



D5.2 Demonstration of prototype application for automated port reporting

Project no.

636329

Project acronym:

EfficienSea2

EFFICIENSEA2 – efficient, safe and sustainable traffic at sea

Funding scheme:

Innovation Action (IA)

Start date of project:

1 May 2015

End date of project:

30 April 2018

Duration:

36 months

Due date of deliverable:

31.01.2018

Actual submission date:

26.01.2018

Organisation in

charge of deliverable:

BIMCO and GateHouse Maritime A/S

DOCUMENT STATUS

Authors

Name	Organisation
Morten Christian Larsen	GateHouse Maritime A/S
Jeppe Skovbakke Juhl	BIMCO

Document History

Version	Date	Initials	Description
0.1	08.01.2018	MCL	Initial draft
0.2	12.01.2018	JSJ	Updated initial draft
0.3	23.01.2018	MCL	Updated initial draft
Final	25.01.2018	MCL	Final version

Review

Name	Organisation
Ole A. Jensen (Partner, Senior Consultant)	SeaSolutions
Martin Svenning Svanholm Ahl (Naval Staff)	Defence Command Denmark

Contents

DOCUMENT STATUS	2
Authors	2
Document History	2
Review	2
1 Introduction	4
1.1 E2 Work Package 5 – Administrative burdens and exhaust emissions	4
2 Scope	5
3 Use Case and requirements	6
3.1 Technical Requirements	7
4 Service specification	9
5 REST Schema	10
5.1 Port Info	10
5.1.1 Example GET /portinfo/<locode>/	10
5.2 Required information for Port	10
5.2.1 Example GET /port/<locode>/	11
5.2.2 Example GET /port/<locode>/<voyagetype>.....	11
5.3 Submit information to port	12
5.3.1 Example Simple Payload	12
5.3.2 Example complex Payload	12
6 Implementation	14
7 Online test of the demonstrator	15
8 Conclusion and outlook	16

1 Introduction

The EfficienSea2 (E2) project aims at creating and implementing smart solutions for efficient, safe and sustainable traffic at sea through improved connectivity for ships. This is achieved by global collaboration, use of open-source software and an explicit aim for standardized solutions.

This report details an open source demonstrator capable of facilitating automatic reporting from ship to shore.

1.1 E2 Work Package 5 – Administrative burdens and exhaust emissions

WP 5 consists of 3 subtasks, of which 5.1 and 5.2 addresses a reduction of the administrative burdens for seafarers, and 5.3 is about regulatory limitations in SOx content in exhaust from ships.

This report has been prepared by the members of WP5 and consider the exchange of information between ship and shore in relation to port calls, referring to subtasks 5.1 and 5.2.

The first deliverable of WP5 described in detail the concept and structure of the E2 proposed e-solution. This report considered the development of a new common port database concept and structure, D5.3, was delivered in M10, February 2016.

The second deliverable explained in detail the Draft S-100 product specification, D5.1. This report was delivered in M21, January 2017, and explained the technical service and product specification associated with the e-solution. D5.1 was closely associated with the third report, D5.4, providing a brief status on the online test of ship data transfer. D5.4 was delivered in M30, October 2017.

The current report, D5.2, provides a description of the demonstrator, the aims to describe the prototype application that promotes information exchange and reduces administrative burdens with regards to reporting and collecting information during port calls.

The concluding report, D5.5 which will be submitted February 2018, will detail the effect of implementing an e-solution for the exchange of information between ship and shore in terms of delivering pre-arrival documents.

2 Scope

In this report, we detail the prototype demonstrator for exchanging data between ship and shore, aiming to enable widespread use of computers instead of manually entering the same data again and again. The concept builds on the use of computer applications on board the ship as well as ashore for both commercial partners and authorities.

The scope of the demonstrator is limited to the use of a developed prototype, and the current iteration of the list of data element id's (attribute id's) as described in our first report, D5.1 Draft S-100 product specification.

The developed prototype provides a secure REST interface with several specific services available. Access to the service is intended to be provided through Maritime Connectivity Platform, with each party identified using the Maritime Connectivity Platform certificate. The Maritime Connectivity Platform layer is not yet implemented, and is not detailed in this document.



3 Use Case and requirements

To ensure the best possible implementation, a use case has been specified based on the scenarios set out in D5.3. The use case describes a scenario where a container vessel with a crew of 15 people without passengers arrives at and departs from a number of ports:

- Port of Gdansk, Poland
- Port of Aarhus, Denmark
- Port of Bremerhaven, Germany
- Port of Rotterdam, the Netherlands.

On this trip, the ship will have different reporting tasks to perform – between some ports, the tasks will be more or less the same and between others, they will vary. This constitutes a burden for the crew on board, but also for the stakeholders ashore receiving the information.

In the list of data element, it is described what information must be exchanged between the ship and shore from the time the ship approaches Gdansk, Poland, with its final destination in Rotterdam, the Netherlands via port calls at Aarhus, Denmark and Bremerhaven, Germany. The information is to a large extent based on filling in pre-defined forms, either from EU SafeSeaNet (SSN) or FAL forms from IMO. Additionally, a number of certificates must be sent from the ship to shore.

To ensure proper implementation, the use case has been further tightened, and the reporting to shore now mimics the Danish National Single Window, through the SafeSeaNet.dk.

The sequence diagram below details the calls and responses (APR is Automated Port Reporting prototype).

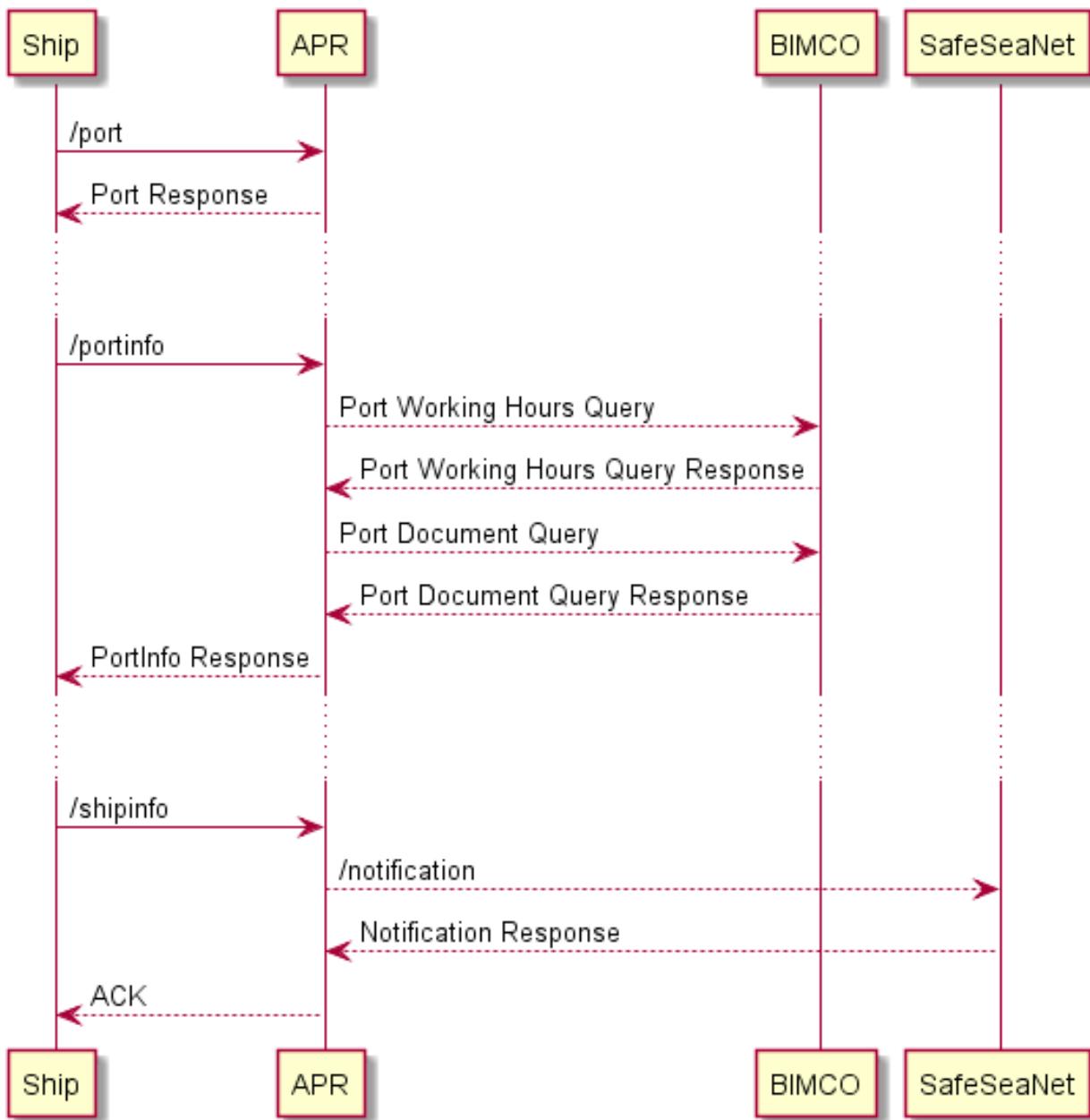


Figure 1 Sequence Diagram.

3.1 Technical Requirements

From the use case, a number of detailed requirements have been extracted and are detailed below.

- Requesting reporting formalities. A user/vessel must be able to request and receive a list of required reporting formalities.
- Requesting port information. A user/vessel must be able to request and receive a list of port information including port services. In case of the current solution, the BIMCO

port database will illustrate the exchange of information on opening hours from the port.

- Submitting arrival information. A user/vessel must be able to ready the required information, and upon approval from the Captain, send the information to the prototype for distribution.
- Acknowledgement. A user/vessel must be notified, that the transaction has been completed successfully.
- Data structure. Data must be sent in JSON format to ensure compatibility with SafeSeaNet, which is used as testing partner.



4 Service specification

The service specification describes in general the given service at the logical level in a technological manner, but at the meta level, by providing, for instance:

- the operational context of the service, e.g. requirements, use cases,
- the service interface descriptions, e.g. operations, parameters,
- the data structures used by the service (the service data model),
- the dynamic behaviour of the service (sequence of operations),
- the author of the service specification (organisation, contact person).

The service specification does, however, not describe the details of a specific service implementation. For that purpose, a service description of the technical design has to be provided, where the actual realisation of the service with a dedicated technology is described.

This report detailing the prototype application is to be considered a service technical design document. The design is based upon the Service Specification, delivered in D5.1.

5 REST Schema

The prototype interface will be implemented using a REST interface. This section describes the implemented REST calls.

5.1 Port Info

This call enables the user/vessel to retrieve additional information about any port in the BIMCO database. The data is delivered to the user/vessel as is (JSON or HTML), and the receiver can update the local database for future use. Retrieve information about a port. The information is retrieved from the BIMCO API.

The call facilitated using the LOCODE of the port.

5.1.1 Example GET /portinfo/<locode>/

Call:

```
GET /portinfo/DKAAR/
```

Response:

```
{  
  "locode": "DKAAR",  
  "information": "Working hours:\nMonday - Thursday: Normal hours 07.00-16.00. Overtime 06.00-07.0  
0 and 16.00-18.00 or 16.00-19.00 at 50% extra. 00.00-06.00 and 18.00-24.00 or 19.00-07.00 at 100% ex  
tra.\nFriday: Normal hours 07.00-15.30. Overtime 06.00-07.00 and 15.30-17.30 or 15.30-18.30 at 50% e  
xtra. 00.00-06.00 and 18.00-24.00 or 18.30-07.00 at 100% extra.\nMeal breaks 08.30-09.00, 12.00-13.0  
0, 16.30-17.30, 22.00-22.30, 02.30-03.00. Meal break surcharge minimum 50%. Bulk cargoes usually dis  
charged/loaded without meal breaks.\nNo shift work, but work can be performed round the clock.\nSatu  
rday: 07.00-10.00 or 06.00-09.00 at 50% extra. 10.00-06.00 or 09.00-06.00 at 100% extra.\nSundays an  
d holidays: Work is available 00.00-24.00 at 100% extra.\nSaturday, Sunday, holidays: Work must be o  
rdered latest 11.00 on preceding normal working day.\nWork after 12.00 at 100% extra on 1 May (Labou  
r Day) and 5 June (Constitution Day).\nWork is available 00.00-24.00 at 100% extra on 24 December (C  
hristmas Eve) and 31 December (New Year's Eve).\n"}  
}
```

5.2 Required information for Port

This call enables the user/vessel to retrieve information about the elements that each port requires when submitting arrival information. The data is delivered to the user/vessel as JSON. By specifying only the LOCODE, basic information is returned, see example 5.2.1. Note that the `requestelements` array is always empty in this case. When using the call with a `voyagetype` the call returns a list of requested elements, see example 5.2.2.

5.2.1 Example GET /port/<locode>/

Call:

```
GET /port/DKAAR/
```

Response:

```
{
  "id":1,
  "locode":"DKAAR",
  "name":"Aarhus",
  "requestelements":[]
}
```

5.2.2 Example GET /port/<locode>/<voyagetype>

Call:

```
GET /port/DKAAR/Arrival
```

Response:

```
{
  "id": 1,
  "locode": "DKAAR",
  "name": "Aarhus",
  "requestelements": [
    {
      "name": "Ship name",
      "type": "String",
      "DataElementId": "Ship.Name",
      "schedule": 72,
      "voyagetype": "Arrival",
      "required": true,
      "description": "Given name of the ship in the ship registry"
    },
    {
      "name": "IMO number",
      "type": "String",
      "DataElementId": "Ship.Imo",
      "schedule": 72,
      "voyagetype": "Arrival",
      "required": true,
      "description": "Unique ship identification number assigned by Lloyd's Register - Fairplay in accordance with IMO resolution A.600(15)."
    },
    ...
  ]
}
```



5.3 Submit information to port

This call enables the user/vessel to POST information to a port or to SafeSeaNet. Implemented in the prototype is only the reporting to SafeSeaNet. It is always a requirement to include the Ship.Imo field in every POST, to uniquely identify the ship.

The contents of the payload are converted to SSN format, forwarded to SafeSeaNet, and the SafeSeaNet response is returned.

The POST can be used to make a SafeSeaNet draft, that can be updated, and completed at a later time. This functionality requires the acknowledgement reference to be added to future POST calls.

5.3.1 Example Simple Payload

Call:

```
{  
  "Ship": {  
    "Imo": "1110001"  
  },  
  "Port": {  
    "PortOfCall": {  
      "Locode": "DKAAL"  
    },  
    "EtaToPortOfCall": "2017-11-16T13:04:30Z"  
  }  
}
```

Returns:

```
{"success": true, "$reference": "W50"}
```

5.3.2 Example complex Payload

Call with reference:

```
{  
  "$reference": "W6P",  
  "Ship": {  
    "Imo": "1110001"  
  },  
  "Port": {  
    "PortOfCall": {  
      "Locode": "SWGOT"  
    },  
    "EtaToPortOfCall": "2017-11-18T13:04:30Z"  
  },  
  "person": {  
    "NumberOfCrew": 2,  
    "Crew": [  
      {  
        "name": "John Doe",  
        "position": "Captain",  
        "rank": "Master",  
        "age": 45  
      },  
      {  
        "name": "Jane Smith",  
        "position": "First Officer",  
        "rank": "First Officer",  
        "age": 35  
      }  
    ]  
  }  
}
```



```

    "FamilyName": "Testersson",
    "GivenName": "Test",
    "Gender": {
        "Name": "Female"
    },
    "Rank": {
        "Name": "Second Mate"
    },
    "Nationality": {
        "Code": "IS"
    },
    "DateOfBirth": "1966-04-25",
    "PlaceOfBirth": "Reykjavik"
},
{
    "FamilyName": "Dente",
    "GivenName": "Al",
    "Gender": {
        "Name": "Male"
    },
    "Rank": {
        "Name": "Captain"
    },
    "Nationality": {
        "code": "IT"
    },
    "DateOfBirth": "1951-07-21",
    "PlaceOfBirth": "Milano"
}
]
}
}

```

Returns:

```
{"success": true, "$reference":"W6P"}
```



6 Implementation

The implementation of the prototype has been realized using Django and Django REST Framework. This means that it requires Python to run.

The prototype consists of three logical parts:

1. A database which associates a port with a list of request elements. For each association, a flag/tick indicates whether the element is required, and a number indicates how many hours prior to arrival/departure it must be sent. The database is exposed via a REST API (web service), intended to facilitate building a graphical application for data entry. (Such an application has been developed during the project, but is not contained in this repository, as it is not open source).
2. An API for retrieving information about a port. This information is retrieved from another set of web services provided by BIMCO.
3. An API for submitting information to a port. In the current version of the prototype, the information is simply forwarded to the Danish National Single Window (SafeSeaNet) after being converted to an appropriate format.

The implementation is Open Source, and is available for use by other partners or other commercial parties at GitHub: https://github.com/gatehouse/apr_prototype



7 Online test of the demonstrator

For the proof of concept, WP5 have teamed up with a global provider of specialized maritime IT solutions and services in order to get a correct baseline of the setup. The service provider, SeaSolutions, has more than 25 years of experience within core shipping business areas, with its headquarters in Copenhagen and with offices in India and Malaysia. Providing software for more than 25 global shipowners and 800 ships, among others A.P.Moller-Maersk, SeaSolutions was the best match for the online test of the demonstrator.

As part of the test, WP5 got access to a version of the SeaSolutions software whereby we could build the connection to the APR. SeaSolutions generously offered to develop a special interface for connecting through the MCP. This was agreed upon in spring 2017.

In June 2017, MAERSK was hit by the Petya cyber-attack. The attack caused outages of the company's computer systems across the world. Computer servers across Europe and in India were hit by a major ransomware attack. Globally, the cyber-attack was affecting multiple business units at Maersk, including container shipping, port and tug boat operations, oil and gas production, drilling services, and oil tankers. The Petya attack also affected the E2 test cases.

As a result of the attack, the scheduled E2 onboard tests of the demonstrator was put on hold, as MAERSK was not interested to give access to any external systems at the time the online tests were scheduled.

The E2 were however allowed to make use of an equivalent online test version of the very same system, although the test system was not allowed onboard the ships.

Having considered the options, WP5 accepted the generous offer, by establishing a test protocol for the online test version – based on an exchange of information between two platforms (machine to machine) both placed ashore. The test environment was deemed very similar the onboard tests.

8 Conclusion and outlook

As expressed in the outlook in D5.1 “Draft S-100 product specification”, the scope of the demonstrator has been fairly limited. A reporting demonstrator has been developed, as well as testing stubs to simulate the vessel reporting, and port information distribution. Due to time restraints the demonstrator has not yet been fully implemented into the Maritime Connectivity Platform, but still we have achieved a successful proof of concept.

The demonstrator and associated testing stubs have shown that it is possible to automate some, if not all types of reporting, to e.g. SafeSeaNet. Testing was done with real sample data, but used via the online sandbox of the actual Danish SafeSeaNet, thus giving weight to the conclusion that machine-to-machine reporting is possible and desirable.

The source code is obviously open source and has been published online to enable any interested parties to further develop on the concepts proved here, or to extend the features to involve other maritime single windows etc.

A great deal of work has also been put into unifying the vessel attributes, and this a great step toward making machine-to-machine feasible and commercially viable.

The final report from the WP5 group, D5.5 will describe the outcome of the study on administrative tasks onboard. The objective of this study has been to establish quantitative estimates of time needed to complete 32 selected administrative tasks. The method was based on expert evaluation through a questionnaire based survey.

