

*Document No:*

*Title: Ship Port Information Service REST Design - Private*

*Date: 2017-04-03*



Authors

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

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Initials** | **Description** |
| Version 0.1 | 2016-05-16 | MO | Created based on SPIS Specification 1.2  + Updated the private API with same API as VIS  - publishMessage to send PCM  - authorizeIdentities to set a access and distribution list |
| Version 0.2 | 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 0.3 | 2016-11-14 | MO | Update after review   * findService updated, both input and output * findIdentities updated, parameters removed, the result is all identities the requester have access to |
| Version 1.0 | 2017-03-31 | MO | Update after review   * Swagger and document aligned * Swagger: getMessage, parameter dataId removed |

Review

|  |  |
| --- | --- |
| **Name** | **Organisation** |
|  |  |

**Table of contents**

1 Introduction 5

1.1 Purpose of the document 5

1.2 Intended readership 5

1.3 Inputs from other projects 6

2 Service Design Identification 7

3 Technology Introduction 8

3.1 REST 8

3.2 Swagger 8

4 Service Design Overview 9

4.1 Service Interface Design - Private side 9

4.1.1 Service Interfaces 11

5 Physical Data Model - Private side 12

5.1.1 MessageEnvelope 13

5.1.2 Message 13

5.1.3 enumeration\_messageType 14

5.1.4 portCallMessage 15

5.1.5 Notification 15

5.1.6 enumeration\_notificationType 17

5.1.7 identityDescriptionObject 17

5.1.8 subscriptionObjectSPIS 18

5.1.9 ResponseObj 19

5.2 Physical Data Model - Private side SSC related 20

5.2.1 callServiceRequestObj 21

5.2.2 header 21

5.2.3 enumeration\_requestType 22

5.2.4 callServiceResponseObj 22

5.2.5 findIdentitiesResponseObj 22

5.2.6 Organization 23

5.2.7 findServicesRequestObj 24

5.2.8 Filter 24

5.2.9 FindServicesRequestObjFilterCoverageArea 26

5.2.10 findServicesResponseObj 27

5.2.11 ServiceInstance 27

5.2.12 Xml 28

6 Service Interface Design - Private side 30

6.1 Ship Port Information Service Private REST 30

6.1.1 SPIS Private Interface 30

6.1.2 SPIS Private Subscription Interface 33

6.1.3 SPIS Private ACL Interface 34

6.1.4 SPIS Private Service Support Interface 36

7 Service Dynamic Behaviour 38

7.1 Service Dynamic Behaviour 38

7.1.1 Service Sequence diagrams 39

7.1.2 Logging 45

8 References 45

9 Acronyms and Terminology 46

9.1 Acronyms 46

9.2 Terminology 46

10 APPENDIX API Swagger (JSON) 51

11 Document lifecycle 68

11.1 Maturity 68

11.2 Forecast 68

# Introduction

## Purpose of the document

The purpose of this service design description document is to provide a detailed description of the service, realized by using a specific the technology, according to the guidelines given in Service Documentation Guidelines. It describes a well-defined baseline of the service design by clearly identifying the service design version.

The aim is to document the key aspects of the service technical design. This includes:

* identification and summary of the service design
  + reference to the service specification
  + identification of the service design
* identification and summary of chosen technology
* detailed description about the realization of each service interface and service operation
  + mapping of interfaces to the chosen technology
  + mapping of operations to the chosen technology
  + mapping of the message exchange patterns to the chosen technology
* detailed description of the physical data model
  + mapping to the service data model of the service specification.

## 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 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 projects

No information.

# Service Design Identification

|  |  |
| --- | --- |
| **Name** | Ship Port Information Service Design - Private side |
| **ID** | urn:mrn:stm:service:design:sma:spis:private |
| **Version** | 1.0 |
| **Description** | Service (private side) for publish Port Call Messages to authorized actors, and to receive Port Call Messages |
| **Keywords** | Ship Port Information Service, SPIS, PCM |
| **Architect(s)** | Per Löfbom, Mikael Olofsson |
| **Status** | Provisional |

# Technology Introduction

This service design is realized using RESTful API’s described in JSON using the Swagger interface.

## REST

REST (REpresentational State Transfer) is an architectural style, and an approach to communications that is often used in the development of [Web services](http://searchsoa.techtarget.com/definition/Web-Services-Glossary). The use of REST in preferred over the more heavyweight [SOAP](http://searchsoa.techtarget.com/definition/SOAP) (Simple Object Access Protocol) style 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](http://searchwindevelopment.techtarget.com/definition/HTTP) (Hypertext Transfer Protocol), has several architectural constraints:

* *Decoupling* – Decouples consumers from producers which suits SeaSWIM decentralized architecture well.
* [*Stateless*](http://whatis.techtarget.com/definition/stateless) *existence* – Also a good prerequisite for a decentralized architecture design.
* *Able to leverage a* [*cache*](http://searchstorage.techtarget.com/definition/cache) – Probably less important in SeaSWIM since most of the interaction is between machines, although for services with man-machine interfaces this is of importance.
* *Leverages a layered system* – SeaSWIM is dependant on good scaling capabilities which has REST support.
* *Leverages a uniform interface* – Again since SeaSWIM defines the available services centrally in a Service registry this constraint supports implementations being decoupled from the services they provide.

## Swagger

Swagger is a simple yet powerful representation of RESTful API. With the largest ecosystem of API tooling on the planet, thousands of developers are supporting Swagger in almost every modern programming language and deployment environment. With a Swagger-enabled API, you get interactive documentation, client and server SDK generation together with discoverability.

A reference to provided Swagger JSON file is included in the Service Design Xml description.

References:

* Fielding, Roy Thomas (2000). [*"Chapter 5: Representational State Transfer (REST)"*](http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm). Architectural Styles and the Design of Network-based Software Architectures (Ph.D.). University of California, Irvine.
* Richardson, Leonard; Ruby, Sam (2007), [*RESTful Web service*](http://books.google.com/books?id=XUaErakHsoAC), O'Reilly Media, [*ISBN*](https://en.wikipedia.org/wiki/International_Standard_Book_Number/oInternationalStandardBookNumber)[*978-0-596-52926-0*](https://en.wikipedia.org/wiki/Special:BookSources/978-0-596-52926-0/oSpecial:BookSources/978-0-596-52926-0)*, retrieved 18 January 2011*.
* Richardson, Leonard; Amundsen, Mike (2013), [*RESTful Web APIs*](http://www.amazon.com/RESTful-Web-APIs-Leonard-Richardson/dp/1449358063/ref=sr_1_1?ie=UTF8&qid=1442372039&sr=8-1&keywords=restful+web+apis), O'Reilly Media, [*ISBN*](https://en.wikipedia.org/wiki/International_Standard_Book_Number/oInternationalStandardBookNumber)[*978-1-449-35806-8*](https://en.wikipedia.org/wiki/Special:BookSources/978-1-449-35806-8/oSpecial:BookSources/978-1-449-35806-8)*, retrieved 15 September 2015*
* Swagger Open API specification - http://swagger.io/specification/

# Service Design Overview

## Service Interface Design - Private side

The service is designed in REST and each operation in each interface has a defined REST endpoint.

The private interface exposes service endpoints in REST for publish and get messages, Access Control and support to search for identities and services and to consume other services.



### Service Interfaces

The table below shows the REST interface designed for the corresponding operation in the Service Specification.

In the table, only the mandatory parameters are shown. For detailed description of each operation including optional parameters, see chapter 6.

| **Service Specification** | **Service Design** |
| --- | --- |
| **Service Interface** | **Service REST Operation** |
| SPIS Private Interface | |  |  | | --- | --- | | **REST** | **Operation-id** | | POST /InstanceURL/publishMessage | publishMessage | | GET /InstanceURL/getPublishMessages | getPublishedMessages | | DELETE /InstanceURL/publishMessage | removePublishedMessage | | GET /InstanceURL/notification | getNotification | | GET /InstanceURL/getMessage | getMessage | |
| SPIS Private Subscription Interface | |  |  | | --- | --- | | **REST** | **Operation-id** | | POST /instanceURL/subscription | addSubscriptions | | GET /instanceURL/subscription | getSubscriptions | | DELETE /instanceURL/subscription | removeSubscriptions | |
| SPIS Private ACL Interface | |  |  | | --- | --- | | **REST** | **Operation-id** | | POST /InstanceURL/authorizeIdentities | authorizeIdentities | | GET /InstanceURL/authorizeIdentities | findAuthorizedIdentitites | | DELETE /InstanceURL/authorizeIdentities | removeAuthorizedIdentitites | |
| SPIS Private Service Support Interface | |  |  | | --- | --- | | **REST** | **Operation-id** | | GET /InstanceURL/findServices | findServices | | POST /InstanceURL/callService | callService | | GET /InstanceURL/findIdentities | findIdentities | |

# Physical Data Model - Private side

The service exchanges the following data.



### MessageEnvelope

Container for messages to STM Module

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **MessageEnvelope** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | numberOfMessages | int | | Total number of messages in this container | | | remainingNumberOfMessages | int | | Total number of remaining messages waiting to be fetched | | | message | Message | | Uploaded message(s) | | |

**Data definition in JSON**

"MessageEnvelope": {

"description": "",

"type": "object",

"properties": {

"numberOfMessages": {

"format": "int32",

"description": "Gets or Sets NumberOfMessages",

"type": "integer"

},

"remainingNumberOfMessages": {

"format": "int32",

"description": "Gets or Sets RemainingNumberOfMessages",

"type": "integer"

},

"message": {

"description": "Gets or Sets Message",

"type": "array",

"items": {

"$ref": "#/definitions/Message"

}

}

}

}

### Message

Message to the STM Module

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **Message** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | id | MRN | | Identity of the message | | | receivedAt | dateTime | | Date and time of reception in UTC (e.g. 2017-03-29T11:33:00Z | | | fromServiceId | MRN | | Entity Identity in MRN format of the message source | | | fromOrgId | MRN | | Organisation Identity in MRN format of the message source | | | fromOrgName | string | | Organisation Identity in readable format | | | callbackEndpoint | URL | | callbackEndpoint as baseURL where to upload expected result | | | messageType | enumeration\_messageType | | Type of STM message | | | message | string | | The message of any STM format | | |

**Data definition in JSON**

"Message": {

"description": "",

"type": "object",

"properties": {

"CallbackEndpoint": {

"description": "Gets or sets CallbackEndpoint",

"type": "string"

},

"id": {

"description": "Gets or Sets Id",

"type": "string"

},

"receivedAt": {

"format": "date-time",

"description": "Gets or Sets ReceivedAt",

"type": "string"

},

"FromOrgId": {

"description": "Gets or Sets FromOrgId",

"type": "string"

},

"FromOrgName": {

"description": "Gets or Sets FromOrgName",

"type": "string"

},

"FromServiceId": {

"description": "Gets or Sets FromServiceId",

"type": "string"

},

"messageType": {

"description": "Gets or Sets MessageType",

"type": "string"

},

"stmMessage": {

"$ref": "#/definitions/StmMessage",

"description": "Gets or Sets StmMessage"

}

}

}

### enumeration\_messageType

Type of messages

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **enumeration\_messageType** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | PCM | string | | Port Call Message in XML | | |

### portCallMessage

Port Call Message Format (PCMF). See references for more detailed information.

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **portCallMessage** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | portCallId | pcm:PortCallIdentifier | |  | | | messageId | pcm:PortCallMessageIdentifier | |  | | | groupWith | pcm:PortCallMessageIdentifier | |  | | | reportedAt | dateTime | |  | | | reportedBy | string | |  | | | timeType | pcm:TimeType | |  | | | time | dateTime | |  | | | comment | string | |  | | | locationState | LocationState | |  | | | serviceState | ServiceState | |  | | |

### Notification

To inside/private application, such as STM Module

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **Notification** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | dataId | string | | Identity of the notification and the stored message which can be retrieved with "getMessage(dataId)", mandatory | | | notificationType | enumeration\_notificationType | | Type of notification by enumeration | | | notificationCreatedAt | dateTime | | Notification created at date and time in UTC (e.g. 2017-03-29T11:33:00Z), mandatory | | | fromOrgId | MRN | | Organisation identifier of the source in focus  Example: urn:mrn:stm:org:sma | | | fromOrgName | string | | Friendly name of sender organisation for presentation | | | receivedAt | dateTime | | Date and time for the reception of the message in UTC (e.g. 2017-03-29T11:33:00Z), mandatory | | | messageWaiting | int | | >0 if a message is waiting in server, otherwise 0, mandatory | | | subject | string | | Notification subject, mandatory | | | body | string | | Notification body, optional | | |

**Data definition in JSON**

"Notification": {

"required": ["Body", "FromOrgId", "FromServiceId", "MessageWaiting", "NotificationCreatedAt", "NotificationType", "ReceivedAt", "Subject", "NotificationSource"],

"type": "object",

"properties": {

"Body": {

"type": "string"

},

"FromOrgId": {

"type": "string"

},

"FromOrgName": {

"type": "string"

},

"FromServiceId": {

"type": "string"

},

"MessageWaiting": {

"format": "int32",

"type": "integer"

},

"NotificationCreatedAt": {

"format": "date-time",

"type": "string"

},

"NotificationType": {

"enum": ["MESSAGE\_WAITING", "UNAUTHORIZED\_REQUEST", "ACKNOWLEDGEMENT\_RECEIVED", "ERROR\_MESSAGE"],

"type": "string"

},

"ReceivedAt": {

"format": "date-time",

"type": "string"

},

"Subject": {

"type": "string"

},

"NotificationSource": {

"enum": ["VIS", "SPIS"],

"type": "string"

}

}

}

### enumeration\_notificationType

Types of notifications

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **enumeration\_notificationType** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | MESSAGE\_WAITING | int | | Uploaded message queued in VIS which can be retrieved with getMessage. Messages can be of types RTZ, TXT, S124 or PCM. | | | UNAUTHORIZED\_REQUEST | int | | Request for information from unauthorized actor. | | | ACKNOWLEDGEMENT\_RECEIVED | int | | Acknowledgement received | | | ERROR\_MESSAGE | int | | Subject describes type of failure, such as | | |

### identityDescriptionObject

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **identityDescriptionObject** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | identityId | MRN | | Identity in MRN format registered in Identity Registry. | | | identityName | string | | Human readable name of the identity | | |

**Data definition in JSON**

"IdentityDescriptionObject": {

"description": "",

"type": "object",

"properties": {

"identityId": {

"description": "identity in urn format according to ID registry",

"type": "string"

},

"identityName": {

"description": "Identity shortname in ID registry",

"type": "string"

}

}

}

### subscriptionObjectSPIS

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **subscriptionObjectSPIS** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | identityId | MRN | | Identity of the subscriber in MRN format | | | identityName | string | | Identity name | | | AmssEndpointURL | URL | | Subscribers endpointURL (callback baseURL) where to upload subscribed messages | | | MbEndpointURL | URL | | Subscribers endpointURL (callback baseURL) where to create and poll queue for retrieving messages | | |

**Data definition in JSON**

"SubscriptionObject": {

"description": "",

"type": "object",

"properties": {

"IdentityId": {

"description": "",

"type": "string"

},

"IdentityName": {

"description": "",

"type": "string"

},

"MbEndpointURL": {

"description": "",

"type": "string"

},

"AmssEndpointURL": {

"description": "",

"type": "string"

}

}

}

### ResponseObj

Generic response object

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **ResponseObj** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | body | string | | Error (message) text to complement status code | | |

**Data definition in JSON**

"ResponseObj": {

"description": "",

"type": "object",

"properties": {

"dataId": {

"description": "Gets or Sets DataId",

"type": "string"

}

}

}

## Physical Data Model - Private side SSC related



### callServiceRequestObj

Contain the data needed by the callService function to execute the request. The parameters contained in the callServiceRequestObj are the general parameters contained in a standard HTTP request.

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **callServiceRequestObj** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | requestType | enumeration\_requestType | | Type of request according to enumeration | | | endpoint\_method | URL | | URL contains <ipaddress>:<port>/<request> | | | body | string | | Request body needed by the requested service | | | headers | string | | Request headers needed by the requested service | | |

**Data definition in JSON**

"CallServiceRequestObj": {

"type": "object",

"properties": {

"body": {

"type": "string"

},

"endpointMethod": {

"type": "string"

},

"headers": {

"type": "array",

"items": {

"$ref": "#/definitions/Header"

}

},

"requestType": {

"type": "string"

}

}

}

### header

Header to be included in HTTP call

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **header** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | key | string | |  | | | value | string | |  | | |

**Data definition in JSON**

"Header": {

"type": "object",

"properties": {

"key": {

"type": "string"

},

"value": {

"type": "string"

}

}

}

### enumeration\_requestType

Type of request to be used in callService

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **enumeration\_requestType** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | GET | string | |  | | | POST | string | |  | | | DELETE | string | |  | | | PATCH | string | |  | | |

**Data definition in JSON**

### callServiceResponseObj

This object is a response container with response from called service

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **callServiceResponseObj** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | statusCode | int | | HTTP Status codes as described by the consumed service | | | body | string | | Request Body as described in swagger of the consumed service | | |

**Data definition in JSON**

"CallServiceResponseObj": {

"type": "object",

"properties": {

"body": {

"type": "string"

},

"statusCode": {

"format": "int64",

"type": "integer"

}

}

}

### findIdentitiesResponseObj

Contains the result from search in Identity Registry.

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **findIdentitiesResponseObj** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | statusCode | int | | HTTP Status code | | | statusMessage | string | | Error message | | | organization | Organization | | Object with response | | |

**Data definition in JSON**

"FindIdentitiesResponseObj": {

"type": "object",

"properties": {

"organizations": {

"type": "array",

"items": {

"$ref": "#/definitions/Organization"

}

},

"statusCode": {

"format": "int64",

"type": "integer"

},

"statusMessage": {

"type": "string"

}

}

}

### Organization

Response from search in Identity Registry.

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **Organization** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | country | string | | Country of organization | | | email | string | | The email to organization contact | | | mrn | MRN | | The organization identity in MRN format | | | name | string | | Short name for organization | | |

**Data definition in JSON**

"Organization": {

"type": "object",

"properties": {

"country": {

"type": "string"

},

"email": {

"type": "string"

},

"mrn": {

"type": "string"

},

"name": {

"type": "string"

}

}

}

### findServicesRequestObj

Contains parameters for search in Service Registry

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **findServicesRequestObj** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | filter | Filter | | Set of search criterias provided. If none given, "all" service instances will be returned | | | page | int | | If provided, the response will start at given page.  Enables consumer to limit response for each call | | | pageSize | int | | If provided, the response will be limited to given number of pages (service instance descriptions) | | |

**Data definition in JSON**

"FindServicesRequestObj": {

"type": "object",

"properties": {

"filter": {

"$ref": "#/definitions/FindServicesRequestObjFilter"

},

"page": {

"format": "int64",

"type": "integer"

},

"pageSize": {

"format": "int64",

"type": "integer"

}

}

}

### Filter

Filter parameters correlated with the elastic search in Service Registry.

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **Filter** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | keyWords | string | | Search values separated by space for search in keywords section.  Can be used in combination with all other parameters except freeText. | | | ServiceProviderIds | URN | | Array of Service provider identities (organizationId) in MRN format.  Can be used in combination with all other parameters except freeText. | | | UnLoCode | string | | Search value for UN/LOCODE (5 characters without space)  Can be used in combination with all other parameters except freeText and coverageArea | | | serviceDesignId | URN | | Service Design Id in MRN format  Can be used in combination with all other parameters except freeText. | | | serviceInstanceId | URN | | Service instance id in MRN format  Can be used in combination with all other parameters except freeText. | | | mmsi | string | | MMSI with only numbers [0-9]  Can be used in combination with all other parameters except freeText. | | | imo | string | | IMO with numbers [0-9]  Can be used in combination with all other parameters except freeText. | | | serviceType | string | | Service type [VIS | ROS | RCS | EMS tbd]  Can be used in combination with all other parameters except freeText. | | | serviceStatus | string | | Search value for service [Provision | Live | Simulation]  Can be used in combination with all other parameters except freeText. | | | freeText | string | | Search value for free text search.  Cannot be combined with any other search criteria. | | | coverageArea | FindServicesRequestObjFilterCoverageArea | | Search value for UN/LOCODE (5 characters without space)  Can be used in combination with all other parameters except freeText and UnLoCode | | |

**Data definition in JSON**

"FindServicesRequestObjFilter": {

"type": "object",

"properties": {

"coverageArea": {

"$ref": "#/definitions/FindServicesRequestObjFilterCoverageArea"

},

"UnLoCode": {

"type": "string"

},

"ServiceProviderIds": {

"type": "array",

"items": {

"type": "string"

}

},

"serviceDesignId": {

"type": "string"

},

"serviceInstanceId": {

"type": "string"

},

"mmsi": {

"type": "string"

},

"imo": {

"type": "string"

},

"serviceType": {

"type": "string"

},

"serviceStatus": {

"type": "string"

},

"keyWords": {

"type": "array",

"items": {

"type": "string"

}

},

"freeText": {

"type": "string"

}

}

}

### FindServicesRequestObjFilterCoverageArea

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **FindServicesRequestObjFilterCoverageArea** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | coverageType | string | |  | | | value | string | |  | | |

**Data definition in JSON**

"FindServicesRequestObjFilterCoverageArea": {

"type": "object",

"properties": {

"coverageType": {

"type": "string"

},

"value": {

"type": "string"

}

}

}

### findServicesResponseObj

Contains the result from search in Identity Registry.

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **findServicesResponseObj** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | statusCode | int | | HTTP Status codes | | | statusMessage | string | | Following the Service Registry Specification | | | serviceInstanceDescription | ServiceInstance | | Object with response | | |

**Data definition in JSON**

"FindServicesResponseObj": {

"type": "object",

"properties": {

"servicesInstances": {

"type": "array",

"items": {

"$ref": "#/definitions/ServiceInstance"

}

},

"statusCode": {

"format": "int64",

"type": "integer"

},

"statusMessage": {

"type": "string"

}

}

}

### ServiceInstance

Response from search in Service Registry

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **ServiceInstance** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | comment | string | | Description of the service instance | | | endpointType | string | | ?? | | | endpointUri | URL | | The base URL to the service instance | | | instanceAsXml | string | | The service instance as XML, as registered by service provider | | | instanceId | MRN | | Service instance ID in MRN format | | | keywords | string | | Keywords for the service instance | | | name | string | | Name of the service instance | | | organizationId | MRN | | Providing organization in MRN format | | | status | string | | Status on the service instance; provisional, released, deleted, deprecated, simulated | | | unlocode | string | | UN/LOCODE as 5 capital characters | | | version | string | | Version of the service instance | | |

**Data definition in JSON**

"ServiceInstance": {

"type": "object",

"properties": {

"comment": {

"type": "string"

},

"endpointType": {

"type": "string"

},

"endpointUri": {

"type": "string"

},

"instanceAsXml": {

"$ref": "#/definitions/Xml"

},

"instanceId": {

"type": "string"

},

"keywords": {

"type": "string"

},

"name": {

"type": "string"

},

"organizationId": {

"type": "string"

},

"status": {

"type": "string"

},

"unlocode": {

"type": "string"

},

"version": {

"type": "string"

}

}

}

### Xml

|  |  |
| --- | --- |
| **Element Name** | **Attributes** |
| **Xml** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Name** | | **Type** | | **Description** | | comment | string | |  | | | content | string | |  | | | contentContentType | string | |  | | | id | int | |  | | | name | string | |  | | |

**Data definition in JSON**

"Xml": {

"type": "object",

"properties": {

"comment": {

"type": "string"

},

"content": {

"type": "string"

},

"contentContentType": {

"type": "string"

},

"id": {

"format": "int64",

"type": "integer"

},

"name": {

"type": "string"

}

}

}

# Service Interface Design - Private side

The service is designed in REST and each operation in each interface has a defined REST endpoint.

## Ship Port Information Service Private REST

The service private side provides interfaces for publish and get messages, handle authorization, searching for identities and services and to consume/invoke services in the SeaSWIM domain.

### SPIS Private Interface

Message exchange pattern: REQUEST\_RESPONSE

Private interface to sending and receiving messages.

#### POST /publishMessage

Operation id from specification: publishMessage()

Enables publish messages in STM format to the service which will store and forward it to subscribers.

When a publishMessage request is received, the message payload is validated and subsequently stored and distributed.

Endpoints for subscribers are read from the subscription table and the SSC exposed interface for consuming a service, callService is invoked. Hereby passing the relevant payload received in the STMMessage together with endpoint (URI) for the called service found in the subscription table.

In case the schema validation fails a message informing the caller of an invalid format is returned.

**Request type POST**

Endpoint path: /publishMessage

**In Parameters**

dataId in MRN format. The same dataId shall be used in authorizeIdentities, mandatory

messageType "PCM", mandatory

**In Body**

PCM in JSON (application/json) according to schema, mandatory

**Return**

http code

**Where return codes are**

200=Successful

400=Bad Request

500=Internal Server Error

**Operation functionality**

Validate the schema

Store the message

Forward the message to all subscribers according to subscription parameters and authorization (Access Control List)

#### GET /getPublishedMessages

Operation id from specification: getPublishedMessages()

Enable private application to fetch published messages from SPIS to check consistency.

**Request type GET**

Endpoint path: /getPublishedMessages

**In Parameters**

-

**In Body**

-

**Return**

http code

array of publishedMessages in JSON (application/json)

**Where return codes are**

200=Successful

400=Bad Request

401=Unauthorized

403=Forbidden

500=Internal Server Error

**Operation functionality**

#### DELETE /publishedMessage

Operation id from specification: removePublishedMessage()

Enables removing a published message in SPIS,

**Request type DELETE**

Endpoint path: /publishedMessage

**In Parameters**

dataId in MRN format is mandatory, e.g. "urn:mrn:stm:voyage:id:sma:voyage-001"

**In Body**

-

**Return**

http code

**Where return codes are**

200=Successful

400=Bad Request

401=Unauthorized

403=Forbidden

500=Internal Server Error

**Operation functionality**

#### GET /getMessage

Operation id from specification: getMessage()

Enables the internal application (such as STM Module) to retrieve incoming messages waiting in the service. The messages can be of type any valid STM message format. A limit can be given which allows setting a limit on number of messages in the answer. Information on remaining messages waiting is also returned.

**Request type GET**

Endpoint path: /getMessage

**In Parameters**

limitQuery is optional.

**In Body**

No body

**Return**

http code

MessageEnvelope object including a sequence of Message, where number is limited to limitQuery

**Operation functionality**

All messages that has not been fetched is sent as response limited by the parameter.

All messages sent is flagged as fetched with timestamp

The response contains information of remaining number of messages that are waiting.

#### GET /getNotification

Operation id from specification: getNotification()

Enables the STM Module to check for notifications as complement to push functionality of notifications.

All non-fetched notifications will be returned ordered by event time.

**Request type GET**

Endpoint path: /getNotification

**In Parameters**

None

**In Body**

No body

**Return**

http code

Sequence of Notification object (0..\*) in JSON format (application/json)

**Operation functionality**

Return all non-fetched notifications

### SPIS Private Subscription Interface

Message exchange pattern: REQUEST\_RESPONSE

#### POST /subscription

Operation id from specification: addSubscriptions()

Facilitates adding (nominating) actor as subscriber to a certain information object. In SPIS the information object is Port Call Messages related to a Port Call.

When successfully added consumer (service) as subscriber, the consumer receives latest published PortCallMessage for the related Port Call (identified by dataId).

**Internal logic!**

+ The service type AMSS shall be added as subscriber and thus shall receive PCM on the state\_update operation.

+ The service type MB shall be used to create queue(Filter) and initiate timer to poll the queue regularly.

Thus the parent application needs to add both PortCDM AMSS and PortCDM MB services as subscribers (in the current solution).

The function should be called in collaboration with authorizeIdentities and publishMessage.

**Request type POST**

Endpoint path: /subscription

**In Parameters**

dataId is required in MRN format. The same dataId shall be used in publishMessage. In SPIS this would be Port Call Identifier, global or local.

**In Body**

array of subscriptionObject in JSON (application/json) required

**Return**

http code

**Operation functionality**

Add AMSS as subscriber to PCID

Create queue on MB

Initiate poll of queue

Send latest published PCM related to PCID

#### GET /subscription

Operation id from specification: getSubscriptions()

Facilitates retrieving actors subscribing to a certain information object. In SPIS the information object is Port Call Messages related to a Port Call.

**Request type GET**

Endpoint path: /subscription

**In Parameters**

dataId is required in MRN format. In SPIS this would be Port Call Identifier, global or local.

**In Body**

-

**Return**

http code

array of subscriptionObject in JSON (application/json)

#### DELETE /subscription

Operation id from specification: removeSubscriptions()

Facilitates removing actor(s) as subscriber to a certain information object. In SPIS the information object is Port Call Messages related to a Port Call.

Internal logic:

+ Remove corresponding queue

**Request type DELETE**

Endpoint path: /subscription

**In Parameters**

dataId is required in MRN format. In SPIS this would be Port Call Identifier, global or local.

**In Body**

array of subscriptionObject in JSON (application/json) required

**Return**

http code

**Operation functionality**

Remove subscriber at PCID

Remove queue

### SPIS Private ACL Interface

Message exchange pattern: REQUEST\_RESPONSE

Private interface to handle authorization and distribution list.

#### POST /authorizeIdentities

Operation id from specification: authorizeIdentities()

Give access (authorization) to a specific information object to a set of identities.

The function should be called in collaboration with publishMessage.

**Request type POST**

Endpoint path: /authorizeIdentities

**In Parameters**

dataId is required in MRN format. The same dataId shall be used in publishMessage

**In Body**

identityDescriptionObject in JSON (application/json) is required with 0 or more identities in MRN format

**Return**

http code

**Operation functionality**

Create a POSIX Access Control List of the given identities and data object id

#### DELETE /authorizeIdentities

Operation id from specification: removeAuthorizedIdentitites()

Removes a set of identitites from Access Control List for dataId.

**Request type DELETE**

Endpoint path: /authorizeIdentities

**In Parameters**

dataId is required in MRN format.

**In Body**

identityDescriptionObject in JSON (application/json) is required with 0 or more identities in MRN format

**Return**

http code

**Operation functionality**

Remove records from stored Access Control list for a specific dataId, a specific organizationId, a combination of dataId and organizationId or all Access Control records.

#### GET /authorizeIdentities

Operation id from specification: findAuthorizedIdentitites()

Returns the set of identitites that are authorized to dataId

**Request type GET**

Endpoint path: /authorizeIdentities

**In Parameters**

dataId is required in MRN format.

**In Body**

No body

**Return**

http code

Sequence of identityDescriptionObjects

**Operation functionality**

Find authorized organizations for a specific dataId in the Access Control list stored in VIS.

### SPIS Private Service Support Interface

Message exchange pattern: REQUEST\_RESPONSE

Provides support interface to SeaSWIM core services via SeaSWIM Connector. Includes search for identities and services, and consume/invoke services in the SeaSWIM domain.

#### GET /findIdentities

Operation id from specification: findIdentities()

The operation is directly propagated to SeaSWIM Connector service operation findIdentities.

Find identities in Identity Registry to be used in e.g. authorization.

Returns a list of organizations.

**Request type GET**

Endpoint path: /findIdentities

**In Parameters**

-

**In Body**

-

**Return**

http code

JSON object findIdentitiesResponseObj

**Where return codes are**

200=Successful

400=Bad Request

401=Unauthorized

403=Forbidden

500=Internal Server Error

**Operation functionality**

Call SeaSWIM Connector to find organizations in SeaSWIM ID registry and return the response.

#### GET /findServices

Operation id from specification: findServices()

The operation is directly propagated to SeaSWIM Connector service operation findService.

Facilitates service discovery in the central SeaSWIM service registry using query service parameters like service type, service category, location, service id etc.

The response is a list of endpoints and corresponding service descriptions.

**Request type GET**

Endpoint path: /findServices

**In Parameters**

-

**In Body**

JSON object findServicesRequestObj

**Return**

http code

JSON object findServicesResponseObj

**Where return codes are**

200=Successful

400=Bad Request

401=Unauthorized

403=Forbidden

500=Internal Server Error

**Operation functionality**

Call SeaSWIM Connector to find services in SeaSWIM service registry and return the response.

#### POST /callService

Operation id from specification: callService()

The operation is directly propagated to SeaSWIM Connector service operation callService.

Supports consumption of other information services through the service and SSC assisting with authentication and secure transfer.

Invokes a service instance and returns the result. This service is transparent regarding both input and output to the consumed service.

**Request type POST**

Endpoint path: /callService

**In Parameters**

-

**In Body**

JSON object callServiceRequestObj

**Return**

http code

JSON object callServiceResponseObj

**Where return codes are**

200=Successful

400=Bad Request

401=Unauthorized

403=Forbidden

500=Internal Server Error

**Operation functionality**

Call SeaSWIM Connector to invoke SeaSWIM published services and return the response.

# Service Dynamic Behaviour

## Service Dynamic Behaviour

The diagram describes the expected internal functionality of the Ship-Port Information Service based on present decisions.



### Service Sequence diagrams

This section contains interaction diagrams (sequence diagrams) to show the interactions to SPIS interface.

#### Interaction Private authorizeIdentities

Message exchange pattern: REQUEST\_RESPONSE

The STM Module (private consumer) authorizes a dataId (created by STM Module or port) to a set of identities that shall have access to the published information.

SPIS creates an Access Control List (ACL) and searches for PortCDM service endpoints based on authorized identities. Each PortCDM endpoint is stored in a subscription list that is used when publishing PCM messages.



#### Interaction Private addSubscriber

Message exchange pattern: REQUEST\_RESPONSE

Add PortCDM MB and AMSS services for a Port as subscribers to published PCM messages related to a Port Call ID. The MB (Message Broker) service is used for creating and later polling PortCDM queue in the Port to retrieve messages from Port to ship. The AMSS (Assisted Message Submission Service) is used for sending Port Call Messages to PortCDM.

The organisation must be authorized before accepting to be subscriber.



#### Interaction Private publishMessage

Message exchange pattern: REQUEST\_RESPONSE

The STM Module (private consumer) publishes messages in PCM format (Port Call Message) with same dataId as was used in authorizedIdentities. The PCM messages is forwarded by SPIS to the PortCDM services declared in the subscription related to the ID.



#### Interaction Private Notify

Message exchange pattern: REQUEST\_RESPONSE

When a message is received in SPIS, either retrieved from polling a queue or pushed to SPIS, the message generates a notification which is sent to the STM Module. If push of notification to STM Module fails, it is stored and can be checked with getNotification.

If there are messages waiting in SPIS, the STM Module calls getMessage to retrieve waiting messages.



#### Interaction Private getMessage

Message exchange pattern: REQUEST\_RESPONSE

The private interface getMessage can be called any time by the STM Module. Normally it's related to a received Notification but can also be called at start-up or re-connection to purge messages from SPIS.

The call to getMessage can contain a upper boundary to limit the number of messages returned.



#### Service orchestration - Poll PortCDM queue (Get RTA )

Message exchange pattern: N/A

Shows the interactions when a timer event is fired and the PortCDM queues is polled. Messages in PortCDM queue will be forwarded to the ship.



#### Service Orchestration - Ship Port Synchronization with PCM

Message exchange pattern: N/A



#### Service Orchestration - overview with RTZ and PCM

Message exchange pattern: N/A



### Logging

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

#### 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

|  |  |
| --- | --- |
| **Incoming service call on SeaSWIM side** | |
| **Event** | **Log description** |
| uploadPCM | Log event for incoming data  Log event with notification to STM Module |
| **Incoming service call on private side** | |
| **Event** | **Log description** |
| authorizeIdentities | Log event for incoming request  Log event for added row in ACL  Log event for added actor in subscription list |
| removeAuthorizedIdentitites | Log event for incoming request  Log event for removed row in ACL  Log event for removed actor in subscription list |
| findAuthorizedIdentitties | Log event for incoming request |
| publishMessage | Log event for incoming data  Log event for outgoing data |
| getMessage | Log event for incoming request  Log event with returned data |
| callService | Log event for incoming request  Log event with returned data |
| findOrganizations | Log event for incoming request  Log event with returned data |
| findServices | Log event for incoming request  Log event with returned data |
| **Internal events** | |
| **Event** | **Log description** |
| Start-up | Log event start-up |
| Close-down | Log event close-down |

# References

|  |  |  |
| --- | --- | --- |
| **Reference name** | **Comment** | **Link** |
| STM Governance Handbook |  | - |
| Port Call Message Format | PCMF | <http://stmvalidation.eu/schemas/> |
| SPIS Specification |  | <http://stmvalidation.eu/service-catalogue/> |

# 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** | **Acronym** | **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* |

# APPENDIX API Swagger (JSON)

{

"swagger": "2.0",

"info": {

"version": "SPIS\_v1\_0\_0",

"title": "STM Ship Port Information Service STM Onboard API"

},

"host": "localhost",

"basePath": "/STM.SPIS.Services.Private",

"schemes": ["http", "https"],

"paths": {

"/authorizeIdentities": {

"get": {

"tags": ["AuthorizeIdentities"],

"summary": "",

"description": "Find list of authorized identities to data",

"operationId": "FindAuthorizedIdentities",

"consumes": [],

"produces": ["application/json"],

"parameters": [{

"name": "dataId",

"in": "query",

"description": "Data id of PCM message",

"required": true,

"type": "string"

}

],

"responses": {

"200": {

"description": "OK",

"schema": {

"type": "array",

"items": {

"$ref": "#/definitions/IdentityDescriptionObject"

}

}

},

"400": {

"description": "Bad Request"

},

"404": {

"description": "Not Found ( requested identities are not found)"

},

"500": {

"description": "Internal Server Error"

},

"default": {

"description": "unexpected error"

}

}

},

"post": {

"tags": ["AuthorizeIdentities"],

"summary": "",

"description": "Authorize identities to Voyage Plan",

"operationId": "AuthorizeIdentities",

"consumes": ["application/json"],

"produces": ["application/json"],

"parameters": [{

"name": "dataId",

"in": "query",

"description": "Data id of the PCM message",

"required": true,

"type": "string"

}, {

"name": "identityDescriptionObjects",

"in": "body",

"description": "Attributes to describe an organization in STM",

"required": true,

"schema": {

"type": "array",

"items": {

"$ref": "#/definitions/IdentityDescriptionObject"

}

}

}

],

"responses": {

"200": {

"description": "OK",

"schema": {

"$ref": "#/definitions/ResponseObj"

}

},

"400": {

"description": "Bad Request"

},

"403": {

"description": "Forbidden"

},

"500": {

"description": "Internal Server Error"

},

"default": {

"description": "unexpected error"

}

}

},

"delete": {

"tags": ["AuthorizeIdentities"],

"summary": "Remove Authorize identities to data",

"operationId": "AuthorizeIdentities\_RemoveAuthorizedIdentitites",

"consumes": ["application/json"],

"produces": ["application/json"],

"parameters": [{

"name": "dataId",

"in": "query",

"description": "Data id of the PCM message",

"required": true,

"type": "string"

}, {

"name": "identityDescriptionObjects",

"in": "body",

"description": "Attributes to describe an organization in STM",

"required": true,

"schema": {

"type": "array",

"items": {

"$ref": "#/definitions/IdentityDescriptionObject"

}

}

}

],

"responses": {

"200": {

"description": "Success",

"schema": {

"$ref": "#/definitions/ResponseObj"

}

},

"400": {

"description": "Bad Request"

},

"404": {

"description": "Not Found ( requested identities are not found)"

},

"500": {

"description": "Internal Server Error"

},

"default": {

"description": "unexpected error"

}

}

}

},

"/callService": {

"post": {

"tags": ["CallService"],

"operationId": "CallService\_CallService",

"consumes": ["application/json"],

"produces": ["application/json"],

"parameters": [{

"name": "callServiceObj",

"in": "body",

"required": true,

"schema": {

"$ref": "#/definitions/CallServiceRequestObj"

}

}

],

"responses": {

"200": {

"description": "Success",

"schema": {

"$ref": "#/definitions/CallServiceResponseObj"

}

},

"400": {

"description": "Bad Request"

},

"401": {

"description": "Unauthorized (the user cannot be auhtenticated in the Identity Registry)"

},

"403": {

"description": "Forbidden (the user is not authorized to requested service)"

},

"404": {

"description": "Not Found (the requested service endpoint is not found)"

},

"500": {

"description": "Internal Server Error"

},

"default": {

"description": "unexpected error"

}

}

}

},

"/findIdentities": {

"get": {

"tags": ["FindIdentities"],

"summary": "",

"description": "Seacrh for identities in STM identity registry",

"operationId": "FindIdentities",

"consumes": [],

"produces": ["application/json"],

"responses": {

"200": {

"description": "OK",

"schema": {

"type": "array",

"items": {

"$ref": "#/definitions/FindIdentitiesResponseObj"

}

}

},

"400": {

"description": "Bad Request"

},

"401": {

"description": "Unauthorized (the user cannot be auhtenticated in the Identity Registry)"

},

"403": {

"description": "Forbidden (the user is not authorized to requested organization)"

},

"404": {

"description": "Not Found (the requested identity is not found)"

},

"500": {

"description": "Internal Server Error"

},

"default": {

"description": "unexpected error"

}

}

}

},

"/findServices": {

"post": {

"tags": ["FindServices"],

"summary": "Discover services in SeaSWIM service registry",

"operationId": "FindServices\_FindServices",

"consumes": ["application/json"],

"produces": ["application/json"],

"parameters": [{

"name": "findServicesObj",

"in": "body",

"description": "Filter keywords and organizationId for service instances",

"required": true,

"schema": {

"$ref": "#/definitions/FindServicesRequestObj"

}

}

],

"responses": {

"200": {

"description": "Success",

"schema": {

"$ref": "#/definitions/FindServicesResponseObj"

}

},

"400": {

"description": "Bad Request"

},

"401": {

"description": "Unauthorized (the user cannot be authenticated in the Service Registry)"

},

"403": {

"description": "Forbidden (the user is not authorized to requested service)"

},

"500": {

"description": "Internal Server Error"

},

"default": {

"description": "unexpected error"

}

}

}

},

"/getMessage": {

"get": {

"tags": ["GetMessage"],

"summary": "Retrieve received messages from SPIS database",

"operationId": "GetMessage\_GetMessage",

"consumes": [],

"produces": ["application/json"],

"parameters": [{

"name": "limitQuery",

"in": "query",

"description": "Limit the number of messages to be received",

"required": false,

"type": "string"

}

],

"responses": {

"200": {

"description": "Success",

"schema": {

"$ref": "#/definitions/MessageEnvelope"

}

},

"default": {

"description": "Unexpected error"

}

}

}

},

"/getNotification": {

"get": {

"tags": ["Notification"],

"summary": "",

"description": "Get all awaiting notifications",

"operationId": "GetNotification",

"consumes": [],

"produces": ["application/json"],

"responses": {

"200": {

"description": "OK",

"schema": {

"type": "array",

"items": {

"$ref": "#/definitions/Notification"

}

}

},

"401": {

"description": "Unauthorized (the user cannot be auhtenticated"

},

"400": {

"description": "Bad Request"

},

"500": {

"description": "Internal Server Error"

},

"default": {

"description": "unexpected error"

}

}

}

},

"/getPublishedMessages": {

"get": {

"tags": ["PublishMessage"],

"summary": "Get all currently published messages",

"operationId": "PublishMessage\_GetPublishedMessages",

"consumes": [],

"produces": ["application/json"],

"responses": {

"200": {

"description": "Success",

"schema": {

"type": "array",

"items": {

"$ref": "#/definitions/PublishedMessageContract"

}

}

},

"500": {

"description": "Internal Server Error"

},

"default": {

"description": "unexpected error"

}

}

}

},

"/publishedMessage": {

"delete": {

"tags": ["PublishMessage"],

"summary": "Remove published message",

"operationId": "PublishMessage\_RemovePublishedMessage",

"consumes": ["application/json"],

"produces": ["application/json"],

"parameters": [{

"name": "dataId",

"in": "query",

"description": "",

"required": true,

"type": "string"

}

],

"responses": {

"200": {

"description": "OK",

"schema": {

"$ref": "#/definitions/ResponseObj"

}

}

}

}

},

"/publishMessage": {

"post": {

"tags": ["PublishMessage"],

"summary": "Publish message to SPIS database for subsequent sending to subscribers",

"operationId": "PublishMessage\_PublishMessage",

"consumes": ["application/json"],

"produces": ["application/json"],

"parameters": [{

"name": "dataId",

"in": "query",

"description": "data Id for published message",

"required": true,

"type": "string"

}, {

"name": "messageType",

"in": "query",

"description": "Message type for published message (PCM)",

"required": true,

"type": "string"

}, {

"name": "message",

"in": "body",

"description": "Typically STM payload data (PCM)",

"required": true,

"schema": {

"type": "string"

}

}

],

"responses": {

"200": {

"description": "Success",

"schema": {

"$ref": "#/definitions/ResponseObj"

}

},

"400": {

"description": "Bad Request"

},

"403": {

"description": "Forbidden"

},

"500": {

"description": "Internal Server Error"

},

"default": {

"description": "unexpected error"

}

}

}

},

"/subscription": {

"get": {

"tags": ["Subscription"],

"summary": "",

"description": "Find list of subscriber identities to data",

"operationId": "GetSubscriptions",

"consumes": [],

"produces": ["application/json"],

"parameters": [{

"name": "dataId",

"in": "query",

"description": "Data id",

"required": false,

"type": "string"

}

],

"responses": {

"200": {

"description": "OK",

"schema": {

"type": "array",

"items": {

"$ref": "#/definitions/SubscriptionObject"

}

}

},

"400": {

"description": "Bad Request"

},

"404": {

"description": "Not Found ( requested identities are not found)"

},

"500": {

"description": "Internal Server Error"

},

"default": {

"description": "unexpected error"

}

}

},

"post": {

"tags": ["Subscription"],

"summary": "",

"operationId": "AddSubscription",

"consumes": ["application/json"],

"produces": ["application/json"],

"parameters": [{

"name": "subscriptions",

"in": "body",

"description": "Identities to add as subscribers on dataId",

"required": true,

"schema": {

"type": "array",

"items": {

"$ref": "#/definitions/SubscriptionObject"

}

}

}, {

"name": "dataId",

"in": "query",

"description": "",

"required": true,

"type": "string"

}

],

"responses": {

"200": {

"description": "OK",

"schema": {

"$ref": "#/definitions/ResponseObj"

}

},

"400": {

"description": "Bad Request"

},

"403": {

"description": "Forbidden"

},

"500": {

"description": "Internal Server Error"

},

"default": {

"description": "unexpected error"

}

}

},

"delete": {

"tags": ["Subscription"],

"summary": "",

"operationId": "RemoveSubscriptions",

"consumes": ["application/json"],

"produces": ["application/json"],

"parameters": [{

"name": "dataId",

"in": "query",

"description": "",

"required": true,

"type": "string"

}, {

"name": "subscriptionObjects",

"in": "body",

"description": "",

"required": true,

"schema": {

"type": "array",

"items": {

"$ref": "#/definitions/SubscriptionObject"

}

}

}

],

"responses": {

"200": {

"description": "OK",

"schema": {

"type": "array",

"items": {

"$ref": "#/definitions/SubscriptionObject"

}

}

},

"400": {

"description": "Bad Request"

},

"404": {

"description": "Not Found ( requested identities are not found)"

},

"500": {

"description": "Internal Server Error"

},

"default": {

"description": "unexpected error"

}

}

}

}

},

"definitions": {

"IdentityDescriptionObject": {

"description": "IdentityDescriptionObject",

"type": "object",

"properties": {

"identityId": {

"description": "identity in urn format according to ID registry",

"type": "string"

},

"identityName": {

"description": "Identity shortname in ID registry",

"type": "string"

}

}

},

"ResponseObj": {

"description": "ResponseObj",

"type": "object",

"properties": {

"dataId": {

"description": "Data Id typically a ID",

"type": "string"

}

}

},

"CallServiceRequestObj": {

"type": "object",

"properties": {

"body": {

"type": "string"

},

"endpointMethod": {

"type": "string"

},

"headers": {

"type": "array",

"items": {

"$ref": "#/definitions/Header"

}

},

"requestType": {

"type": "string"

}

}

},

"Header": {

"type": "object",

"properties": {

"key": {

"type": "string"

},

"value": {

"type": "string"

}

}

},

"CallServiceResponseObj": {

"type": "object",

"properties": {

"body": {

"type": "string"

},

"statusCode": {

"format": "int64",

"type": "integer"

}

}

},

"FindIdentitiesResponseObj": {

"type": "object",

"properties": {

"organizations": {

"type": "array",

"items": {

"$ref": "#/definitions/Organization"

}

},

"statusCode": {

"format": "int64",

"type": "integer"

},

"statusMessage": {

"type": "string"

}

}

},

"Organization": {

"type": "object",

"properties": {

"country": {

"type": "string"

},

"email": {

"type": "string"

},

"mrn": {

"type": "string"

},

"name": {

"type": "string"

}

}

},

"FindServicesRequestObj": {

"type": "object",

"properties": {

"filter": {

"$ref": "#/definitions/FindServicesRequestObjFilter"

},

"page": {

"format": "int64",

"type": "integer"

},

"pageSize": {

"format": "int64",

"type": "integer"

}

}

},

"FindServicesRequestObjFilter": {

"type": "object",

"properties": {

"coverageArea": {

"$ref": "#/definitions/FindServicesRequestObjFilterCoverageArea"

},

"UnLoCode": {

"type": "string"

},

"ServiceProviderIds": {

"type": "array",

"items": {

"type": "string"

}

},

"serviceDesignId": {

"type": "string"

},

"serviceInstanceId": {

"type": "string"

},

"mmsi": {

"type": "string"

},

"imo": {

"type": "string"

},

"serviceType": {

"type": "string"

},

"serviceStatus": {

"type": "string"

},

"keyWords": {

"type": "array",

"items": {

"type": "string"

}

},

"freeText": {

"type": "string"

}

}

},

"FindServicesRequestObjFilterCoverageArea": {

"type": "object",

"properties": {

"coverageType": {

"type": "string"

},

"value": {

"type": "string"

}

}

},

"FindServicesResponseObj": {

"type": "object",

"properties": {

"servicesInstances": {

"type": "array",

"items": {

"$ref": "#/definitions/ServiceInstance"

}

},

"statusCode": {

"format": "int64",

"type": "integer"

},

"statusMessage": {

"type": "string"

}

}

},

"ServiceInstance": {

"type": "object",

"properties": {

"comment": {

"type": "string"

},

"endpointType": {

"type": "string"

},

"endpointUri": {

"type": "string"

},

"instanceAsXml": {

"$ref": "#/definitions/Xml"

},

"instanceId": {

"type": "string"

},

"keywords": {

"type": "string"

},

"name": {

"type": "string"

},

"organizationId": {

"type": "string"

},

"status": {

"type": "string"

},

"unlocode": {

"type": "string"

},

"version": {

"type": "string"

}

}

},

"Xml": {

"type": "object",

"properties": {

"comment": {

"type": "string"

},

"content": {

"type": "string"

},

"contentContentType": {

"type": "string"

},

"id": {

"format": "int64",

"type": "integer"

},

"name": {

"type": "string"

}

}

},

"MessageEnvelope": {

"description": "MessageEnvelope",

"type": "object",

"properties": {

"numberOfMessages": {

"format": "int32",

"description": "Gets or Sets NumberOfMessages",

"type": "integer"

},

"remainingNumberOfMessages": {

"format": "int32",

"description": "Gets or Sets RemainingNumberOfMessages",

"type": "integer"

},

"message": {

"description": "Gets or Sets Message",

"type": "array",

"items": {

"$ref": "#/definitions/Message"

}

}

}

},

"Message": {

"description": "Message",

"type": "object",

"properties": {

"messageType": {

"description": "Gets or Sets MessageType",

"type": "string"

},

"id": {

"description": "Gets or Sets Id",

"type": "string"

},

"receivedAt": {

"format": "date-time",

"description": "Gets or Sets ReceivedAt",

"type": "string"

},

"fromOrgId": {

"description": "Gets or Sets FromOrgId",

"type": "string"

},

"fromOrgName": {

"description": "Gets or Sets FromOrgName",

"type": "string"

},

"fromServiceId": {

"description": "Gets or Sets fromServiceId",

"type": "string"

},

"stmMessage": {

"$ref": "#/definitions/StmMessage",

"description": "Gets or Sets StmMessage"

}

}

},

"StmMessage": {

"description": "StmMessage",

"type": "object",

"properties": {

"message": {

"description": "Message contents",

"type": "string"

}

}

},

"Notification": {

"required": ["Body", "FromOrgId", "FromServiceId", "MessageWaiting", "NotificationCreatedAt", "NotificationType", "ReceivedAt", "Subject", "NotificationSource"],

"type": "object",

"properties": {

"Body": {

"type": "string"

},

"FromOrgId": {

"type": "string"

},

"FromOrgName": {

"type": "string"

},

"FromServiceId": {

"type": "string"

},

"MessageWaiting": {

"format": "int32",

"type": "integer"

},

"NotificationCreatedAt": {

"format": "date-time",

"type": "string"

},

"NotificationType": {

"enum": ["MESSAGE\_WAITING", "UNAUTHORIZED\_REQUEST", "ACKNOWLEDGEMENT\_RECEIVED", "ERROR\_MESSAGE"],

"type": "string"

},

"ReceivedAt": {

"format": "date-time",

"type": "string"

},

"Subject": {

"type": "string"

},

"NotificationSource": {

"enum": ["VIS", "SPIS"],

"type": "string"

}

}

},

"PublishedMessageContract": {

"description": "Contains published messages from STM Module",

"type": "object",

"properties": {

"Message": {

"description": "The actual message in raw format",

"type": "string"

},

"MessageID": {

"description": "Identity of the message (extracted from the message.",

"type": "string"

},

"MessageLastUpdateTime": {

"format": "date-time",

"description": "Time of update of the message (extracted from the message)",

"type": "string"

},

"MessageStatus": {

"format": "int32",

"description": "Status on the message (extracted or derived from the message.",

"type": "integer"

},

"MessageType": {

"description": "Type of message (enumeration)",

"type": "string"

},

"MessageValidFrom": {

"format": "date-time",

"description": "",

"type": "string"

},

"MessageValidTo": {

"format": "date-time",

"description": "",

"type": "string"

},

"PublishTime": {

"format": "date-time",

"description": "Publish time to SPIS (set by SPIS when received)",

"type": "string"

}

}

},

"SubscriptionObject": {

"description": "",

"type": "object",

"properties": {

"IdentityId": {

"description": "",

"type": "string"

},

"IdentityName": {

"description": "",

"type": "string"

},

"MbEndpointURL": {

"description": "",

"type": "string"

},

"AmssEndpointURL": {

"description": "",

"type": "string"

}

}

}

}

}

# 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 |  |
|  |  |  |