EMSA Mobile Façade

Architecture V1.0

1. Distribution List

2. Revision History

Date	Version	Description	Revised By
30-09-2014	0.1	First written	Innovagency
15/10/2014	0.2	EMSA comments to architecture	EMSA
20/10/2014	0.3	New version by Innovagency	Innovagency
30/10/2014	0.4	Final comments by EMSA	EMSA
30-11-2014	0.9	New version by Innovagency	Innovagency
30-11-2014	V1.0	Final Version Approved by EMSA	EMSA

3. Referenced Documents

Referenced using [Nr.] throughout the document

Nr.	Document Title	Author	Description	File
1	EMSA - Integrated Maritime Data Environment [IMDatE] - Interface Control Document [ICD]	EMSA	EMSA Design Specification v1	IMDATE-TD-ACS-EMSA-118
2	EMSA CleanSeaNet Data Centre [CSN-DC] - External Interface Control Document [EICD]	EMSA	CSNDC-ID-ACS-EMSA-0104 Issue: 1.3.3	CSNDC-ID-ACS-EMSA-0104
3	Technical Design Specification ENCWS	EMSA	ENCWS Technical Design Specification V 3.0.0	ENC_WS

4. Definitions, Acronyms and Abbreviations

Ref	Title
OSB	Oracle Service Bus
ESB	Enterprise Service Bus

5. Table of Contents

EMSA Mobile Façade 1
Architecture V1.0 1
1. Distribution List 2
2. Revision History 2
3. Referenced Documents 2
4. Definitions, Acronyms and Abbreviations 4
6. Introduction 8
6.1. Solution 8
6.2. Oracle Service Bus 8
6.3. Protocol and Messages Format 8
6.4. Project Structure 9
6.4.1. OSB Project Structure 9
6.5. Public Service 10
6.6. Service Implementation 10
6.7. Proxy Service Configuration 11
6.8. Proxy Implementation 12
6.9. Business Services 14
6.10. Component Diagram 15
6.11. Deployment Diagram 15
6.12. Common Domain 16
6.12.1. Common 16
6.12.2. XQuery 16
6.12.2.1. generateErrorMessage 17
6.12.2.2. generateErrorMessageFromFault 17
6.13. IMDATE Domain 17
6.13.1. Proxy Services 17
6.13.1.1. positions 17
6.13.1.1.1. Activity Diagram 20
6.13.1.1.2. Route(sPrepareRoute) 20
6.13.1.1.3. Mapping 20
6.13.1.1.4. Error Codes 21
6.13.1.2. grids 21

6.13.1.2.1. Activity Diagram 22
6.13.1.2.2. Route(sPrepareRoute) 23
6.13.1.2.3. Error Codes 23
6.13.1.3. tracks 23
6.13.1.3.1. Activity Diagram 26
6.13.1.3.2. Route(sPrepareRoute) 27
6.13.1.3.3. Error Codes 27
6.13.1.4. voyages 27
6.13.1.4.1. Activity Diagram 29
6.13.1.4.2. Route(sPrepareRoute) 30
6.13.1.4.3. Error Codes 30
6.13.1.5. alerts 30
6.13.1.5.1. Activity Diagram 31
6.13.1.5.2. Route(sPrepareRoute) 32
6.13.1.5.3. Error Codes 32
6.13.1.6. incidents 32
6.13.1.6.1. Activity Diagram 35
6.13.1.6.2. Route(sPrepareRoute) 35
6.13.1.6.3. Error Codes 35
6.13.1.7. particulars 36
6.13.1.7.1. Activity Diagram 37
6.13.1.7.2. Route(sPrepareRoute) 38
6.13.1.7.3. Mapping 38
6.13.1.7.4. Error Codes 39
6.13.2. Business Services 39
6.13.2.1. getPositions 39
6.13.2.2. grid 39
6.13.2.3. tracks 39
6.13.2.4. voyages 40
6.13.2.5. alerts 40
6.13.2.6. getIncidentDetail 40
6.13.2.7. getShipParticulars 40
6.13.2.8. getActiveIncidents 41

6.13.2.9. getIncidentsInAreaByBB 41
6.13.2.10. getIncidentsInAreaByWKT 41
6.14. CSNDC Domain 41
6.14.1. Proxy Service 41
6.14.1.1. oilSpills 41
6.14.1.1.1. Activity Diagram 42
6.14.1.1.2. Route(sPrepareRoute) 43
6.14.1.1.3. Mapping 43
6.14.1.1.4. Error Codes 43
6.14.2. Business Service 44
6.14.2.1. getOilSpills 44
6.15. ENCWS Domain 44
6.15.1. Proxy Services 44
6.15.1.1. cmap 44
6.15.1.1.1. Activity Diagram 46
6.15.1.1.2. Route(sPrepareRoute) 46
6.15.1.1.3. Mapping 46
6.15.1.1.4. Error Codes 47
6.15.2. Business Services 47
6.15.2.1. getMap 47

6. Introduction

This document provides the low level implementation details of the “Mobile Facade”.

6.1. Solution

The OSB in this solution will be used to provide multiple rest services in EMSA ESB, making those services available for use by multiple mobile clients. Since most of those service already exists and in order to reduce dependencies between systems, on the new OSB design approach services we are creating only the required functionality by other systems at this moment. This kind of action may lead to later changes on the ESB services, but will minimize effort spent at this stage, while it also simplifies the exposed service interface.

6.2. Oracle Service Bus

The Oracle Service Bus (OSB) is a component of the Oracle SOA portfolio. The OSB’s task is to publish access to internal resources and perform necessary mediation to make sure that the client systems can access the resources using the transport technology of preference. The core tasks of the OSB are:

- To receive, mediate and proxy requests of external systems for internal resources;
- To perform minor orchestrations – such as callouts to log or validate a message received, for example;
- To proxy requests to back-end systems – hiding the internal interfaces behind a proxy layer;

  To allow easily aggregate – with no need of modifying the internal resources – new external client systems;

6.3. Protocol and Messages Format

The following principles apply to the described API:

- The unitary (operating on a single object) interfaces:
  - The interfaces are exposed in a RESTful fashion.
  - All the communication is performed over the secure HTTP (HTTPS) protocol. Certificates are installed at F5 load balance.
  - All https are terminated at the F5 – internal communications are performed over HTTP.
  - Only GET methods are used.
  - The server accepts the HTTP requests with following headers (properties):
    - Content-Type: application/json

- Following HTTP status codes will be returned:
  - 200 in case of successful message processing
  - 400 if a request received by the “Mobile Façade” is invalid (either format or data is incorrect). The message should be corrected by the client before re-sending.
  - 500 the “Mobile Façade” server failed to process a message for a technical or functional reason (and the operation may be retried without data changes)
- Apart from the HTTP status code each response message contains a Json object with detailed error information, when applicable.
  - This Json object has the following structure:

Output			
Argument	Type	Occurs	Description
status	String	1..1	Indicates the status of the request. (success or error)
message	Array	1..1	Detailed message of the error.

- The message bodies, both response, contain Json objects defined later in this document

6.4. Project Structure

6.4.1. OSB Project Structure

Each OSB Project will translate in a functional domain, allowing system grouping based on common functionality regardless the specific end-system implementation, reducing the impact of system changes across the organization, while improve common functionality re-use.

On the following diagram we may find a common project structure:

Folder	Description	Layout
business	Contains OSB business service used to connect to the target system, each connector should be contained on a specific folder with the following rules: v_<version number>/<service_name> This kind of configuration improve the organization of the services on the ESB service catalogue, simplifying the identification of the available interfaces to the end systems and grouping external resources from the ESB private layer. Additional business services for different services location, like specific country services should be grouped inside the same service / version folder in order to simplify the identification of the available final services and reduce the number of services available when no additional	

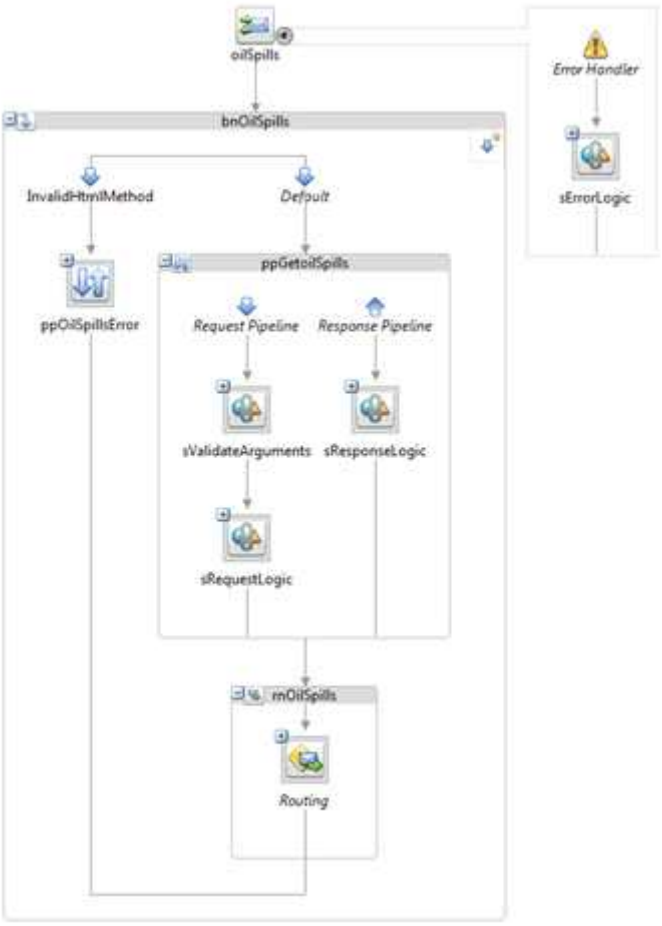
	functionality is provided.	
common	Folder containing common resources used on the domain scope, like a common schema description, or a common transformation used for target system integration. E.g. XQuery used to create a common header on a specific system that will be used along all proxies	
pub	The following folder should contain all public proxies that will be linked to a specific transport protocol, like HTTP, JMS and other. The sub-folders created should follow the following format: v_<version number>/<service_name>This will allow a clear vision over the available protocols on the given for each service on the ESB catalogue. Additionally the service folder should contain the service specific wsdl / xsd, and related XQueries used for data transformation	

6.5. Public Service

Public services provide an interface to external clients in this case using REST over HTTP.

6.6. Service Implementation

On the next diagram we may found an implementation example, identifying the common steps.

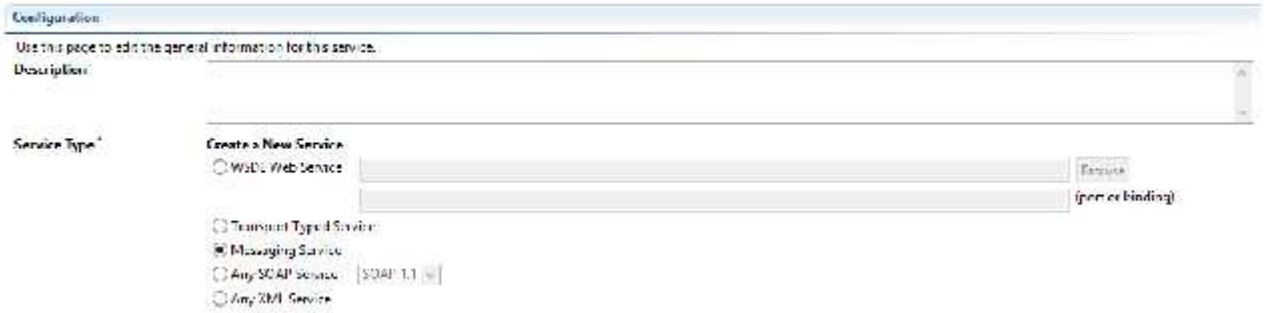


Service steps	
Step Id	**Description**
sRequestLogic	No more than one assign/replace action should be made on a staging area for the same variable, this kind of constraint is not an OSB limitation, but a suggestion for better error tracking, since $fault information while transforming variable in OSB doesn't provide a custom final activity name, but based on the previous suggestion is possible to clearly identify in which activity the error occurred.
sResponseLogic	No more than one assign/replace action should be made on a staging area for the same variable, this kind of constraint is not an OSB limitation, but a suggestion for better error tracking, since $fault information while transforming variable in OSB doesn't provide a custom final activity name, but based on the previous suggestion is possible to clearly identify in which activity the error occurred.
sErrorLogic	Finally, technical errors may be automatically managed by the common functions, simplifying the error decoding and masking, in order to manage external technical errors check 6.12.2.2. generateErrorMessageFromFault. A custom error management can be added, but it's advised to identify the actual activity error and service, like it's ensured in the previous XQuery.

6.7. Proxy Service Configuration

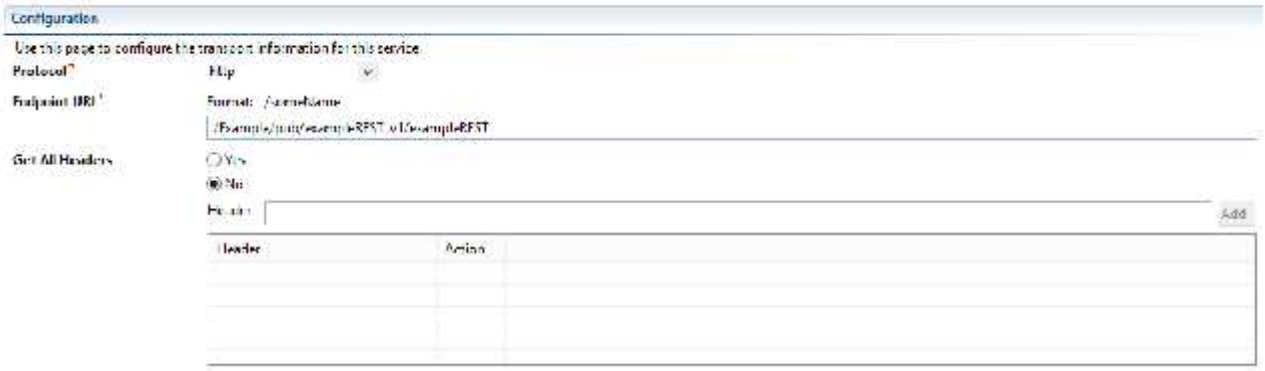
In order to create a public service offering a REST service type, the following service configuration should be made:

General Configuration



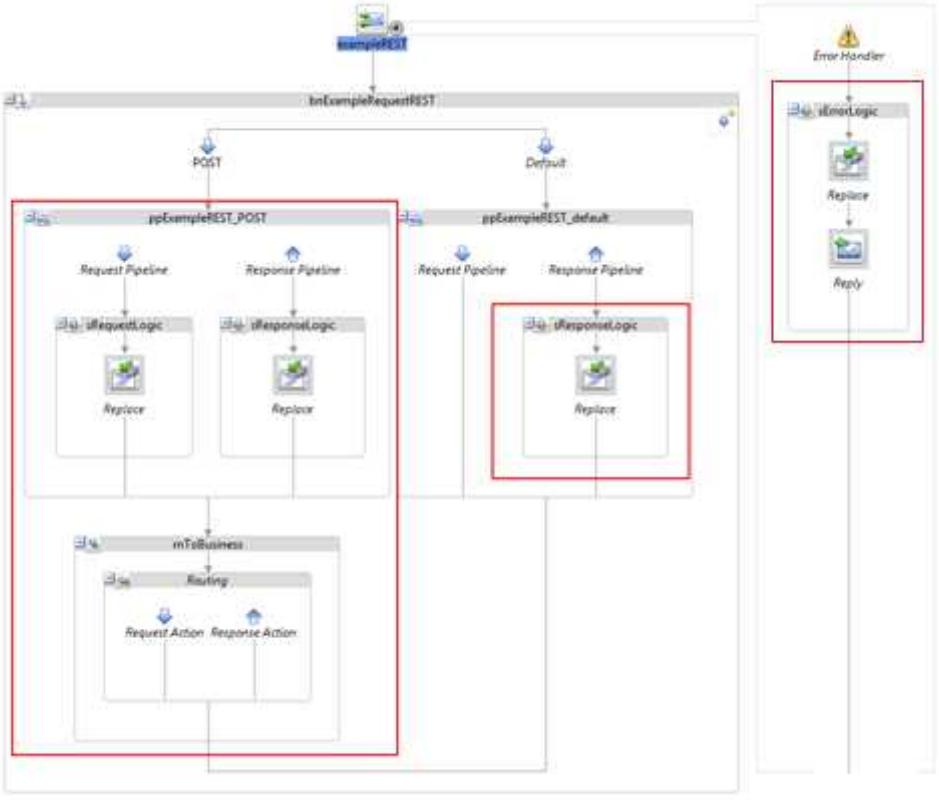
Message Type Configuration

Transport Configuration



The default endpoint URI should be kept (/<Domain>/pub/v_<version_number>/<service_name>) since it will be shorter and best suitable to rest APIs ().

6.8. Proxy Implementation



REST request can be submitted through multiple HTTP method actions, like GET, POST, and other. To retrieve the kind of HTTP method OSB provides an inbound execution context (http://docs.oracle.com/cd/E13171_01/alsb/docs30/userguide/context.html#wp1100201), available through internal variable $inbound, the following variable provides additional information which can be used to retrieve service request parameters. On the following is provided useful XQueries expression that can be used to retrieve relevant information for the service execution:

Field	XQuery	Description
Relative URI	$inbound/*:transport[1]/*:request[1] /*:relative-URI	The relative URI map to the requested URL submission made by client application after the proxy URI listener point. This kind of property allows REST requests to be made based on URL action, encoding additional parameters by URI context. For example, we have a listener service in: http://localhost:7001/Example/pub/v1/example/ The following request http://localhost:7001/Example/pub/v1/example/OPERATION?PARAMETER=VALUE will provide the following internal information: Relative URI = OPERATION This will allow proxy dispatching based on Relative URI request

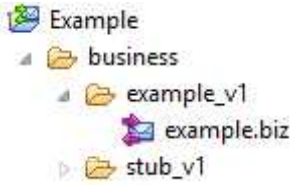
URL Query Parameters	$inbound/*:transport[1]/*:request[1]/*:query-parameters/*:parameter	Additional parameters can be URL encoded and processed by proxy services, in order to retrieve those additional parameters, refer to @name and @value attribute inside parameter element. Executing the previous example, we'll be able to retrieve: @name = PARAMETER @value = VALUE
HTTP Method	$inbound/*:transport[1]/*:request[1]/*:http-method	Load method type used in the HTTP request

Example

```
<con:endpoint name="ProxyService$Example$pub$exampleRESTGet_v1$exampleRESTGet" xmlns:con="http://www.bea.com/wli/sb/context">
        <con:service/>
        <con:transport>
                <con:uri>/Example/pub/v1/example</con:uri>
                <con:mode>request-response</con:mode>
                <con:qualityOfService>best-effort</con:qualityOfService>
                <con:request xsi:type="http:HttpRequestMetaData" xmlns:http="http://www.bea.com/wli/sb/transports/http"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
                        <tran:headers xsi:type="http:HttpRequestHeaders" xmlns:tran="http://www.bea.com/wli/sb/transports">
                                <tran:user-header name="Cookie"
value="ADMINCONSOLESESSION=N1TvTfQWThxmfl1rnfv8bSfyfZsn4h2ZGXyZc1nC3xQ9jXy9y152!-1885622139"/>
                                <http:Accept>text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8</http:Accept>
                                <http:Accept-Encoding>gzip,deflate,sdch</http:Accept-Encoding>
                                <http:Accept-Language>en-US,en;q=0.8,pt-BR;q=0.6,pt;q=0.4</http:Accept-Language>
                                <http:Connection>keep-alive</http:Connection>
                                <http:Cookie/>
                                <http:Host>localhost:7001</http:Host>
                                <http:User-Agent>Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/33.0.1750.146 Safari/537.36</http:User-Agent>
                        </tran:headers>
                        <tran:encoding xmlns:tran="http://www.bea.com/wli/sb/transports">iso-8859-1</tran:encoding>
                        <http:relative-URI>WAUBA24B3XN104537</http:relative-URI>
                        <http:query-string>device=telematics&amp;info=status</http:query-string>
                        <http:query-parameters>
                                <http:parameter name="device" value="telematics"/>
                                <http:parameter name="info" value="status"/>
                        </http:query-parameters>
                        <http:client-host>0:0:0:0:0:0:0:1</http:client-host>
                        <http:client-address>0:0:0:0:0:0:0:1</http:client-address>
                        <http:http-method>GET</http:http-method>
                </con:request>
                <con:response xsi:type="http:HttpResponseMetaData" xmlns:http="http://www.bea.com/wli/sb/transports/http"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
                        <tran:headers xsi:type="http:HttpResponseHeaders" xmlns:tran="http://www.bea.com/wli/sb/transports">
                                <http:Content-Type>text/plain</http:Content-Type>
                        </tran:headers>
                        <tran:response-code xmlns:tran="http://www.bea.com/wli/sb/transports">0</tran:response-code>
                </con:response>
        </con:transport>
        <con:security>
                <con:transportClient>
                        <con:username>&lt;anonymous></con:username>
        </con:transportClient>
</con:security>
</con:endpoint>
```

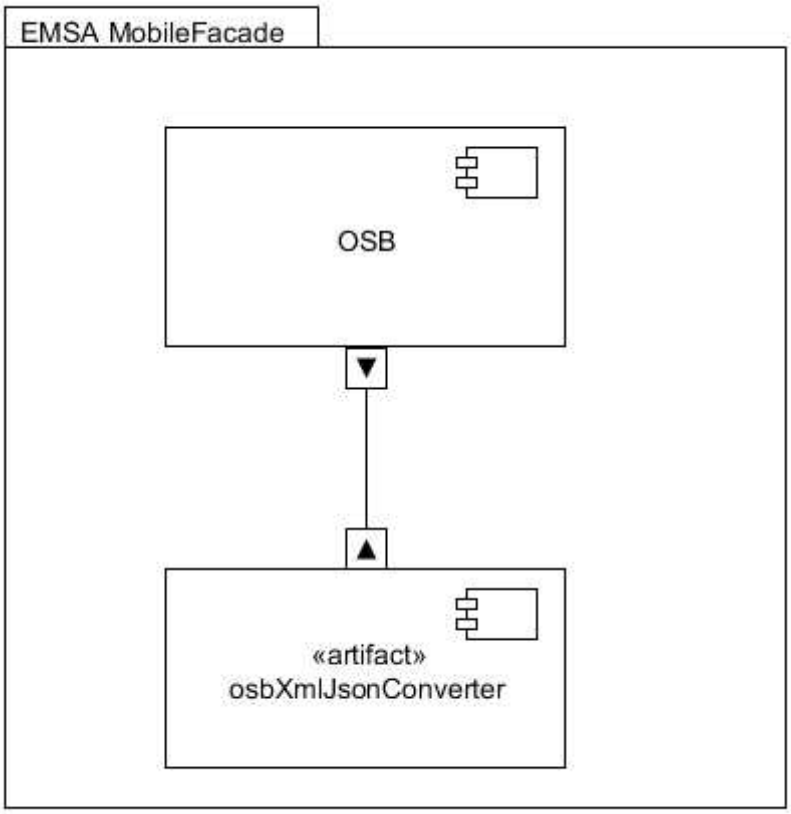
6.9. Business Services

Business service will contain the final integration with the target system.



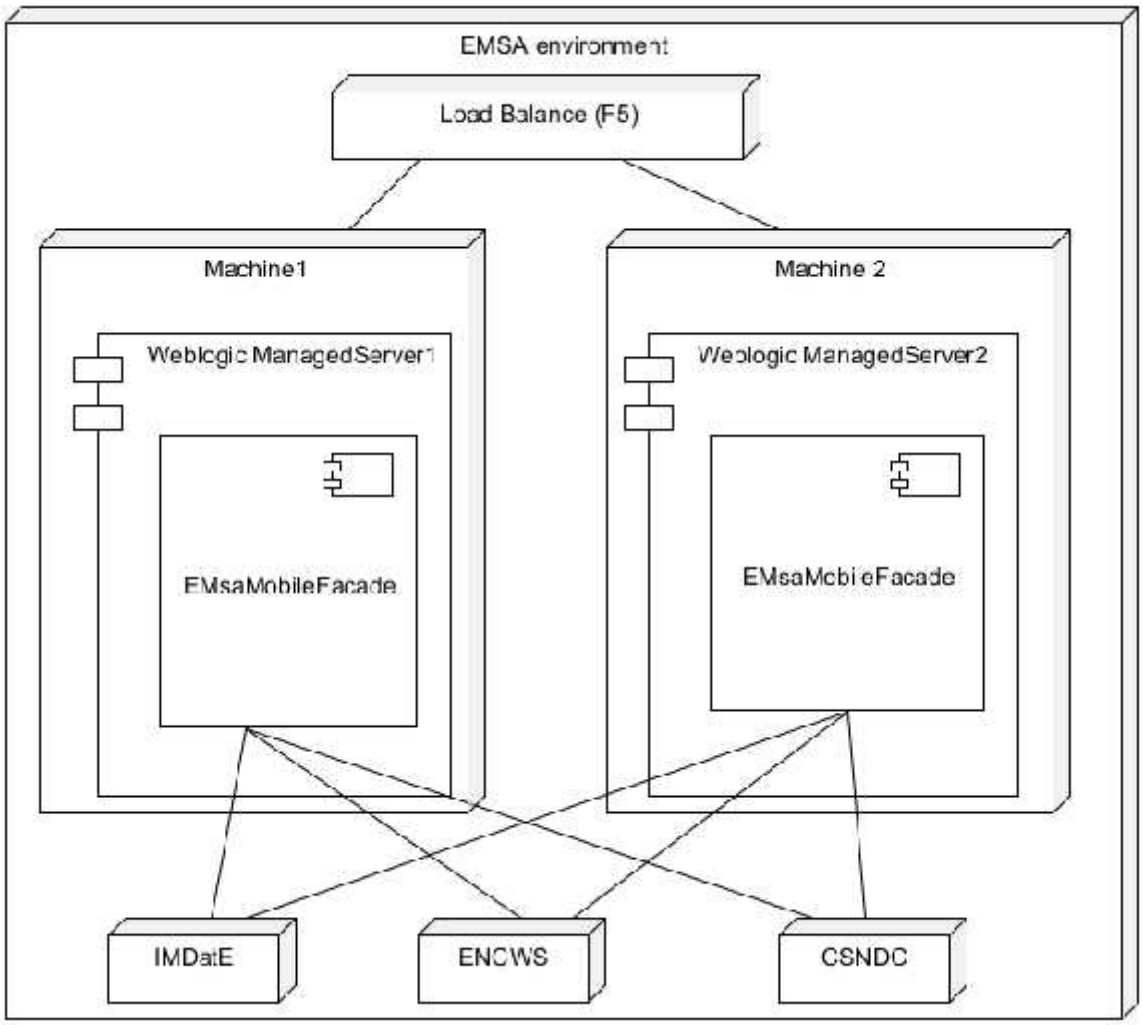
6.10. Component Diagram

This solution consists in two software components that will be deployed together as an OSB jar file. The first component is the OSB layer that delivers the new endpoints and where the logic is implemented. The second component it's a java utility that is used to convert xml to Json when applicable.



6.11. Deployment Diagram

Following diagram presents physical components and their placement in EMSA environment.



All the EMSAMobileFacade run on the same set of hardware. The proposed setup consists of two machines. Each of the machines runs one managed servers of the Web Logic server that perform work.

Access to this configuration is provided through a central load balancer and guarded by a firewall, already present in EMSA's environment.

6.12. Common Domain

6.12.1. Common

The following chapter describes the common resources, like messages definitions and XQuery that can be used by all ESB services, simplifying the ESB services creation through the re-use of common functions and capabilities.

6.12.2. XQuery

6.12.2.1. generateErrorMessage

Service	generateErrorMessage will provide a common error function for all domains		
Input Parameters			
Name	Type	Description	
message	String	1..1	Message to be display in error message.
Output			
Name	Type	Occurs	Description
error	String	1..1	Returns error message formatted. E.g {"status":"error", "message":"invalid arguments"}

6.12.2.2. generateErrorMessageFromFault

Service	generateErrorMessage will provide a common error function for all domains		
Input Parameters			
Name	Type	Description	
fault	String	1..1	OSB fault variable
Output			
Name	Type	Occurs	Description
error	String	1..1	Returns error message formatted. E.g {"status":"error", "message":"invalid arguments"}

6.13. IMDATE Domain

6.13.1. Proxy Services

6.13.1.1. positions

Service	positions	Version	1
Description	Returns the ships counts in the specified time range over a 4-level	Http Method	GET

	geohash grid.		
Endpoint	https://<host>[:<port>]/EMSAMobile/v1/positions?user=<user>&project=<project>&minX=<minX>&maxX=<maxX>&minY=<minY>&maxY=<maxY>&beginTimestamp=<beginTimestamp>&maxRecords=<maxRecords>&offset=<offset>		
Input Arguments			
Argument	Type	Occurs	Description
user	String	1..1	User name
project	String	0..1	Name of the project. If not provided the default project is assumed.
beginTimestamp	date	0..1	Begin time window. If not provided the time range for the ships will not be restricted.
endTimestamp	date	0..1	End time window. If not provided the time range for the ships will not be restricted.
minX	Float	1..1	Minimum longitude
minY	Float	1..1	Minimum latitude
maxX	Float	1..1	Maximum longitude
maxY	Float	1..1	Maximum latitude
maxRecords	Integer	0..1	If indicated, controls the maximum number of records returned by the request.
offset	Integer	0..1	An offset to be used in the response. This can be used in combination with the max number of records, in case pagination shall be implemented.
Output			
Argument	Type	Occurs	Description
Status	String	1..1	Indicates the status of the request. It can be: • success • fail
result	object	1..1	Array of grid elements
Vid	Integer	1..1	Vessel ID
Mmsi	Integer	0..1	Vessel MMSI
Imo	Integer	0..1	Vessel IMO
Ir	String	0..1	Vessel IR
Sn	String	0..1	Ship name
Fs	String	0..1	Flag state
Cs	String	0..1	Call Sign

St	String	0..1	Ship PSC Type
Lat	Float	1..1	Position latitude
Lon	Float	1..1	Position longitude
Hdg	Float	0..1	Position heading
Vf	String	1..1	Position validity flag (V = validated, I = non validated, U = unable to validate)
Ts	Date	1..1	Position time stamp
Npre	String	0..1	NPR enrichment
Speed	Float	0..1	Position speed
Ns	Integer	0..1	Position AIS navigational status
Orig	String	0..1	Position originator
Th	Float	0..1	Position true heading
Pid	Integer	1..1	Position ID (internal to IMDatE)
Loc	String	0..1	Destination locode
Sai	String	0..1	Project OVR specific info

Request Example

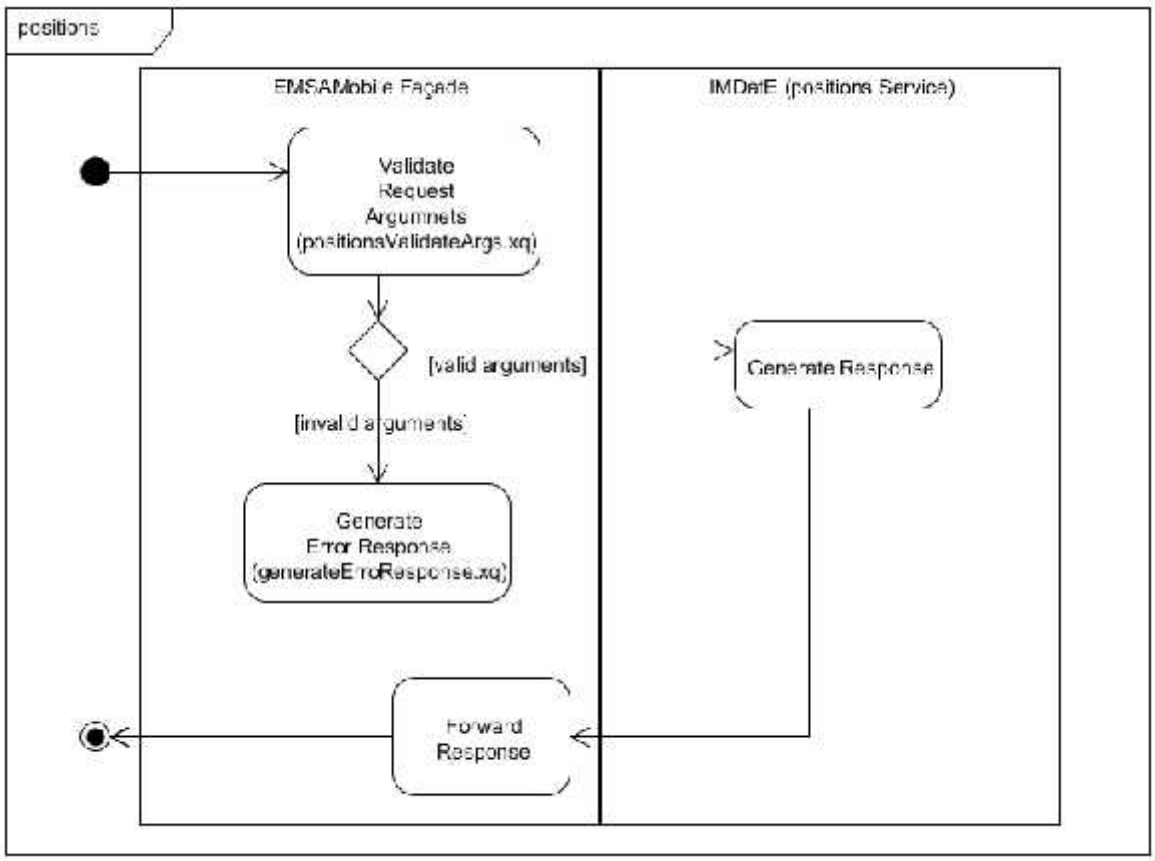
GET:
https://<host>[:<port>]/EMSAMobile/v1/positions?user=BILL&project=default&minX=5.625&maxX=22.5&minY=33.75&maxY=45&beginTimestamp=2014-05-20T10:25:23Z&maxRecords=1000&offset=0

Response Example


```
{
	"status": "success",
	"result": [{
		"vid": 107465,
		"mmsi": 319053000,
		"sn": "M/V HAKUNA MATATA",
		"ts": "2014-05-20T12:20:36Z",
		"speed": 0,
		"src": "T-AIS",
		"vf": "V",
		"hdg": 0,
		"lat": 43.817673,
		"lon": 7.7906733,
		"orig": "ou=AISCLASSB,dc=acsys,dc=it",
		"pid": 287047328
	},
	{
		"vid": 37648,
		"imo": 9559822,
		"mmsi": 235082478,
		"fs": "GB",
		"cs": "2DWB9",
```

```
                    "sn": "NATORI",
                    "ts": "2014-05-20T12:20:34Z",
                    "speed": 0.1,
                    "src": "T-AIS",
                    "ns": 1,
                    "vf": "U",
                    "hdg": 235.8,
                    "lat": 43.539974,
                    "lon": 7.02376,
                    "orig": "ou=AISA,dc=acsys,dc=it",
                    "st": 319,
                    "npre": 0,
                    "th": 64,
                    "pid": 287047313
            }]
}
```


6.13.1.1.1. Activity Diagram



6.13.1.1.2. Route(sPrepareRoute)

Condition	All	System	IMDatE	Type	rest
		Service	positions	Http Method	GET

6.13.1.1.3. Mapping

to

From Field	To Field
N/A	
from	
From Field	To Field
N/A	

6.13.1.1.4. Error Codes

The service returns following HTTP status codes:

- 200 – if the request was fulfilled successfully.
- 500 – if a problem on the server side occurred
  400 – if the request sent by the client system cannot be interpreted.

6.13.1.2. grids

Service	grids	Version	1
Description	Returns the ships counts in the specified time range over a 4-level geohash grid.	Http Method	GET
Endpoint	https://<host>[:<port>]/EMSAMobile/v1/grids?project=<project>&beginTimestamp=<begindate>&endTimestamp=<endDdate>&user=<user>		

Input Arguments			
Argument	Type	Occurs	Description
user	String	1..1	User name
project	String	0..1	Name of the project. If not provided the default project is assumed.
beginTimestamp	DateTime	0..1	Begin time window. If not provided the time range for the ships will not be restricted.
endTimestamp	DateTime	0..1	End time window. If not provided the time range for the ships will not be restricted.
Output			
Argument	Type	Occurs	Description
status	String	1..1	Indicates the status of the request. It can be: • success • fail
result	Array	1..1	Array of grid elements
Gh	String	1..1	Geohash code of the current cell

C	Integer	1..1	Count of ships for the current geohash cell

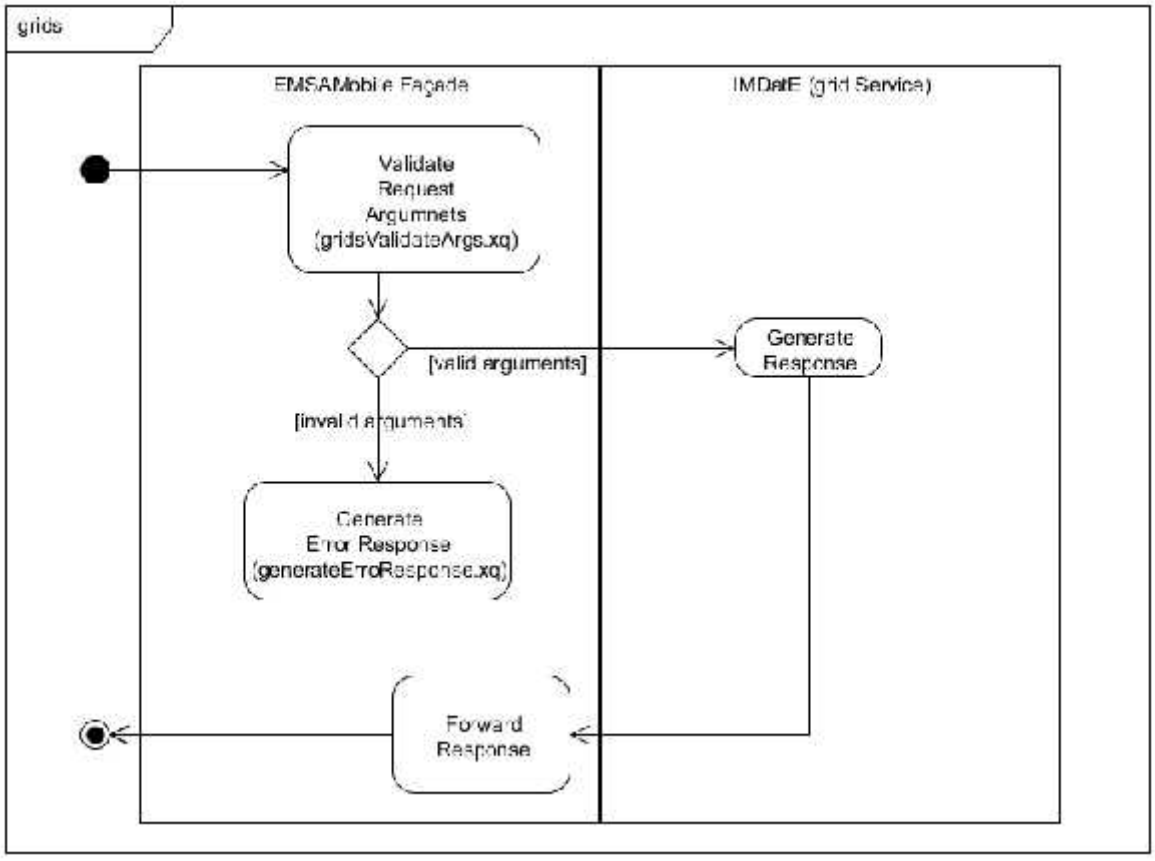
Request Example

GET: https://<host>[:<port>]/EMSAMobile/v1/grids?user=BILL

Response Example


```
{
	"status": "success",
	"result": [{
		"gh": "7zzz",
		"c": 1
	},
	{
		"gh": "9khm",
		"c": 1
	},
	{
		"gh": "dne4",
		"c": 1
	},
	{
		"gh": "e6v9",
		"c": 1
	},
	{
		"gh": "e6xk",
		"c": 1
	},
	{
		"gh": "e6xv",
		"c": 1
	}]
}
```


6.13.1.2.1. Activity Diagram



6.13.1.2.2. Route(sPrepareRoute)

Condition	All	System	IMDatE	Type	Rest
		Service	grid	Http Method	GET

6.13.1.2.3. Error Codes

The service returns following HTTP status codes:

- 200 – if the request was fulfilled successfully.
- 500 – if a problem on the server side occurred
- 400 - if the request sent by the client system cannot be interpreted.

6.13.1.3. tracks

Service	tracks	Version	1
Description	This method gets the ship positions tracks defined by request	Http Method	GET

	option		
Endpoint	https://<host>[:<port>]/EMSAMobile/v1/tracks/<request>?beginTimestamp=<beginTimestamp>&endTimestamp=<endTimestamp>&maxX=<maxX>&maxY=<maxY>&minx=<minx>&minY=<minY>&extrapolated=<extrapolated>&fusedAndSmoothed=<fusedAndSmoothed>&source=<source>&project=<project>&stepInSeconds=<stepInSeconds>&vesselId=<vesselId>&criteria=<criteria>		
Input Arguments			
Argument	Type	Occurs	Description
request	string	1.. 1	Defines filter type. This argument could be one of the following: • vessel • bbox
beginTimestamp	Date	1..1	Begin time stamp of the request
endTimestamp	Date	1..1	End time stamp of the request
maxX	float	Required if request argument has the value “bbox”.	Max Longitude of the bounding box in EPSG 4326
maxY	float	Required if request argument has the value “bbox”.	Max Latitude of the bounding box in EPSG 4326
minX	float	Required if request argument has the value “bbox”.	Min Longitude of the bounding box in EPSG 4326
minY	float	Required if request argument has the value “bbox”.	Min Latitude of the bounding box in EPSG 4326
extrapolated	boolean	0..1	Flag indicating if the track positions must be extrapolated.
fusedAndSmoothed	boolean	0..1	Flag indicating if the track positions must be interpolated and smoothed.
source	String	0..*	Common separated value of one or many sources as defined by the CDF. If more than one source is set, they are put in logical OR.
project	String	0..1	Project in the scope of which the request is made.
stepInSeconds	int	0..1	Interpolation step.
vesselId	String	Required if request argument has the value “vessel”.	The vessel ID used for retrieving the track. It can be any of the IDs managed by the system.
criteria	String	Required if request	Defines vesselId criteria. Could be one of the following values:

		argument has the value “vessel”.	• emsaId • mmsi • imo • irNumber
user	String	0..1	User name
maxTimeDifferenceInSeconds	Integer	0..1	

Output

Argument	Type	Occurs	Description
status	String	1..1	Indicates the status of the request.
result	Array	1..1	Array of grid elements

Request Example

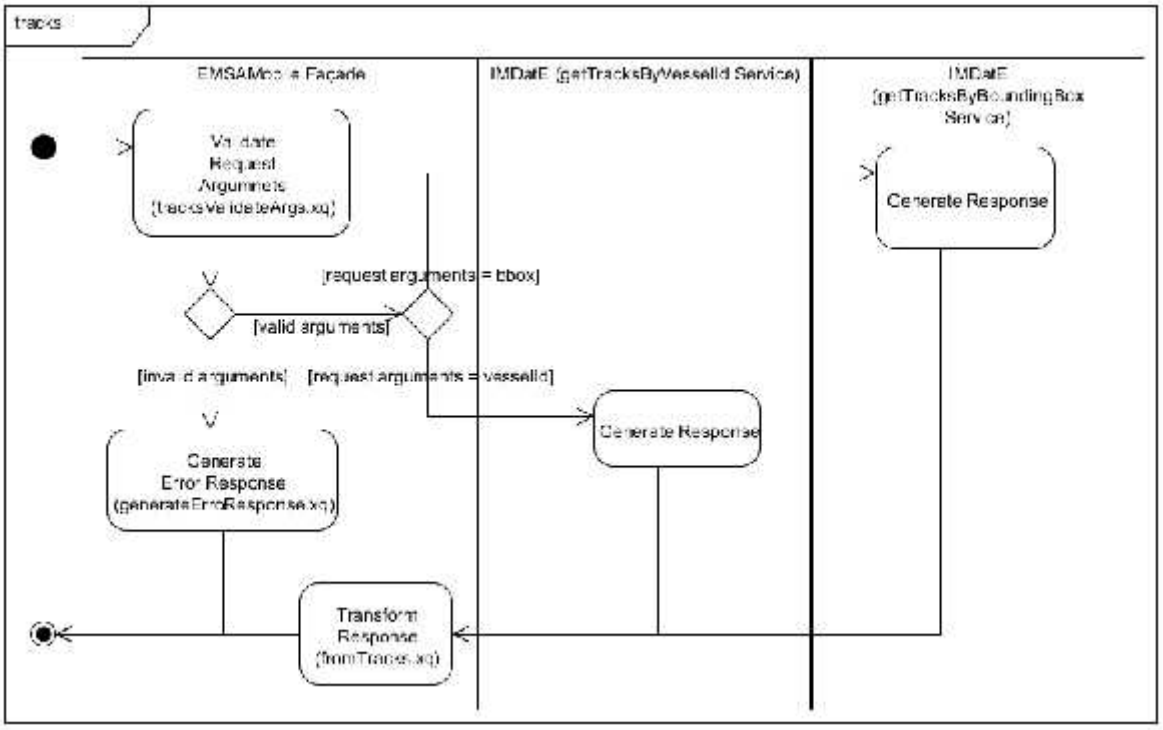
GET: http://tesb01:7101/EMSAMobile/v1/tracks?beginTimestamp=2014-11-01T10:44:42Z&endTimestamp=2014-11-02T10:44:42Z&vesselId=123456789&criteria=mmsi&user=CARREPH1

Response Example


```
{
	"result": {
		"EMSA": {
			"timestamp": "2014-11-21T16:29:09.735Z",
			"id": 1416587349734163,
			"PositionMessage": {
				"PositionEnrichmentInfo": {
					"ShipInfo": {
						"source": "IMDatE",
						"ShipType": 80,
						"ShipParticulars": ""
					}
				},
				"PositionReport": [{
					"timestamp": "2014-11-02T00:52:36.000Z",
					"source": "T-AIS",
					"COG": 0,
					"AdditionalInfo": {
						"source": "IMDatE",
						"PositionId": 138796989,
						"PreProcessingSpecific": {
							"KalmanValidationOutput": {
								"ValidityFlag": "U"
							}
						}
					},
					"Latitude": 38.91356,
					"SOG": 0,
					"Longitude": "1.443930",
					"DataAccessRights": {
						"Originator": "AIS-A"
					}
				},
```

```
                                        {
                                                "timestamp": "2014-11-02T01:43:37.000Z",
                                                "source": "T-AIS",
                                                "COG": 0,
                                                "AdditionalInfo": {
                                                        "source": "IMDatE",
                                                        "PositionId": 138805220,
                                                        "PreProcessingSpecific": {
                                                                "KalmanValidationOutput": {
                                                                        "ValidityFlag": "V"
                                                                }
                                                        }
                                                },
                                                "Latitude": 38.91359,
                                                "SOG": 0,
                                                "Longitude": "1.443940",
                                                "DataAccessRights": {
                                                        "Originator": "AIS-A"
                                                }
                                        }],
                                        "ShipParticulars": {
                                                "Name": "VOLGONEFT-107",
                                                "EMSAId": 21739,
                                                "CallSign": "UBBF-8",
                                                "IMO": 8869397,
                                                "MMSI": 123456789
                                        }
                                }
                        }
                },
                "status": "success"
}
```


6.13.1.3.1. Activity Diagram



6.13.1.3.2. Route(sPrepareRoute)

Condition	request argument = bbox	System	IMDatE	Type	SOAP
		Service	getTracksByBoundingBox	Http Method	
Condition	request argument = vesselId	System	IMDatE	Type	SOAP
		Service	getTracksByVesselId	Http Method	

6.13.1.3.3. Error Codes

The service returns following HTTP status codes:

- 200 – if the request was fulfilled successfully.
- 500 – if a problem on the server side occurred
- 400 - if the request sent by the client system cannot be interpreted.

6.13.1.4. voyages

Service	voyages	Version	1
Description	Returns all the voyage info for the vessel specified by the id in the	Http Method	GET

	time window specified		
Endpoint	https://<host>[:<port>]/EMSAMobile/v1/voyages/<shipId>?begin=<date>&end=<date>		

Input Arguments			
Argument	Type	Occurs	Description
shipId	Date	1..1	Imdate Id.
begin	Date	1..1	Begin time window
end	float	1..1	End time window

Output			
Argument	Type	Occurs	Description
status	String	1..1	Indicates the status of the request. (success)
result	Array	1..1	Array of voyages
id	Integer	1..1	Id of the voyage.
ts	DateTime	1..1	Timestamp of the voyage information
origin	String	1..1	Origin of the voyage.
destination	String	1..1	Destination of the voyage.
eta	Interger	1..1	Estimated time of arrival.

Request Example

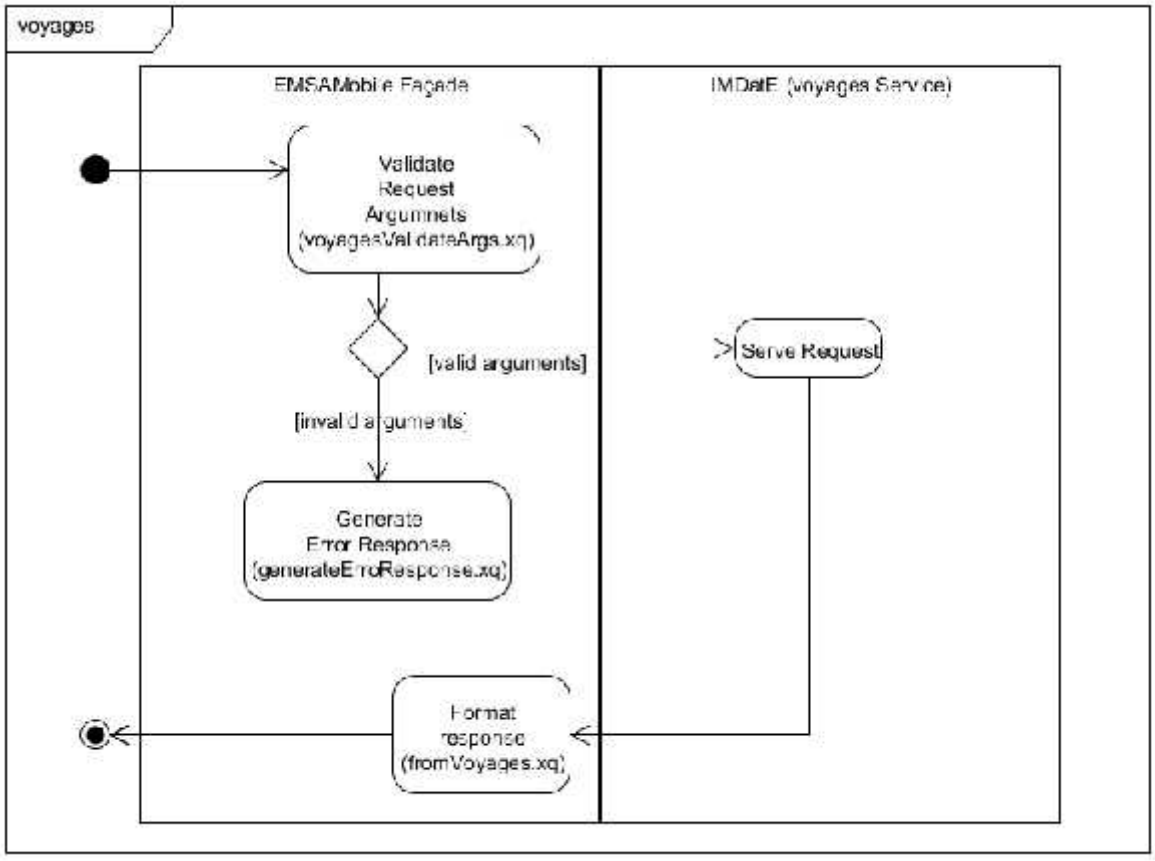
GET:
https://<host>[:<port>]/EMSAMobile/v1/voyages?/EMSAMobile/v1/voyages/80000?begin=2014-06-11T10:28:42Z&end=2014-10-12T10:33:22Z

Response Example


```
{
        "status": "success",
        "result": [{
                "id": 80000,
                "ts": "2014-09-18T07:59:49Z",
                "origin": "N.A.",
                "destination": "ROTTERDAM VIA NOK",
                "eta": "2014-09-19T01:30:00Z"
        },
        {
                "id": 80000,
                "ts": "2014-09-21T01:46:34Z",
                "origin": "ROTTERDAM VIA NOK",
                "destination": "ROTTERDAM VIA NOK",
                "eta": "2014-07-09T01:30:00Z"
        },
        {
```

```
                        "id": 80000,
                        "ts": "2014-09-22T00:28:29Z",
                        "origin": "ROTTERDAM VIA NOK",
                        "destination": "KOTKA VIA NOK",
                        "eta": "2014-09-25T05:00:00Z"
                },
                {
                        "id": 80000,
                        "ts": "2014-09-26T15:55:48Z",
                        "origin": "KOTKA VIA NOK",
                        "destination": "HELSINKI",
                        "eta": "2014-09-26T14:30:00Z"
                },
                {
                        "id": 80000,
                        "ts": "2014-09-26T19:38:54Z",
                        "origin": "HELSINKI",
                        "destination": "MUUGA",
                        "eta": "2014-09-26T23:30:00Z"
                },
                {
                        "id": 80000,
                        "ts": "2014-10-02T22:47:32Z",
                        "origin": "NLRTM VIA NOK",
                        "destination": "ST.PETERSBURG",
                        "eta": "2014-10-06T14:00:00Z"
                }]
}
```


6.13.1.4.1. Activity Diagram



6.13.1.4.2. Route(sPrepareRoute)

Condition	All	System	IMDate	Type	Rest
		Service	GetVoyages	Http Method	GET

6.13.1.4.3. Error Codes

The service returns following HTTP status codes:

- 200 – if the request was fulfilled successfully.
- 500 – if a problem on the server side occurred
- 400 – if the request sent by the client system cannot be interpreted.

6.13.1.5. alerts

Service	alerts	Version	1
Description	Returns all the alerts specified by the search criteria.	Http Method	GET
Endpoint	https://<host>[:<port>]/EMSAMobile/v1/alerts?user=<user>&project=<project>&startTime=<startTime>&endTime=<endTime>&wkt=<wkt>		

Input Arguments			
Argument	Type	Occurs	Description
user	String	1..1	User name
project	String	0..1	Name of the project. If not provided the default project is assumed.
startTime	DateTime	1..1	Begin time window
endTime	DateTime	1..1	End time window
wkt	WKT Polygon	0.. 1	Polygon to be specified using the WKT convention.
Output			
Argument	Type	Occurs	Description
status	String	1..1	Indicates the status of the request.
result	Array	1..1	
startTime	DateTime	1..1	Date time of start event
endTime	DateTime	1..1	Date time of end event
lat	Float	1..1	Latitude of the event
lon	Float	1..1	Longitude of the event

eventType	String	1..1	Type of event
Id	Interger	1..1	Unique internal ID of the event
report	String	1..1	Boolean flag. True if the alert has an associated report, false otherwise.
vessel	Object	0.1	A sequence defining some key characteristics of the vessel.
id	Interger	1..1	Internal vessel Id.
mmsi	String	0..1	Vessel MMSI
imo	String	0..1	Vessel IMO
name	String	0..1	Ship name
fs	String	0..1	Flat state of vessel.
cs	String	0..1	Call Sign

Request Example

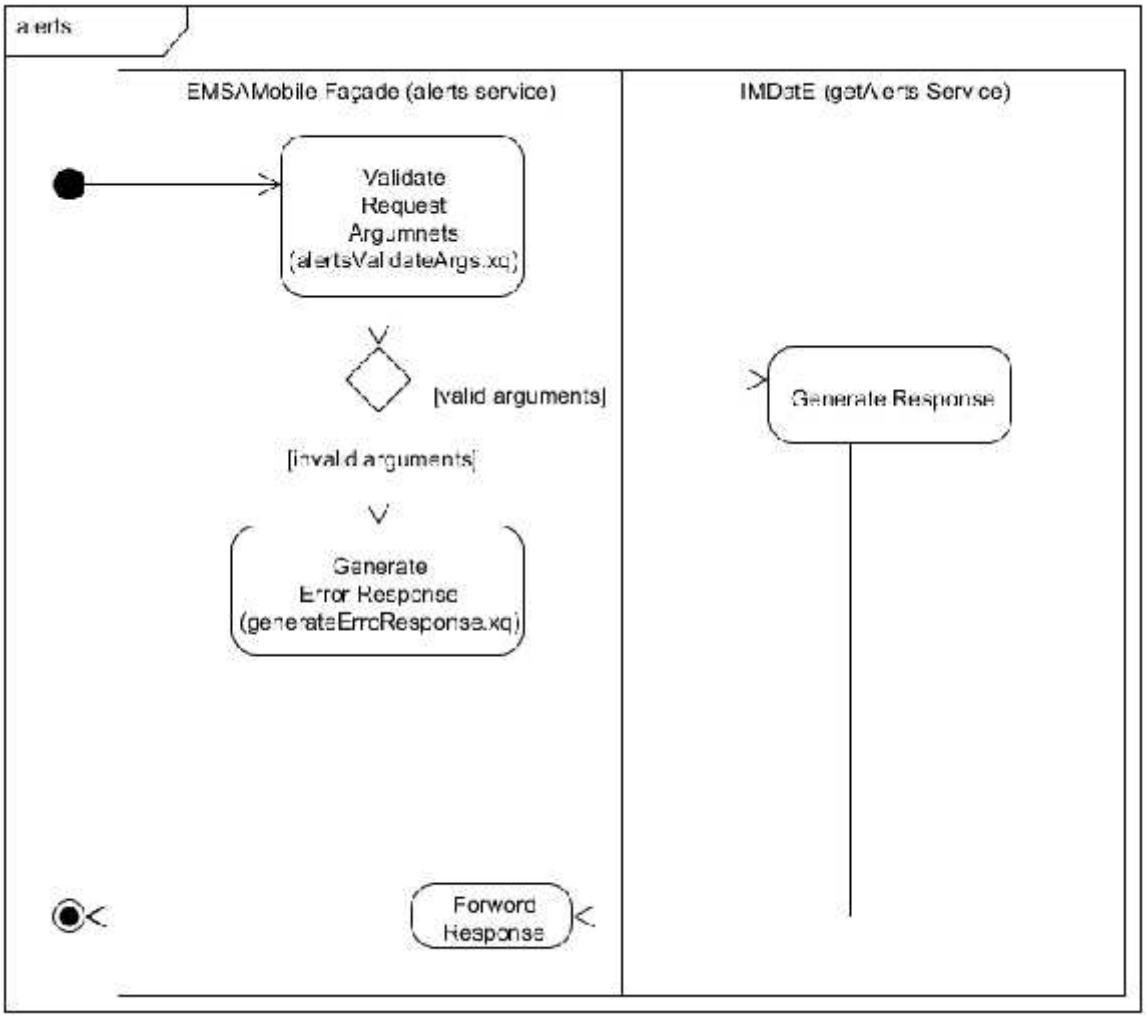
GET:

https://<host>[:<port>]/ EMSAMobile/v1/alerts?user=BILL&project=default&startTime=2014-05-06T12:21:56Z&endTime=2014-05-20T12:21:56Z&wkt=POLYGON((14.879317400542782%2037.727134127779024,%2016.407540635752195%2037.727134127779024,%2016.407540635752195%2038.56162257041281,%2014.879317400542782%2038.56162257041281,%2014.879317400542782%2037.727134127779024))

Response Example


```
{
	"status": "success",
	"result": [{
		"startTime": "2014-05-15T14:59:36Z",
		"endTime": "2014-05-15T14:59:36Z",
		"lat": 38.21733,
		"lon": 15.59798,
		"eventType": "anc_15",
		"id": "1400165996305",
		"report": true,
		"vessel": {
			"imo": "9487380",
			"id": "122973",
			"name": "ANZER",
			"fs": "TR",
			"mmsi": "271043399",
			"cs": "TCVR5"
		}
	}]
}
```


6.13.1.5.1. Activity Diagram



6.13.1.5.2. Route(sPrepareRoute)

Condition	All	System	IMDatE	Type	Rest
		Service	getAlerts	Http Method	GET

6.13.1.5.3. Error Codes

The service returns following HTTP status codes:

- 200 – if the request was fulfilled successfully.
- 500 – if a problem on the server side occurred
- 400 – if the request sent by the client system cannot be interpreted.

6.13.1.6. incidents

Service	tracks	Version	1
Description	This method can be used for	Http Method	GET

	getting the incidents.		
Endpoint	https://<host>[:<port>]/EMSAMobile/v1/incidents/<incidentId>?user=<user>&project=<project>&beginTimeStamp=< beginTimeStamp>&endTimeStamp=<endTimeStamp>&polygon=<>&category=<category>&maxX=<maxX>&maxY=<maxY>&minX=<minX>&minY=<minY>		
Input Arguments			
Argument	Type	Occurs	Description
incidentId	String	0..1	incident id
user	String	0..1	User performing the request
project	String	0..1	Project performing the request. If not provided assumed default.
beginTimeStamp	DateTime	0..1	Begin time stamp of the request.
endTimeStamp	DateTime	0..1	End time stamp of the request.
polygon	Polygon	0..1	Polygon expressed in WKT convention.
category	String	0..1	Incident category (possible values are: AREA_INCIDENT, LOCAL_INCIDENT, SHIP_INCIDENT)
maxX	String	0..1	Max longitude of the request.
maxY	String	0..1	Max latitude of the request.
minX	String	0..1	Min longitude of the request.
minY	String	0..1	Min latitude of the request.
Output			
Argument	Type	Occurs	Description
status	String	1..1	Indicates the status of the request. Always return success backend system return data.
result	Array	1..1	Array of grid elements
id	Integer	1..1	Unique identifier of the incident.
category	String	1..1	Category of incident, as defined by the CDF.
source	String	1..1	Source of the incident.
TS	DateTime	1..1	Time stamp of the incident.
description	String	0..1	Description of the incident.
incidentTypeId	Integer	1..1	Internal ID mapping the incident types (1=MotherShip, 2=PAG, 3=Attack, 4=SuspiciousApproach, 5=other).
incidentTypeCode	String	1..1	Incident type as defined by the CDF element IncidentTypeType.
expireDate	DateTime	1..1	Expiration date of the incident (defined by the endTime of the CDF).

radius	Integer	0..1	Radius of the incident (to be used for the display).
position	String	1..1	Position of the incident.

Request Example

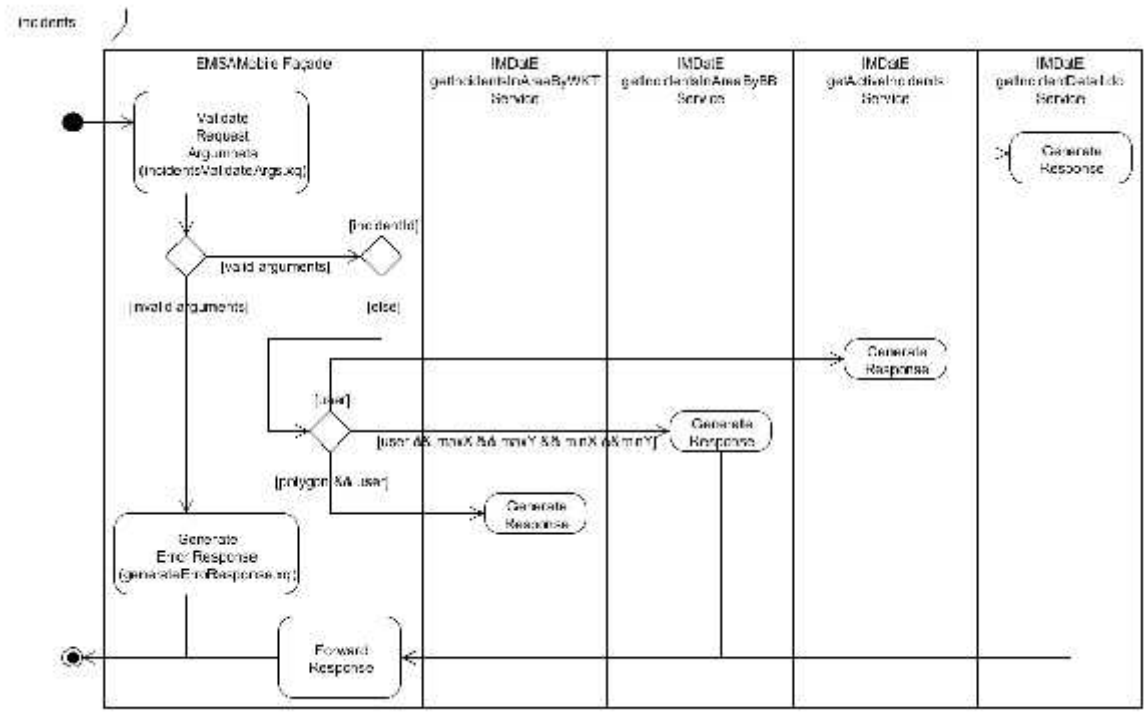
GET:

http://tesb01:7101/EMSAMobile/v1/incidents?user=CARREPH1

Response Example


```
{
 "result": [{
  "position": "POINT (13.5 40.5)",
  "id": 1703,
  "category": "SHIP_INCIDENT",
  "source": "MSCHOA",
  "description": "Test n. 2",
  "expireDate": "2014-12-20T16:34:57.000Z",
  "incidentTypeId": 4,
  "TS": "2014-11-21T11:34:57.000Z",
  "incidentTypeCode": "SuspiciousApproach"
 },
 {
  "position": "POINT (13.7 40.7)",
  "id": 1704,
  "category": "SHIP_INCIDENT",
  "source": "MSCHOA",
  "description": "Test n. 2",
  "expireDate": "2014-12-20T16:34:57.000Z",
  "incidentTypeId": 4,
  "TS": "2014-11-21T11:34:57.000Z",
  "incidentTypeCode": "SuspiciousApproach"
 },
 {
  "position": "POINT (13.9 40.9)",
  "id": 1705,
  "category": "SHIP_INCIDENT",
  "source": "MSCHOA",
  "description": "Test n. 2",
  "expireDate": "2014-12-20T16:34:57.000Z",
  "incidentTypeId": 4,
  "TS": "2014-11-21T11:34:57.000Z",
  "incidentTypeCode": "SuspiciousApproach"
 },
 {
  "position": "POLYGON ((8.9 35.9, 10.9 37.9, 8.9 33.9, 8.9 35.9, 8.9 35.9))",
  "id": 1706,
  "category": "AREA_INCIDENT",
  "description": "Area n.1",
  "expireDate": "2014-12-20T16:34:57.000Z",
  "incidentTypeId": 4,
  "TS": "2014-11-21T11:35:57.000Z",
  "incidentTypeCode": "SuspiciousApproach"
 }],
 "status": "success"
}
```

6.13.1.6.1. Activity Diagram



6.13.1.6.2. Route(sPrepareRoute)

Condition	arguments defined: maxX && maxY && minx && minY && user	System	IMDatE	Type	Rest
		Service	getIncidentsInAreaByBB.do	Http Method	GET
Condition	arguments defined: user	System	IMDatE	Type	Rest
		Service	getActiveIncidents.do	Http Method	GET
Condition	arguments defined: incidentId	System	IMDatE	Type	Rest
		Service	getIncidentDetail.do	Http Method	GET
Condition	arguments defined: polygon && user	System	IMDatE	Type	Rest
		Service	getIncidentsInAreaByWKT.do	http Method	GET

6.13.1.6.3. Error Codes

The service returns following HTTP status codes:

- 200 – if the request was fulfilled successfully.

- 500 – if a problem on the server side occurred
- 400 – if the request sent by the client system cannot be interpreted.

6.13.1.7. particulars

Service	particulars	Version	1
Description	Retrieve Ship particulars from Imdate OVR	Http Method	GET
Endpoint	https://<host>[:<port>]/EMSAMobile/v1/particulars/<id>?user=<user>&project=<project>		

Input Arguments			
Argument	Type	Occurs	Description
user	String	0..1	User who made the request
project	String	0..1	if specified, additional info for specific projects are retrieved
id	String	1..1	Imdate ID

Output			
Argument	Type	Occurs	Description
status	String	1..1	Indicates the status of the request.
result	Object	0..1	
thetisTypeCode	integer	1..1	PSC Ship type
thetisTypeDescription	string	1..1	PSC type Description
callSign	string	0..1	Ship Call Sign
countryCode	string	0..1	ISO Code for the country
dimension	integer	0..1	Vessel length
id	integer	1..1	imdate ID
identitySource	string	1..1	source of the Identity
imo	integer	0..1	IMO Number
ir	String	0..1	IR number, for fishing vessels
mmsi	Integer	0..1	MMSI
shipName	String	0..1	Ship Name
singleHull	String	0..1	Flag indicating if it is a Single Hull tanker. It can be: true/false.

otherIds	String	0..*	Other IDs, identified by a number key values pairs.

Request Example

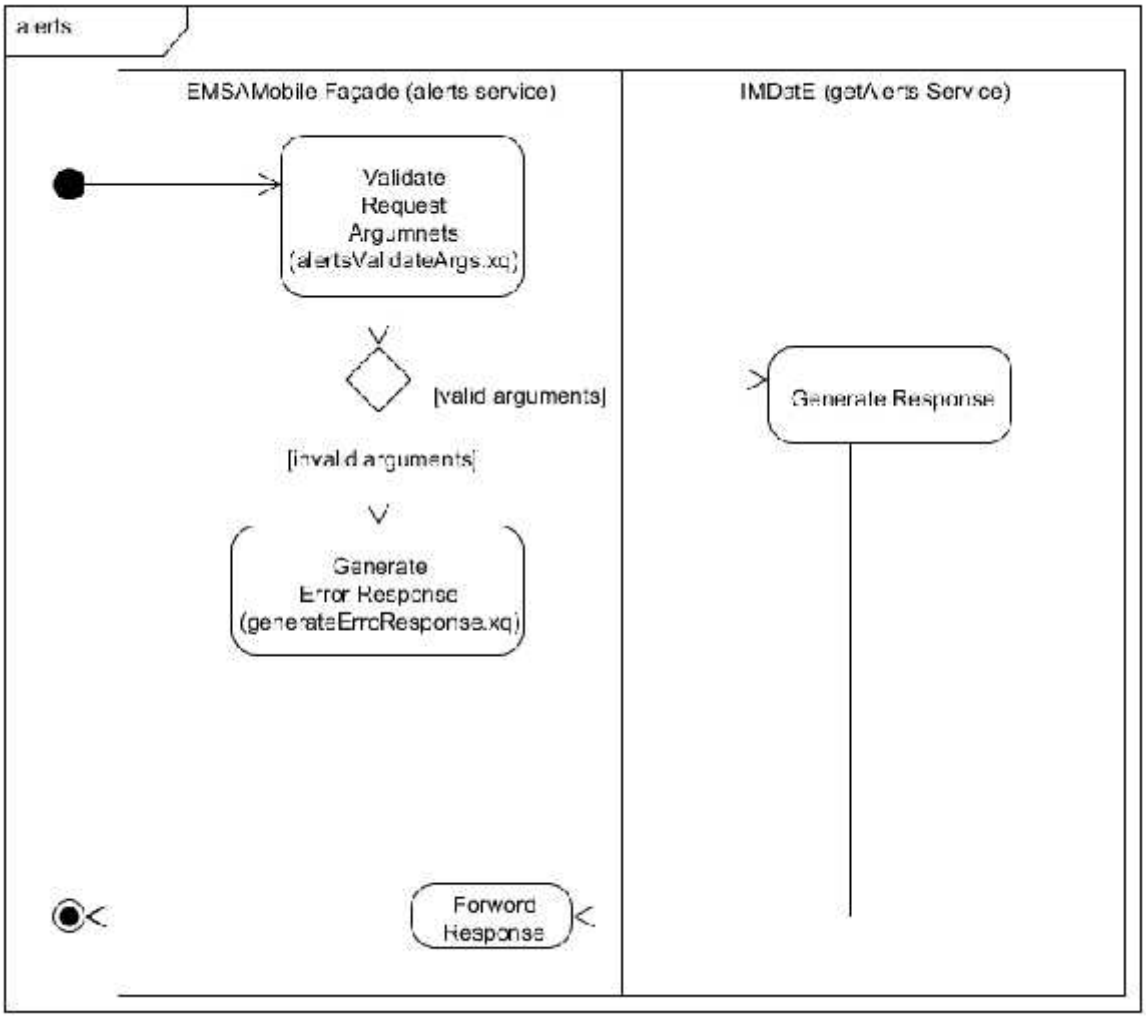
GET:

https://<host>[:<port>]/EMSAMobile/v1/particulars?id=1&user=CARREPH1

Response Example


```
{
	"status": "success",
	"result": {
		"imo": 5045811,
		"shipName": "CANADIAN VOYAGER",
		"identitySource": "SSN",
		"id": 1
	}
}
```


6.13.1.7.1. Activity Diagram



6.13.1.7.2. Route(sPrepareRoute)

Condition	All	System	IMDatE	Type	Rest
		Service	ovrInfo	Http Method	GET

6.13.1.7.3. Mapping

to	
From Field	To Field
N/A	
from	
From Field	To Field
N/A	

6.13.1.7.4. Error Codes

The service returns following HTTP status codes:

- 200 – if the request was fulfilled successfully.
- 500 – if a problem on the server side occurred
- 400 – if the request sent by the client system cannot be interpreted.

6.13.2. Business Services

6.13.2.1. getPositions

Service	getPositions	System	IMDatE	Version	
		Service Type	Rest	Protocol	HTTP
Description	Returns the ships counts in the specified time range over a 4-level geohash grid.				
Endpoint URI	http://<host>:<port>/<system servlet listener>/<service name>				

6.13.2.2. grid

Service	grid	System	IMDatE	Version	
		Service Type	Rest	Protocol	HTTP
Schemas		System	GDSP		
Description	Returns the ships counts in the specified time range over a 4-level geohash grid.				
Endpoint URI	http://<host>:<port>/<system servlet listener>/<service name>				

6.13.2.3. tracks

Service	tracks	System	CustomerManagement	Version	
		Service Type	SOAP	Protocol	HTTP
Description	This method gets the ship positions tracks defined by request option				
Endpoint URI	http://<host>:<port>/<system servlet listener>/<service name>				
Operations					

getTracksByVesselId
getTracksByBoundingBox

6.13.2.4. voyages

Service	voyages	System	CustomerManagement	Version	
		Service Type	Rest	Protocol	HTTP
Description	Returns all the voyage info for the vessel specified by the id in the time window specified				
Endpoint URI	http://<host>:<port>/<system servlet listener>/<service name>				

6.13.2.5. alerts

Service	getAlerts	System	IDDatE	Version	
		Service Type	Rest	Protocol	HTTP
Description	Returns all the alerts specified by the search criteria.				
Endpoint URI	http://<host>:<port>/<system servlet listener>/<service name>				

6.13.2.6. getIncidentDetail

Service	getIncidentDetail.do	System	IDDatE	Version	
		Service Type	Rest	Protocol	HTTP
Description	This method can be used for getting additional information for the incidents of type SHIP_INCIDENT, e.g. data about the ship position, etc.				
Endpoint URI	http://<host>:<port>/<system servlet listener>/<service name>				

6.13.2.7. getShipParticulars

Service	ovrInfo	System	IMDatE	Version	
		Service Type	Rest	Protocol	HTTP

Description	Retrieve Ship particulars from Imdate OVR.
Endpoint URI	http://<host>:<port>/<system servlet listener>/<service name>

6.13.2.8. getActiveIncidents

Service	getActiveIncidents.do	System	IMDatE	Version	1
		Service Type	Rest	Protocol	HTTP
Schemas		System	GDSP		
Description	This method can be used for getting the incidents.				
Endpoint URI	http://<host>:<port>/<system servlet listener>/<service name>				

6.13.2.9. getIncidentsInAreaByBB

Service	getIncidentsInAreaByBB.do	System	IMDatE	Version	
		Service Type	Rest	Protocol	HTTP
Description	This method can be used for getting the incidents.				
Endpoint URI	http://<host>:<port>/<system servlet listener>/<service name>				

6.13.2.10. getIncidentsInAreaByWKT

Service	getIncidentsInAreaByWKT.do	System	IMDatE	Version	1
		Service Type	Rest	Protocol	HTTP
Description	This method can be used for getting the incidents.				
Endpoint URI	http://<host>:<port>/<system servlet listener>/<service name>				

6.14. CSNDC Domain

6.14.1. Proxy Service

6.14.1.1. oilSpills

Service	incident	Version	1
Description	This method can be used for getting information for oil spills	Http Method	GET
Endpoint	https://<host>[:<port>]/EMSAMobile/v1/oilSpills?beginDate=<beginDate>&endDate=<endDate>&minX=<minX>&minY=<minY>&maxX=<maxX>&maxY=<maxY>&maxFeatures=<maxFeatures>		

Input Arguments			
Argument	Type	Occurs	Description
beginDate	Datetime	1..1	
enddate	Datetime	1..1	
minX	Integer	0..1	
minY	Integer	0..1	
maxX	Integer	0..1	
maxY	Integer	0..1	
maxFeatures	Integer	1..1	

Output			
Argument	Type	Occurs	Description
status	String	1..1	Indicates the status of the request.
result	Array	1..1	

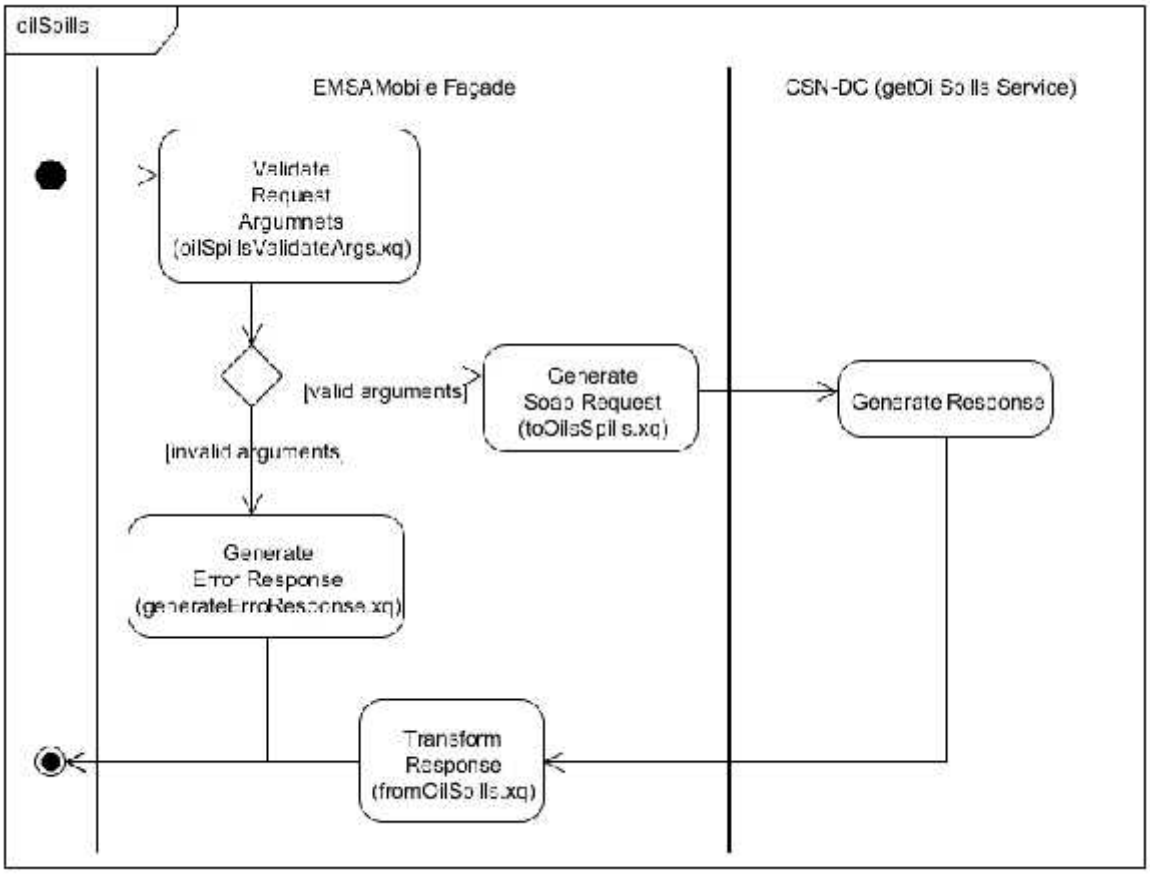
Request Example

GET: https://<host>[:<port>]/EMSAMobile/v1/oilSpills?beginDate=2014-11-01T10:44:42Z&endDate=2014-11-02T10:44:42Z&maxFeatures=10&user=CARREPH1

Response Example

{"result":{"FeatureCollection":{"numberOfFeatures":0}},"status":"success"}

6.14.1.1.1. Activity Diagram



6.14.1.1.2. Route(sPrepareRoute)

Condition	All	System	CSNDC	Type	Rest
		Service	getOilSpills	Operation	GET

6.14.1.1.3. Mapping

to	
From Field	To Field
N/A	
from	
From Field	To Field
N/A	

6.14.1.1.4. Error Codes

The service returns following HTTP status codes:

- 200 – if the request was fulfilled successfully.
- 500 – if a problem on the server side occurred
- 400 – if the request sent by the client system cannot be interpreted.

6.14.2. Business Service

6.14.2.1. getOilSpills

Service	getOilSpills	System	CSNDC	Version	
		Service Type	Rest	Protocol	HTTP
Description	This operation returns a map				
Endpoint URI	http://<host>:<port>/<system servlet listener>/<service name>				

6.15. ENCWS Domain

6.15.1. Proxy Services

6.15.1.1. cmap

Service	cmap	Version	1
Description	This operation returns a map	Http Method	GET
Endpoint	https://<host>[:<port>]/EMSAMobile/v1/cmap?LAYERS=<LAYER>&STYLES=<STYLES>&SRS=<SRS>&FORMAT=<FORMAT>&VERSION=<VERSION>&REQUEST=<REQUEST>&BBOX=<BBOX>&WIDTH=<WIDTH>&HEIGHT=<HEIGHT>		

Input Arguments			
Argument	Type	Occurs	Description
VERSION	String	1..1	Request version
REQUEST	String	1..1	Request name
LAYERS	String	1.1	Comma separated list of one or more layers
STYLES	String	1..1	Comma-separated list of one rendering style per requested layer

SRS	String	1..1	Coordinate reference system
BBOX	String	1..1	Bounding box corners (lower left, upper right) in CRS units
WIDTH	Integer	1..1	Width in pixels of map picture
HEIGHT	Integer	1..1	Height in pixels of map picture
FORMAT	String	1..1	Output format of map
TRANSPARENT	String	0..1	Background transparency of map (default=FALSE).
BGCOLOR	String	0..1	Hexadecimal red-green-blue colour value for the background color (default=0xFFFFFF)
TIME	Time	0..1	Time value of layer desired
ELEVATION	String	0..1	Elevation of layer desired

Output

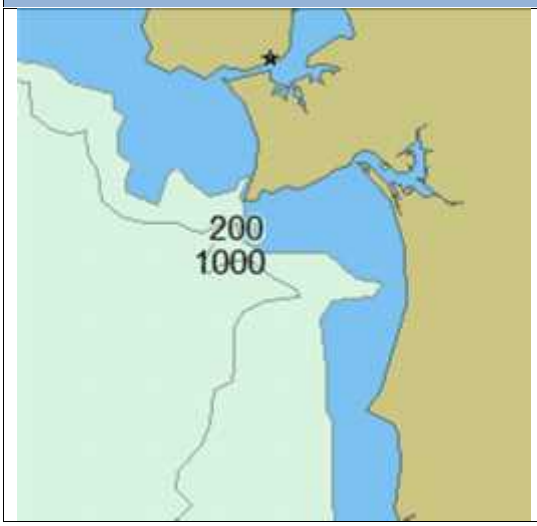
Argument	Type	Occurs	Description
Map	Binary File	1..1	Image that satisfies input arguments

Request Example

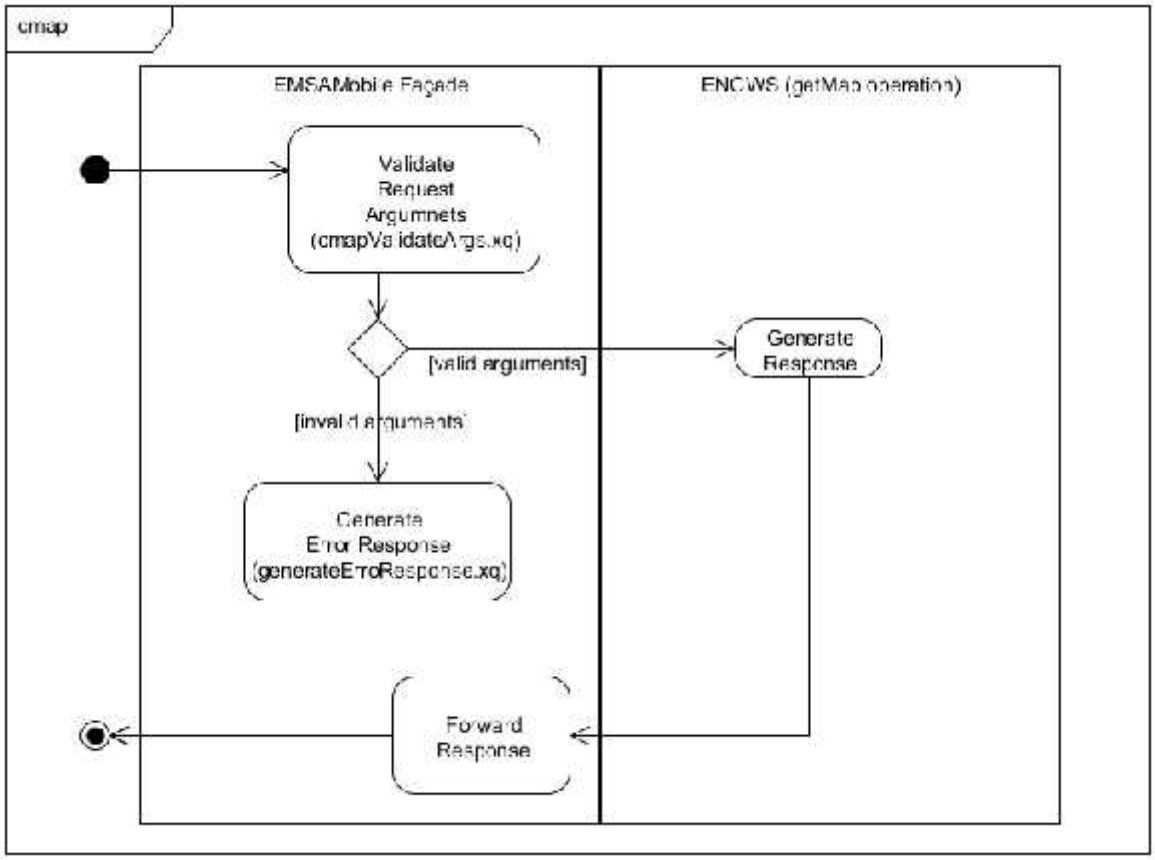
GET:

https://<host>[:<port>]/EMSAMobile/v1/cmap?LAYERS=S52%20Base&STYLES=S52&SRS=EPSG%3A3395&FORMAT=image%2Fpng&VERSION=1.1.1&REQUEST=GetMap&BBOX=-939258.203568,4357728.578321,-782715.169640,4513604.392918&WIDTH=256&HEIGHT=256

Response Example



6.15.1.1.1. Activity Diagram



6.15.1.1.2. Route(sPrepareRoute)

Condition	All	System	ENCWS	Type	Rest
		Service	getMap	Operation	GET

6.15.1.1.3. Mapping

to	
From Field	To Field
N/A	
from	
From Field	To Field
N/A	

6.15.1.1.4. Error Codes

The service returns following HTTP status codes:

- 200 – if the request was fulfilled successfully.
- 500 – if a problem on the server side occurred
- 400 – if the request sent by the client system cannot be interpreted.

6.15.2. Business Services

6.15.2.1. getMap

Service	getMap	System	ENCWS	Version	
		Service Type	Rest	Protocol	HTTP
Description	This operation returns a map				
Endpoint URI	http://<host>:<port>/<system servlet listener>/<service name>				