Operation Parameters

**POST baseUrl/v1/subscription/notification {body} : response**

Contains the identity of the information object in focus and type of event; Create or Delete.

|  |  |  |  |
| --- | --- | --- | --- |
| Body | Type | Format | Require |
| subscriptionIdentifier | string | uuid | required |
| eventEnum | integer | enum | required |

Table 19 Service body

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 200 | application/json | string | OK |
| message | string |  |
| subscriptionIdentifier | string |  |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 20 Service response (complete list in RFC 9110)

### Interpretation guidelines

The subscriptionIdentifier is the reference to the specific subscription in focus.

The eventEnum describes the type of event, e.g. 1=subscription created, 2=subscription removed.

## Operation REMOVE SUBSCRIPTION

This interface is called when the client or server removes a subscription. Response is given as callback to interface Subscription Notification.

Subscription(s) can be removed either internally by the information owner, or externally by the consumer. This interface shall be used by the consumer to request removal of a subscription.

### Operation Functionality

The operation removes the subscription attached to the subscriptionIdentifier. The operation generates a callback call to SubscriptionNotification.

### Operation Parameters

**DELETE baseUrl/v1/** **subscription {body} : response**

The specific identity of the information object to remove subscription for. If no id entity is provided, all subscriptions for the caller are removed.

|  |  |  |  |
| --- | --- | --- | --- |
| Body | Type | Format | Require |
| subscriptionIdentifier | string | uuid | required |

Table 21 Service body

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 200 | application/json | string | OK |
| message | string | Subscription \*identifier\* removed |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 403 | application/json | string | Forbidden |
| 404 | application/json | string | Not Found |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 22 Service response (complete list in RFC 9110)

### Interpretation guidelines

The incoming subscriptionIdentifier given when the subscription was created, and provided in the SubscriptionNotification are used to remove the corresponding subscription.

## Operation CAPABILITY

This interface is called when the client asks for the service capabilities.

The purpose of the interface is to provide a dynamic method to ask a service instance at runtime what interfaces that are accessible, and what payload formats and versions that are valid.

### Operation Functionality

The operation responds with all operations and data products that the service offers.

### Operation Parameters

**GET baseUrl/v1/capability: response**

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 200 | application/json |  | OK |
| array of CapabilityObject { |  |  |
| dataProductType | integer |  |
| containerType | integer |  |
| productSchemaUrl | string |  |
| serviceVersion | string |  |
| implementedInterfaces  upload | boolean |  |
| implementedInterfaces  uploadLink | boolean |  |
| implementedInterfaces  get | boolean |  |
| implementedInterfaces  getByLink | boolean |  |
| implementedInterfaces  getSummary | boolean |  |
| implementedInterfaces  subscription | boolean |  |
| implementedInterfaces  access | boolean |  |
| implementedInterfaces  encryptionKey | boolean |  |
| } |  |  |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 23 Service response (complete list in RFC 9110)

### Interpretation guidelines

The response contains one object for each unique dataProductType that the service can provide.

For each dataProductType, the available operations are listed with (as?) true or false.

## Operation PING

This interface is called when the client checks the availability of the service.

The purpose of the interface is to provide a dynamic method to ask for the technical status of the specific service instance.

### Operation Functionality

The operation response with 200 to inform the user that the service is available and ready to serve.

### Operation Parameters

**GET baseUrl/v1/**ping**/ pathParam?queryParam : response**

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 200 | application/json |  | OK |
| message | string | Response message |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 24 Service Response (complete list in RFC 9110)

### Interpretation guidelines

No specific input or output are given from the operation. The HTTP response 200 shall be interpreted as the service is ready to serve the consumer.

## Operation ENCRYPTIONKEY

This interface is called when sending (pushing) an encryption key to a receiver.

### Operation Functionality

The operation receives an encryptionKey to be used for decryption of uploaded data that is encrypted; dataProtection = true.

The encryptionKey is protected with a Diffie\_hellman procedure where the secret key is created by pairing public key with private key and the iv. Only the actor holding the opposite pair of PrivateKey-A - PublicKey-B and PrivateKey-B – PublicKey-A can re-create the secret (key?) and decrypt the data encryptionKey.

### Operation Parameters

**POST baseUrl/v1/encryptionKey {body} : response**

Object with symmetric encryption key

|  |  |  |  |
| --- | --- | --- | --- |
| Body | Type | Format | Require |
| envelope  encryptionKey | string | Base64 | required |
| envelope  iv | string | Base64 | required |
| envelope  transactionIdentifier | string | uuid | optional |
| envelope  digitalSignatureValue  publicRootCertificateThumbprint | string | SHA1 and HEX | optional |
| envelope  digitalSignatureValue  publicCertificate | string | PEM | required |
| envelope  digitalSignatureValue  digitalSignature | string | HEX | required |
| envelope  envelopeSignatureCertificate | string | PEM | required |
| envelope  envelopeRootCertificateThumbprint | string | SHA1 and HEX | required |
| envelope  envelopeSignatureTime | string | date-time | required |
| envelopeSignature | string | HEX | required |

Table 25 Service body

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 200 | application/json |  | OK |
| message | string | Response message |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 26 Service response (complete list in RFC 9110)

### Interpretation guidelines

The encryptionKey and iv represent the encryptionKey.

The transactionIdentifier is a reference to what data the key can decrypt.

## Operation ENCRYPTIONKEY NOTIFICATION

This interface is called when asking for an encryption key to a receiver.

The purpose of this interface is to receive a notification request for an exchange of an encrypted secret key. The response is sent asynchronously through the consumer’s POST encryption key operation.

### Operation Functionality

The operation receives an encryptionKey event notification.

### Operation Parameters

**POST baseUrl/v1/encryptionKey/notify {body} : response**

EncryptionKey notification object

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Body | Type | Format | | Require |
| envelope  dataReference | string | | uuid | required |
| envelope  publicCertificate | string | | PEM | required |
| envelope  envelopeSignatureCertificate | string | | PEM | required |
| envelope  envelopeSignatureTime | string | | date-time | required |
| envelopeSignature | string | | HEX | required |

Table 27 Service body

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 202 | application/json |  | Request accepted |
| message | string | Response message |
| Secom\_ResponseCodeEnum | integer |  |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 28 Service response (complete list in RFC 9110)

### Interpretation guidelines

-

## Operation GET PUBLICKEY

This interface is called when the client gets (pulls) the public certificate(s) from the service.

The purpose of the interface is to request a public key.

### Operation Functionality

The operation is expected to return the requested public key, or the latest valid public key wrapped in X509 Certificate for the service.

REQUEST-RESPONSE pattern.

### Operation Parameters

**GET baseUrl/v1/PublicKey/pathParam?queryParam : response**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Param | REST | Type | Format | Require |
| dataProtection | queryParam | boolean |  | optional |
| certificateThumbprint | queryParam | string | SHA | optional |

Table 29 Service parameters

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 200 | application/json |  | OK |
| publicCertificate | string | PEM |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 403 | application/json | string | Forbidden |
| 404 | application/json | string | Not found |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 30 Service Response (complete list in RFC 9110)

### Interpretation guidelines

The dataProtection flag indicates as information whether the key is needed for an exchange of encryption key using Diffie-Hellman procedure or not.

The certificateThumbprint is a reference to what public key that is requested. If not provided, the latest valid public key wrapped in a X.509 Certificate is returned.

## Operation UPLOAD PUBLIC KEY

The purpose of the interface is to upload a consumers public key for e.g. exchanging an encryption key using Diffie-Hellman procedure.

### Operation Functionality

The operation receives a public key wrapped in X.509 Certificate.

The operation shall/should verify that the caller can be trusted.

### Operation Parameters

**POST baseUrl/v1/PublicKey { body } : response**

Public certificate x.509 in PEM format, Base64 encoded byte array.

|  |  |  |  |
| --- | --- | --- | --- |
| Body | Type | Format | Require |
| publicCertificate | string | PEM |  |

Table 31 Service body

|  |  |  |  |
| --- | --- | --- | --- |
| HTTP Code | Content | Type | Description |
| 200 | application/json |  | OK |
| 400 | application/json | string | Bad request |
| 401 | application/json | string | Unauthorized |
| 405 | application/json | string | Method not allowed |
| 500 | application/json | string | Internal server error |
| 501 | application/json | string | Not implemented |

Table 32 Service response (complete list in RFC 9110)

### Interpretation guidelines

The public key wrapped in publicCertificate.

# SERVICE TEMPLATE DYNAMIC BEHAVIOUR

**Reference: IEC 63173-2 SECOM v1.0.0 Clause 5.7 SECOM service interface definitions**

## Generic sequence for signatures

This sequence assumes the following actors:

Data Provider 🡪 Data Sender 🡪 Data Receiver 🡪 Data Consumer

The sequence describes the data from its source through transport using SECOM to the end-user.

Provider preparation:

* The data provider must have an identity in trusted PKI system
* The data provider creates a private-public key pair
* The data provider sends the public key to PKI system to get it signed and connected to a specific identity/identifier in the PKI system (Certificate Signing Request)
* The data sender must have an identity in trusted PKI system
* The data sender creates a private-public key pair
* The data sender sends the public key to PKI system to get it signed and connected to a specific identity/identifier in the PKI system (Certificate Signing Request)

Data Provider and Data Sender:

* The data provider selects identifier and certificate to use for the data signing
* The data provider creates a signature for the data
* The data provider transfers the data with its signature to the data sender.
* The data sender protects the data transferring by the selected communication mechanism, e.g. SECOM. SECOM Upload wraps the data, the signature and metadata into an envelope. The envelope is signed by the data sender.
* The data sends contacts the data receiver and transfers the data

Data Receiver and Data Consumer

* The data receiver receives the transferred object with envelope. The envelope signature is verified.
* The data sender forwards the data and data signature to the data consumer
* The data consumer verified the data signature

# DEFINITIONS

The definitions of terms used in this IALA Guideline can be found in the International Dictionary of Marine Aids to Navigation (IALA Dictionary) at <http://www.iala-aism.org/wiki/dictionary> and were checked as correct at the time of going to print. Where conflict arises, the IALA Dictionary shall be considered as the authoritative source of definitions used in IALA documents.

## Terminology

Persons producing the Technical Service are invited to add definitions to the following list as appropriate.

1. Definition of terminology – Technical Service

| Term | Definition |
| --- | --- |
| **External Data Model** | Describes the semantics of the ‘maritime world’ (or a significant part thereof) by defining data structures and their relations. This could be at logical level (e.g. in UML) or at physical level (e.g. in XSD schema definitions), as for example standard data models, or S-100 based data produce specifications. |
| **Message Exchange Pattern** | Describes the principles two different parts of a message passing system (in our case: the service provider and the service consumer) use to interact and communicate with each other. Examples:  In the Request/Response MEP, the service consumer sends a request to the service provider to obtain certain information; the service provider provides the requested information in a dedicated response.  In the Publish/Subscribe MEP, the service consumer establishes a subscription with the service provider to obtain certain information; the service provider publishes information (either in regular intervals or upon change) to all subscribed service consumers. |
|  |  |

# ACRONYMS

Persons producing the Technical Service are invited to provide a list of acronyms as appropriate.

|  |  |
| --- | --- |
| **REST** | REpresentational State Transfer |
| **SECOM** | Secure Communication |
|  |  |

# REFERENCES

1. IALA Guidelineline 1128 11?? on Specification of e-Navigation Technical Services
2. IEC 63173-2 SECOM v1.0.0
3. IHO S-100 ed 5.0.0
4. NIST Digital Signature Standard (DSS–FIPS Publication 186)
5. RFC 9110 HTTP Semantics

# APPENDIX - SERVICE DESIGN TEMPLATE OpenAPI (Swagger) in JSON

