

*Document No:*

*Title: Voyage Information Service Specification - Public*

*Date: 2017-06-12*



Authors

|  |  |
| --- | --- |
| **Name** | **Organisation** |
| Per Löfbom | SMA |
| Mikael Olofsson | SMA |
| Per de Flon | SMA |

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Initials** | **Description** |
| Version 1.0 | 2016-05-16 | MO, PL, PD | Used in tendering documentation |
| Version 1.1 | 2016-09-23 | MO, PL | + Updated dynamic behaviour (sequence diagrams)  + Added possibility to request ACKNOWLEDGE when uploading message  + Added interface VIS Acknowledgement Interface  + Consolidated getVoyagePlan interfaces into one interface with optional parameters  + Consolidated subscribeToVoyagePlan into one interface with optional parameters  + Consolidated removeVoyagePlanSubscription into one interface with optional parameters  + Changed name of interface uploadPolygon -> uploadArea to match S124 Area Exchange format |
| Version 1.2 | 2016-09-23 | MO, PL | + Changed name on private operation findIdentities -> findOrganizations to harmonize with SeaSWIM Connector  + Allows multiple voyageplans in return from getVoyageplans  + Added removeAuthorizedIdentitites  + Added findAuthorizedIdentitites |
| Version 1.3 | 2016-11-14 | MO | Update after review   * Private interface Service Support interface findOrganisations changed name to findIdentities in harmony with SSC Input and output harmonised with SSC |
| Version 2.0 | 2016-11-16 | MO | Accepted on PMT   * Changed name on findOrganisations to findIdentities to match SSC * Updated sequence diagrams to show SSC |
| Version 2.1 | 2017-02-01 | MO | Updates in public interface (towards SeaSWIM)   * Business rules in getVoyagePlan and subscribeToVoyagePlan clarified * uploadVoyagePlan updated  UVID removed as parameter  callbackEndpoint added as parameter in uploadVoyagePlan * Document only covers the public side of VIS (seen from SeaSWIM) |
| Version 2.2 | 2017-03-10 | MO | Added getSubscriptionToVoyagePlan to enable a consumer to get a list of active subscriptions on voyage plans. |

Review

|  |  |
| --- | --- |
| **Name** | **Organisation** |
| Håkan Heurlin | SMA |
| PMT 2016 Nov 11 |  |
| Per Löfbom | SMA |
| Oliver Norkus | University of Oldenburg |

**Table of contents**

1 Introduction 6

1.1 Purpose of the document 6

1.2 Intended readership 6

1.3 Inputs from other projects 6

2 Service identification 7

3 Operational Context 8

3.1 Functional and non-functional Requirements 9

3.2 Other Constraints 11

3.2.1 Relevant Industrial Standards 11

3.3 Operational Nodes 12

3.4 Operational Activities 14

4 Service Overview 15

4.1 Service Interfaces 15

4.2 Consumer Interfaces 15

5 Service Data Exchange Model 17

5.1 Service Data Exchange Model SeaSWIM interface 17

5.1.1 ResponseObj 18

5.1.2 S124 18

5.1.3 enumeration\_messageType 18

5.1.4 DeliveryAck 18

5.1.5 GetVPResponseObject 19

5.1.6 enumeration\_routeStatus 19

5.1.7 GetVoyagePlanObject 20

5.1.8 textMessage 20

5.1.9 MRN 20

5.1.10 URL 21

5.1.11 route 21

6 Service Interface Specification 22

6.1 Voyage Information Service 22

6.1.1 VIS Get Interface 22

6.1.2 VIS Subscription Interface 22

6.1.3 VIS Upload Interface 23

6.1.4 VIS Acknowledgement Interface 25

7 Service Dynamic Behaviour 26

7.1 VIS SeaSWIM Interface 26

7.1.1 VIS v3 Public 26

7.1.2 VIS v4 Public 27

7.1.3 VIS Get Interface 27

7.1.4 VIS Upload Interface 29

7.1.5 VIS Subscription Interface 33

7.2 Logging 35

7.2.1 VIS Event Log 35

8 Service Provisioning 37

9 References 38

10 Acronyms and Terminology 39

10.1 Acronyms 41

10.2 Terminology 41

11 Service Specification as XML 46

12 Document lifecycle 54

12.1 Maturity 54

12.2 Forecast 54

# Introduction

## Purpose of the document

The purpose of this service specification document is to provide a holistic overview of the Voyage Informationservice and its building blocks in a technology-independent way, according to the guidelines. It describes a well-defined baseline of the service by clearly identifying the service version.

The aim is to document the key aspects of the Information service at the logical level:

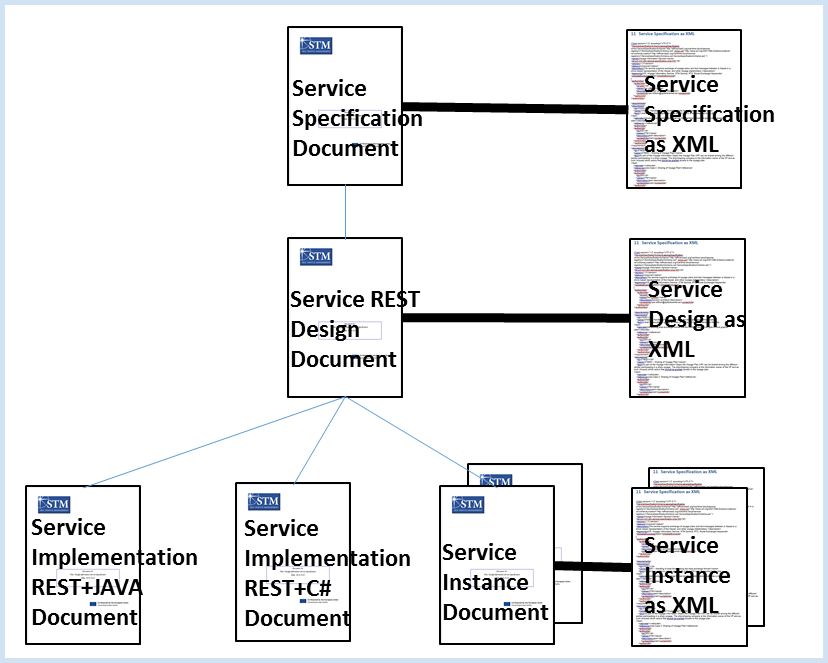
1. the operational and business context of the service
   * requirements for the service (e.g., information exchange requirements)
   * involved nodes: which operational components provide/consume the service
   * operational activities supported by the service
   * relation of the service to other services
2. the service description
   * service interface definitions
   * service interface operations
   * service payload definition
   * service dynamic behaviour description
3. service provision and validation aspects

## Intended readership

This service specification is intended to be read by service architects, system engineers and developers in charge of designing and developing an instance of the Information service.

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

This document contains specification of the service in focus.



## Inputs from other projects

No Information.

# Service identification

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

|  |  |
| --- | --- |
| **Name** | Voyage Information Service |
| **ID** | urn:mrn:stm:service:specification:sma:vis |
| **Version** | 2.2 |
| **Description** | The service supports exchange of voyage plans, text messages and areas. |
| **Keywords** | VIS, Voyage Information Service, STM Service, RTZ, Route Exchange |
| **Status** | released |

# Operational Context

The main purpose with Voyage Information Service is to support sharing of voyage plans to authorized actors. Sharing of voyage plan is primarily initiated by the ship by authorizing the voyage plan to concerned actors and by direct accessing e.g. route optimization or route check, but sharing can also be on request by other service providers such as enhanced monitoring.

The Voyage Information Service can be used both to support exchange of voyage plans from ship as well as other service providers and consumers such as shore centers and route optimization providers.

The Voyage Information Service is specified in such way that by using VIS on all consumers and providers that intend to share/exchange voyage plans, interoperability can be reached. That enables new services to arise in Service Registry based on VIS Design for voyage plan exchange to be used without new implementation on consumer side.

Each Voyage Plan shall refer to a UVID (Unique Voyage Identity) generated by the service provider and contain status on the voyage/route.



## Functional and non-functional Requirements

Requirements briefly

* VIS has a **storage** (for storing sent and received messages, XML schemas, logs)
* VIS is an **information service registered** in SeaSWIM central Service Registry
* VIS has service **endpoints** for exposing **methods**
* VIS has a **function to validate** message payload according to the following predefined schemas (rtz, text, area)
* **All communication** between VIS and SeaSWIM Central services or and other information services is achieved using SeaSWIM connector.



|  |  |
| --- | --- |
| **Requirement Id** | VIS002 |
| Requirement Name | VIS002 - Handling of area format using the Area exchange format |
| Requirement Text | It shall be possible to upload an Area (S-124) to VIS. |
| Rationale | Introducing area exchange format into the maritime domain will give a more graphic view |
| Author | STM |
| Reference |  |

|  |  |
| --- | --- |
| **Requirement Id** | STM001 |
| Requirement Name | STM001 - Sharing of Voyage Plan |
| Requirement Text | As part of the Voyage Information Object the Voyage Plan (VP) can be shared among the different parties participating in a ships voyage. The ship/shipping company is the information owner of the VP and as such chooses which actors that should be granted access to the voyage plan. |
| Rationale |  |
| Author | STM |
| Reference | Use-Case 2: Sharing of Voyage Plan |

|  |  |
| --- | --- |
| **Requirement Id** | VIS003 |
| Requirement Name | VIS003 - Handling of exchange of text messages |
| Requirement Text | It shall be possible to upload a STM Text Message to VIS. |
| Rationale |  |
| Author | STM |
| Reference |  |

|  |  |
| --- | --- |
| **Requirement Id** | VIS005 |
| Requirement Name | VIS005 - Message transfer status |
| Requirement Text | Handling of message statuses sent to be able to support messages transferred ok |
| Rationale |  |
| Author |  |
| Reference |  |

|  |  |
| --- | --- |
| **Requirement Id** | VIS006 |
| Requirement Name | VIS006 - Save timestamp for sent and received messages |
| Requirement Text | The age of information shall be known by VIS |
| Rationale |  |
| Author |  |
| Reference |  |

|  |  |
| --- | --- |
| **Requirement Id** | VIS001 |
| Requirement Name | VIS001 - Handling of voyage plan using route exchange (RTZ) format |
| Requirement Text | Handle voyage plans in RTZ format identified by UVID for sending on request, publish to subscribers and forward incoming from external parties. |
| Rationale |  |
| Author | STM |
| Reference |  |

|  |  |
| --- | --- |
| **Requirement Id** | VIS007 |
| Requirement Name | VIS007 - Events and data exchanged shall be stored and logged. |
| Requirement Text | All data to and from VIS shall be stored and logged with metadata to support the validation of STM. |
| Rationale |  |
| Author |  |
| Reference |  |

## Other Constraints

Currently there is no subscription or request operations for STM Text Message and Area message. Only receive uploaded messages and send (callService) is supported.

### Relevant Industrial Standards

IEC 61174:2015

## Operational Nodes

The section describes the context surrounding the information service.

The Voyage Information Service, VIS, is a generic service for exchange of voyage information, primarily in RTZ format. Thus the VIS can be used both representing a ship, but also representing a Shore Centre or a Service Provider of e.g. Route Optimization.

In the picture below, the SSC (SeaSWIM Connector) is shown to represent the functionality required by the SeaSWIM infrastructure for mainly security reasons. The SSC can be either a separate service or component, or built-in functionality in the Voyage Information Service. For further details related to SeaSWIM and SeaSWIM Connector, please see http://stmvalidation.eu/seaswim-overview



| **Operational Node/Activity** | **Remarks** |
| --- | --- |
| Ship | A floating structure designed for the transport of cargo and/or passengers.  The operational node represents a collection of services, activities and procedures of Ship. |
| Service provider | Organizations/ authorities offering e.g. route optimization services possible consumers of voyage plans provided by a vessel or a representation thereof. SMHI (Swedish Metrological & Hydrological Institute) is one example. |
| Shore Center | Collection of services, activities and procedures of Shore Center  Refers to entities offering services such as route check and/ or enhanced monitoring. |
| SeaSWIM | SeaSWIM enables information security and service lookup in a structured and governed manner. |

## Operational Activities

Operational Activities (processes) has not been more elaborated than on Use Case level.



# Service Overview

The main purpose with VIS is to handle the communication around voyage information and the main artefact Voyage Plan (VP) in RTZ format. VIS implements methods for exposing new and updated VP’s and to consume external VP’s. VIS also supports subscription of voyage plans.

In addition to voyage plans (RTZ), VIS also supports exchange of STM Text Message and area message (S-124).

VIS is also consuming the same Upload and Acknowledgement interface specified in VIS, hence VIS assumes VIS or equal on subscriber consumer.



## Service Interfaces

The following set of interfaces and operations are provided by VIS.

| **Service Interface** | **Role** | **Service Operation** |
| --- | --- | --- |
| VIS Get Interface | Provided | getVoyagePlans |
| VIS Subscription Interface | Provided | subscribeToVoyagePlan  getVoyagePlanSubscription  removeVoyagePlanSubscription |
| VIS Upload Interface | Provided | uploadVoyagePlan  uploadTextMessage  uploadArea |
| VIS Acknowledgement Interface | Provided | acknowledgement |

## Consumer Interfaces

The following set of interfaces and operations are consumed by VIS.

| **Service Interface** | **Role** | **Service Operation** |
| --- | --- | --- |
| VIS Upload Interface | Consumed | uploadVoyagePlan |
| VIS Acknowledgement Interface | Consumed | acknowledgement |

# Service Data Exchange Model

## Service Data Exchange Model SeaSWIM interface



### route

RTZ files contain a number of waypoints, followed with auxiliary schedules.

For detailed information, see <http://stmvalidation.eu/schemas/>

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **route** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | routeInfo | RouteInfo | | Generic route information. | | | waypoints | Waypoints | | A list of waypoints. | | | schedules | Schedules | | Optional list of schedules. | | | extensions | Extensions | | You can add extend RTZ by adding your own elements from another schema  here. | | | version | NonEmptyString | | Format version | | |

### textMessage

Text message defined in STM project.

For detailed information, see <http://stmvalidation.eu/schemas/>

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **textMessage** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | textMessageId | string | | Identifier of the text message, mandatory. | | | informationObjectReferenceId | string | | A reference to an information object, optional. | | | author | string | | The message author, mandatory. | | | from | string | | The sending actor, mandatory. | | | serviceType | string | | The service type of the sender, optional. | | | createdAt | dateTime | | The message creation dateTime, mandatory. | | | subject | string | | The message subject, mandatory. | | | body | string | | The message body,mandatory. | | | position | GM\_Point | | Position, optional | | | area | GM\_Surface | | Area, optional | | |

### S124

S124 area message

For detailed information, see <http://stmvalidation.eu/schemas/>

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **S124** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | dataSet | string | | S124 area message as defined at STM Developer Forum site http://stmvalidation.eu. | | |

### DeliveryAck

Object for message ACK

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **DeliveryAck** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | id | string | | Id for the ACK | | | referenceId | MRN | | Reference to delivered message according to the STM MRN identifier. For example an unique voyage identifier: urn:mrn:stm:voymgt:uvid:<organizationId>:<local voyagenumber> | | | timeOfDelivery | dateTime | | Time of delivery | | | fromId | MRN | | Identity of source (sender) of message that have been delivered according to the STM MRN identifier. Example: urn:mrn:stm:org:<organizationId> | | | fromName | string | | Friendly name of sender | | | toId | MRN | | Identity of target (receipient) of message delivery according to the STM MRN identifier. Example: urn:mrn:stm:org:<organizationId> | | | toName | string | | Friendly name of recipient | | | ackResult | string | |  | | |

### GetVPResponseObject

Response object from request for voyage plan

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **GetVPResponseObject** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | lastInteractionTime | dateTime | | Last interaction time with private application | | | route | rtz:route | | Sequence of 0 or more route messages (RTZ) in XML format | | |

### GetVoyagePlanObject

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **GetVoyagePlanObject** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | UVID | MRN | | UVID in MRN format, optional  routeStatus as string "1" to "8", optional | | | routeStatus | enumeration\_routeStatus | |  | | |

### MRN

Marine Resource Name identifier, based on URN.

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **MRN** | |  |  |  | | --- | --- | --- | | **Name** | **Type** | **Description** | |

### URL

Uniform Resource Identifier

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **URL** | |  |  |  | | --- | --- | --- | | **Name** | **Type** | **Description** | |

# Service Interface Specification

The main purpose with VIS is to handle the communication around voyage information and the main artefact Voyage Plan (VP) in RTZ format. VIS implements methods for exposing new and updated VP’s and to consume external VP’s. VIS also supports subscription of voyage plans.

In addition to voyage plans (RTZ), VIS also supports exchange of STM Text Message and area message (S-124).

VIS is also consuming the same Upload and Acknowledgement interface specified in VIS, hence VIS assumes VIS or equal on subscriber consumer.

## Voyage Information Service

The Voyage Information Service provides interfaces for requesting voyage plan (Get), requesting subscription of voyage plans (Subscription) and to upload voyage plan, text message and areas (Upload).

### VIS Get Interface

Facilitates operations for requesting a Voyage Plan.

#### getVoyagePlans()

Returns active voyage plans according to parameters that the requester is authorized to. The response can contain 0 or more (0..\*) voyage plans, but only 0 or one (0..1) voyage plan with same UVID.

An active voyage is a voyage plan with routeStatus not equal 8.

Will return the latest published voyage plan with routeStatus not equal 8,for each UVID (vesselVoyage). If two or more voyage plans have routeStatus=7, only the latest published voyage plan for each ship shall be returned.

If not authorized, a message will be returned and a message will be sent on private side as a authorization request.

Also returns “Last known time” with interaction on private side. The intention is to give a consumer a possibility to assess validity of given voyage plan(s) if the e.g. ship is offline.

**Operation functionality**

Retrieve requester identity

Retrieve messages from cache/repository that apply to get criterias (parameters)

Check authorization to data against ACL

If authorized, return VPs

**Operation Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter Name** | **Direction** | **Data Type** | **Description** |
| getVoyagePlanObj | Input | GetVoyagePlanObject | Request can be of a specific UVID and/or a specific route status |

|  |  |  |  |
| --- | --- | --- | --- |
| **Return** | **Direction** | **Data Type** | **Description** |
|  | Return | GetVPResponseObject |  |

### VIS Subscription Interface

Facilitates operations for subscribing and unsubscribing to a Voyage Plan.

#### subscribeToVoyagePlan()

Facilitates request for subscription of voyage plans to the given callbackEndpoint.

If UVID is empty, VIS will search and add a subscription of all active voyageplans (UVIDs) the requester has access to.

If the requester doesn't have access to any UVIDs, a error message will be returned and private application will be notified.

If the requester have access to some of the active UVIDs, a subscription will be added to the authorized UVID(s). The private application will be notified regarding the non-authorized UVID(s).

If the requester have access to all active UVIDs, a subscription for each authorized UVID will be created.

An active voyage is a voyage plan with routeStatus not equal 8.

**Operation functionality**

The received request is stored in VIS database.

The authorization to data is checked and if authorized an OK is given and the callbackEndpoint from calling party is stored in a subscription table in VIS database together with subscription parameters.

The latest active Voyage Plan is sent to callbackEndpoint

**Operation Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter Name** | **Direction** | **Data Type** | **Description** |
| callbackEndpoint | Input | URL |  |
| dataId | Input | MRN |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Return** | **Direction** | **Data Type** | **Description** |
|  | Return | void |  |

#### getVoyagePlanSubscription()

Get information on subscribed voyage plans.

**Operation Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter Name** | **Direction** | **Data Type** | **Description** |
| callbackEndpoint | Input | URL |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Return** | **Direction** | **Data Type** | **Description** |
|  | Return | GetSubscriptionResponseObj |  |

#### removeVoyagePlanSubscription()

Remove subscription of Voyage Plans

**Operation functionality**

The subscription attached to the subscription parameters is removed

**Operation Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter Name** | **Direction** | **Data Type** | **Description** |
| callbackEndpoint | Input | URL |  |
| dataId | Input | MRN |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Return** | **Direction** | **Data Type** | **Description** |
|  | Return | void |  |

### VIS Upload Interface

Facilitates operations for uploading a Voyage Plan, Text Message and Area.

#### uploadVoyagePlan()

Facilitates sending (uploading) a voyage plan to VIS. The route shall be uncompressed (RTZ).

If endpoint/URL provided for deliveryACK, an ACK will be sent when message has been forwarded to private application.

**Operation functionality**

The uploaded message is stored in cache

The voyage plan is checked against the RTZ schema

If correct, a notification is sent to STM Module

If delivery ACK is requested, the flag is set in cache and when the STM Module calls getMessage, VIS sends the message to STM Module and a delivery ACK to the requested endpoint.

**Operation Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter Name** | **Direction** | **Data Type** | **Description** |
| deliveryAckEndpoint | Input | URL | Name of ACK callback endpoint if ACK is requested, otherwise empty |
| callbackEndpoint | Input | URL |  |
| voyagePlan | Input | rtz:route | The route in RTZ format |

|  |  |  |  |
| --- | --- | --- | --- |
| **Return** | **Direction** | **Data Type** | **Description** |
|  | Return | ResponseObj |  |

#### uploadTextMessage()

Facilitates sending (uploading) a text message to VIS to be forwarded to the ship (STM Module).

If endpoint/URL provided for deliveryACK, an ACK will be sent when message has been forwarded to private application.

**Operation functionality**

The uploaded message is stored in cache

The textMessage is checked against the textMessage schema

If correct, a notification is sent to STM Module

If delivery ACK is requested, the flag is set in cache and when the STM Module calls getMessage, VIS sends the message to STM Module and a delivery ACK to the requested endpoint.

**Operation Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter Name** | **Direction** | **Data Type** | **Description** |
| textMessage | Input | stm:textMessage | The text message in STM Text Message format |
| deliveryAckEndpoint | Input | URI | Name of ACK callback endpoint if ACK is requested, otherwise empty |

|  |  |  |  |
| --- | --- | --- | --- |
| **Return** | **Direction** | **Data Type** | **Description** |
|  | Return | ResponseObj |  |

#### uploadArea()

Facilitates sending (uploading) an area to VIS to be forwarded to the ship (STM Module).

If endpoint/URL provided for deliveryACK, an ACK will be sent when message has been forwarded to private application.

**Operation functionality**

The uploaded message is stored in cache

The message is checked against the area schema

If correct, a notification is sent to STM Module

If delivery ACK is requested, the flag is set in cache and when the STM Module calls getMessage, VIS sends the message to STM Module and a delivery ACK to the requested endpoint.

**Operation Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter Name** | **Direction** | **Data Type** | **Description** |
| area | Input | S124:DataSet | The area in S-124 format |
| deliveryAckEndpoint | Input | URI | Name of ACK callback endpoint if ACK is requested, otherwise empty |

|  |  |  |  |
| --- | --- | --- | --- |
| **Return** | **Direction** | **Data Type** | **Description** |
|  | Return | ResponseObj |  |

### VIS Acknowledgement Interface

#### acknowledgement()

During upload of message, an acknowledgement can be requested by giving a callback endpoint to a Acknowledge interface. If such an acknowledgement has been requested, the service will send an Acknowledgement message when the uploaded message has been retrieved/sent to private application.

**Operation functionality**

AckDelivery endpoint stored from upload interface

When uploaded message to VIS is sent to STM Module, VIS sends an ACK to uploaded Ack endpoint.

**Operation Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter Name** | **Direction** | **Data Type** | **Description** |
| DeliveryAck | Input | DeliveryAck |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Return** | **Direction** | **Data Type** | **Description** |
|  | Return | ResponseObj |  |

# Service Dynamic Behaviour

## VIS SeaSWIM Interface

This section contains sequence diagrams related to VIS SeaSWIM interface.



### VIS Get Interface

This section contains sequence diagrams related to VIS SeaSWIM Get interface.

A service consumer may request a voyage plan any time, either asking for a known UVID or just ask for any voyage plan published in VIS instance.

A service consumer can ask for voyage plans in a certain status, according to routeStatus enumeration, or ask for any voyage plan.

If the service consumer is not authorized by the "owner" of the VIS instance, a notification is forwarded to the "owner" and the service consumer don't get any voyage plans back until "owner" has authorized the service consumer.

If several unique voyage plans have been published in the VIS instance, all will be returned in the request. This enables the VIS to be deployed as a catalogue of voyage plans and routes.

However be aware that only zero or one (0..1) voyage plans in routeStatus=7 (Used for monitoring) can be returned from a VIS.

The service consumer must always check the routeStatus and act according the purpose by the service consumer. If the service consumer only wants "Used for monitoring", the request should be for routeStatus="7".

VIS will only handle requests from service consumer that are authenticated in STM.

#### Interaction getVoyagePlan

Message exchange pattern: REQUEST\_RESPONSE

At receipt of request for a voyage plan in VIS, the user authorization is checked using an Access Control List (ACL). In case of successful authorization, the requested voyage plan(s) are fetched and returned to the calling service. If unsuccessful authorization, a non-authorized error response is sent. See further diagram Not authorized.



#### Service orchestration - Not authorized

In case the consumer is not authorized to requested data, the private application is notified hereof. The service consumer receives a message “Not authorized, request forwarded to operator”.

If no UVID is provided as parameter, a notification is sent to the private application only if the requester is not authorized to the voyage plan Used for monitoring (latest published voyage plan with routeStatus="7" for one ship).

It is then up to the user operating the private application to authorize the consumer to the requested voyage plan. Hereby creating a record in VIS ACL for the consumer identity.

In the case the operator chooses not to authorize the consumer, a textMessage can be sent to the consumer with notification of an unsuccessful authorization.



### VIS Upload Interface

This section contains sequence diagrams related to VIS SeaSWIM Upload interface.

Asynchronous Acknowledgement can be requested.

A service provider can always upload a voyageplan, text message or area message to VIS.

The service provider can always request an acknowledge message by providing an acknowledgement endpoint in the upload service request. When the message has been delivered to VIS private side, an acknowledgement is sent to the service provider. This acknowledgement however does not ensure that the message have reached the end user. This depends on the deployment on the private side where the STM Module may be an application on shore side and proprietary system to the end user, such as a ship.

VIS will only handle uploaded messages from service providers that are authenticated in STM.

#### Interaction uploadVoyagePlan

Message exchange pattern: REQUEST\_CALLBACK



#### Interaction uploadTextMessage

Message exchange pattern: REQUEST\_CALLBACK

After receipt of a text message, the originating user organization is authenticated. Following a successful authentication the payload of the received message is validated against the schema.

In the case a deliveryAckEnpoint is supplied as parameter, an acknowledgement message is returned to the consumer after the delivery to the private application.



#### Interaction uploadArea

Message exchange pattern: REQUEST\_CALLBACK

After receipt of a area message (S-124), the originating user organization is authenticated. Following a successful authentication the payload of the received message is validated against the schema.

In the case a deliveryAckEnpoint is supplied as parameter, an acknowledgement message is returned to the consumer after the delivery to the private application.



#### Service orchestration - Upload with ACK

The acknowledgement interface VIS exposes, is the endpoint for acknowledgement messages optionally requested by use of parameter deliveryAckEndpoint at upload of messages to VIS. The acknowledgement message is created for a specific message when it is successfully retrieved by the STM Module using VIS private interface getMessage, i.e. forwarded to the vessel. When the ACK is received, a notification is sent to the STM Module. The STM Module is responsible for checking and acting if ACK is not received.



#### Service orchestration - Upload with Callback

The callbackEndpoint can be provided for two purposes; inform that callback with information is expected, such as an optimized route; and inform to which endpoint the data is expected to be delivered to. The service responding on the provided callbackEndpoint shall still be an authenticated service.



### VIS Subscription Interface

This section contains sequence diagrams related to VIS SeaSWIM Subscription interface.

A service consumer can always ask to subscribe to voyageplans. Either a known specific UVID or all voyageplans published in the VIS instance.

If the service consumer is not authorized by the "owner" of the VIS instance, a notification is forwarded to the "owner" and the service consumer don't get eny voyage plans back until "owner" has authorized the service consumer.

VIS will only handle requests from service consumer that are authenticated in STM.

#### Interaction subscribeToVoyagePlan

Message exchange pattern: REQUEST\_CALLBACK

Consumer requests subscription by invoking interface subscribeToVoyagePlan providing the URI (address to consuming service uploadVoyagePlan interface - callbackEndpoint), optionally an uvid parameter can be passed for subscription on a specific voyagePlan. Following a successful authorization the subscriber identity and corresponding callbackEndpoint is stored in VIS dB subscription table and a voyagePlan is sent to the added subscriber. Every time a voyagePlan is published in VIS, the voyagePlan is forwarded to all selected subscribers. In response to the subscription request a responseObj is returned with statusCode=200, successful.

If UVID is not provided (is blank), VIS will try to set up a subscription to all "active" UVID with route with routeStatus 1-7 the requester has access to.

If there are 2 ore more voyage plans with routeStatus="7" for one ship, only the latest published of them will generate a subscription.

I.e. if there are one VP with routeStatus=7 and one in routeStatus=3, subscription will be enabled for both UVIDs.



#### Interaction removeVoyagePlanSubscription

Message exchange pattern: ONE\_WAY

At removal of a subscription the removeVoyagePlanSubscription is invoked by the consumer. Parameters are the consumer callBackendpoint (mandatory) and optionally a specific uvid. At receipt of the subscription removal request VIS deletes all subscriptions for the callBackendpoint or a specific subscription for an uvid. In response to the subscription removal request a responseObj is returned with statusCode=200, successful.



#### Service orchestration - Not authorized

In case the consumer is not authorized the STM Module operator onboard the vessel is notified hereof (message includes the consumer STM identity). The consumer receives a message “Not authorized request forwarded to operator”.

If authorized it is up to the user operating the STM Module to authorize the consumer to the requested voyage plan using VIS private interface authorizeIdentities. Hereby creating a record in VIS ACL for the consumer identity. VIS then searches SeaSWIM service registry in order to find the consuming service endpoint for receiving voyagePlans (findServices) and sends the requested voyagePlan to the consumer.

In the case the operator chooses not to authorize the consumer a textMessage is sent to the consumer with notification of an unsuccessful authorization.



## Logging

Logging in the service is required for validation purposes to enable analysis of data in order to assess the STM Concept.

### VIS Event Log

Message exchange pattern:

The following events are proposed to generate a log:

* Messages in and out of the service
* Failure events (Schema validation failure, Service operation failure)
* Authorization events

The following events are proposed to be logged:

* Messages in and out of the service
* Failure events (Schema validation failure, Service operation failure)

|  |  |
| --- | --- |
| **Incoming service calls on SeaSWIM side** | |
| **Event** | **Log description** |
| getVoyagePlans | Log event for incoming request  Log event with returned data |
| subscribeToVoyagePlan | Log event for incoming request  Log event with returned data |
| uploadVoyagePlan | Log event with incoming data |
| uploadTextMessage | Log event with incoming data |
| uploadArea | Log event with incoming data |
| acknowledgement | Log event with incoming data |

|  |  |
| --- | --- |
| **Outgoing service calls on SeaSWIM side** | |
| **Event** | **Log description** |
| <callService> | Log event with outgoing data |

# Service Provisioning

The VIS service is intended to be provided by either shore side server or onboard server depending on available connectivity and architecture.



The goal when specifying the Voyage Information Service was to support exchange of voyage information both on a ship, from ship operator planning centre but also a shore centre such as VTS with Enhanced Monitoring and 3rd party service providers for e.g. Route Optimization.

Using the same uploadVoyagePlan interface at all parties enables interoperability and scalability. When new services arises that are built around sharing of voyage information, they will be found in the service registry and can be used without new software.



# References

|  |  |  |
| --- | --- | --- |
| **Reference name** | **Comment** | **Link** |
| Route Exchange format (IEC 61174 App S) | IEC 61174:2015 Appendix S - Route plan exchange format - RTZ | http://stmvalidation.eu/schemas/ |
| STM Governance Handbook |  | - |
| STM Text Message |  | http://stmvalidation.eu/schemas/ |
| Area Exchange Format.pdf | Proposed to IHO as S124 | http://stmvalidation.eu/schemas/ |
| SeaSWIM Connector Specification |  | http://stmvalidation.eu/seaswim-overview/ |

# Acronyms and Terminology

## Acronyms

|  |  |
| --- | --- |
| **Term** | **Definition** |
| SSC | SeaSWIM Connector |
| URN | Uniform Resource Locator |
| UVID | Unique Voyage Identity |
| VIS | Voyage Information Service |
| VP | Voyage Plan |
| XML | Extendible Mark-up Language |
| XSD | XML Schema Definition |

## Terminology

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Service Specification | Describes one dedicated service at logical level. The Service Specification is technology-agnostic. The Service Specification includes (but is not limited to) a description of the Service Interfaces and Service Operations with their data payload. The data payload description may be formally defined by a Service Data Model.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service Technical Design | The technical design of a dedicated service in a dedicated technology. One service specification may result in several technical service designs, realising the service with different or same technologies.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service Implementation | The provider side implementation of a dedicated service technical design (i.e., implementation of a dedicated service in a dedicated technology).  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service Instance | One service implementation may be deployed at several places by same or different service providers; each such deployment represents a different service instance, being accessible via different URLs.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service Endpoint | A Service Endpoint is the URL where your service can be accessed by a client application. The same web service can have multiple endpoints, for example in order to make it available using different protocols.  *Source*  *http://stackoverflow.com/questions/9807382/what-is-a-web-service-endpoint* |
| Service Interface | The communication mechanism of the service, i.e., interaction mechanism between service provider and service consumer. A service interface is characterised by a message exchange pattern and consists of service operations that are either allocated to the provider or the consumer of the service.  *Source*  *E2 D3.4 Service Documenation Guidelines*  *v01.01* |
| Service Operation | Functions or procedure which enables programmatic communication with a service via a service interface.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service Parameters | Service Parameters are input to a Service Operation and can be described formally in a data exchange model as e.g. XML Schemas.  *Source*  *MO* |
| Service Response | Service Response are output from a Service Operation and can be described formally in a data exchange model as e.g. XML Schemas.  *Source*  *MO* |
| Authentication | Authentication is the process of determining whether someone or something is, in fact, who or what it is declared to be.  *Source*  *http://searchsecurity.techtarget.com/definition/authentication* |
| Authorization | Authorization is the process of giving someone permission to do or have something.  *Source*  *http://searchsoftwarequality.techtarget.com/definition/authorization* |
| Service Consumer | A service consumer uses service instances provided by service providers. All users within the maritime domain can be service customers, e.g., ships and their crew, authorities, VTS stations, organizations (e.g., meteorological), commercial service providers, etc.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service Provider | A service provider provides instances of services according to a service specification and service instance description. All users within the maritime domain can be service providers, e.g., authorities, VTS stations, organizations (e.g., meteorological), commercial service providers, etc.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Proxy Service | A proxy service is an intermediary role played by software or a dedicated computer system between an endpoint device and a client which is requesting the service. The proxy service may exist on the same machine or on a separate server. The proxy service enables the client to connect to a different server and provides easy access to services like Web pages, connections or files.  *Source*  *https://www.techopedia.com/definition/31705/proxy-service* |
| Service Request | *Source* |
| Operational Activity | An activity performed by an operational node. Examples of operational activities in the maritime context are: Route Planning, Route Optimization, Logistics, Safety, Weather Forecast Provision, …  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Operational Model | A structure of operational nodes and associated operational activities and their inter-relations in a process model.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Operational Node | A logical entity that performs activities. Note: nodes are specified independently of any physical realisation.  Examples of operational nodes in the maritime context are: Maritime Control Center, Maritime Authority, Ship, Port, Weather Information Provider, …  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service | The provision of something (a non-physical object), by one, for the use of one or more others, regulated by formal definitions and mutual agreements. Services involve interactions between providers and consumers, which may be performed in a digital form (data exchanges) or through voice communication or written processes and procedures.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service Data Model | Formal description of one dedicated service at logical level. The service data model is part of the service specification. Is typically defined in UML and/or XSD. If an external data model exists (e.g., a standard data model), then the service data model shall refer to it: each data item of the service data model shall be mapped to a data item defined in the external data model.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service Implementer | Implementers of services from the service provider side and/or the service consumer side. Anybody can be a service implementer but mainly this will be commercial companies implementing solutions for shore and ship.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service Instance Description | Documents the details of a service implementation (most likely documented by the service implementer) and deployment (most likely documented by the service provider). The service instance description includes (but is not limited to) service technical design reference, service provider reference, service access information, service coverage information, etc.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service Instance Model | Describes the implementation of a dedicated service instance in a dedicated technology. This includes a detailed description of the data payload to be exchanged by this service instance. The actual format of the service instance model depends on the chosen technology. Examples may be WSDL and XSD files (e.g., for SOAP services) or swagger (Open API) specifications (e.g., for REST services). If an external data model exists (e.g., a standard data model), then the service instance model shall refer to it: each data item of the service instance model shall be mapped to a data item defined in the external data model.  In order to prove correct implementation of the service specification, there shall exist a mapping between the service instance model and the service data model. This means, each data item used in the service instance model shall be mapped to a corresponding data item of the service data model. (In case of existing mappings to a common external (standard) data model from both the service data model and the service instance model, such a mapping is implicitly given.)  *Source* |
| Service Technology Catalogue | List and specifications of allowed technologies for service implementations. Currently, SOAP and REST are envisaged to be allowed service technologies. The service technology catalogue shall describe in detail the allowed service profiles, e.g., by listing communication standards, security standards, stacks, bindings, etc.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service Design Description | Documents the details of a service technical design (most likely documented by the service implementer). The service design description includes (but is not limited to) a service physical data model and describes the used technology, transport mechanism, quality of service, etc.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service Physical Data Model | Describes the realisation of a dedicated service data model in a dedicated technology. This includes a detailed description of the data payload to be exchanged using the chosen technology. The actual format of the service physical data model depends on the chosen technology. Examples may be WSDL and XSD files (e.g., for SOAP services) or swagger (Open API) specifications (e.g., for REST services). If an external data model exists (e.g., a standard data model), then the service physical data model shall refer to it: each data item of the service physical data model shall be mapped to a data item defined in the external data model.  In order to prove correct implementation of the service specification, there shall exist a mapping between the service physical data model and the service data model. This means, each data item used in the service physical data model shall be mapped to a corresponding data item of the service data model. (In case of existing mappings to a common external (standard) data model from both the service data model and the service physical data model, such a mapping is implicitly given.)  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Service Specification Producer | Producers of service specifications in accordance with the service documentation guidelines.  *Source*  *E2 D3.4 Service Documentation Guidelines*  *v01.01* |
| Authentication | The process of verifying the identity claimed by an entity based on its credentials.  *Source*  *developers.maritimecloud.net*  *2016-11-11* |

# Service Specification as XML

<?xml version="1.0" encoding="UTF-8"?>

<ServiceSpecificationSchema:serviceSpecification xmlns:ServiceSpecificationSchema="http://efficiensea2.org/maritime-cloud/service-registry/v1/ServiceSpecificationSchema.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://efficiensea2.org/maritime-cloud/service-registry/v1/ServiceSpecificationSchema.xsd ServiceSpecificationSchema.xsd ">

<name>Voyage Information Service</name>

<id>urn:mrn:stm:service:specification:sma:vis</id>

<version>2.2</version>

<status>released</status>

<description>The service supports exchange of voyage plans, text messages and area messages.</description>

<keywords>VIS, Voyage Information Service, STM Service, RTZ, Route Exchange, Area, S-124, TXT, Text Message</keywords>

<isSpatialExclusive>false</isSpatialExclusive>

<authorInfos>

<authorInfo>

<id>urn:mrn:stm:org:sma:pelo</id>

<name>Per Löfbom</name>

<description>Solution architect</description>

<contactInfo>per.lofbom@sjofartsverket.se</contactInfo>

</authorInfo>

</authorInfos>

<requirements>

<requirement>

<id>VIS002</id>

<name>VIS002 - Handling of area format using the Area exchange format</name>

<text>It shall be possible to upload an Area (S-124) to VIS.

</text>

<rationale>Introducing area exchange format into the maritime domain will give a more graphic view</rationale>

<reference></reference>

<authorInfos>

<authorInfo>

<id>000</id>

<name>STM</name>

<description>text</description>

<contactInfo>mail</contactInfo>

</authorInfo>

</authorInfos>

</requirement>

<requirement>

<id>STM001</id>

<name>STM001 - Sharing of Voyage Plan</name>

<text>As part of the Voyage Information Object the Voyage Plan (VP) can be shared among the different parties participating in a ships voyage. The ship/shipping company is the information owner of the VP and as such chooses which actors that should be granted access to the voyage plan.

</text>

<rationale></rationale>

<reference>Use-Case 2: Sharing of Voyage Plan</reference>

<authorInfos>

<authorInfo>

<id>000</id>

<name>STM</name>

<description>text</description>

<contactInfo>mail</contactInfo>

</authorInfo>

</authorInfos>

</requirement>

<requirement>

<id>VIS003</id>

<name>VIS003 - Handling of exchange of text messages</name>

<text>It shall be possible to upload a STM Text Message to VIS.

</text>

<rationale></rationale>

<reference></reference>

<authorInfos>

<authorInfo>

<id>000</id>

<name>STM</name>

<description>text</description>

<contactInfo>mail</contactInfo>

</authorInfo>

</authorInfos>

</requirement>

<requirement>

<id>VIS005</id>

<name>VIS005 - Message transfer status</name>

<text>Handling of message statuses sent to be able to support messages transferred ok

</text>

<rationale></rationale>

<reference></reference>

<authorInfos>

<authorInfo>

<id>000</id>

<name></name>

<description>text</description>

<contactInfo>mail</contactInfo>

</authorInfo>

</authorInfos>

</requirement>

<requirement>

<id>VIS006</id>

<name>VIS006 - Save timestamp for sent and received messages</name>

<text>The age of information shall be known by VIS

</text>

<rationale></rationale>

<reference></reference>

<authorInfos>

<authorInfo>

<id>000</id>

<name></name>

<description>text</description>

<contactInfo>mail</contactInfo>

</authorInfo>

</authorInfos>

</requirement>

<requirement>

<id>VIS001</id>

<name>VIS001 - Handling of voyage plan using route exchange (RTZ) format</name>

<text>Handle voyage plans in RTZ format identified by UVID for sending on request, publish to subscribers and forward incoming from external parties.

</text>

<rationale></rationale>

<reference></reference>

<authorInfos>

<authorInfo>

<id>000</id>

<name>STM</name>

<description>text</description>

<contactInfo>mail</contactInfo>

</authorInfo>

</authorInfos>

</requirement>

<requirement>

<id>VIS007</id>

<name>VIS007 - Events and data exchanged shall be stored and logged.</name>

<text>All data to and from VIS shall be stored and logged with metadata to support the validation of STM.

</text>

<rationale></rationale>

<reference></reference>

<authorInfos>

<authorInfo>

<id>000</id>

<name></name>

<description>text</description>

<contactInfo>mail</contactInfo>

</authorInfo>

</authorInfos>

</requirement>

</requirements>

<serviceDataModel>

<definitionAsXSD>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:simpleType name="messageType">

<xs:annotation>

<xs:documentation>Type of messages (including version)</xs:documentation>

</xs:annotation>

<xs:restriction base="xs:string">

<xs:enumeration value="RTZ">

<xs:annotation>

<xs:documentation>Route Exchange Format in XML</xs:documentation>

</xs:annotation>

</xs:enumeration>

<xs:enumeration value="TXT">

<xs:annotation>

<xs:documentation>STM Textmessage in XML</xs:documentation>

</xs:annotation>

</xs:enumeration>

<xs:enumeration value="S124">

<xs:annotation>

<xs:documentation>S124 in XML</xs:documentation>

</xs:annotation>

</xs:enumeration>

<xs:enumeration value="PCM">

<xs:annotation>

<xs:documentation>Port Call Message in XML</xs:documentation>

</xs:annotation>

</xs:enumeration>

</xs:restriction>

</xs:simpleType>

<xs:element name="DeliveryAck" type="DeliveryAck"/>

<xs:complexType name="DeliveryAck">

<xs:annotation>

<xs:documentation>Object for message ACK</xs:documentation>

</xs:annotation>

<xs:sequence>

<xs:element name="id" type="xs:string" minOccurs="1" maxOccurs="1">

<xs:annotation>

<xs:documentation>Id for the ACK</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="referenceId" type="URN" minOccurs="1" maxOccurs="1">

<xs:annotation>

<xs:documentation>Reference to delivered message according to the STM MRN identifier. For example an unique voyage identifier: urn:mrn:stm:voymgt:uvid:&lt;organizationId&gt;:&lt;local voyagenumber&gt;</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="timeOfDelivery" type="xs:dateTime" minOccurs="1" maxOccurs="1">

<xs:annotation>

<xs:documentation>Time of delivery</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="fromId" type="URN" minOccurs="1" maxOccurs="1">

<xs:annotation>

<xs:documentation>Identity of source (sender) of message that have been delivered according to the STM MRN identifier. Example: urn:mrn:stm:org:&lt;organizationId&gt;</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="fromName" type="xs:string" minOccurs="1" maxOccurs="1">

<xs:annotation>

<xs:documentation>Friendly name of sender</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="toId" type="URN" minOccurs="1" maxOccurs="1">

<xs:annotation>

<xs:documentation>Identity of target (receipient) of message delivery according to the STM MRN identifier. Example: urn:mrn:stm:org:&lt;organizationId&gt;</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="toName" type="xs:string" minOccurs="1" maxOccurs="1">

<xs:annotation>

<xs:documentation>Friendly name of recipient</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="ackResult" type="xs:string" minOccurs="1" maxOccurs="1"/>

</xs:sequence>

</xs:complexType>

<xs:element name="GetVPResponseObject" type="GetVPResponseObject"/>

<xs:complexType name="GetVPResponseObject">

<xs:sequence>

<xs:element name="code" type="xs:int" minOccurs="1" maxOccurs="1">

<xs:annotation>

<xs:documentation>Status code (20x, 40x)</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="subject" type="xs:string" minOccurs="1" maxOccurs="1">

<xs:annotation>

<xs:documentation>Message e.g. error text</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="lastInteractionTime" type="xs:dateTime" minOccurs="1" maxOccurs="1">

<xs:annotation>

<xs:documentation>Last interaction time with ship or ship representative.</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="route" type="rtz:route" minOccurs="0" maxOccurs="unbounded"/>

<xs:element name="route" type="route" minOccurs="1" maxOccurs="1"/>

</xs:sequence>

</xs:complexType>

<xs:simpleType name="routeStatus">

<xs:restriction base="xs:int">

<xs:enumeration value="Original">

<xs:annotation>

<xs:documentation>1</xs:documentation>

</xs:annotation>

</xs:enumeration>

<xs:enumeration value="Planned\_for\_voyage">

<xs:annotation>

<xs:documentation>2</xs:documentation>

</xs:annotation>

</xs:enumeration>

<xs:enumeration value="Optimized">

<xs:annotation>

<xs:documentation>3</xs:documentation>

</xs:annotation>

</xs:enumeration>

<xs:enumeration value="Cross\_Checked">

<xs:annotation>

<xs:documentation>4</xs:documentation>

</xs:annotation>

</xs:enumeration>

<xs:enumeration value="Safety\_Checked">

<xs:annotation>

<xs:documentation>5</xs:documentation>

</xs:annotation>

</xs:enumeration>

<xs:enumeration value="Approved">

<xs:annotation>

<xs:documentation>6</xs:documentation>

</xs:annotation>

</xs:enumeration>

<xs:enumeration value="Used\_for\_monitoring">

<xs:annotation>

<xs:documentation>7</xs:documentation>

</xs:annotation>

</xs:enumeration>

<xs:enumeration value="Inactive">

<xs:annotation>

<xs:documentation>8</xs:documentation>

</xs:annotation>

</xs:enumeration>

</xs:restriction>

</xs:simpleType>

<xs:element name="GetVoyagePlanObject" type="GetVoyagePlanObject"/>

<xs:complexType name="GetVoyagePlanObject">

<xs:sequence>

<xs:element name="UVID" type="URN" minOccurs="0" maxOccurs="1"/>

<xs:element name="routeStatus" type="routeStatus" minOccurs="0" maxOccurs="1"/>

</xs:sequence>

</xs:complexType>

<xs:element name="GetSubscriptionResponseObj" type="GetSubscriptionResponseObj"/>

<xs:complexType name="GetSubscriptionResponseObj">

<xs:annotation>

<xs:documentation>Object with array of dataId, in MRN format, such as a list of UVIDs.</xs:documentation>

</xs:annotation>

<xs:sequence>

<xs:element name="dataId" type="MRN" minOccurs="0" maxOccurs="unbounded">

<xs:annotation>

<xs:documentation>Data id in MRN format, such as UVID.</xs:documentation>

</xs:annotation>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:schema>

</definitionAsXSD>

</serviceDataModel>

<serviceInterfaces>

<serviceInterface>

<name>VIS Get Interface</name>

<description>Facilitates operations for requesting a Voyage Plan.

</description>

<dataExchangePattern>REQUEST\_RESPONSE</dataExchangePattern>

<operations>

<operation>

<name>getVoyagePlans</name>

<description>Returns active voyage plans according to parameters that the requester is authorized to. The response can contain 0 or more (0..\*) voyage plans, but only 0 or one (0..1) voyage plan with same UVID.

</description>

<returnValueType>

<typeReference>GetVPResponseObject</typeReference>

</returnValueType>

<parameterTypes>

<parameterType>

<typeReference>GetVoyagePlanObject</typeReference>

</parameterType>

</parameterTypes>

</operation>

</operations>

</serviceInterface>

<serviceInterface>

<name>VIS Subscription Interface</name>

<description>Facilitates operations for subscribing and unsubscribing to a Voyage Plan.

</description>

<dataExchangePattern>REQUEST\_RESPONSE</dataExchangePattern>

<operations>

<operation>

<name>subscribeToVoyagePlan</name>

<description>Facilitates request for subscription of voyage plans to the given callbackEndpoint.

</description>

<returnValueType>

<typeReference>void</typeReference>

</returnValueType>

<parameterTypes>

<parameterType>

<typeReference>URL</typeReference>

</parameterType>

<parameterType>

<typeReference>MRN</typeReference>

</parameterType>

</parameterTypes>

</operation>

<operation>

<name>getVoyagePlanSubscription</name>

<description>Get information on subscribed voyage plans.

</description>

<returnValueType>

<typeReference>GetSubscriptionResponseObj</typeReference>

</returnValueType>

<parameterTypes>

<parameterType>

<typeReference>URL</typeReference>

</parameterType>

</parameterTypes>

</operation>

<operation>

<name>removeVoyagePlanSubscription</name>

<description>Remove subscription of Voyage Plans

</description>

<returnValueType>

<typeReference>void</typeReference>

</returnValueType>

<parameterTypes>

<parameterType>

<typeReference>URL</typeReference>

</parameterType>

<parameterType>

<typeReference>MRN</typeReference>

</parameterType>

</parameterTypes>

</operation>

</operations>

<consumerInterface>

<name>VIS Upload Interface</name>

<description></description>

<operations>

<operation>

<name>uploadVoyagePlan</name>

<description></description>

<returnValueType>

<typeReference>void</typeReference>

</returnValueType>

<parameterTypes>

<parameterType>

<typeReference>URL</typeReference>

</parameterType>

<parameterType>

<typeReference>URL</typeReference>

</parameterType>

<parameterType>

<typeReference>rtz:route</typeReference>

</parameterType>

</parameterTypes>

</operation>

</operations>

</consumerInterface>

</serviceInterface>

<serviceInterface>

<name>VIS Upload Interface</name>

<description>Facilitates operations for uploading a Voyage Plan, Text Message and Area.

</description>

<dataExchangePattern>REQUEST\_RESPONSE</dataExchangePattern>

<operations>

<operation>

<name>uploadVoyagePlan</name>

<description>Facilitates sending (uploading) a voyage plan to VIS. The route shall be uncompressed (RTZ).

</description>

<returnValueType>

<typeReference>ResponseObj</typeReference>

</returnValueType>

<parameterTypes>

<parameterType>

<typeReference>URL</typeReference>

</parameterType>

<parameterType>

<typeReference>URL</typeReference>

</parameterType>

<parameterType>

<typeReference>rtz:route</typeReference>

</parameterType>

</parameterTypes>

</operation>

<operation>

<name>uploadTextMessage</name>

<description>Facilitates sending (uploading) a text message to VIS to be forwarded to the ship (STM Module).

If endpoint/URL provided for deliveryACK, an ACK will be sent when message has been forwarded to private application.

</description>

<returnValueType>

<typeReference>ResponseObj</typeReference>

</returnValueType>

<parameterTypes>

<parameterType>

<typeReference>stm:textMessage</typeReference>

</parameterType>

<parameterType>

<typeReference>URI</typeReference>

</parameterType>

</parameterTypes>

</operation>

<operation>

<name>uploadArea</name>

<description>Facilitates sending (uploading) an area to VIS to be forwarded to the ship (STM Module).

If endpoint/URL provided for deliveryACK, an ACK will be sent when message has been forwarded to private application.

</description>

<returnValueType>

<typeReference>ResponseObj</typeReference>

</returnValueType>

<parameterTypes>

<parameterType>

<typeReference>S124:DataSet</typeReference>

</parameterType>

<parameterType>

<typeReference>URI</typeReference>

</parameterType>

</parameterTypes>

</operation>

</operations>

<consumerInterface>

<name>VIS Acknowledgement Interface</name>

<description></description>

<operations>

<operation>

<name>acknowledgement</name>

<description></description>

<returnValueType>

<typeReference>void</typeReference>

</returnValueType>

<parameterTypes>

<parameterType>

<typeReference>DeliveryAck</typeReference>

</parameterType>

</parameterTypes>

</operation>

</operations>

</consumerInterface>

</serviceInterface>

</serviceInterfaces>

</ServiceSpecificationSchema:serviceSpecification>

# Document lifecycle

## Maturity

The document describes a feature for future use to enable push from a producer of port call messages to a consumer. The interface may need to be revised when such service producers have been designed.

## Forecast

The foreseen updates on the document is the following:

|  |  |  |
| --- | --- | --- |
| **Chapter** | **Rationale** | **Time and version** |
|  | Document reviewed and accepted as version 1.0 | 2016 November in v1.0 |
|  | Revisit the service design when push functionality implemented in producer end |  |
|  |  |  |