SERVICE TECHNICAL DESIGN TEMPLATE

Service Design Description for the Route Exchange Service

<*xyz Technology*>

# INTRODUCTION

This design is intended to support all of the different Service Specifications that will be create where Route Exchange is involved.

The data product exchanged in this design is S-421 Route Plan.

The service can be part of a service orchestrated architecture.

## Purpose of the Document

The purpose is to describe the design of service specification as a REST service that facilitates development of service instances.

## Intended Readership

This service design description document is intended to be read by service architects, designers, system engineers and developers in charge of designing and developing an instance of the <XYZ> service.

Furthermore, this service design description is intended to be read by service architects, information architects, system engineers and developers in pursuing architecting, design and development activities of other related services.

## Inputs from Other Sources

This section provides an overview of activities, which are dealing with similar topics and lists already finished ones that provided inputs to this activity.

# Service Design 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 | Route Exchange REST Service Design |
| ID | Urn:mrn:iala:techsvc:design:routeexchange |
| Version | 0.0.1 |
| Technology | REST |
| Service Specification ID | Reference to the service specification |
| Service Specification Version | Reference to the service specification |
| Description | Service design to exchange route plans |
| Keywords | Route Plan, Route Exchange |
| Architect(s) | IALA TG 2. |
| Status | Provisional |

# TECHNOLOGY INTRODUCTION

## Service technology and service transportation protocol

[Soure IEC 63173-1]

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

REST is an architectural style, and an approach to communications that is often used in the development of Web services. The use of REST in SECOM is preferred over other more heavyweight protocols such as e.g. SOAP (Simple Object Access Protocol) because REST does not leverage as much bandwidth, which makes it a better fit for use in communication between vessels and shore based representation of the same.

REST, which typically runs over HTTP (Hypertext Transfer Protocol), has several architectural constraints:

* Decoupling – Decouples consumers from producers which suits SECOM decentralized architecture well.
* Stateless existence – Also a good prerequisite for a decentralized architecture design.
* Able to leverage a cache – Probably less important in SECOM since most of the interaction is between machines, although for services with man-machine interfaces this is of importance.
* Leverages a layered system – SECOM is dependent on good scaling capabilities which REST supports.
* Leverages a uniform interface – Again since SECOM defines the available services centrally in one or several Service registry(s) this constraint supports implementations being decoupled from the services they provide.

The definition of each operation defines additional error messages that the operation specifically shall respond with as a complement to the HTTP response codes defined by HTTP/1.1 (RFC 7231).

## Security

### Communication channel security

[Source IEC 63173-1]

When consuming service instances according to SECOM, the internet transport shall be protected with TLS and valid certificates from trusted party as stated in SECOM communication channel security. The protection of the channel is a complement to the protection of the data itself and is necessary to be included to secure also other service requests, such as subscription request, access request and notifications. The SECOM communication channel security describes the usage of certificate obtained from a trusted identity registry and thereby enables authentication on the service interaction itself as depicted in Figure 2.

Graphical user interface

Description automatically generated

Figure 2 - Secure communication channel

The SECOM communication channel security relies on a SECOM Public Key Infrastructure (SECOM PKI) or Public-Private Key management using X.509 Certificates to exchange the keys.

The SECOM communication channel security scheme does not comprise the “last mile” links between the SECOM information service interface and the end-user application. In Annex D there are informative examples and guidance of how such protection could be achieved.

### Data protection

<describe the possibility and the procedure to send encrypted route.>

Refer to S-100 Part 15-6 Data encryption

# SERVICE DESIGN OVERVIEW

## General

[Source IEC 63173-2 SECOM]

Table 15 gives an overview of the service interfaces that constitutes the SECOM information service interface. Service interfaces Capability and Ping shall be supported while the remaining interfaces are optional to implement.

Table 15 - Service interfaces overview

|  |  |  |
| --- | --- | --- |
| Service Interface | Exchange Pattern | Definition |
| Upload | ONE\_WAY | Interface to send (push) information to consumer |
| Acknowledgement | ONE\_WAY | Interface to send acknowledgement on uploaded information. |
| Get | REQUEST\_RESPONSE | Interface to ask for (pull) information from provider |
| Get Summary | REQUEST\_RESPONSE | Interface to ask for (pull) an information list from provider |
| Subscription | PUBLISH\_SUBSCRIBE | Interface to create subscription of information |
| Remove Subscription | ONE\_WAY | Interface to remove subscription |
| Subscription Notification | ONE\_WAY | Interface for notification from subscription events |
| Access | REQUEST\_CALLBACK | Interface to ask for access to information |
| Access Notification | ONE\_WAY | Interface for notification from access request |
| Capability | REQUEST\_RESPONSE | Interface to ask for the interface capabilities. Mandatory to implement. |
| Ping | REQUEST\_RESPONSE | Interface to check status on the service instance. Mandatory to implement. |
| Encryption Key | ONE\_WAY, REQUEST\_CALLBACK | Interface to securely send symmetric key for data encryption |
| PublicKey | ONE\_WAY,  REQUEST\_RESPONSE | Interface to request (pull) and send (push) a public certificate |

## Service Interfaces

[Source IEC 63173-2 SECOM v1.0.0 Clause 5 SECOM Information service interface]

To be discussed: The REST API is described in IEC 63173-2 SECOM. The princip to discuss is how much information is expected in this design document?

Shall it be possible to take this design alone and make a service implementation without having the SECOM document? Or is it expected that the SECOM document is available?

Is there a benefit of just take a version of SECOM and handle it further in this version of a service design? Keep the reference to SECOM for the rationale?

TODO: Make a table with interfaces and describe them applied for S-421 Route Plan

### Service interface – Upload

**Specification**

The purpose with this interface is to upload (push) information that shall not be larger than a maximum size of 350kb (Base64 encoded) to an information consumer. An information consumer shall implement this interface in order to receive information while an information provider may implement it.

When uploading (sending) information, acknowledgement can be requested. See interface Acknowledgement for the details of sending acknowledgement.

Uploading (sending) of information can either be within an active subscription of information or uploaded (sent) once as a single message as indicated by the attribute fromSubscription.

The data can be either XML, compressed XML or encrypted binary, typically compressed, as described in the exchange metadata object. The data can contain either one message, such as an S-421 Route Plan (IEC 63173-1) dataset, or a set of messages (files) compressed in a ZIP container and described by an ExchangeSet as described in S-100. The data is encoded as one row Base64 before transferred.

Attached to the data there is a set of metadata describing the protection of the data. The metadata refers to the content in the data attribute, independent of its type. For exchange of an S-100 DataSet this means that the dataset might be in XML (compressionFlag=false) or in ZIP (compressionFlag=true), or optionally compressed and encrypted (compressionFlag = true and dataProtection=true) and the data is signed (hash calculated and signed). For exchange of an S-100 ExchangeSet (exchange catalogue) the same procedure is used. The S-100 ExchangeCatalogue contains a set of files, the folder is compressed, and optionally encrypted and the data is signed (hash calculated and signed).

Figure 11 shows the interface in UML.

Diagram

Description automatically generated

Figure 11 - Upload interface UML diagram

### Service interface - Acknowledgement

### Service interface - Get

### Service interface - Get Summary

### Service interface - Access

### Service interface - Access Notification

### Service interface – Subscription

### Service interface - Remove Subscription

### Service interface - Subscription Notification

### Service interface - Capability

### Service interface - Ping

### Service interface – EncryptionKey

### Service interface – PublicKey

# PHYSICAL DATA MODEL

**[Source IEC 63173-1 S-421]**

Refer to URL for XSD file to S-421

Refer to IEC 63173-1 Route Plan

Diagram

Description automatically generated

1. <Service Name> Service Data Model diagram

An XML schema for this data model is included in the formal service design xml file attached in APPENDIX 1.

# SERVICE INTERFACE DESIGN

This section describes the details of each service interface. One sub-section is provided for each Service Interface.

The Service Interface design covers the static design description while the dynamic design (behaviour) is described in section E 7.

To be discussed: Add information here in this document or refer to IEC 63173-2?

## Service Interface Upload

[Source IEC 63173-1 v1.0.0 Clause ]

#### Upload REST design

##### General

The service interface and its data exchange model is defined with REST technology.

See Annex A for formal and detailed definition of the service interface as OpenAPI format.

##### Operation POST /object

The interface shall be used for uploading (pushing) data to a consumer. The operation expects one single data object and its metadata. The details are described in Table 18. For detailed description of data exchange model see clause 5.7.2.2.

Table 18 - REST implementation of Upload

|  |  |  |  |
| --- | --- | --- | --- |
| REST Operation | | | |
| POST URL/v1/object {body} : return | | | |
| REST Parameter (in) | REST Encoding | Mult | Definition |
| No parameters defined | n/a | n/a | n/a |
| REST Body (in) | REST Encoding | Mult | Definition |
| UploadObject | application/json | 1 | The data (payload) package with its metadata |
| Return (out) | REST Encoding | Mult | Definition |
| UploadResponseObject | application/json | 1 | Confirmation of upload or error message, including description of incorrect or missing values in payload for the specific service. |

#### Data exchange model

Table 16 and Table 17 describe the data exchanged in the interface.

Table 16 – Information input for Upload interface

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UploadObject | | | | |
| Attribute | Mult | Processing | Type | Definition |
| envelope | 1 | Mandatory | EnvelopeUploadObject | The complete EnvelopeObject being uploaded to receiver including data and message properties |
| envelopeSignature | 1 | Mandatory | CharacterString | The signature of the EnvelopeObject in HEX format without whitespace or linebreaks |
| EnvelopeUploadObject | | | | |
| **Attribute** | **Mult** | **Processing** | **Type** | **Definition** |
| data | 1 | Mandatory | Base64 | The payload XML dataProductType, base64 encoded (e.g. inside a S100\_ExchangeSet, S100\_DataSet), ZIP or binary. The data can be open, protected and/or compressed. |
| containerType | 1 | Mandatory | ContainerTypeEnum | Container type of message in the data object  Table 7 shows the values of the enum. |
| dataProductType | 1 | Mandatory | SECOM\_DataProductType.Name | The name column of the SECOM\_DataProductType in Table 8 |
| exchangeMetadata | 1 | Mandatory | SECOM\_ExchangeMetadataObject | The exchange metadata contains information regarding protection scheme, compression, signature and claimed identity according to Table 4 |
| fromSubscription | 1 | Optional | Boolean | Flag to indicate whether the data has been uploaded within an active subscription or not. |
| ackRequest | 1 | Mandatory | AckRequestEnum | Flag to indicate that acknowledgement is expected to be returned when the data is delivered to the end user, and/or when the content of the data is processed (opened) by the end user. |
| transactionIdentifier | 1 | Mandatory | UUID | Transaction identifier to be used e.g. in acknowledgement |
| envelopeSignatureCertificate | 1 | Mandatory | CharacterString | The public certificate of the sender. used to verify the envelopeObject signature |
| envelopeRootCertificateThumbprint | 1 | Optional | CharacterString | Claimed Thumbprint for Signed Root Key (X.509 Certificate) |
| envelopeSignatureTime | 1 | Optional | DateTime | Time stamp when the envelope is signed |

Table 17 - Information output for Upload interface

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UploadResponseObject | | | | |
| Attribute | Mult | Processing | Type | Definition |
| SECOM\_ResponseCode | 0..1 | Mandatory | SECOM\_ResponseCodeEnum | Additional error code for internal errors  SECOM\_ResponseCodeEnum  The enum in Table 9 is added to the synchronous HTTP response object if validation error occurs at the SECOM instance level for information to the service consumer. |
| message | 1 | Optional | CharacterString | Success or error response message |

##### Service response

The REST service instance shall respond with HTTP codes and message according to Table 19.

Table 19 describes internal error codes and messages generated by the service implementer. Error codes and messages generated by the deployment environment shall follow RFC 7231 HTTP1/1 and is out of the scope for SECOM.

See also Table 11 for codes common to all interfaces. For this interface with an extra attribute, SECOM\_ResponseCodeEnum, this attribute is set to null for the common response codes.

For tests related to these error codes, see 10.14

Table 19 – HTTP Response codes and message in response object

|  |  |  |
| --- | --- | --- |
| HTTP Code | SECOM\_ResponseCodeEnum | Message examples |
| 200 | null | Message successfully uploaded |
| 400 | 0 | Missing required data for the service |
| 400 | 1 | Failed signature verification |
| 400 | 2 | Invalid certificate |
| 400 | 3 | Schema validation error |
| 403 | null | Not authorized to upload |
| 413 | null | Request entity too large |

Please explain the purpose, messaging pattern and architecture of the Interface.

A Service Interface supports one or several service operations. Each operation in the service interface shall be described in the following sections.

* + 1. Operation <Operation Name>

Give an overview of the operation: Include here a textual description of the operation functionality. In most instances this will be the same as the operation description taken from the UML modelling tool.

1. Operation Functionality

Describe here the functionality of the operation, i.e. how does it produce the output from the input payload.

* + - 1. Operation Parameters

Describe the logical data structure of input and output parameters of the operation (payload) by using an explanatory table (see below) and optionally UML diagrams (which are usually sub-sets of the service data model described in previous section above).

Figure 3 shows an example of a UML diagram (subset of the service data model, related to one operation).



1. <Service name> Interface Parameter Definition diagram for <operation name>

It is mandatory to provide a table with a clear description of each service operation parameter and the information about which data types defined in the service data mode are used by the service operation in its input and output parameters.

**Note:** While the descriptions provided in the physical data model shall explain the data types in a neutral format, the descriptions provided here shall explicitly explain the purpose of the parameters for the operation.

Table 4 shows an example operation parameter description table.

1. Payload description of <operation name> operation

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter Name | Direction | Data Type | Description |
| person | Input | Person | The ‘person’ parameter specifies the person for which the address is being looked for. |
| <none> | Return | Address | The return value provides the address of the person. |

1. Operation <Operation Name>

Repeat previous section for every operation defined in the service interface definition operation.

# Service Dynamic Behaviour

To be discussed: Add information here in this document or refer to IEC 63173-2?

#### Upload Dynamic behaviour

Figure 12 describes the dynamic use of the Upload interface and the relation to the Acknowledgement interface.

The Upload interface is used when data is pushed (sent) to another actor. The size of the data is technically constrained by the chosen REST technology to messages less than 350 kB. In the sequence diagram below, the left side actor, SECOM service consumer, produces the data to be pushed to the right side actor. The SECOM service consumer invokes the Upload interface (consumes the Upload service interface) and sends data, data signature and complementary metadata to the receiving actor. The complementary metadata contains attributes for the acknowledgement request (optional), transaction identifier, subscription flag and the signature for the complete transferred data package.

The receiver checks the size of data received (normally by the web server) and will respond with HTTP response 413 if the size is too large for the current configuration. A service consumer shall always be prepared to handle such error response. The receiver of data checks the integrity as part of the Verify Signature step of the data package and responds accordingly with either HTTP response 2xx if OK, or with 4xx if not OK.

The data is then forwarded to a user application, either directly or through vendor specific communication channels. If delivery acknowledgement has been requested, “delivery ACK” is sent asynchronously to the acknowledgement endpoint appointed in the metadata when the data is correctly forwarded to end-user.

The end-user application verifies access rights and verifies the signatures as received, both as integrity check and authentication of data received.

If opened acknowledgement has been requested, “opened ACK” is expected to be sent asynchronously to the acknowledgement endpoint appointed in the metadata when the data is correctly opened or processed by the end-user. SECOM does not specify a time limit between receiving a message and sending an “opened ACK”.

Graphical user interface, diagram

Description automatically generated

Figure 12 – Sequence diagram for upload signed unclassified data with acknowledgement

NOTE The acknowledgements can be used for supervision and diagnostic purposes.

# 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.

1. 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) 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. |
| **Route Plan** |  |
|  |  |
|  |  |

# ACRONYMS

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

|  |  |
| --- | --- |
| **REST** |  |
| **SECOM** | Secure Communication |
|  |  |

# REFERENCES

1. IALA Guidelineline 1128 11?? on Specification of e-Navigation Technical Services
2. IEC 63173-1 S-421 Route Plan
3. IEC 63173-2 SECOM

# SERVICE DESIGN DESCRIPTION XML

This appendix contains the formal definition of the service design description.

It is up to the author whether the service design description xml file (which includes the technology dependent definition of the physical data model) is presented in full text or just as an embedded file.