

From: IALA
To: IEC

C80-12.3.2 rev1

LIAISON NOTE

Request to use elements of IEC 63173-2 in IALA document

1 INTRODUCTION

IALA has made a guideline for creating technical service specifications. These services can be using many different technologies, SECOM (IEC 63173-2), being one. In order to lower the barrier for creating these rather extensive specifications, IALA wishes to provide a template to be used, when making specifications - again using specific technologies. The annex to this liaison note is a draft template for making a specification using SECOM. As evident in the annex, this template uses text and drawings from the SECOM standard - in order to ease the writing of the specifications. Thus IALA seeks approval from IEC to use these elements from SECOM in a IALA document.

2 ACTION REQUESTED

IALA seeks acceptance from IEC to use elements from IEC 63173-2, as evident in the annex.

Annex

Service Design Template for SECOM Service

Contents

LIAISON NOTE	1
Request to use elements of IEC 63173-2 in IALA document.....	1
1 INTRODUCTION.....	1
2 ACTION REQUESTED	1
Annex	2
Service Design Template for SECOM Service	2
1 INTRODUCTION.....	7
1.1 General.....	7
1.2 Intended Readership.....	7
1.3 Inputs from Other Sources	7
2 SERVICE DESIGN TEMPLATE IDENTIFICATION.....	7
3 TECHNOLOGY INTRODUCTION.....	8
3.1 General.....	8
3.2 Service technology and service transportation protocol	8
3.3 Security.....	8
3.3.1 Communication channel security.....	8
3.3.2 Data protection.....	8
3.3.3 Data Signature	9
3.3.4 Data Encryption.....	9
4 SERVICE DESIGN TEMPLATE OVERVIEW.....	9
4.1 General.....	9
4.2 Service Template Interfaces	9
4.3 Template User Instructions (PERHAPS MOVE SECTION)	11
5 PHYSICAL SERVICE TEMPLATE DATA MODEL	11
6 SERVICE TEMPLATE INTERFACE DESIGN.....	11
6.1 Service Interface SECOM REST.....	12
6.2 Operation UPLOAD	12
6.2.1 Operation Functionality	12
6.2.2 Operation Parameters	12
6.2.3 Interpretation guidelines	12
6.3 Operation ACKNOWLEDGEMENT	13
6.3.1 Operation Functionality	13
6.3.2 operation Parameters.....	13
6.3.3 Interpretation guidelines	13
6.4 Operation GET.....	13
6.4.1 Operation Functionality	13
6.4.2 Operation Parameters	13
6.4.3 Interpretation guidelines	14

6.5	Operation UPLOAD LINK	14
6.5.1	Operation Functionality	14
6.5.2	Operation Parameters	14
6.5.3	Interpretation guidelines	14
6.6	Operation GET BY LINK	15
6.6.1	Operation Functionality	15
6.6.2	Operation Parameters	15
6.6.3	Interpretation guidelines	15
6.7	Operation GET SUMMARY	15
6.7.1	Operation Functionality	15
6.7.2	Operation Parameters	15
6.7.3	Interpretation guidelines	16
6.8	Operation ACCESS	16
6.8.1	Operation Functionality	16
6.8.2	Operation Parameters	16
6.8.3	Interpretation guidelines	16
6.9	Operation ACCESS NOTIFICATION	17
6.9.1	Operation Functionality	17
6.9.2	Operation Parameters	17
6.9.3	Interpretation guidelines	17
6.10	Operation SUBSCRIPTION	17
6.10.1	Operation Functionality	17
6.10.2	Operation Parameters.....	17
6.10.3	Interpretation guidelines.....	17
6.11	Operation SUBSCRIPTION NOTIFICATION	18
6.11.1	Operation Functionality	18
6.11.2	Operation Parameters.....	18
6.11.3	Interpretation guidelines.....	18
6.12	Operation REMOVE SUBSCRIPTION	18
6.12.1	Operation Functionality	18
6.12.2	Operation Parameters.....	18
6.12.3	Interpretation guidelines.....	19
6.13	Operation CAPABILITY	19
6.13.1	Operation Functionality	19
6.13.2	Operation Parameters.....	19
6.13.3	Interpretation guidelines.....	19
6.14	Operation PING	19

6.14.1	Operation Functionality	19
6.14.2	Operation Parameters.....	19
6.14.3	Interpretation guidelines.....	19
6.15	Operation ENCRYPTIONKEY	19
6.15.1	Operation Functionality	20
6.15.2	Operation Parameters.....	20
6.15.3	Interpretation guidelines.....	20
6.16	Operation ENCRYPTIONKEY NOTIFICATION.....	20
6.16.1	Operation Functionality	20
6.16.2	Operation Parameters.....	20
6.16.3	Interpretation guidelines.....	20
6.17	Operation GET PUBLICKEY	20
6.17.1	Operation Functionality	20
6.17.2	Operation Parameters.....	20
6.17.3	Interpretation guidelines.....	21
6.18	Operation UPLOAD PUBLIC KEY.....	21
6.18.1	Operation Functionality	21
6.18.2	Operation Parameters.....	21
6.18.3	Interpretation guidelines.....	21
7	SERVICE TEMPLATE DYNAMIC BEHAVIOUR.....	21
7.1	Generic sequence for signatures	21
7.2	Dynamic Behavior of the Service Interface	22
7.2.1	Upload interface	22
7.2.2	Upload Link interface.....	24
7.2.3	Get Interface	25
7.2.4	Get By Link interface	27
7.2.5	Request Access interface.....	27
7.2.6	Subscription interface.....	28
7.2.7	EncryptionKey interface.....	29
7.2.8	Public Key Interface	30
8	DEFINITIONS	30
8.1	Terminology	30
9	ACRONYMS	31
10	REFERENCES.....	31
11	APPENDIX - SERVICE DESIGN TEMPLATE OpenAPI (Swagger) in JSON	32
12	APPENDIX – Guidance on Implementation	32
12.1	Example on shore Service Deployment	32
12.2	Example on Ship – SECOM Service outside Ship.....	33

12.3	Example on Ship – Consuming SECOM Service	33
12.4	Example on Ship – Service Onboard	34

1 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.

1.1 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.

1.2 Intended Readership

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

1.3 Inputs from Other Sources

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

2 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.

Table 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

3 TECHNOLOGY INTRODUCTION

3.1 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.

3.2 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).

3.3 Security

3.3.1 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 1.2 (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.

3.3.2 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.

3.3.3 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-384-SHA2 but the design supports also other algorithms.

The signature is transported in HEX.

3.3.4 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.

4 SERVICE DESIGN TEMPLATE OVERVIEW

4.1 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.

4.2 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.
-----------	--	---

4.3 Template User Instructions (PERHAPS MOVE SECTION)

This document is intended to be used 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 can be omitted and the HTTP return code should be HTTP 405 Method not allowed or HTTP 501 Not implemented.

Additions to the SECOM interface 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 are to be defined by the service provider, and the operation path, such as “/v1/object”, is defined by the SECOM standard and shall not be change.

5 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 in this document.

6 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.

6.1 Service Interface SECOM REST

6.2 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.

6.2.1 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.

6.2.2 Operation Parameters

POST [baseUrl]/v1/object {body} : response

Input is the data (payload) and its metadata wrapped and signed in the envelope.

For details, see 0

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

6.2.3 Interpretation guidelines

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

The data content 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 an encryptionKey is needed

Data signature shall be provided in a one-line HEX format using algorithm stated in dataSignatureReference.

The value for protectionScheme is dynamically set based on the scheme administrator used for the data signing.

Exchange information

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

Envelope signature

The envelope signature shall be made on a dot (.) separated “CSV” data structure of the envelope in the exact same order as the fields are defined in the table (OpenAPI):

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)

The envelope is signed using the ECDSA-384-SHA2 algorithm.

6.3 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.

6.3.1 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.

6.3.2 operation Parameters

POST baseUrl/v1/acknowledgement {body} : response

Input is the acknowledgement object with reference to information and time when delivered, wrapped in envelope and signed.

For details, see 0

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

6.3.3 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.

6.4 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.

6.4.1 Operation Functionality

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

Request-Response pattern.

6.4.2 Operation Parameters

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

Input is the filtering parameters for the wanted data and data formats.

Output is none or many base64 encoded data objects matching the filter.

For details, see 0

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

6.4.3 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 containerType 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.

6.5 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.

6.5.1 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.

6.5.2 Operation Parameters

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

The input is the link to the data.

For details, see 0

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

6.5.3 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.

6.6 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.

6.6.1 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.

6.6.2 Operation Parameters

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

Input is the filter parameters.

For details, see 0

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

6.6.3 Interpretation guidelines

The unique transactionIdentifier given by the Upload By Link call.

6.7 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.

6.7.1 Operation Functionality

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

6.7.2 Operation Parameters

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

Input is the filtering parameters for the wanted data and data formats.

Output is a list of data information matching the filter.

6.7.3 Interpretation guidelines

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

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_lastModifiedData 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.

6.8 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.

6.8.1 Operation Functionality

The operation checks the availability for the caller to requested information.
Request-Callback pattern. (explain)

6.8.2 Operation Parameters

POST baseUrl/v1/access {body} : response

Input is the reason for requesting access to information.

Output is given in an asynchronous callback to Access Notification.

For details, see 0

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

6.8.3 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.

6.9 Operation ACCESS NOTIFICATION

This interface is called as callback response to interface Access.

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

6.9.1 Operation Functionality

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

6.9.2 Operation Parameters

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

Input is the decision from the request for access; True or False and a reason.

For details, see 0

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

6.9.3 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.

6.10 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.

6.10.1 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 (SubscriptionNotification) and Publish-Subscribe pattern (Upload).

6.10.2 Operation Parameters

POST baseUrl/v1/ subscription {body} : response

Input is filtering parameters for the requested subscription.

Output is a subscription identifier, if successful.

For details, see 0

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

6.10.3 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.

6.11 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.

6.11.1 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.

6.11.2 Operation Parameters

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

Input is the subscription identifier and type of event; Create or Delete.

For details, see 0

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

6.11.3 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.

6.12 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.

6.12.1 Operation Functionality

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

6.12.2 Operation Parameters

DELETE baseUrl/v1/ subscription {body} : response

Input is the subscription identifier.

For details, see 0

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

6.12.3 Interpretation guidelines

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

6.13 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.

6.13.1 Operation Functionality

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

6.13.2 Operation Parameters

GET baseUrl/v1/capability: response

Output is lists of all capabilities the service has implemented.

For details, see 0

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

6.13.3 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.

6.14 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.

6.14.1 Operation Functionality

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

6.14.2 Operation Parameters

GET baseUrl/v1/ping/ pathParam?queryParams : response

No input.

Output is the service HTTP response.

For details, see 0

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

6.14.3 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.

6.15 Operation ENCRYPTIONKEY

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

6.15.1 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.

6.15.2 Operation Parameters

POST baseUrl/v1/encryptionKey {body} : response

Input is the protected symmetric encryption key.

For details, see 0

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

6.15.3 Interpretation guidelines

The encryptionKey and iv represent the encryptionKey.

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

6.16 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.

6.16.1 Operation Functionality

The operation receives an encryptionKey event notification.

6.16.2 Operation Parameters

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

Input is reference to the encrypted data.

For details, see 0

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

6.16.3 Interpretation guidelines

6.17 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.

6.17.1 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.

6.17.2 Operation Parameters

GET baseUrl/v1/PublicKey/pathParam?queryParams : response

Input is filtering parameters for the requested public certificate.

Output is zero or one specific public leaf certificate according to filtering.

For details, see 0

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

6.17.3 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.

6.18 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.

6.18.1 Operation Functionality

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

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

6.18.2 Operation Parameters

POST baseUrl/v1/PublicKey { body } : response

The input is PEM file.

For details, see 0

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

6.18.3 Interpretation guidelines

The public key wrapped in publicCertificate.

7 SERVICE TEMPLATE DYNAMIC BEHAVIOUR

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

7.1 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 sender contacts the data receiver and transfers the data

Data Receiver and Data Consumer

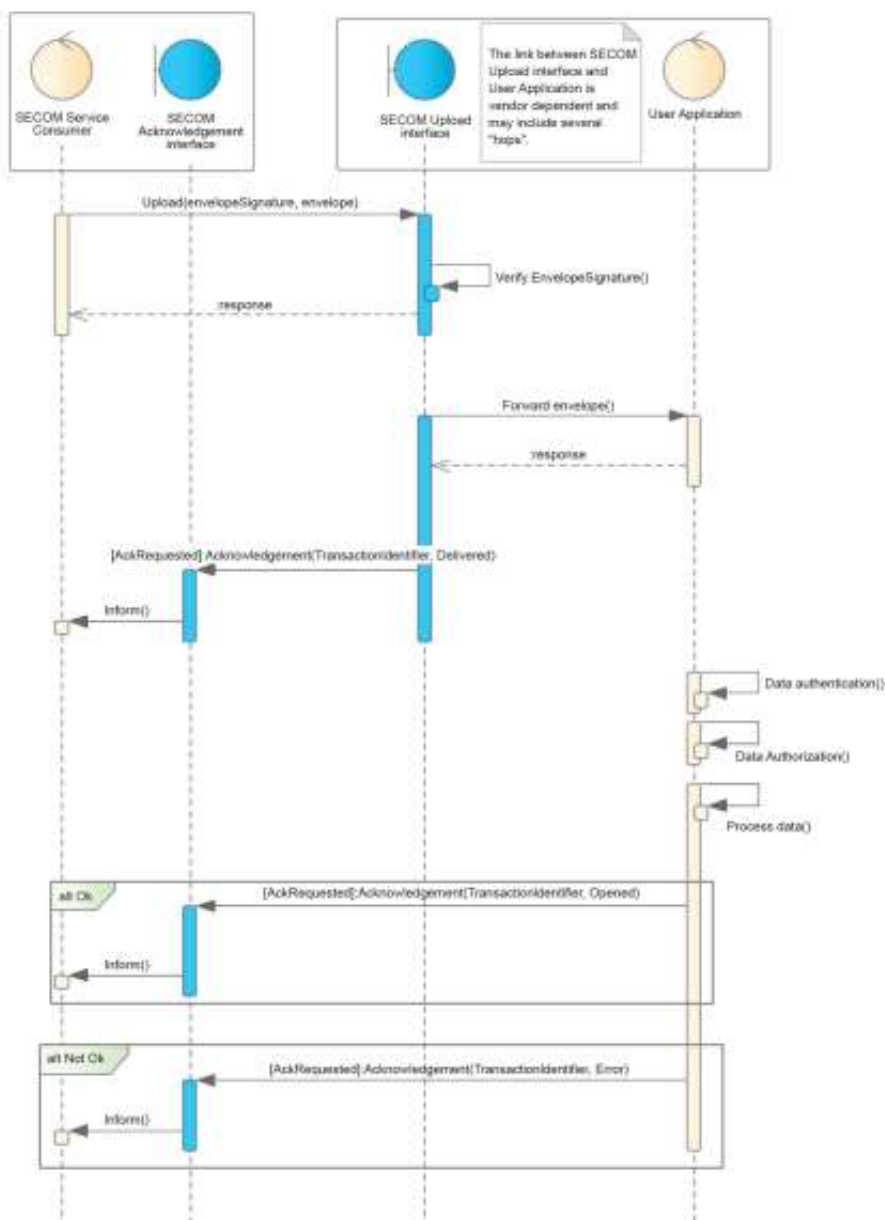
- The data receiver receives the transferred object with envelope. The envelope signature is verified.
- The data receiver forwards the data and data signature to the data consumer
- The data consumer verifies the data signature

7.2 Dynamic Behavior of the Service Interface

The following chapters contain sequence diagrams for the service interfaces.

7.2.1 Upload interface

Reference: IEC 63173-2 SECOM v1.0.0 Clause 5.7.2.4 Dynamic behavior



IEC

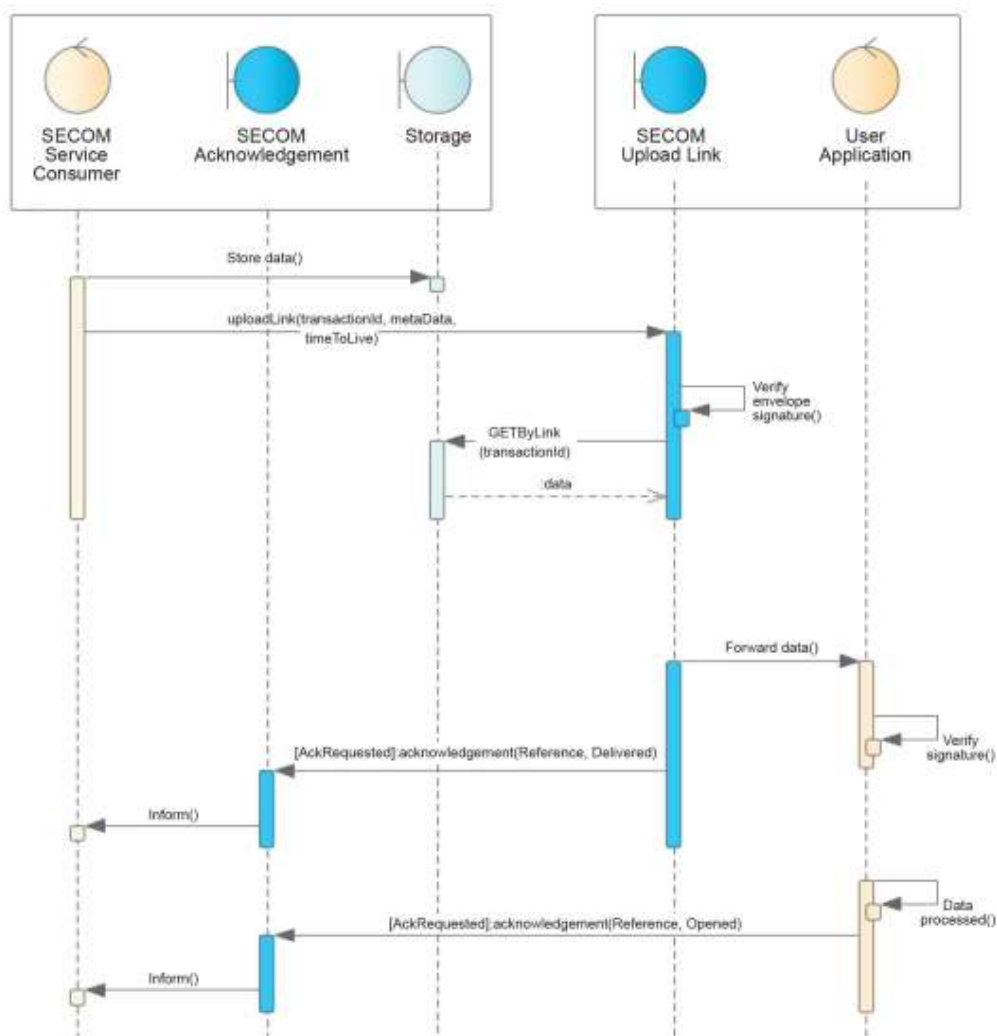
Figure 1 Upload data interface

7.2.2 Upload Link interface

Reference: IEC 63173-2 SECOM v1.0.0 Clause 5.7.3.4 Dynamic behavior

IEC FDIS 63173-2 © IEC 2022

– 51 –

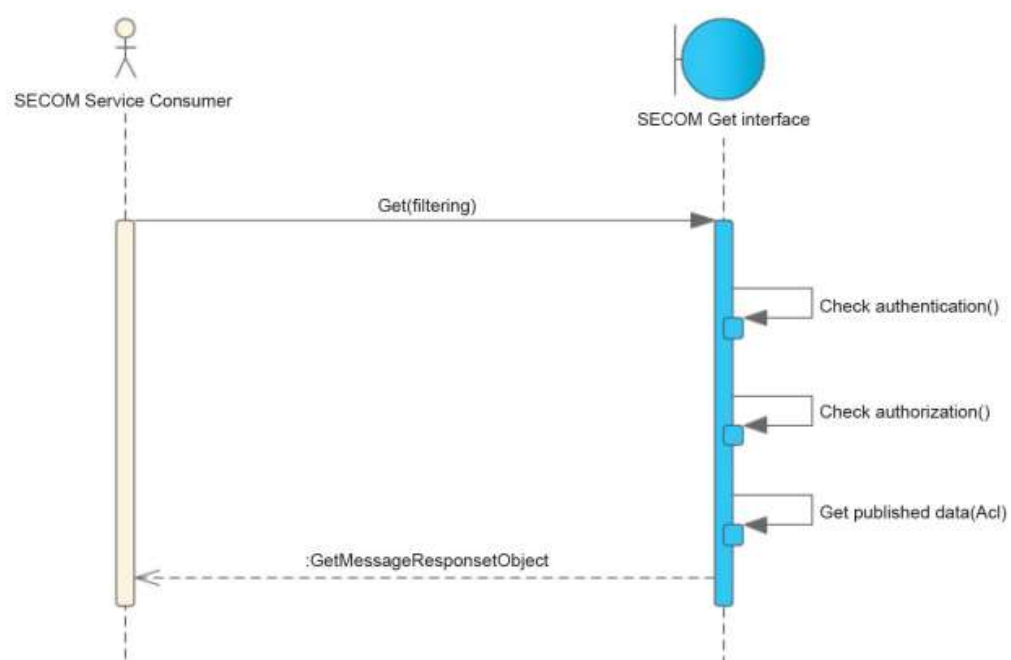


IEC

Figure 2 Upload link interface

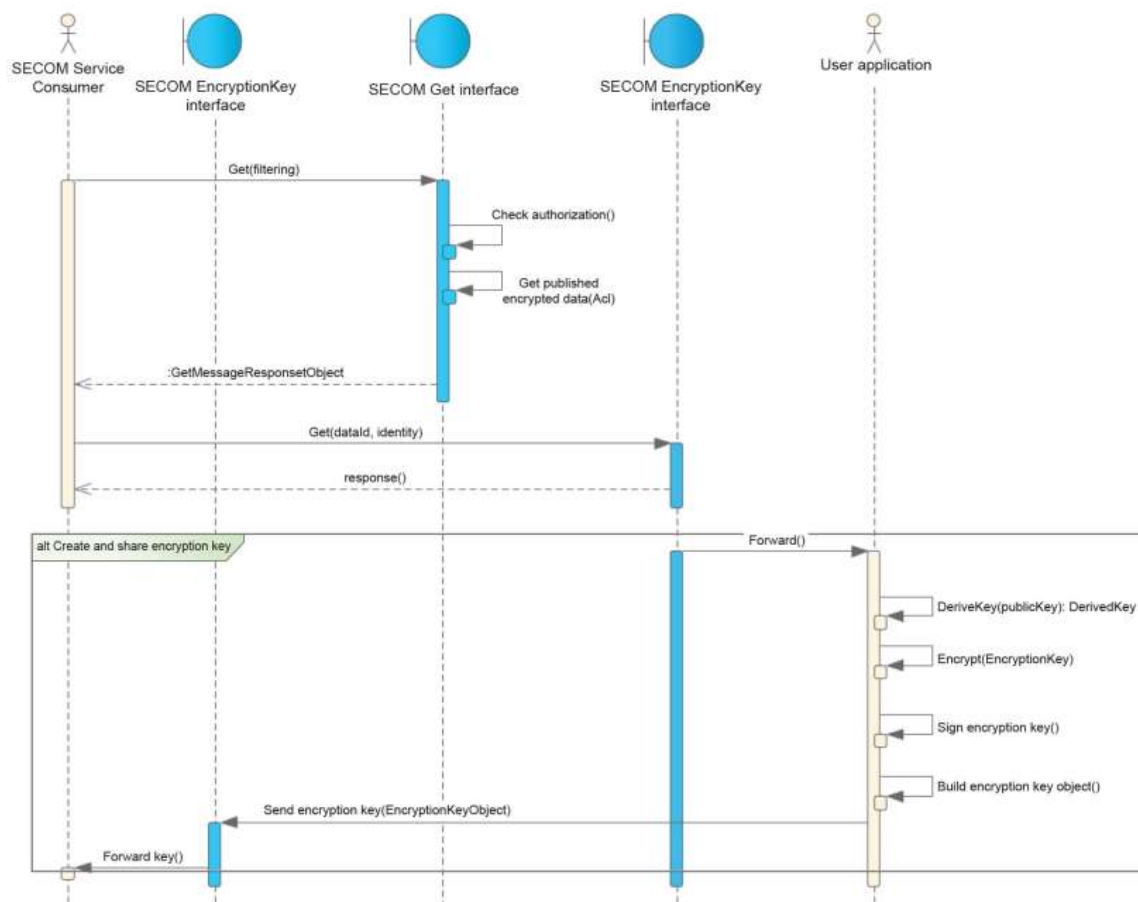
7.2.3 Get Interface

Reference: IEC 63173-2 SECOM v1.0.0 Clause 5.7.5.4 Dynamic behavior



IEC

Figure 3 Get data interface

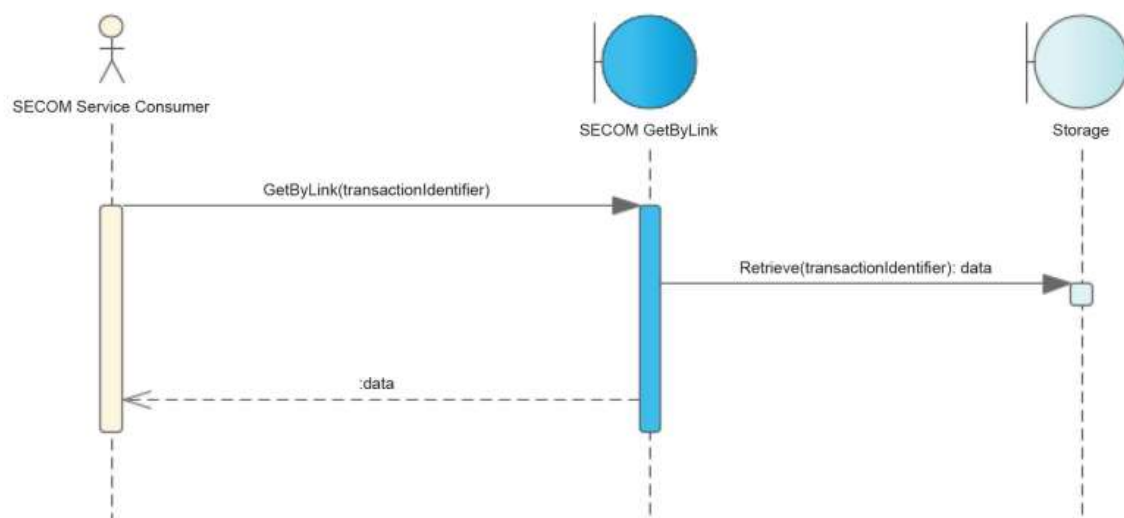


IEC

Figure 4 *Get encrypted data interface*

7.2.4 Get By Link interface

Reference: IEC 63173-2 SECOM v1.0.0 Clause 5.7.7.4 Dynamic behavior

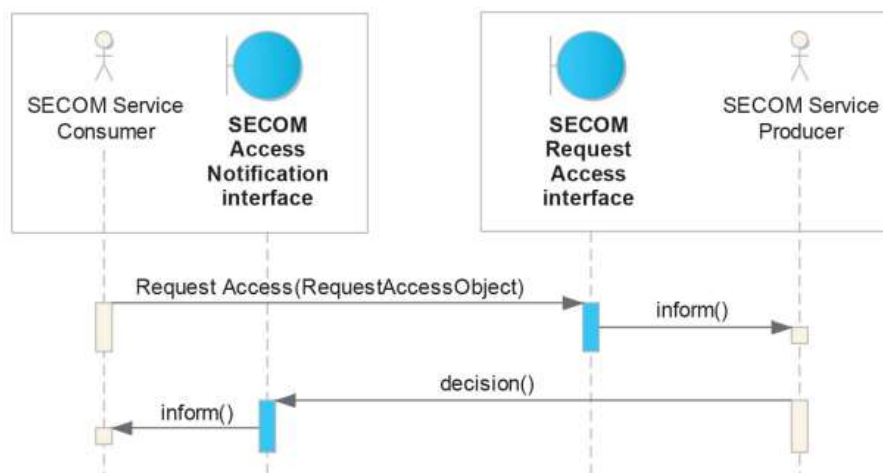


IEC

Figure 5 Get by link interface

7.2.5 Request Access interface

Reference: IEC 63173-2 SECOM v1.0.0 Clause 5.7.8.4 Dynamic behavior



IEC

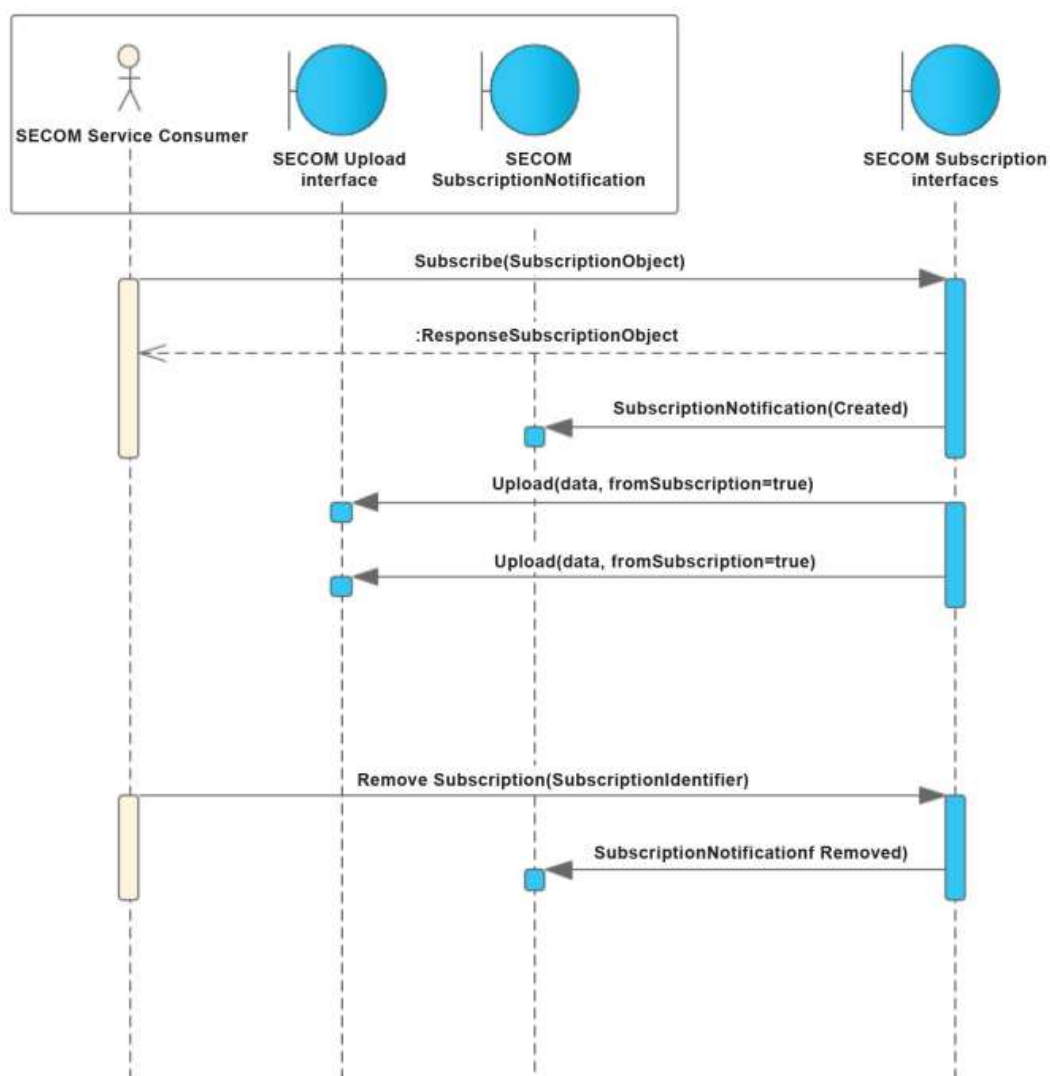
Figure 6 Access request interface

7.2.6 Subscription interface

Reference: IEC 63173-2 SECOM v1.0.0 Clause 5.7.10.4 Dynamic behavior

– 76 –

IEC FDIS 63173-2 © IEC 2022



IEC

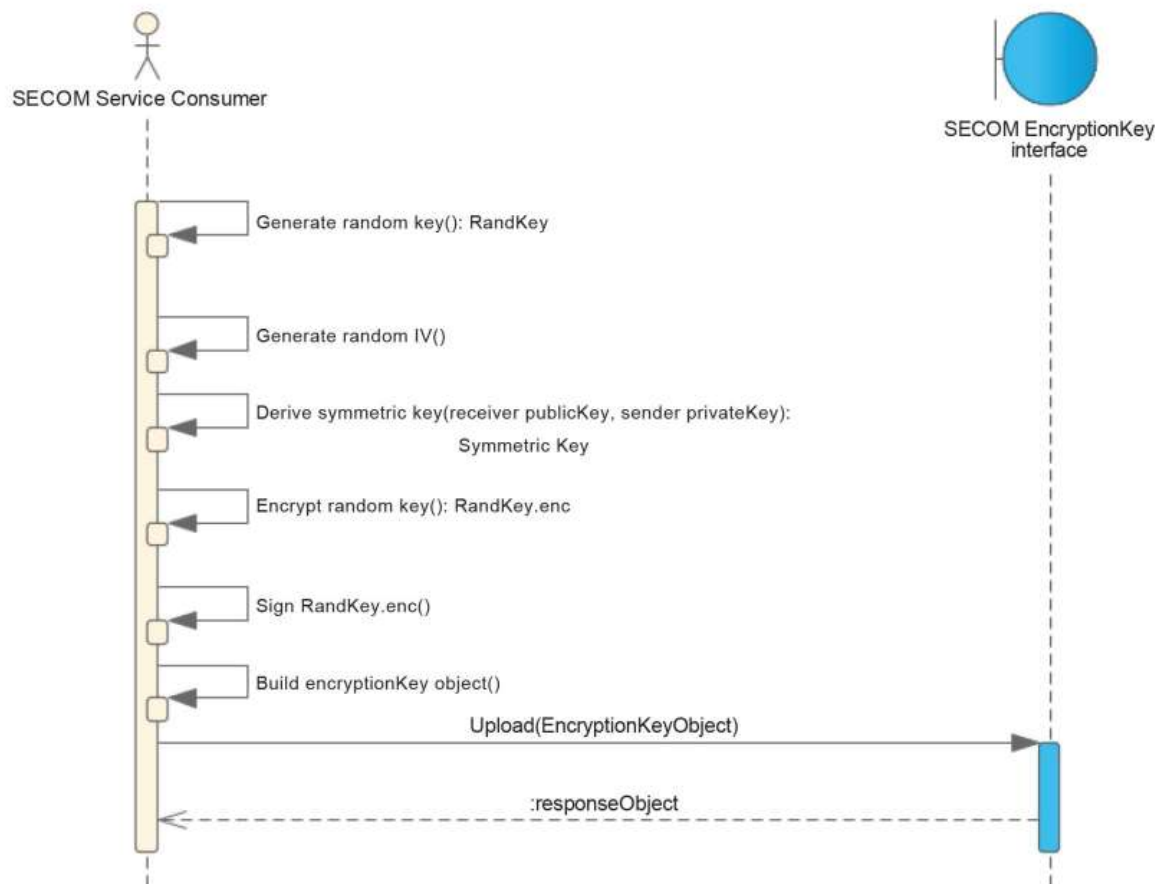
Figure 7 Subscription interface

7.2.7 EncryptionKey interface

Reference: IEC 63173-2 SECOM v1.0.0 Clause 5.7.8.4 Dynamic behavior

IEC FDIS 63173-2 © IEC 2022

– 91 –



IEC

Figure 8 EncryptionKey interface

7.2.8 Public Key Interface

Reference: IEC 63173-2 SECOM v1.0.0 Clause 5.7.16.4 Dynamic behavior

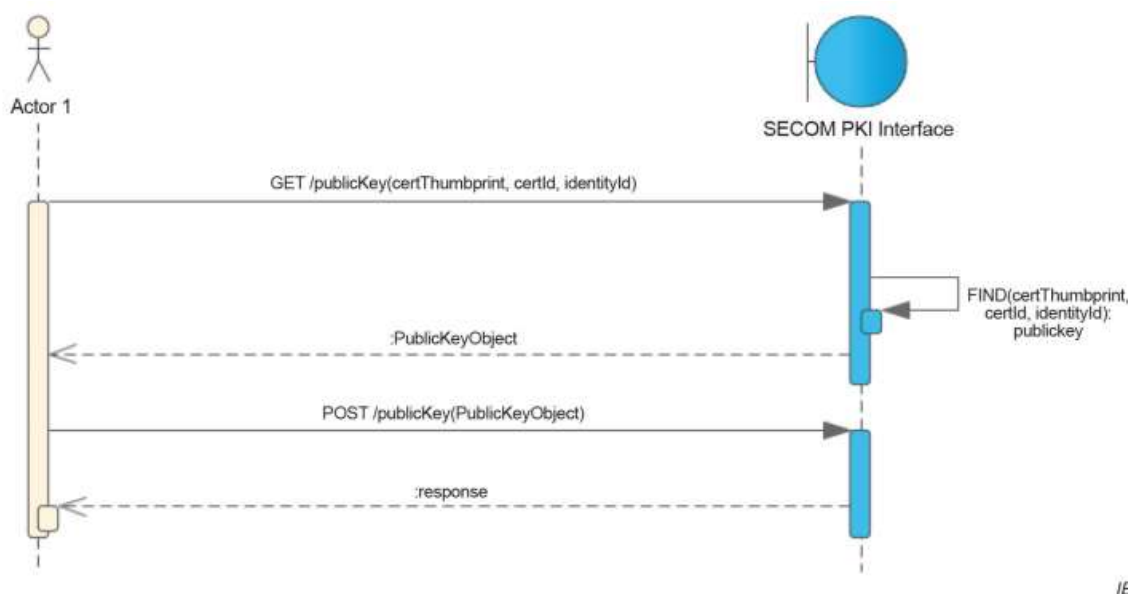


Figure 9 Public Key interface

8 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.

8.1 Terminology

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

Table 2 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.

Term	Definition
Message Exchange Pattern	<p>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:</p> <p>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.</p> <p>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.</p>

9 ACRONYMS

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

REST	REpresentational State Transfer
SECOM	Secure Communication

10 REFERENCES

- [1] IALA Guideline 1128 on Specification of e-Navigation Technical Services
- [2] IEC 63173-2 SECOM v1.0.0
- [3] SECOM Service Interface OpenAPI (<https://cirm.org/secom>)
- [4] IHO S-100 ed 5.2.0
- [5] NIST Digital Signature Standard (DSS–FIPS Publication 186)
- [6] RFC 9110 HTTP Semantics

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

The included OpenAPI JSON file can also be found in ref [3].



Service Design
Template - SECOM Se

12 APPENDIX – Guidance on Implementation

The following pictures are copied from ref [2].

12.1 Example on shore Service Deployment

SECOM Service deployed close to shoreside application.

Reference: IEC 63173-2 SECOM v1.0.0 Appendix D.3

– 174 –

IEC FDIS 63173-2 © IEC 2022

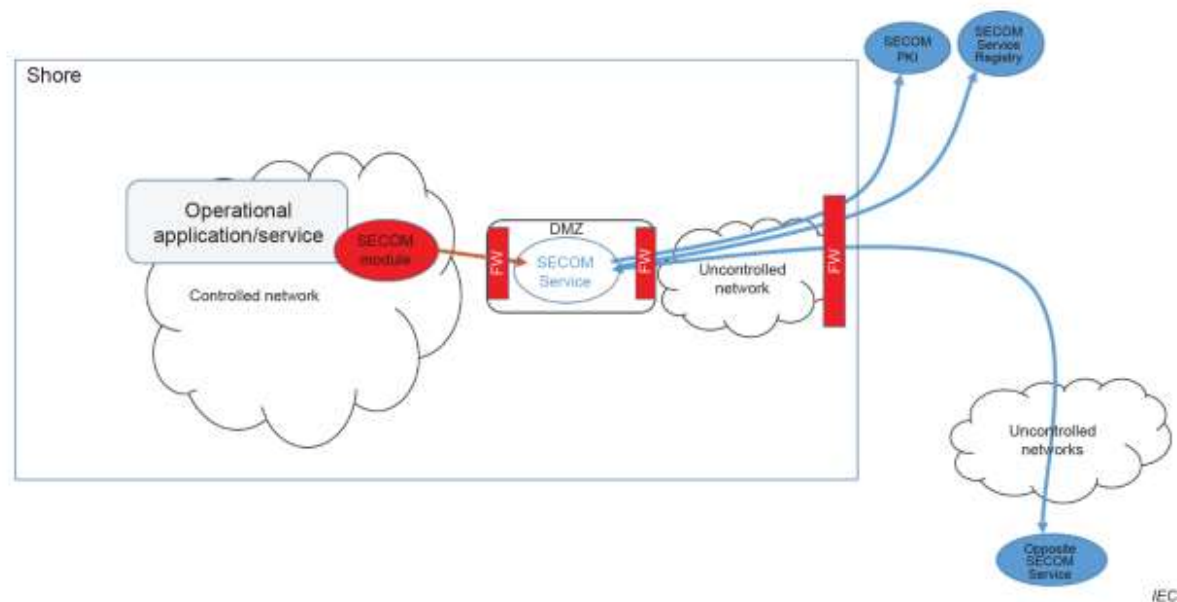


Figure 10 Deployment example for SECOM on shore

12.2 Example on Ship – SECOM Service outside Ship

SECOM Service deployed outside ship on stable network and polled at time intervals from shipside system.

Reference: IEC 63173-2 SECOM v1.0.0 Appendix D.2

IEC FDIS 63173-2 © IEC 2022

– 173 –

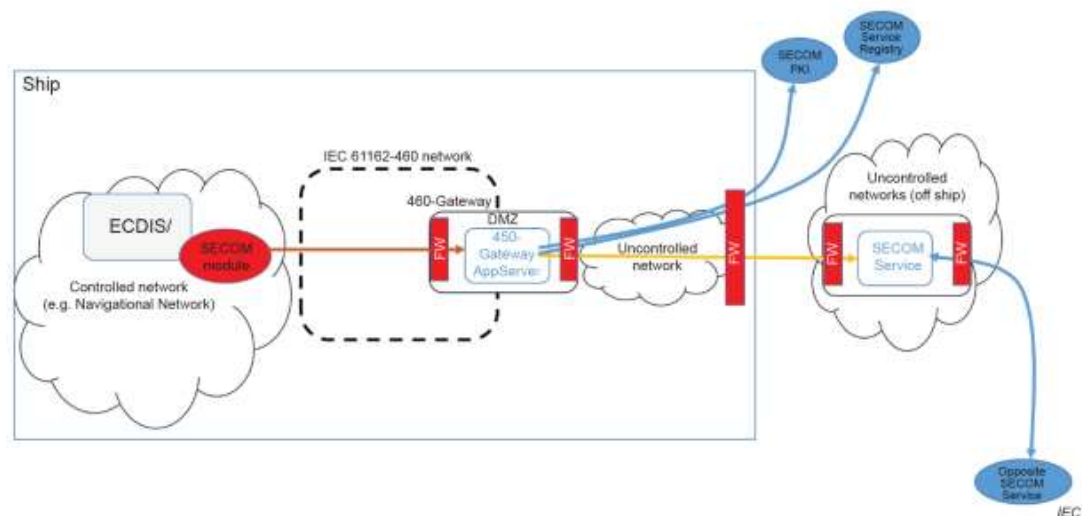


Figure 11 Deployment example for SECOM on ship

12.3 Example on Ship – Consuming SECOM Service

Shipside system consumes other SECOM Services directly from shipside system.

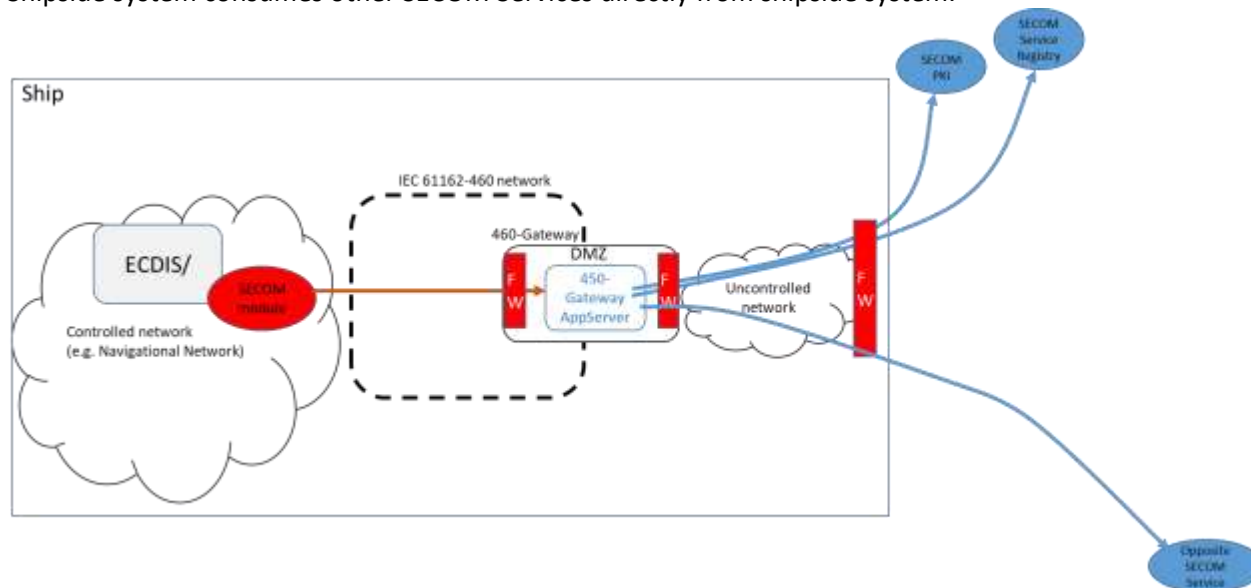


Figure 12 Deployment example for SECOM on ship

