BLAST

Bringing Land and Sea Together

The Interreg IVB North Sea Region Programme

Investing in the future by working together for a sustainable and competitive region

BLAST Work Package 5 (WP 5) Validation of the End Result

Date:

09032012

Author:

Ulrich Tagne

Authors

Name	*Organisation*
Ulrich Tagne	Norwegian Coastal Administration, NCA
Finn Martin Vallersnes	NCA

Reviewers

Name	*Organisation*
Jon Leon Ervik	NCA
Jarle Hauge	NCA

Approval of report

Name	*Organisation*	*Signature*	*Date*

Document history

Revision	*Date*	*Organisation*	*Initials*	*Revised pages*	*Short description of changes*
First Draft	05032012	NCA	fmv		
Draft 2					
Draft 3					
Draft 3.2					
Draft 3.3					
Draft 3.4					

Contents at a Glance

Contents

Chapter 1

Introduction

1.1 Background and Motivation

The rationale for Work Package 5 (WP5) was given in the project description of "Interreg IVB North Sea Programme: Bringing Land and Sea Together: BLAST", - version 01-12-2009:

"The WP addresses the need for wider interoperability of maritime traffic information in the North Sea area for use in decision making and resource management to improve the maritime safety and security."

"At present, there is a limited collaboration between Member States and within a Member State, regional, and local level to integrate maritime traffic information. SafeSeaNet covers parts of these needs, but the main purpose is to keep track of polluting and dangerous goods, waste, alert and security information and only National or Local competent Authorities have access. In this WP we will have the traffic players in focus, but the access rights will be a vital element to protect the information integrity.

This WP will concentrate on a best practice network amongst stakeholders from data providers to end users. It will develop pilot studies focusing on the practical issues and solutions of the harmonization of the maritime traffic information and recognize existing formats and propose new formats and/or standards to improve the information exchange".

The objectives of BLAST WP 5 were given as follows:

- Design and develop a regional maritime traffic monitoring platform beneficial for all Member States in the North Sea region.
- Harmonize maritime traffic information formats in the North Sea Region and add new formats where needed.
- Harmonize regional maritime traffic information flow with SafeSeaNet and propose new functionality.
- Develop a network and server platform for development and demonstration.

According to the WP5 work program a validation report should be part of the final evaluation. The present report presents the validation.

1.2 Present Work

1.2.1 Scope of work

This document establishes a verification and validation concept of the software system - North Atlantic Information Management Centre (NAIMC). It also discusses about the quality assurance aspect of the software. Measuring and assessing the quality of a complex software system is not trivial. A case study is made on the validation of a sub-system of the NAIMC and its quality is measured and assessed. Finally, a conclusion is drawn on the validation and quality of the software. Some final thoughts for future improvement are proposed.

1.2.2 Outline Structure of the Document

This document is organized into five chapters.

Chapter 1: Introduction
This chapter presents the background, motivation, and the scope of work for this document. Chapters are briefly summarized to give a quick inside of their contents.

Chapter 2: Software Verification and Validation
This section details the validation concept and methodology for computerized software such as the NAIMC. Applying the methodology is not a trivial and straight forward activity. The importance of collaboration between the modeler and the experimenter to achieve model acceptance or increase the confidence level for model acceptance is explained thoroughly.

Chapter 3: Software Quality Assurance
This section discusses the quality factors and quality criteria for quality software. It shows the importance of measuring the software quality to satisfy the user needs. A quality assurance aspect taking some of the V&V activities is clearly explained.

Chapter 4: Validation of an Alarm Sub-system
This section deals with the verification and validation of an alarm sub-system which is part of the overall NAIMC software system. Several tests and visualizations are conducted to check the sub-system against the standard software quality factors.

Chapter 5: Concluding Remarks
This section presents the specific objectives of this document. It briefly summarizes the driving forces and benefits for the NAIMC V&V and quality assurance.

Chapter 2

Software Verification and Validation

2.1 Software Validation Concept

The software verification and validation (V&V) is a system engineering discipline that uses a well defined methodology for evaluating and assessing the correctness, the reliability, the robustness and the quality of the software throughout the software life cycle.

V&V effort strives to ensure that the quality is built into the software and that the user requirements are satisfied into the software. For critical software, the overall objective should be to reduce the risk of inappropriate decisions caused by incorrect model outputs.
Assuming that the computerized NAIMC software nothing else but a model representing the real world, then it is obvious that the model V&V is what is required in the software V&V activities.

Verification is defined as the process of ensuring that the conceptual model has been carefully transformed into a computer model with sufficient accuracy (Davis, 1992); in other words, it has a mathematical concern of building the model right.

Validation is defined as the process of ensuring that the conceptual model is sufficiently accurate to represent the real world from the perspective of its intended uses; in other words, it has the physical concern of building the right model. It is applied only after the verification has been done.

The model V&V activity can be divided in two main domains, namely the problem domain and the model domain as shown in Fig. 1.

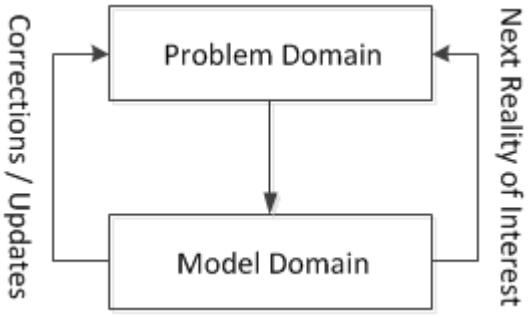


Figure 1: Domain Relationship in V&V

In fact, it is not possible to categorically prove that a model from a complex system is absolutely correct. Therefore the aim of V&V activities should not be to prove the correctness of the model,

but rather to ensure that the model is sufficiently accurate. This can be done by creating enough confidence in a model in order to achieve the acceptance agreement.

From another viewpoint, one may attempt to demonstrate that a model is in fact incorrect. If this exercise and effort cannot be proven, then one will come to the conclusion that the V&V has served to increase confidence in the model result.

Generally, modelers and users may have different opinions on confidence level. Some users may derive their confidence simply from the model's display, others may require more in-depth V&V before they are willing to believe the results. In any scenario, the modeler is responsible for guiding the users and ensuring that sufficient V&V is performed.

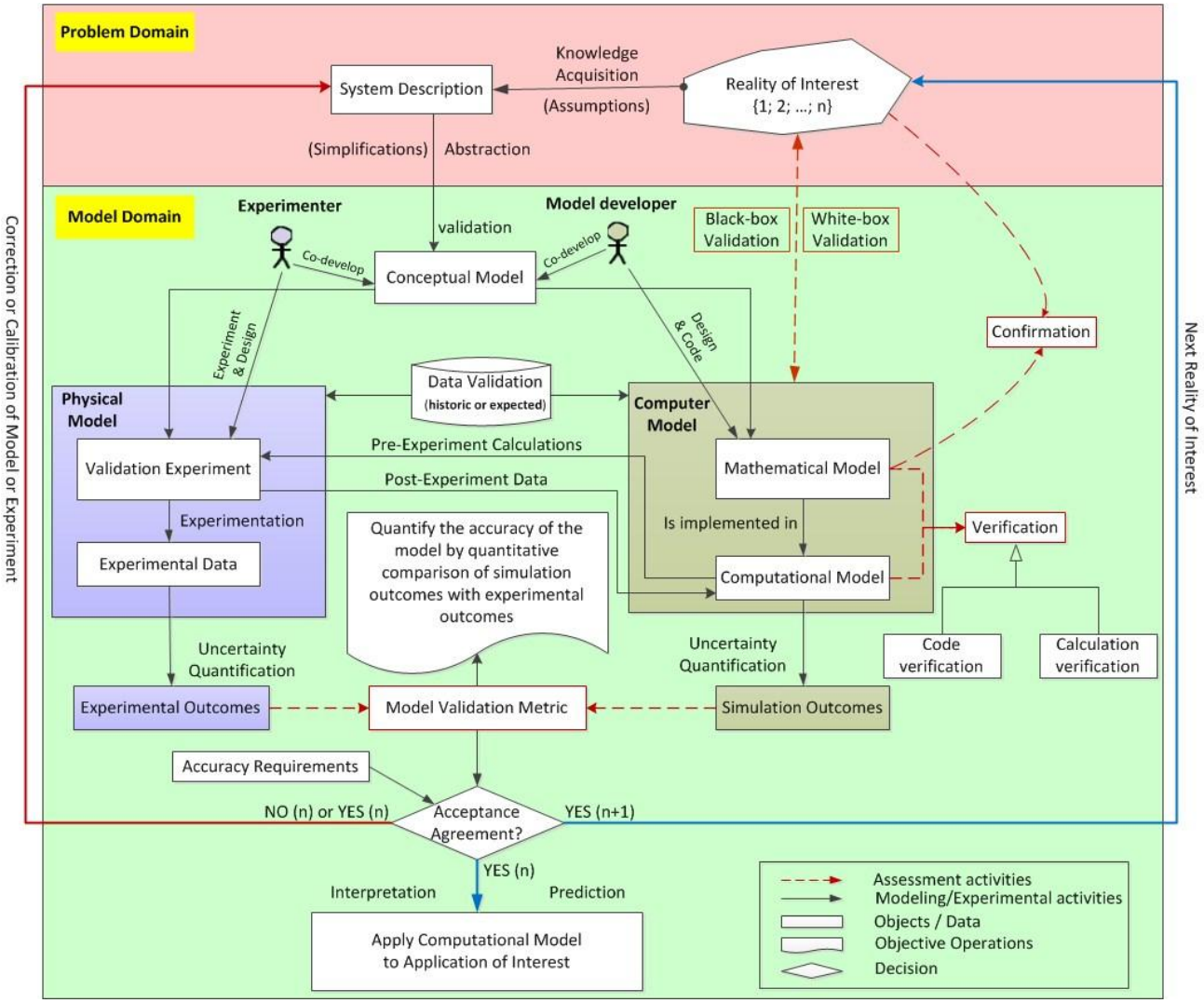


Figure 2: Generic Process of V&V from Unit Problem to Complete System, [1], [2], [3], [8]

This concept will be established in chapter 4 - validation of an alarm sub-system.

In Fig. 2, the *Problem Domain* is located where the problem resides. It is composed of *Reality of Interests* which represent the *Real World*. The Reality of Interests is a set of physical sub-systems from the overall system (ex: the Alarm system in NAIMC software). Each chosen sub-system must be properly described in a *System Description*. The System Description is obtained through *Knowledge Acquisition* and *Assumptions* about the Real World.

The Modeler has to determine what aspects of the Real World to include, or exclude from the *Conceptual Model* to attain an appropriate level of model detail. The decision on the model detail is a join agreement between the *Modeler*, the *Problem Owner*, and the *Stakeholders* who need the model to achieve high accuracy and thus facilitate the decision-making process.

The Conceptual Model belongs to the *Model Domain* (Fig. 2) and describes what part of the System Description to include in the *Computer Model* and more importantly at what level of detail. The Conceptual Model is obtained through *Abstraction* and *Simplifications* of the System Description.
It must be kept in mind that the Real World is not fully knowable. Consequently, the System Description will only be a limited representation of the Real World. Even though all information were available, still assumptions and simplifications are necessary. Some of the main reasons are:

- The Real World has not been observed in all possible states.
- Observations are subject to observer perceptions and viewpoints.
- The Real World has been observed in all possible states, but still abstraction is needed to construct the model.

As a corollary, the presence of uncertainties in the Real World increases the need for establishing assumptions. Both the Problem Owner and the Modeler work closely to establish all necessary assumptions.
It is a good practice to document all the assumptions made. This effort will ease the model calibration / correction activities at a later phase. For example critical assumptions can be assessed at a later development phase by performing sensitivity analysis (SA).

Practically, it is not desirable to model everything that is known about the real world. Sometimes even some which are relevant to the problem situation. Thus, the modeling objectives can be reduced. This is what is called model abstraction. The main objective to perform model abstraction is to achieve model simplifications, for the following reasons:

- Simple models can be developed faster.
- Simple models are more flexible.
- Simple models required less data and resources.
- Simple models run faster.

- Simple models generate results that are easier to understand and have greater chances of acceptance by the client.

As a corollary, the Conceptual Model becomes a partial representation of the System Description. Both the Problem Owner and the Modeler work closely to establish the simplifications.
Briefly, there are two distinct domains in Conceptual Modeling.

a. The Problem Domain, needed to acquire knowledge about the Real World and to derive a System Description. This process requires making assumptions.

b. The Model Domain, needed to abstract the Conceptual Model from the System Description. This process requires making simplifications.

Inside the Model Domain, are two different types of sub-domains: the Computer Model controlled by the Modeler and the Physical Domain controlled by the Experimenter. Both the Modeler and the Experimenter co-develop the Conceptual Model and collaborate closely in the V&V process.

The goal of the Modeler is to generate simulation outcomes based on input to the Computer Model while the Experimenter's goal is to generate experimental outcomes based on input to the Physical Model. The experimental outcomes and the simulation outcomes are then further processed in the *Model Validation Metric*. While these activities might give a deeper knowledge of the system behavior under different operating conditions, some limitations can be observed:

- Misinterpretation of results: Interpretation of simulation results is completely under the responsibility of the User and requires great care. A simulation test might fails, due to wrong input value or a defect in the model itself. Conversely, a simulation test might indicate a valid system when in fact the system may contain a defect that is not revealed by a particular test run.

- Validation Difficulties: Models used for test simulation reflect the level of knowledge of the system under test. Sometimes, aspects of the modeling process are not known precisely and therefore may be decided upon a subjective approach.

- Capturing details of Reality: Model simulations always represent a subset of reality and therefore may obscure some significant problems.

- Human Element: A nontrivial level of knowledge is required for a thorough understanding and correct interpretation of simulation results.

On the right side of Fig. 2, the Computer Model representing the implementation of the Mathematical Model must ensure that the associated Conceptual Model is sufficiently accurate. This assurance is reinforced through verification, confirmation and local validation activities.
In the verification process, *Code Verification* is performed for the removal of errors in the code while *Calculation Verification* is used to quantify the errors introduced during the application of

the code. It is also necessary to check (through *white-box validation*) whether each internal logical part of the Computer Model represents the Reality of Interest with adequate accuracy (correctness). The functional behavior of the complete and overall Computer Model must also be checked (through *black-box validation*) to ensure that the model represents the real world system with adequate accuracy.

On the left side of Fig. 2, there is a Physical Model comprising the *Validation Experiment* (to capture data needed for the model) and the *Experimental Data*.
The Modeler will perform *Pre-Experiment Calculations* to inject the results in the Validation Experiment. In return, the Experimenter will perform *Post-Experiment Data* and introduce the important quantities needed to perform computational simulations.
Although the Modeler and the Experimenter closely work together, they must avoid having knowledge on the each other's simulation or experimental outcomes beforehand.

The relevant data extracted from the real world system to build the model and experimentation must be reliable and sufficiently adequate. The *Data Validation* must be a continuous process as new information might still be available at a later phase. These data are collected through experimentation and produce Experimental Data. Various techniques are available for data collection and among those can be cited *historical methods*, *observational methods*, and *controlled methods*.
Generally for complex systems, no experimentation methods or simulation methods will perfectly match the real world in the Conceptual Model. In other words, uncertainties are always a reality in the simulation results.
The accuracy issue is limited by two main error sources: the *Modeling Error* and the *Processing Error*. A pictorial summary can be given in Fig. 3.

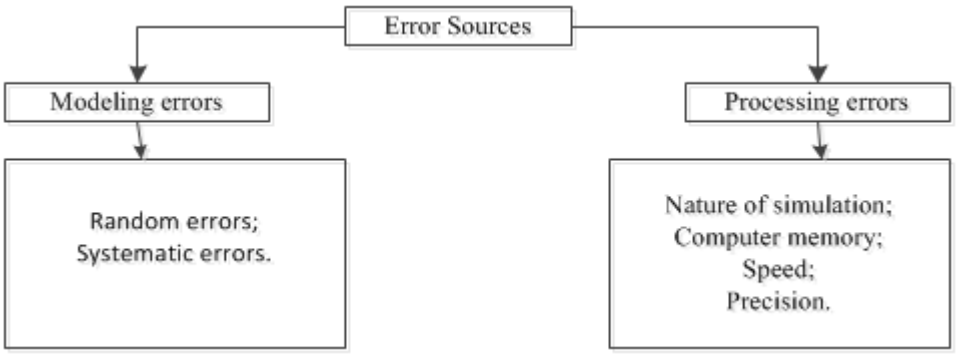


Figure 3: Error Sources in Simulation, [4]

For the Modeler, this might be directly due to processing errors during computations. For the Experimenter, this might be due to modeling errors - systematic error (reducible uncertainties i.e. calibration, test techniques, experience of the experimenter) or random error (irreducible uncertainties i.e. environment, temperature).

Any uncertainty in the simulation outcomes will forcibly originated from the uncertainty in the experimental activities because of the exchange of information between the Modeler and the Experimenter (pre-experimental calculations and post-experimental data).
These uncertainties are quantified both in experimental outcomes and simulation outcomes. The main objective of *Uncertainty Quantification* operations is to reduce the uncertainties in the model so as to increase the model accuracy or the confidence level. The reduction of uncertainty in the model is performed inside the Model Validation Metric where comparison of the two types of result outcomes is being made.

After the Model Validation Metric result is obtained, the next step is to assess the result against the *Accuracy Requirements* which had been predefined for the intended use of the model. The assessment result is used to tell whether or not the model has result in *Acceptance Agreement* with the experiment. If the Acceptance Agreement has not been reached, the experiment or model can be revised or corrected accordingly to achieve adequacy. If the Acceptance Agreement has been reached, the experiment or model can still be update or calibrate with the latest available information for optimization.

Whenever there is Acceptance Agreement, the model can be used as intended, and the next Reality of Interest can be process in the loop.
A common misperception in adopting this V&V concept is to state that if all the sub-models in a model library (ex: an Alarm sub-system in an overall NAIMC software) have passed a certain validation test, then it can be surely stated that the host simulation package is also validated. That is not true.

The host simulation package might remain validated only on an individual model basis. It means that if we string together some number of blocks to form a higher level of system, the simulation of this system will not automatically be validated. This can be due to the accumulation of the lack of relevant information about each subsystem, [4].

Therefore, in order to validate a simulation it is advisable to connect several subsystems whose behavior is known a priori and compare this behavior to the simulation outputs. If the comparisons consistently meet a specified accuracy criterion, then we can feel confident that the simulation package is system-validated to a certain level of system complexity.

Chapter 3

Software Quality Assurance

3.1 Aspect of Quality Assurance

Software quality is subjectively defined as “The degree to which the user perceives that the software meets his / her expectations” and objectively defined as “The degree to which the attributes of the software enable it to perform its specified end product use”

Software quality assurance is "*the assembly of all planned and systematic actions necessary to provide adequate confidence that a product, process, or service will satisfy a given quality requirements*." (McManus and Schulmeyer, 1987)
If software quality satisfaction is the goal, then assurance is the process method involved in the achievement of these goals, as illustrated in Fig. 4.

Figure 4: Generic Quality Process, [5]

The central activities for quality assurance (QA) can be viewed as to ensure that all critical defects causing high disruptions or damages are removed in the software system at the time of release. It executes the planned QA activities and handles discovered defects.
According to Pfleeger et al., 2002 five different view can be identified in software quality:

Transcendental view: here the quality is hard to define, but can it be recognized if it is present.

User view: here the quality is obtained if the product meets user needs and expectations. Quality factors apply.

Manufacturing view: here the quality means the product has followed conformance to standards and procedures.

Product view: here the quality if based on the hypothesis that if a product is manufactured with good internal properties, then it will have good external qualities.

Value-based view: here the quality is the customers' willingness to pay for the software. Quality is pointless if a product does not make economic sense.
Some of these quality views will be revive in Chapter 4 when discussing the quality of an alarm sub-system.
A quality assurance aspect can be view as a string of several activities which can run simultaneously, as shown in Fig. 5.

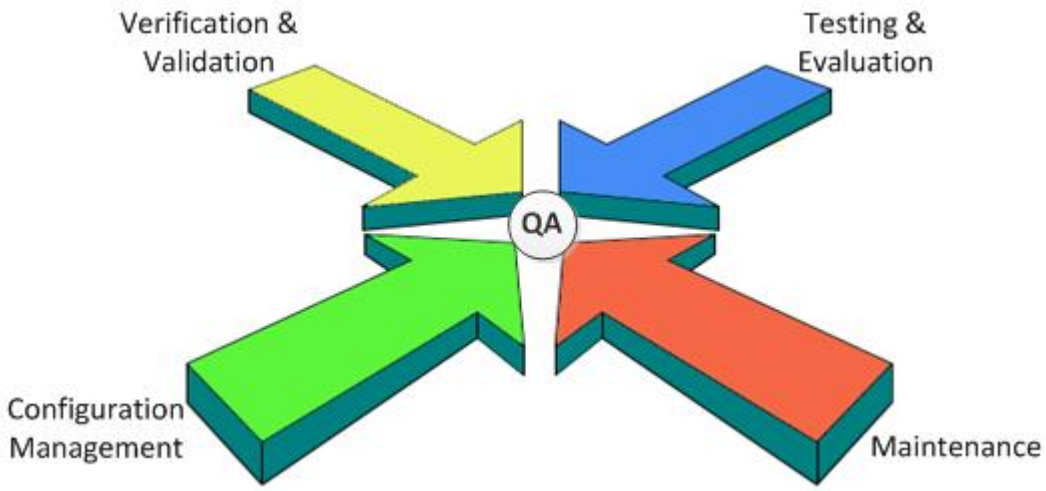


Figure 5: Quality Assurance Aspect, [6]

3.2 Software Quality Factors and Quality Criteria

A quality factor represents a behavioral characteristic of a system. A quality criterion is an attribute of a quality factor. A quality factor is justified by its set of quality criteria. Stated differently, before a product gains one quality factor, it must match its corresponding quality criteria. For example, achieving the software **Correctness** will require software ***completeness***, ***traceability*** and ***consistency***. These quality criteria are used as a verifiable measure in the quality metric which is stated in either quantitative or qualitative terms. The quality factors and criteria can be organized in three groups as illustrated in Tab. 1 or Fig. 6.
If an effort is made to improve one quality factor, another quality factor in the system might be degraded. For example, increasing interoperability will normally reduce the efficiency of a software system.
Some quality factors positively impact others. For example, an effort to enhance the correctness of a system will increase its reliability.

Table 1: HIGH-LEVEL SOFTWARE QUALITY FACTORS AND CRITERIA – McCall (1977); ISO 9126

Quality Categories	Quality Factors	User Concern about the software	Quality Criteria
Software Operation	Correctness	How well does it satisfy user's need?	Traceability; Completeness; Consistency.
	Efficiency	How well does it utilize resources?	Execution efficiency; Storage efficiency.
	Integrity	How secure is the system?	Access Control; Access Audit.
	Reliability	How dependable is it in producing expected performance?	Consistency; Accuracy; Error Tolerance; Simplicity.
	Usability	How easy is it to use the system?	Operability; Training; Communicativeness; Simplicity.
Software Revision	Testability	How easy is it to prove the system?	Simplicity; Instrumentation; Self-descriptiveness; Modularity.
	Verifiability	How easy is it to confirm system performance?	Self-descriptiveness; Modularity; Consistency.
	Expandability	How easy is it to expand the capabilities of the system?	Simplicity; Machine Independence; Software-system Independence; Modularity.
	Flexibility	How easy is it to change the software system?	Self-descriptiveness; Expandability; Generality; Modularity.
	Maintainability	Is the software easy to fix?	Simplicity; Consistency; Conciseness; Self-descriptiveness; Modularity.
Software Transition	Interoperability	Is the software flexible and easy interfaced?	Data Commonality; Communications Commonality; Modularity.
	Reusability	How easy it is to apply the software to another application?	Self-descriptiveness; Generality; Modularity; Software-system Independence; Machine Independence.
	Portability	How easy is it to move to another machine?	Modularity; Self-descriptiveness; Software-system Independence; Machine Independence

Quality Criteria **Quality Factors**

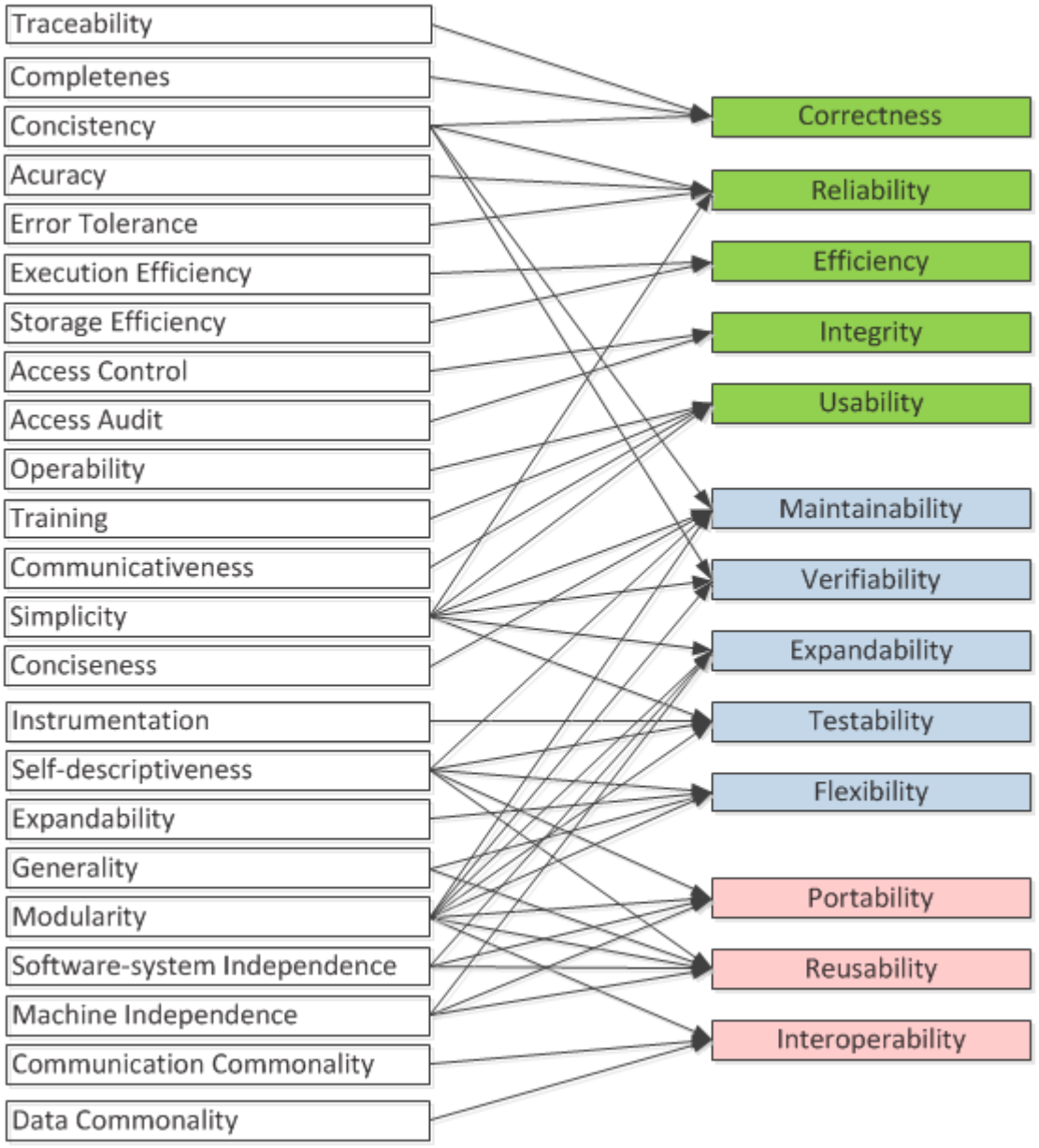


Figure 6: Relationship between Quality Factors and Quality Criteria, [10]

It was previously stated that, for a product to gain a quality factor, it must meet its corresponding quality criteria. But one of the greatest challenges resides in measuring the quality factor.
In the ISO 9126 quality model, the quality characteristics have been broken down into quality sub-characteristics, Fig. 7. This technique allows better comprehension and measurement of quality.

Quality Characteristics	Quality Sub-characteristics

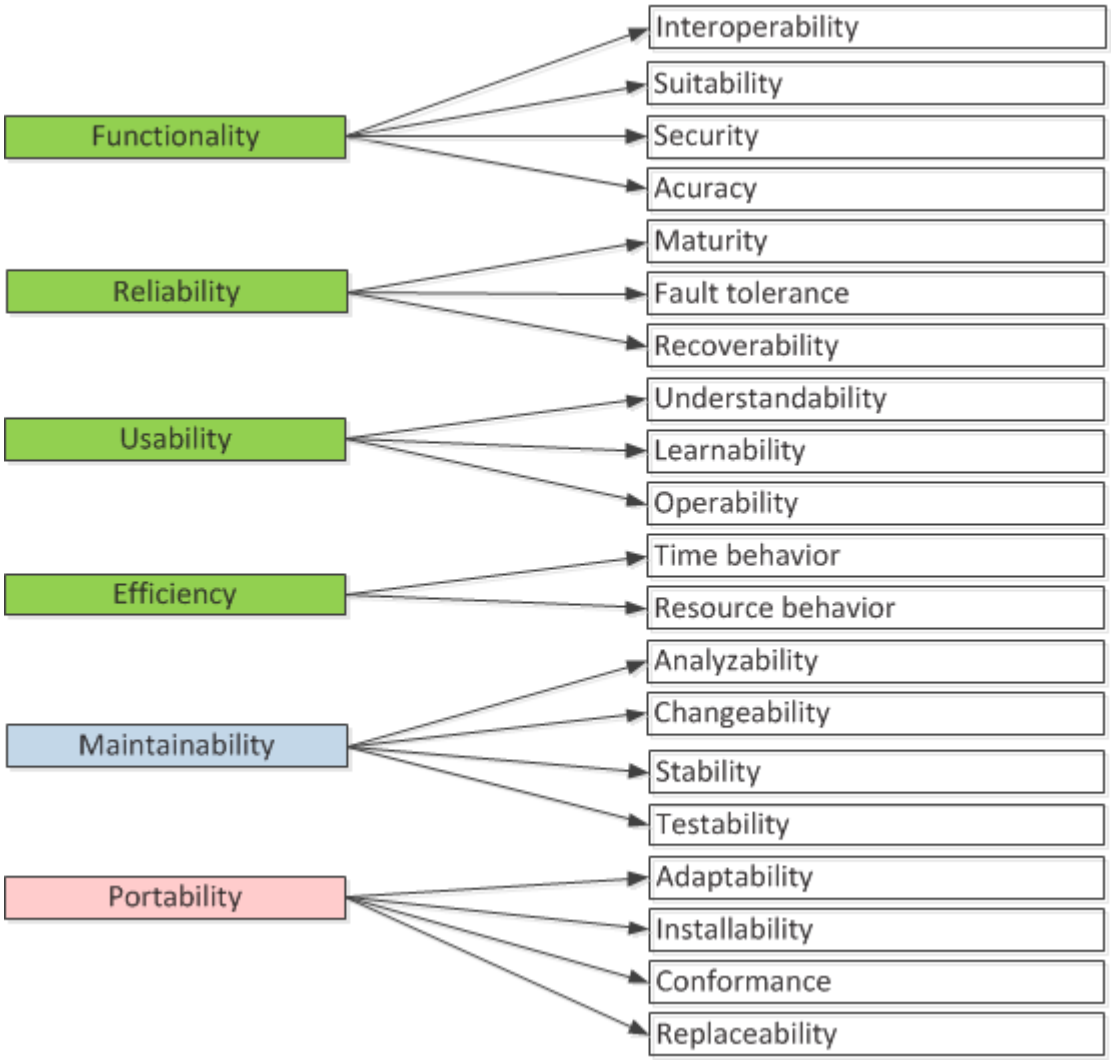


Figure 7: ISO 9126 Sample Quality Model Refines Standard's Features into Sub-characteristics. (From ref. 4. 1996 IEEE), [10]

Chapter 4

Validation: Alarm Sub-system and Traffic Monitoring

4.1 Minimum User Requirements of Alarm in NAIMC - AIS Online.

"The tracking system shall have possibilities for introducing area boundaries, both on more permanent basis and on ad hoc basis. This can be country boarder, Economical Zone boundaries, oil platform security zone etc."

"The tracking system shall be able to create warning when a boundary is crossed. From which direction shall be selectable and programmable".
The warning is automatically send as pure text via email to the authority's email box. The structure of text shall be clear and include vessel's static and dynamic data. In alarm email notification one external link shall be provided to view the corresponding vessel live on map. Another external link shall open the vessel image on click.

4.1.1 Implementation of Alarm and procedures

1. After login into AIS-Server online. Click on ***"Edit Mode Menu"*** and then on ***"Create Filter"***.
It is possible to customize the alarm by creating filter and selecting attributes:

- Create the name of the filter.
- Choose and Attribute.
- Combine constraint with "AND operator" or "OR operator"

Enter the name ***"BoundaryAlarm"*** as filter name and select ***"Polygon"*** in attribute. Select the operation attribute ***"Inside"***, and click on ***"Draw"*** to define the alarm zone.

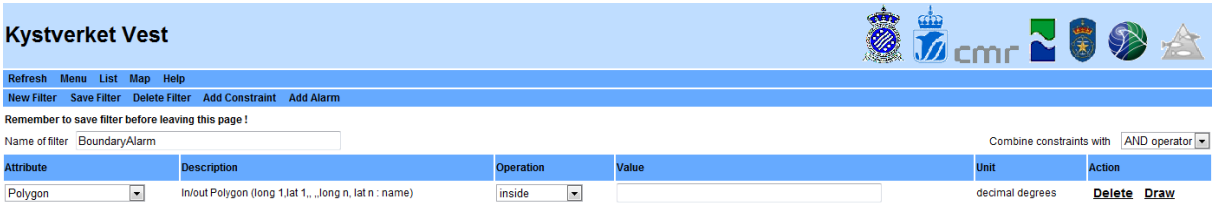


Figure 8: Initiating a Filter.

2. Click ***"Add Points"*** Icon from top menu, draw the polygon by clicking on map and click ***"Done"*** Icon from top menu when finish.

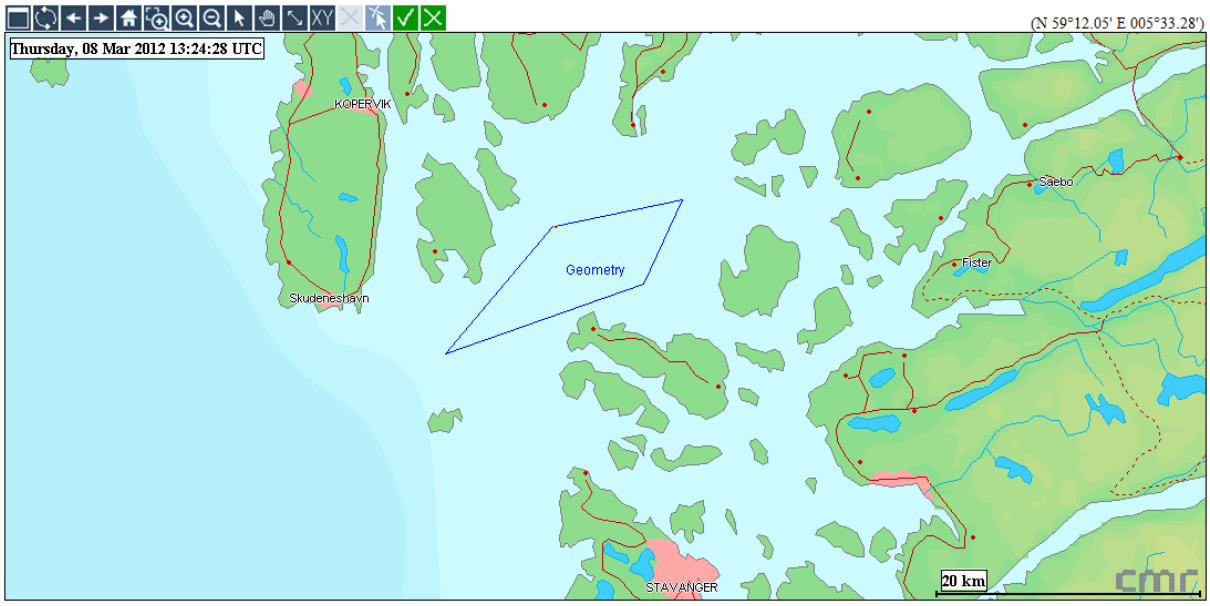


Figure 9: Creating Polygon.

3. The value of the polygon is automatically filled in as shown below. Click on ***"Save Filter"***

5.5517788, 59.200901, 5.4095335, 59.1142426, 5.6720276, 59.1617165, 5.7233534, 59.2189865, 5.5547118, 59.200901
(It is possible to add several new constraints and further filter the alarm).

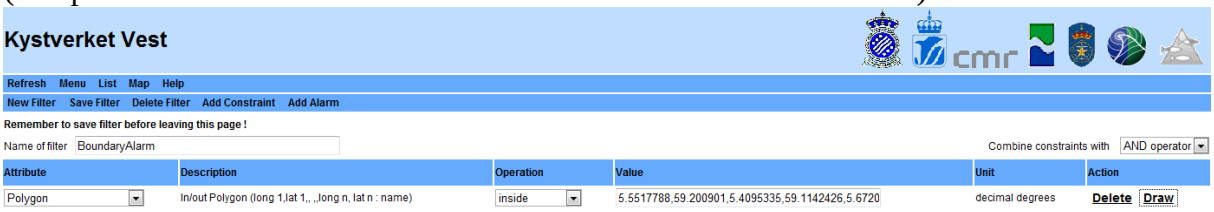


Figure 10: Polygon Input.

4. Now click on ***"Add Alarm"*** and fill in parameter for alarm, afterward click on ***"Save Alarm"***.

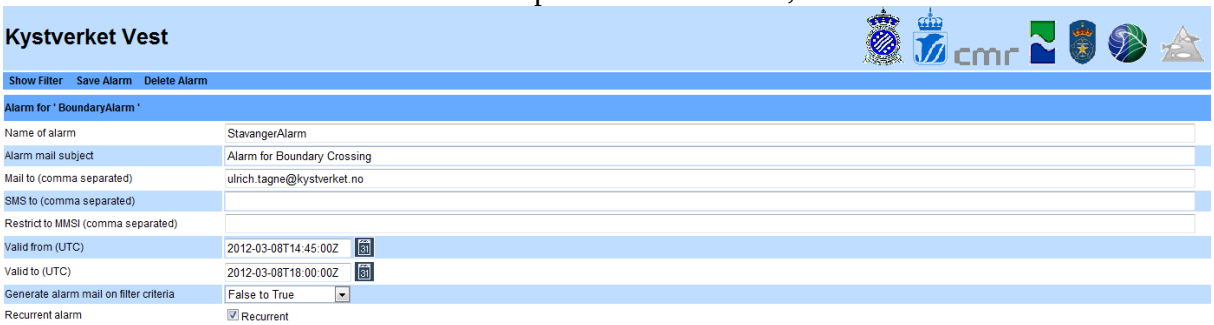


Figure 11: Alarm Parameters

Alarm Interval = [14:45 – 18:00] UTC

4.1.2 Alarm Notification for a Vessel

5. When a vessel has crossed a predefined boundary (step 2), an alarm notification is triggered for the consequent vessel and dispatched to the authority's email box as registered in step 4.

Alarm for Boundary Crossing for MYRE SEADIVER

ulrich.tagne@kystverket.no

Sendt: to 08.03.2012 18:58

Til: Tagne, Ulrich

Vessel Image from Ship Info

Ais Event 'StavangerAlarm' for filter 'BoundaryAlarm' occurred at 2012-03-08T17:56:51 UTC

Vessels that now fulfil filter:

- Vessel name: MYRE SEADIVER
- Vessel type: Cargo ship
- Call sign: E5U2529
- IMO number: 6505662
- SOG: 6.9 kn, COG: 270°
- Time: 2012-03-08T17:56:51 UTC, Longitude: E 005°25.99', Latitude: N 59°07.23'

Departed port: HILLESVAG, NORWAY at: 2012-03-08T15:55:21 UTC

Destination: KALININGRAD, port: KALININGRAD, RUSSIA

Show MYRE SEADIVER on Map

Figure 12: Email Notification.

Shipping Publications AS

Phone/Fax +47 33 18 11 80/ +47 33 18 36 76

Email info@shipping-publ.no

Myre Seadiver

Figure 13: Vessel Image Display.

The structure of the notification (in plain text) is as shown in Fig 12.
Using Tail Map with corresponding date and time interval for the alarm:[14:45 - 18:00]

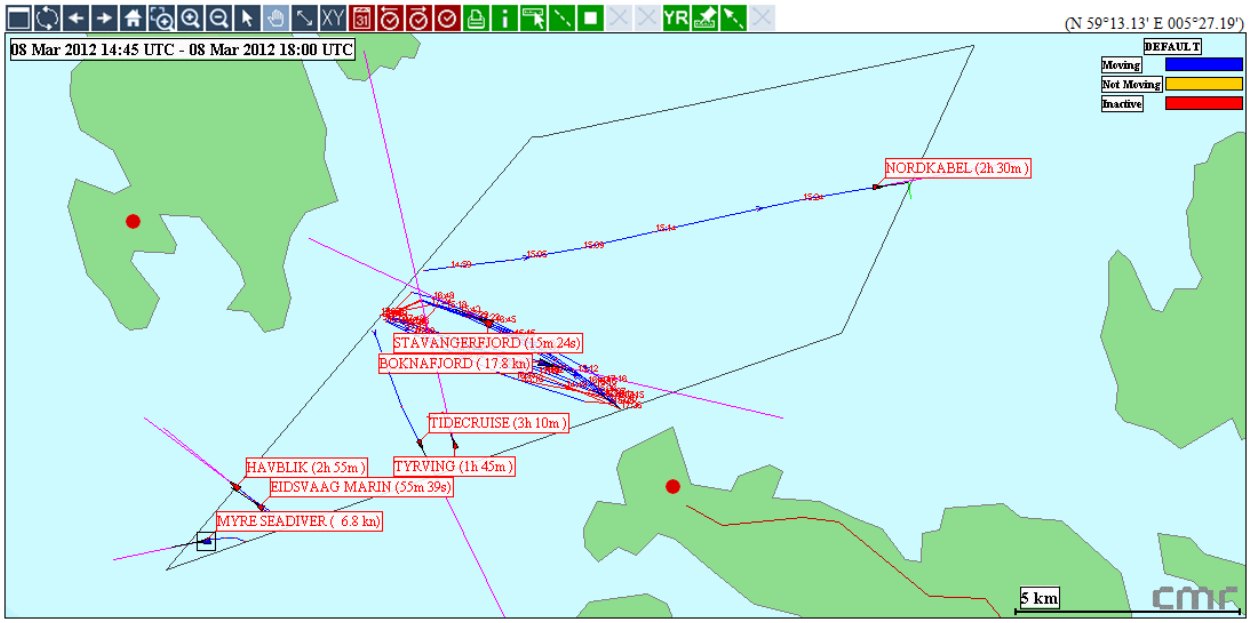


Figure 14: Vessels and Polygon Display.

Statistics: Displaying all vessels with their crossing time inside the BoundaryAlarm domain. For example, the vessel MYRE SEADIVER has crossed once the alarm domain at 18 o'clock.

Kystverket Vest

Menu List Map Export Tails Help

TailStore Targets 'BoundaryAlarm'

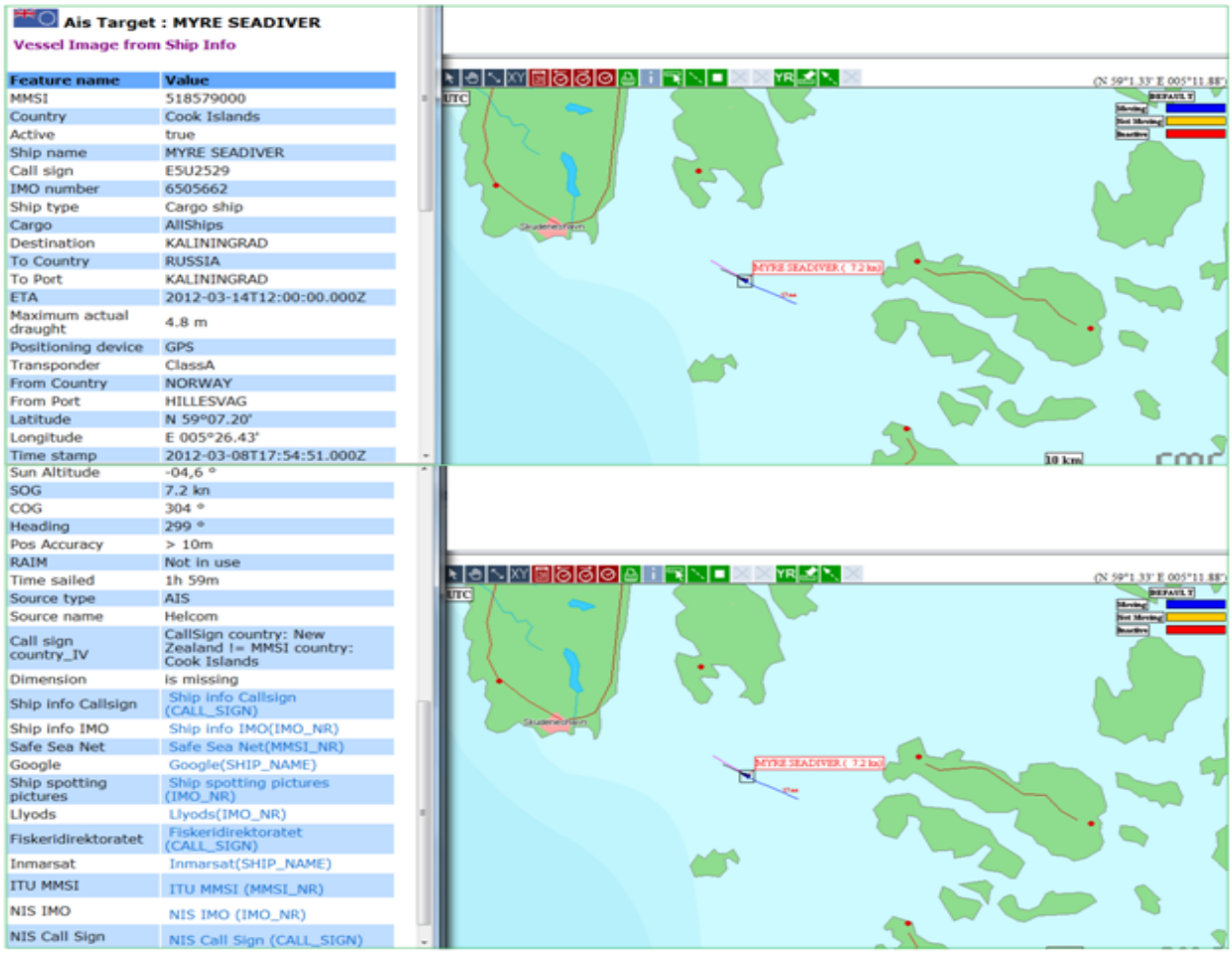
Select attributes for statistics

Name HOUR_OF_DAY

Text	15	16	17	18
HAVBLIK	0	1	0	0
TYRVING	0	0	1	0
STAVANGERFJORD	0	0	0	1
BOKNAFJORD	0	0	0	1
EIDSVAAG MARIN	0	0	0	1
MYRE SEADIVER	0	0	0	1
NORDKABEL	0	1	0	0
TIDECRUISE	1	0	0	0
Total	1	2	1	4

Figure 15: Total Number of Vessels and Crossing Hour Display.

Click on ***"Information icon"*** to display additional vessel data:



Ais Target : MYRE SEADIVER

Vessel Image from Ship Info

Feature name	Value
MMSI	518579000
Country	Cook Islands
Active	true
Ship name	MYRE SEADIVER
Call sign	E5U2529
IMO number	6505662
Ship type	Cargo ship
Cargo	AllShips
Destination	KALININGRAD
To Country	RUSSIA
To Port	KALININGRAD
ETA	2012-03-14T12:00:00.000Z
Maximum actual draught	4.8 m
Positioning device	GPS
Transponder	ClassA
From Country	NORWAY
From Port	HILLESVAG
Latitude	N 59°07.20'
Longitude	E 005°26.43'
Time stamp	2012-03-08T17:54:51.000Z
Sun Altitude	-04,6 °
SOG	7.2 kn
COG	304 °
Heading	299 °
Pos Accuracy	> 10m
RAIM	Not in use
Time sailed	1h 59m
Source type	AIS
Source name	Helcom
Call sign country_IV	CallSign country: New Zealand != MMSI country: Cook Islands
Dimension	is missing
Ship info Callsign	Ship info Callsign (CALL_SIGN)
Ship info IMO	Ship info IMO(IMO_NR)
Safe Sea Net	Safe Sea Net(MMSI_NR)
Google	Google(SHIP_NAME)
Ship spotting pictures	Ship spotting pictures (IMO_NR)
Lloyds	Lloyds(IMO_NR)
Fiskeridirektoratet	Fiskeridirektoratet (CALL_SIGN)
Inmarsat	Inmarsat(SHIP_NAME)
ITU MMSI	ITU MMSI (MMSI_NR)
NIS IMO	NIS IMO (IMO_NR)
NIS Call Sign	NIS Call Sign (CALL_SIGN)

Figure 16: Popup Information on Vessel Click.

Trajectory of Vessel: **MYRE SEADIVER**

Destination: Kaliningrad. It is possible to generate animation (clicking on button ***"Generate Animation"*** from the left menu) of the ship and track its trajectory according to the defined time interval:

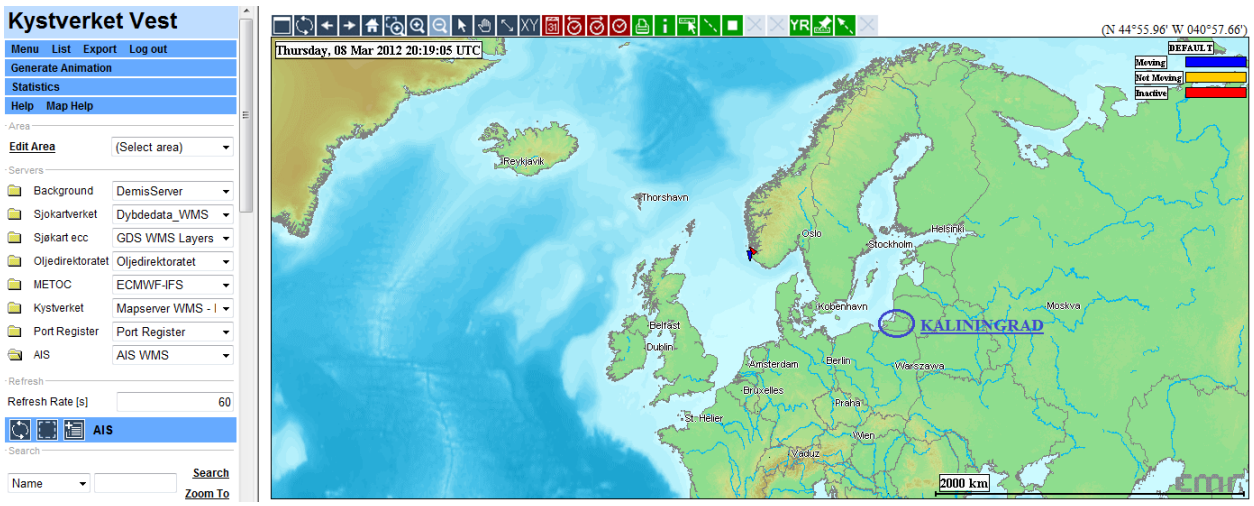


Figure 17: Trajectory Animation of Vessels.

4.1.3 Assessing the Quality of Alarm System

After testing the Alarm sub-system in NAIMC AIS online, the quality of the alarm can be empirically (based on observation and experience) analyzed.

Functionality:

Suitability: The alarm system provides an adequate set of functions that fulfill the minimum user requirements. This includes plotting, tracking, monitoring, and additional information display on screen, and email alarm notifications. These functions are suitable for purposes.

Accuracy: The alarm system is reliable and delivers exactly the needed information as predefined by the user. Simplicity and Consistency is observed when using the system.

Interoperability: The alarm system is a sub-system of the whole NAIMC software system. It is capable of interacting with other internal sub-systems without causing defects or disrupts on its own or on other systems. The alarm therefore runs in modularity.

Security: The alarm system fulfills the integrity requirement. All unauthorized access to the system is avoided and controlled by means of username and password.

Reliability:

Maturity: The alarm system is to some extent dependable to other sub-systems; however it is capable of handling failure caused by errors elsewhere in other system.

Fault Tolerance: The alarm system is highly reliable as it does not deliver any wrong information in case of fault intrusion to its interface. Some logical input error parameters are ignored due to their low impact, otherwise the error is processed and an error message is thrown on the screen.

Recoverability: The alarm system is capable of reinstate itself to its normal functioning condition. Intrusions of non critical faults do not cause system crash or deadlock. If the email server goes down, the email alarm notification will not be cancelled. However critical failure of other systems in which the alarm system heavily depends on may cause disruptions.

Usability:

Understandability: The alarm system is easy to use, and it is easy to understand the results. Textual information and pictorial images are logically arranged and presented on the screen. Chains of operations are simple and consistent.

Learnability: The alarm system can be easily learned quickly by a novice user. The input / output parameters are logical. Colors, legends and other tools are interfaced for better comprehension and most of them are self-explanatory.

Operability: The alarm system has a well organized user interface. The user can control its tasks by selecting, adding, deleting, and updating information input. However, some weakness has been observed – In a consecutive set of operations a single error occurrence (which is being thrown out) will force the user to restart the whole operation, and thus flushing all corrected input in the fields. Another drawback is the mandatory need to work with two or more open browser to achieve one task. For example when defining the ***"Tails"*** function in a browser for plotting operations, another browser must be opened for the plotting according to the tails settings.

Efficiency:

Time Behavior: The alarm system has a processing time which meets user acceptance. The time delay for email notification is less than two minutes provided that the internet network traffic is not overloaded.

Resource Utilization: The alarm system required limited additional resources as it quickly solves the user tasks.

Maintainability:

Analyzability: The alarm system can be easily analyzed. Errors that might cause illogical results are not process. These types of errors are carefully managed and thrown out. Errors message often lead the user to fully or partially diagnose the problem.

Changeability: The alarm system has several user-selectable input parameters. Such as Constraints, Operators, Date, Ship's parameters, etc as seen in many figures above. The user can handle the alarm system by creating new ones or by adjusting those which already exist.

Stability: The alarm system is stable in the manner that it does not keep silence on unrecognized input parameters. Non critical errors are handled accordingly and without system crash.

Testability: The alarm system is easy to test after production. It meets simplicity, communicativeness and self-descriptiveness. The system is verifiable and modifications (corrections or improvements) can be carried out where needed without difficulties.

Portability:

Adaptability: The alarm system can be modified for different specified environments. The effort required to transfer the system from one hardware / software machine to another is not a concern.

Installability: The alarm system is machine-independent because of it modularity, portability and completeness. It is well expandable.

Replaceability: The alarm system is highly modular. New improved versions can replace old one in the overall NAIMC software system, thus increasing its maintainability.

Conformance: The alarm system meets the functional correctness, completeness and integrity as a sub-system. Meanwhile, the process used to develop this system must meet an acceptable standard. Conformance corresponds to the manufacturing view of quality. Non-conformance indicates the presence of faults or errors. For example, a wrong algorithm or an inappropriate data structure is used; some coding standard is violated, etc.
Some modules of the alarm sub-systems might follow a V-model type (Fig. 18) of verification and validation. Especially, it can be the case if the modules are not directly associated with physical and environmental phenomenon.

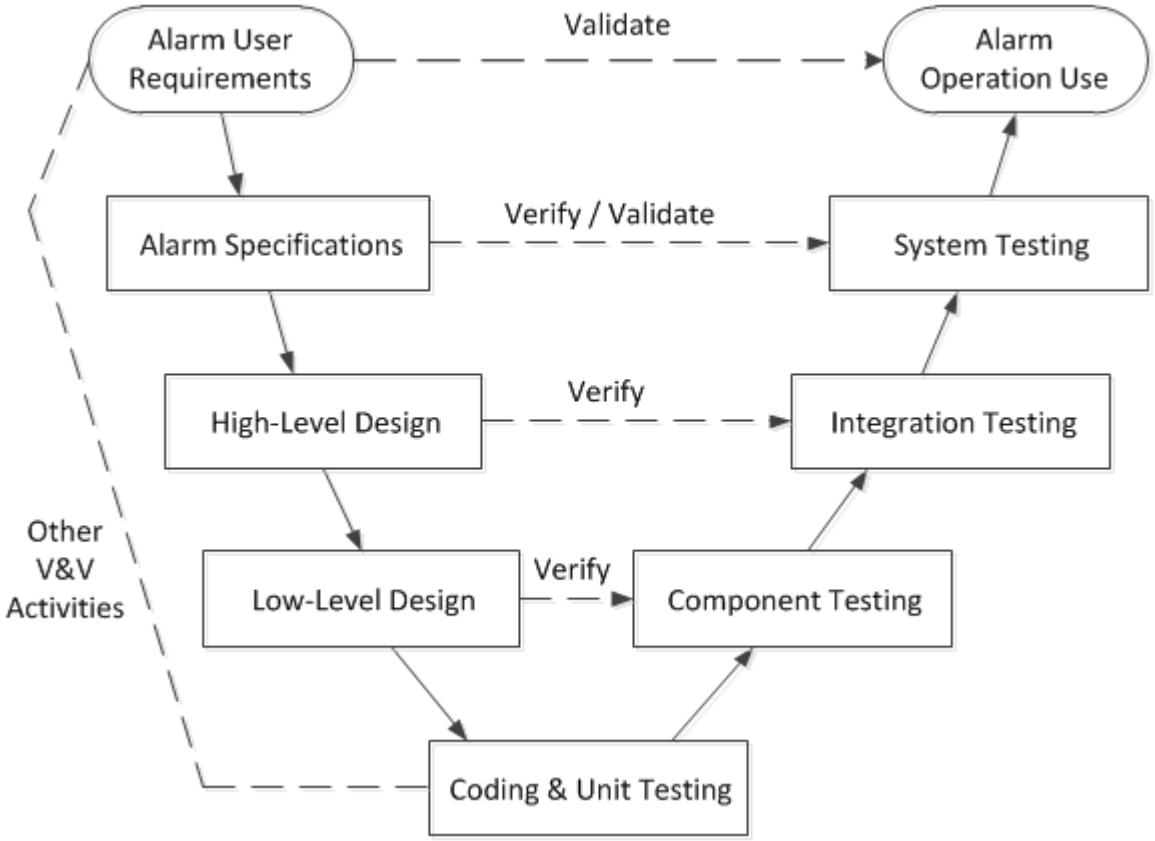


Figure 18: Verification and Validation of Modules in Alarm Sub-system with V-Model, [5]

4.2 Monitoring and Harmonization of Maritime Traffic

With the aim to enforce the maritime shipping safety and security, Search and Rescue, and the marine environmental protection, the exchange of information between national and regional Member States is essential.
The design of such system capability can be generically represent in Fig. 19.
The figure shows communication flows where national AIS servers are interacting with the BLAST regional servers via national proxies.

Considering the Alarm sub-system as described in paragraph 4.1, the design platform is deemed to be beneficial for all Member States in the North Sea Region (Fig. 20).

The regional maritime traffic information is also harmonized with the SafeSeaNet (SSN) system within which (SSN WEB/XML interface) the alert notifications can also be configured with the following content of messages:

Port notification: Used to notify SSN that a given vessel is bound for a particular port with an estimated time of arrival and with a number of persons aboard.

Ship notification: Used to notify SSN about a ship's position. A ship notification is based on AIS information.

Hazmat notification: Used to notify SSN that a given vessel carries dangerous goods and that the Member States owns some detailed information about these dangerous goods.

Alert notification: Used to notify SSN that the sender holds some information about specific incidents like marine SITREP (Pollution **Situation Report** Form), and marine POLREP (**Pollution Report**), Waste, lost/found containers. An alert can be linked or not to a particular vessel.

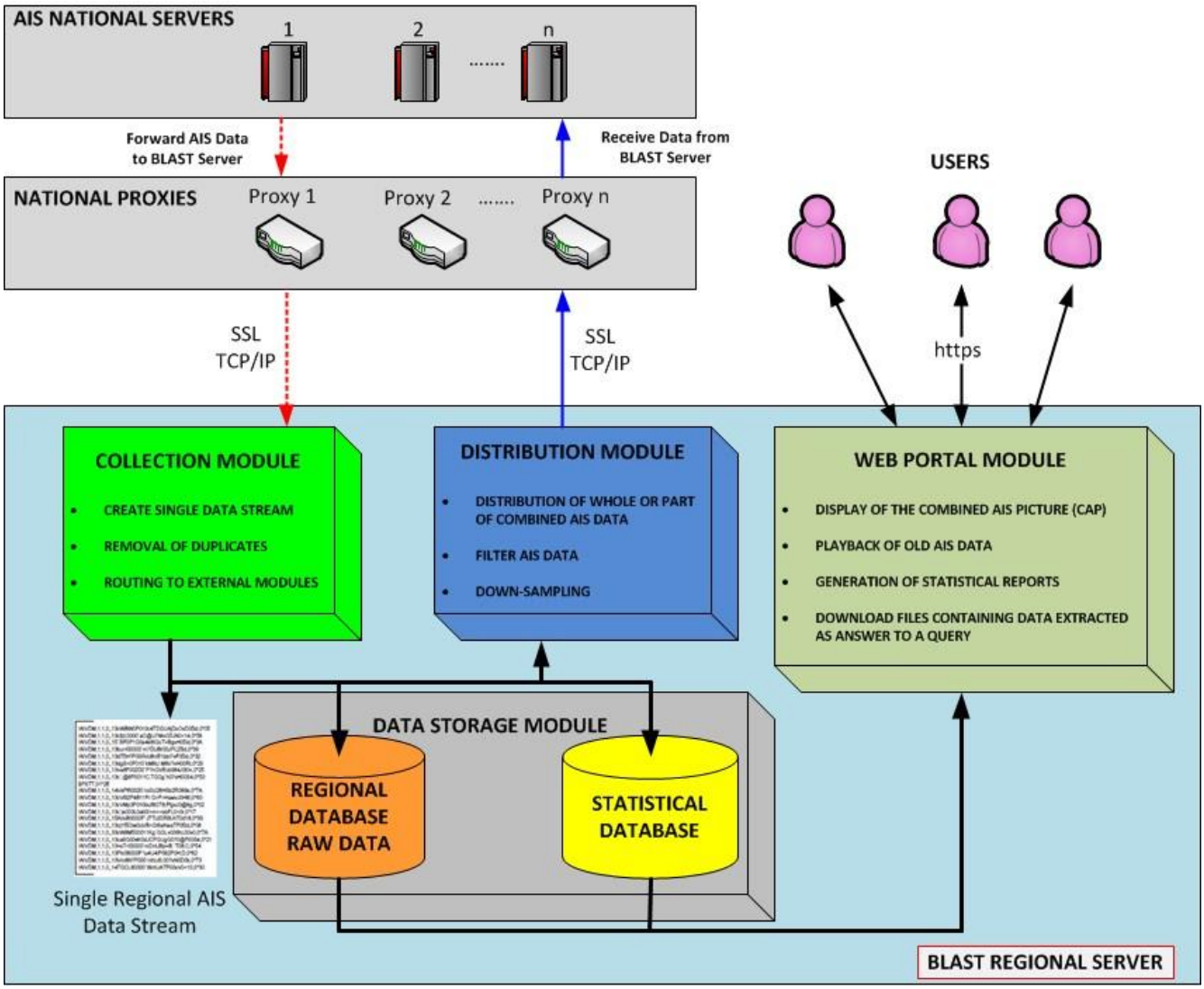


Figure 19: National AIS Server Information Exchange with Regional BLAST Server, [12].

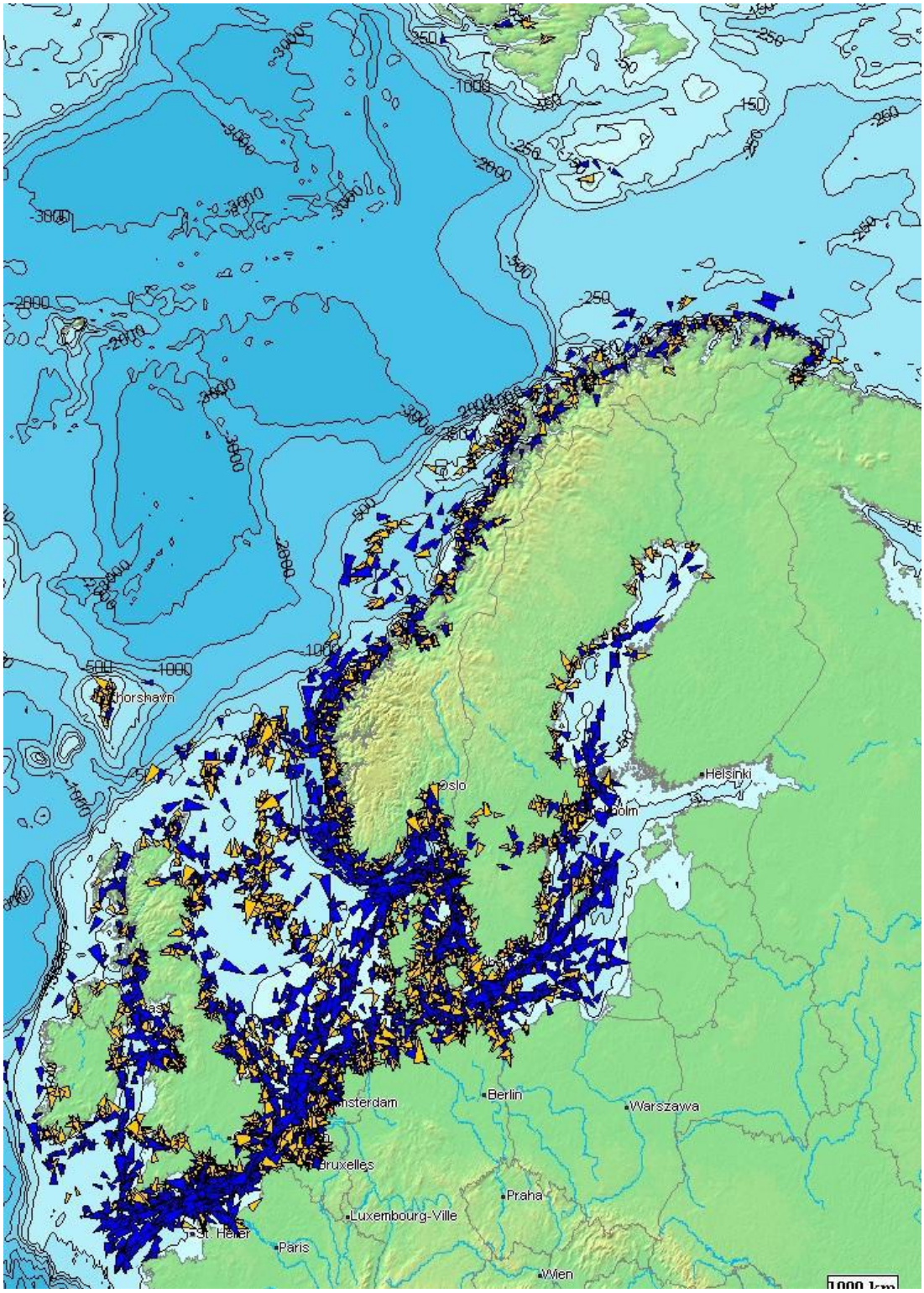


Figure 20: Member State AIS Coverage

Chapter 5

Concluding Remarks

5.1 Conclusions

Conducting software V&V for complex systems is not a trivial task. It required great dedication in performing all the mandatory steps needed to validate the NAIMC software system and demonstrate its quality assurance and user satisfaction.

Unfortunately, it is nearly impossible to measure the quality of a software system without having a well predefined system requirement. Usually, for complex and critical software this document cannot be overlooked as it holds all internal and external behavior of system components.
As a case study, a validation on an alarm sub-system was performed with a minimum predefined user requirements. Some quality in the Alarm system might be present but just hard to define or measure, but it was reasonably possible to identified most of the quality factors and quality criteria after running the Alarm function.

With the assumption that the Alarm sub-system is validated, one can string the Alarm function with other sub-system, incrementally to for a higher level of system. Each new level of system must be checked for validation until the overall NAIMC system achieves validity.

There are valuable benefits in being rigorous with conformance while performing V&V, Quality Assurance activities in complex software. One of the main advantages is to achieve software certification and accreditation. Software certification and accreditation will automatically reduce the time, cost and burden to redo the whole validation process after performing small updates.

The quality system comprising the national AIS Server information exchange with regional BLAST Server will be beneficial for the Member States. The system will achieve a high level of interoperability, meaning modularity, data and communication commonality, in addition to other qualities as elaborated in Fig. 6; all converging to the harmonization and effectiveness of maritime traffic information flow.

References

[1] Stewart Robinson: Simulation Model Verification and Validation – Increasing the Users' Confidence.

[2] Kathy Kotiadis Stewart Robinson: Conceptual Modeling. Knowledge Acquisition and Model Abstraction

[3] William L. Oberkampf, Matthew F. Barone: Measures of agreement between computation and experiment: Validation metrics

[4] Michel C. Jeruchim, Philip Balaban, K. Sam Shanmugan: Simulation of Communication Systems, Modeling, Methodology, and Techniques

[5] Jeff Tian: Software Quality Engineering Testing, Quality Assurance, and Quantifiable Improvement, 2005

[6] Charles K. Njendu: Quality Assurance Framework for Distributed Collaborative Software Development, 1999

[7] Avner Engel:: Vrification, Validation, and Testing of Engineered Systems, 2010

[8] Ben H. Thacker, Scott W. Doebling, Francois M. Hemez, Mark C. Anderson, Jason E. Pepin, Edward A. Rodriguez: Concepts of Model Verification and Validation.

[9] Dr. William L. Oberkampf: Verification and Validation in Computational Simulation

[10] Kshirasagar Naik, Priyadarshi Tripathy: Software Testing and Quality Assurance – Theory and Practice, John Wiley, 2008

[11] Michel C. Jeruchim; Philip Balaban; K. Sam Shanmugan: Simulation of Communication Systems Modeling, Methodology, and Techniques; Second Edition, 2002

[12] Specification BLAST Regional Maritime Traffic Monitoring & Message Reference Guide, BLAST WP5.

Norwegian Hydrographic Service • Aalborg University, Denmark • Agency for Maritime and Coastal Services, Belgium • Danish Coastal Authority • Federal Maritime & Hydrographic Agency, Germany • Hjørring Municipality, Denmark • Jeppesen GmbH, Germany • Local Government, Denmark • Mälardalen University, Sweden • National Space Institute, Denmark • National Survey and Cadastre, Denmark • Natural Environment Research Council, United Kingdom • Norwegian Coastal Administration • Seazone Solutions Limited, United Kingdom •T-Kartor AB, Sweden • TU Delft, the Netherlands • UK Hydrographic Office