



MANATEE

IST-2001-38091

*Maritime Advanced Network for Anticipating Information
Technology Needs for e-work Environment in Safety at Sea*

Deliverable D2.2

Proposal for standardization or acceptance of a defined MSML (maritime safety markup language), an XML derivative adapted for the purposes of maritime applications Specifications of MSML

Report Version: 1.2

Report Preparation Date: 05-11-2004

Classification: Public

Contract Start Date: 01-12-2002

Duration: 24 Months

Project Co-ordinator: METTLE

Partners:	METTLE	France
	SP	Sweden
	BMT	UK
	AIM	Portugal
	MARINTEK	Norway
	EIS	Italy
	MCB	Italy
	HSB	Germany



**Project funded by the European Community
under the "Information Society Technology"
Programme (1998-2002)**

DELIVERABLE SUMMARY SHEET

Deliverable N°:	D2.2
Due date:	30/11/2003
Delivery Date:	27/10/2003 (v. 1.0); 05/11/2004 (v. 1.2)
Classification:	Public

Short Description

This report contains the specification of MSML (Maritime Safety Markup Language) expressed in plain English. From this specification a direct mapping can be made to an XML Schema representation. This description also forms the basis for an ISO standard proposal.

Authors

Name	Company
Lars Strandén	SP
Tore Flobakk	Marintek
Brice Carnicer	Mettle
Marilyne Guevel	Mettle
Isabelle Jamin	Mettle
Håkan Torstensson	SP/HB

Internal Reviewing/Approval of report

Name	Company	Approval	Date
Marielle Labrosse	METTLE	Approved	21/01/2004

Document History

Revision	Date	Company	Initials	Revised pages	Short description of changes
1.	27/10/03	SP	LS		First version
2. Version 1.1	05/04/04	SP	LS	82 - 84	Minor misprints
	05/04/04	SP	LS	84	Added information in Appendix B: Extensions
	05/04/04	SP	LS	Several	Wrong "Structure levels above" in element descriptions
3. Version 1.2	05/11/04	SP	HT	4, App C	The restructuring to a standard proposal added and described
	05/11/04	SP	HT	All	The status changed from confidential to public, new date of issue

DISCLAIMER

Use of any knowledge, information or data contained in this document shall be at the user's sole risk. The members of the MANATEE Consortium accept no liability or responsibility, in negligence or otherwise, for any loss, damage or expense whatsoever incurred by any person as a result of the use, in any manner or form, of any knowledge, information or data contained in this document, or due to any inaccuracy, omission or error therein contained.

The European Commission shall not in any way be liable or responsible for the use of any such knowledge, information or data, or the consequences thereof.

Executive summary

The deliverable D2.2 within the MANATEE project specifies the XML application MSML (Maritime Safety Markup Language). MSML is a language for structuring information and the goal is to create an open standard that can be used generally in the maritime sector. MSML is implemented using XML Schema and is contained in a separate document that could be used at validation. The first purpose of MSML is to make it possible to record safety related information in relation to repair and maintenance. The second purpose is to define an extensible structure that could be developed in future versions of MSML. Note that MSML should not primarily be seen as a support for normal work onboard. Instead it is an add-on support for transfers of safety related information to/from the vessel.

MSML enables security handling and since MSML concerns safety aspects there is information support for:

- preventing accidents
- minimizing extent of damage
- minimizing criticality of consequences

These aspects concern both vessel and bases ashore (denoted shore bases in D2.2).

MSML consists of the following constituents:

- *data model* – that defines the data of interest. The basic parts of the data model are vessel static and dynamic areas, shore base static and dynamic areas and vessel shore base relation. Each of these can be created successively and thus validation can be made even if information is not complete.
- *administrative support* – that defines the handling of the XML application instance as a file.
- *security support* – that defines the handling of data security. MSML enables digital signatures and encryption via the W3C recommendations “XML Encryption Syntax and Processing” and “XML-Signature Syntax and Processing”.

The *data model* can be seen from different *perspectives* and the following are defined:

- *inspection* – that contains information related to externally made inspections.
- *repair and maintenance* – that contains the corresponding information.

A fundamental property of MSML is that it does not consider the actual use of data e.g. there is no specification of MSML messages. This makes it practical to use MSML in a large variety of applications and without modifying the definition of MSML.

The deliverable D2.2 contains nearly the same information as the XML Schema representation but expressed in plain English. In this way it is possible to discuss and evaluate MSML without knowing the syntax details of XML Schema. D2.2 also contains rules and guidelines associated with MSML. Deliverable D2.2 has also been converted to an ISO standard proposal (i.e. presently a PAS, Publicly Available Specification) within ISO TC 8/SC 10.

Contents

1	Introduction (non-normative)	8
2	Background (non-normative)	10
3	Definitions and terminology (non-normative)	12
4	Scope (non-normative)	14
4.1	Included	14
4.2	Limitations	15
4.3	Not included	15
4.4	Summary	15
5	MSML design (non-normative)	17
5.1	Introduction	17
5.2	Data model	18
5.3	Perspectives	19
5.4	Administrative support	20
5.5	Security support	20
5.6	Successive information build-up	21
5.7	MSML vs. standardised maritime terminology	21
5.8	MSML processor	21
6	MSML Specification (normative)	23
6.1	Introduction	23
6.2	Referencing	23
6.3	Security support	24
6.4	Relation to XML Schema	25
6.5	Special considerations	25
6.6	simpleType: units_type	26
6.7	simpleType: element_identity_algorithm_type	27
6.8	simpleType: vessel_type_reference_type	27
6.9	simpleType: waste_type_reference_type	27
6.10	simpleType: dangerous_goods_type_reference_type	27
6.11	simpleType: non_dangerous_cargo_reference_type	28
6.12	simpleType: MSML_non_dangerous_cargo_type	28
6.13	simpleType: certificate_type	28
6.14	simpleType: propulsion_power_type	29
6.15	simpleType: propulsion_principle_type	29
6.16	simpleType: network_power_source_type	30
6.17	simpleType: hull_material_type	30
6.18	simpleType: damage_status_type	30
6.19	simpleType: supply_shortage_type	30
6.20	simpleType: manoeuvrability_type	31
6.21	simpleType: manual_plan_type	31
6.22	simpleType: record_type	31
6.23	simpleType: shore_base_type	32
6.24	simpleType: shore_base_arrival_passing_type	32
6.25	simpleType: vessel_hindrance_reason_type	33
6.26	simpleType: deficiencies_rectified_limit_type	33
6.27	simpleType: cargo_passenger_transfer_type	33
6.28	simpleType: repair_and_maintenance_reason_type	34
6.29	complexType: detailed_information_type	34
6.30	complexType: shore_base_identity_type	34

6.31	complexType: vessel_type	35
6.32	complexType: waste_type	35
6.33	complexType: dangerous_goods_type	35
6.34	complexType: date_and_time_type	35
6.35	complexType: address_information_type	36
6.36	complexType: timed_address_information_type	36
6.37	complexType: address_history_type	36
6.38	complexType: timed_item_type	37
6.39	complexType: item_history_type	37
6.40	complexType: wire_ropes_type	37
6.41	complexType: engine_type	38
6.42	complexType: network_type	38
6.43	complexType: room_type	38
6.44	complexType: hull_mechanical_securing_type	39
6.45	complexType: equipment_type	39
6.46	complexType: crew_group_capability_type	40
6.47	complexType: environmental_condition_type	40
6.48	complexType: dangerous_goods_cargo_type	41
6.49	complexType: non_dangerous_goods_cargo_type	41
6.50	complexType: damage_type	41
6.51	complexType: vessel_hindrance_type	42
6.52	complexType: shore_base_service_type	42
6.53	complexType: vessel_id_type	42
6.54	complexType: vessel_assistance_type	43
6.55	complexType: route_type	43
6.56	complexType: derived_EncryptedType	44
6.57	Element: MSML	44
6.58	Element: administrative_support	45
6.59	Element: security_support	45
6.60	Element: encrypted_element	46
6.61	Element: data_model	46
6.62	Element: vessel_static_data	46
6.63	Element: description	47
6.64	Element: administration	49
6.65	Element: certificate	51
6.66	Element: constituent	52
6.67	Element: hull	53
6.68	Element: mooring	53
6.69	Element: network	53
6.70	Element: construction	54
6.71	Element: propulsion	55
6.72	Element: safety_equipment	55
6.73	Element: communication_equipment	55
6.74	Element: navigation_equipment	56
6.75	Element: supervision_equipment	56
6.76	Element: emergency_equipment	57
6.77	Element: cargo_passenger_equipment	58
6.78	Element: vessel_dynamic_data	58
6.79	Element: crew	58
6.80	Element: route_at_sea	59
6.81	Element: cargo_passenger	60
6.82	Element: status	60
6.83	Element: constituent_status	61
6.84	Element: emergency_status	62
6.85	Element: document_status	62

6.86	Element: overall_status	63
6.87	Element: previous_tasks	64
6.88	Element: shore_base_static_data	64
6.89	Element: description	65
6.90	Element: administration	65
6.91	Element: service	66
6.92	Element: shore_base_dynamic_data	70
6.93	Element: service_status	71
6.94	Element: overall_status	71
6.95	Element: vessel_shore_base_relation	72
6.96	Element: administration	72
6.97	Element: status	72
6.98	Element: vessel_shore_base_task	73
6.99	Element: perspective	75
6.100	Element: repair_and_maintenance	75
6.101	Element: ashore	75
6.102	Element: onboard	77
6.103	Element: status	79
6.104	Element: inspection	80
7	MSML instance processing (non-normative)	82
8	References (non-normative)	83
9	Appendix A: Considered regulations and directives (non-normative)	84
10	Appendix B: Extensions (non-normative)	85
11	Appendix C: Standardisation of MSML	86
11.1	Background	86
11.2	MSML and other initiatives – use of PAS	86
11.3	Overview of the Electronic Port Clearance issue	87
11.4	Rationale for a new ISO work item	87
11.5	Requirements for standardisation success	88
11.6	References	88

1 Introduction (non-normative)

Planning, performing, recording and evaluating repair and maintenance are crucial for safe transports at sea. Preventive actions are especially cost-effective; is it possible to plan repair and maintenance at the optimal place and time? Fulfilling these aspects will prevent accidents and thus save money and effort for all involved parties. For these reasons the Maritime Safety Markup Language (MSML) is defined. It is an XML application specified using XML Schema (see [10] and [11]).

Since the purpose of MSML is to handle safety aspects in relation to repair and maintenance there is information support for:

- preventing accidents e.g. vessel status, previous repairs, remaining deficiencies
- minimizing extent of damage e.g. personnel training, personal equipment, pollution control
- minimizing criticality of consequences e.g. status of emergency equipment

The information support concerns both vessel and shore base (e.g. a port) and is valid also for other safety related aspects than repair and maintenance and thus future extensions can be made smoothly.

MSML is vessel-centric in the sense that all relevant aspects of the vessel and its task are included while only one of possibly many tasks of a port is included (only the berth used for the vessel). Port is a typical example of a *shore base* as defined in this document. By using MSML it is also possible to associate a vessel and a shore base and the information can flow in the following ways:

- *vessel-to-vessel* e.g. support information if communication with shore base cannot be made
- *vessel-to-shore base* e.g. sending status information
- *shore base-to-vessel* e.g. sending recommended actions such as go to nearest dry-dock for inspection
- *shore base-to-shore base* e.g. preparing the next port to visit for the vessel

However, there is no support in MSML for *relating* a vessel with another vessel and *relating* a shore base with another shore base i.e. there is no support for storing information that describes such a relation. For example, a vessel giving instructions to another vessel must be handled outside MSML (but of course a vessel could just send its information to another vessel if necessary). This means that more than one MSML instance has to be used for:

- relating more than one vessel with a shore base
- relating more than one shore base with a vessel
- relating a vessel with another vessel
- relating a shore base with another shore base

There are several reasons for this design:

- keeping the size down of MSML i.e. not making the definition too complex
- keeping the size down of MSML instances i.e. not letting transfers take too long time
- encouraging vessel to vessel communication via shore base
- letting shore base to shore base communication be handled outside MSML

An example of a possible message sequence using MSML is a vessel approaching a port with the intention of delivering its cargo.

1. The vessel stores vessel data and sends the information to the port.
2. The port checks if there is a berth for the vessel, if it is allowed to enter the port, if there are no alarms, if a pilot is available, if repair and maintenance is accurate etc. The port stores port data and sends data to vessel.
3. The vessel checks port data and requests an acknowledgement.
4. The port relates vessel and port data and sends acknowledgement.

MSML puts no requirements on how much information shall be stored before transactions take place. On the contrary, information could be built up successively using a number of information exchanges

between vessel and shore base. This could be made by partly filled in information or by using fragments of information and even using mirrored versions of information. However, a natural unit is a basic MSML instance since it can be validated using the rules specified in MSML. MSML puts no requirements on the originator of the MSML instance; it could be the vessel, the shore base or another party. The picture (Figure 1) below shows an example.

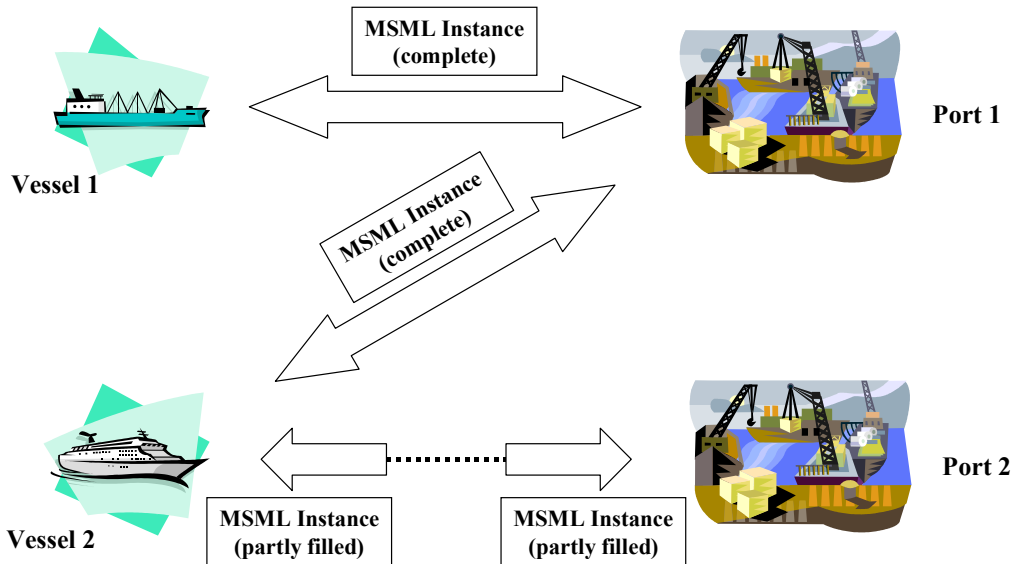


Figure 1 - Example of originator of the MSML instance

Vessel 1 and Port 1 have a mutual agreement as well as Vessel 2 and Port 1. Vessel 2 and Port 2 have not yet established an agreement but both have prepared information relative to the respective side.

2 Background (non-normative)

The creation of MSML is a result of the MANATEE project within the Fifth European Community Framework Programme (IST-2001-38091). A motivation for MSML is given in the MANATEE project description:

- “Enhancing the information and communication channels between ship and shore leading towards a unique e-work platform used and shared between maritime business companies and official Port Authorities, mostly Governmental Bodies as well as all the other interested parties;”
- “Providing simplified access to ship borne and shore based databases and information by users aboard and ashore for decision making support;”
- “Exchanging information on ship’s control system, on-shore supervision and control systems, on-shore and on-ship administrative systems, books, document, circulars, faxes, telexes, improving the connectivity between the on board control systems and the information systems on-shore;”
- “Increasing the use of on-line updated information regarding meteorological data and hazard indication.“

The goal is to create an open standard that can be used generally for safety aspects in the maritime sector. Currently the focus is on repair and maintenance but other aspects can be included in the future.

For the definition of MSML many different resources have been considered. One fundamental source of information is directives and regulations. Those that have been considered relevant concerning maritime use and in relation with repair and maintenance are listed in Appendix A. There are also other initiatives related to MSML however not directly affecting the contents:

- Marine Trading Markup Language (MTML), see [2], is a language for trade and addresses:
 - Trade transactions
 - Price
 - Delivery schedule
 - Goods or servicesMTML is outside the scope of MSML.
- SIRENAC database (see [3]) contains the following information:
 - Ship identification (name, IMO number, flag, ship type, gross registered tonnage, year of build)
 - Class related deficiencies (‘Yes’ or ‘No’), total number of deficiencies,
 - Detention (port of detention, date of release from detention, duration of detention in days, reason(s) for detention)
 - Classification society
 - Owner/operatorSIRENAC contains a subset of the MSML information support.
- EQUASIS database (see [4]) contains the following information:
 - Ship identification
 - Management
 - Classification
 - Safety management certificate
 - P&I information
 - List of Port State Controls
 - Banning orders
 - Association membership
 - Manning information
 - Condensed history

- List of ships under the same management
EQUASIS contains a subset of the MSML information support.
- For the Condition Assessment Program (CAP), see [7], the verification focus is on vessel condition and addresses the following aspects for hull:
 - Rating for each structural group and strength evaluation
 - Survey record
 - Report for fatigue strength assessment
 - Rating for corrosion protection systems of water ballast tanks and coated cargo tanks
 - Photographic report
 - Thickness measurement recordand for machinery/cargo systems
 - Rating for each item
 - Survey record
 - Photographic reportThe CAP results are too detailed for MSML but a reference from MSML can be given to CAP documents.
- ISO 10303, Product Data Representation and Exchange (STEP), see [6], is a set of construction related standards where the following are relevant for maritime use:
 - AP215 Ship arrangement
 - AP216 Ship moulded forms
 - AP218 Ship structures
 - AP226 Ship mechanical systems
 - AP217 Ship pipingThe information support in STEP is for a limited part of the MSML scope and too detailed for MSML.
- The focus of SafeSeaNet (see [5]) is to enable safe transports at sea by keeping better track of vessels and their routes. An important part is the network architecture that defines a distributed database with references to further information. Also, SafeSeaNet defines messages. Since SafeSeaNet concerns safety at sea there is, to a certain extent, an information overlap with MSML. Detailed information is not currently available but is probably a subset of the MSML information support.
- TELEMAS (Tele-maintenance and support through intelligent resource management for ship operation), see [9], aims to increase efficiency and safety of ship operation by combining specific developments together with existing IT systems and tools. Detailed information is not currently available but is probably a subset of the MSML information support.
- OPTIMISE – Optimal Maintenance Intervention of Ships in Europe is focused on hull structural issues such as corrosion, strain damage and cracking (see [8]). Detailed information is not currently available but is probably a subset of the MSML information support.
- System initiatives such as VTS (Vessel Traffic Systems), VTMISS (Vessel Traffic Management and Information Services) and Integrated Ship Control systems (ISC) are not directly considered since the *component* aspects of such systems are addressed by MSML and not the system as such.

If a closer relationship with MSML is needed in the future it could be accomplished by modifying MSML, expanding it or making transformations between different representations. Transformations for XML based information can take place using e.g. XSLT or when accessing a non-native database. Both the underlying data model and grammar could be of interest for modifications.

3 Definitions and terminology (non-normative)

Below definitions and terminology are given specific to this document.

A/A (A/Amax)	Damage stability index, for ro/ro passenger ship
AIS	Automatic Identification System
BCH code	Bulk CHEmical code
Berth	One or more berths can belong to a wharf.
EPIRB	Emergency Position Indicating Radio Beacon
GMDSS	Global Maritime Distress and Safety System
GMT	Greenwich Mean Time
HAZMAT	HAZardous MATerials
Element content	An XML element has element content if the content only consists of other elements (see [1])
IBC	International Bulk Carrier
ICS	International Chamber of Shipping
IGC	International Gas Carrier
IMDG	International Maritime Dangerous Goods
IMO	International Maritime Organization
INF	Irradiated Nuclear Fuel
INMARSAT	INternational MARitime SATellite
ISM	International Safety Management
ISPS	International Ship and Port facility Security
ISS	International Ship Security
MARPOL	MARitime POLLution
MMSI	Maritime Mobile Service Identity
MSML	Maritime Safety Markup Language, an XML application
MSML instance	A document containing vessel and shore base data and is possible to validate using the rules set up by MSML.
OCIMF	Oil Companies International Marine Forum

Paris MOU	Paris Memorandum Of Understanding, on port state control
Port	A port consists of one or more wharfs
SAR	Search And Rescue
SATCOM	SATellite COMmunication
Shore base	A land-based station used for supporting vessels and/or interests on land. In most cases a port is used but it could also be a station just recording vessels passing by and sending information to e.g. authorities.
SOLAS	Safety Of Life At Sea
UN	United Nations
UNCTAD	United Nations Conference on Trade And Disarmament
UNLOCODE	United Nations LOcation CODE
URI	Universal Resource Identifier i.e. a general form of resource address.
Validation	Checking that an MSML instance fulfils the rules given by MSML.
Voyage	A voyage starts from a shore base and can include zero or more intermediate shore bases before reaching the final shore base. Note that different users will have different definitions of the number of shore bases included in the voyage.
VTMIS	Vessel Traffic Management and Information Services
VTs	Vessel Traffic Service
Wharf	A wharf consists of one or more berths.
Wrapper element	An XML element that is only used for enclosing other elements.

4 Scope (non-normative)

4.1 Included

For MSML the following aspects have been especially emphasized:

- The functional applicability in the maritime arena with focus on repair and maintenance and related safety aspects.
- The secure transfer of information between vessel and shore base.
- Extensibility to incorporate increased functionality.
- Use of standardised XML support whenever needed.

The basis of MSML is the data model which defines what kind of maritime data, related to vessel and shore base, that is possible to store. The data model *represents the current state* and only limited historic information is kept in the data model. The following information areas of the data model are defined:

- the vessel
- the actual use and status of the vessel
- the shore base
- the actual use and status of one berth of the shore base
- the relation between the vessel and the shore base
- the history of repair and maintenance and what has been made at each occasion

The data model of MSML makes it possible to describe the following states:

- a vessel with/without defined task
- a berth of a shore base with/without defined task
- a vessel and a berth of a shore base with/without relation

Defining states makes it possible to define *transactions* i.e. sequences of actions for fulfilling specific intentions. We have the general transactions:

- Assigning task to / removing task from vessel
- Assigning task to / removing task from a berth of a shore base (for a specific vessel)
- Assigning relation / removing relation vessel - a berth of a shore base
- Assigning / removing specific pieces of data
- Reading data

Associated with the data model of MSML *perspectives* are defined. The term indicates that the MSML *data model* can be seen from different views. These contain information that is orthogonal to the data model and two are defined:

- *inspection* – this is an inherent perspective and makes it possible to reference results of vessel inspections
- *repair and maintenance* – this is currently the main focus of MSML

There are also other sources of information that are associated with MSML. The data model of MSML assumes that information concerning individual crew and passenger members is handled adequately using the muster list and passenger list respectively. However, for the data model of MSML groups of individuals are considered. The primary purpose of MSML is to make it possible to identify available capabilities and resources and not to handle individuals. However, a reference is included in MSML where to find detailed information.

A certain amount of shore base information is included in the data model of MSML. The main reason to include this information is to support vessel safety related aspects and vessel repair and maintenance.

4.2 Limitations

An MSML instance will contain *extensive* historic information only if included in the definition of a perspective. For example, repair and maintenance contains an extensive history of what has been changed and when, but there is no extensive history information for e.g. bunkering. The reason for including history is that actions made in the past can affect future events and decisions. However, some minor historic information is included in the data model of MSML (i.e. not within perspectives) e.g. the history of vessel name changes. If other types of extensive historic information are needed, a new version of the MSML instance has to be stored for each significant change and this must be handled outside the scope of MSML (probably using a native XML database).

The data model of MSML should not primarily be seen as a support for *normal work onboard*. Instead it is an add-on support for transfers of safety related information to/from the vessel from/to *external* units and within the vessel. For example, current propeller revolutions per minute cannot be extracted from the computerised MSML system, instead the value is read directly from the ordinary equipment. In the same way the shore base part of the MSML data model is seen as a support to the safety related vessel information and not from normal work ashore. Generally, normal dynamic information during a voyage between two shore bases is not included within MSML. On the other hand, alarms and malfunctioning units are generally safety related and of interest internally and externally and thus included in the data model of MSML. If alarms are set automatically or not is a question outside the scope of MSML.

4.3 Not included

The following aspects are *not included* in the scope of MSML:

- Aspects concerning costs and fees
- Geographic information
- Logs e.g. log of communication
- Specific cargo information e.g. tracing, Smart and Secure Tradelanes.
- Presentation of information
- Users and their authorities
- Actual use of data and instances e.g. definition of messages
- Bindings to protocols

4.4 Summary

To sum up the main characteristics of MSML, it

- contains an XML-based data model for information exchange and processing in safety-critical maritime applications.
- does not describe how data is used.
- supports information security and extensibility.
- is a framework and future open standard for the maritime safety sector.

The principal system dependences on MSML are shown below.

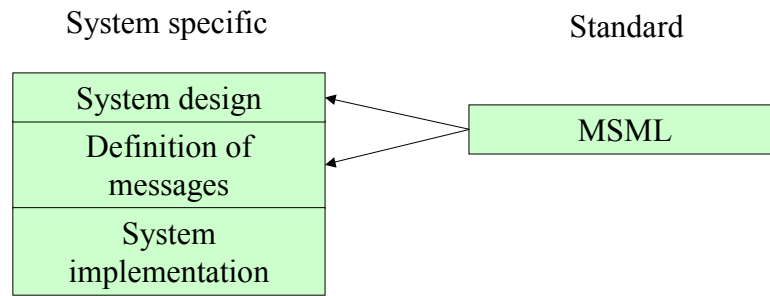


Figure 2 - Principal system dependences on MSML

We see that MSML will affect system design and definition of messages but MSML is not affected. Thus MSML can remain a stable standard for a wide variety of applications.

5 MSML design (non-normative)

5.1 Introduction

MSML is defined using the following primary constituents:

- *data model* – that defines the data of interest. The data model is the basic building block and contains vessel and shore base information. The data model is created top-down where the structure is expanded downwards as much as necessary for fulfilling the intentions.
- *administrative support* – that defines the handling of the XML application instance as a file.
- *security support* – that defines the handling of data security.

For the MSML definition *perspectives* are also included and currently two are defined:

- *inspection* – that contains information related to the data model and to externally made inspections.
- *repair and maintenance* – that contains corresponding information and its relation to the data model.

The figure below shows the building constituents and their connections.

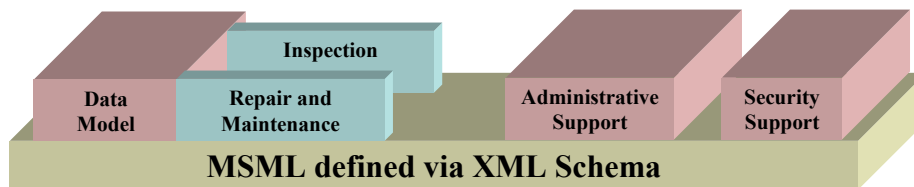


Figure 3 - Building constituents and their connections

There are several advantages using this design and reflecting it in the MSML implementation. A perspective only *references* items of the data model but keep the related information within the perspective as such. Thus a perspective could be changed, removed or added without affecting the data model. Changes could also be made in each block, to a large extent, without affecting the others.

From a tree structure view, i.e. the principle chosen in XML, the picture above is shown below.

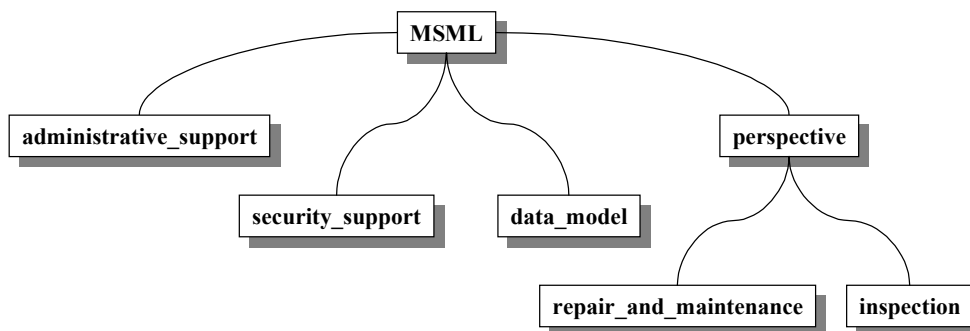


Figure 4 - Tree Structure View

where MSML denotes a top element as required by XML (see [1]).

Future versions of MSML will be created and the major design goal of MSML is to make them *upward compatible* i.e. existing use of MSML shall be unaltered even though extended functionality is included.

5.2 Data model

The data model is the most fundamental block since it sets the scope what could be expressed using MSML. The top view of the data model is described by a number of areas:

- An area containing more or less static data (compared to the number of vessel transports) for the vessel
- A dynamic area containing task related information for vessel
- An area containing more or less static data (compared to the number of handled vessels) for a shore base
- A dynamic area containing service related information for a shore base
- A dynamic area containing information that relates vessel with a shore base

Note that data classified as static also might change, however, in a slow pace. The picture below shows the areas.

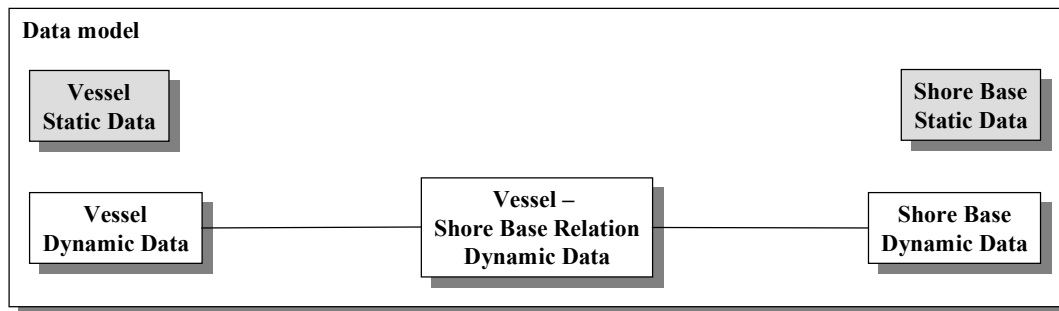


Figure 5 - Areas in the data model

The picture below shows the expanded top view of the areas. A solid line denotes a parent-child relation and a dashed line denotes an association i.e. a relation between two items. Static information is shown in grey.

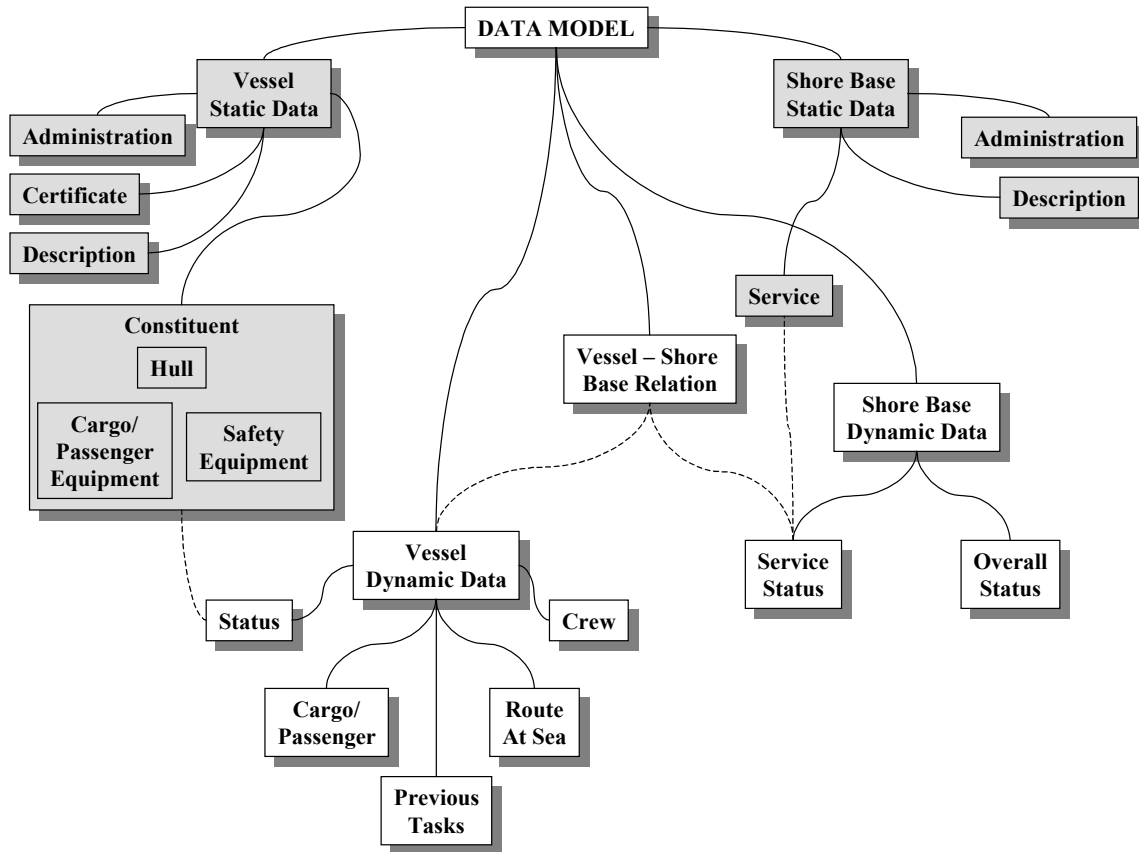


Figure 6 - Expanded top view of the areas in the data model

Each box is represented by an XML *element* and specified in detail below.

5.3 Perspectives

For the *inspection* perspective both static and dynamic vessel data are relevant (not vessel shore base relation).

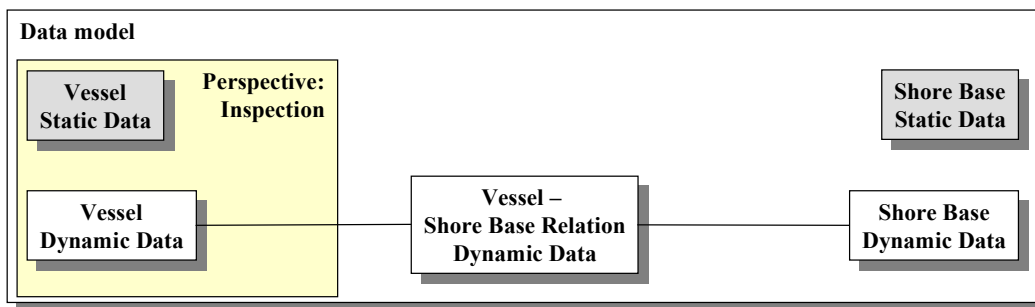


Figure 7 - Static and dynamic vessel data in the data model for the *inspection* perspective

Ship inspections address the status of the vessel and thus overlap the information described by MSML. It is not the intention of MSML that all inspection results shall be included in MSML and the results of inspections can fully be read without using MSML. However, MSML includes references to inspections concerning administrative data and results of inspections. A comparison between MSML and inspections is given below:

- MSML has a wider scope than inspections.
- More detailed information is given by inspections.

- Inspection results can be mapped to MSML generally at a higher level but duplicate information can occur.
- The purpose of inspections is to give reasons *behind* a particular state while the purpose of MSML is to *present* a particular state.
- The inspection results are generally used at non time critical situations while MSML could also be used at time critical situations, e.g. at an emergency, where accurate and concentrated information is of highest priority. MSML can be used for quickly getting an overview of available resources for preventing and/or limiting the effects from an accident.

For the *repair and maintenance* perspective only vessel static data is relevant. Only those events are recorded that result or should/will result in repair or maintenance activities.

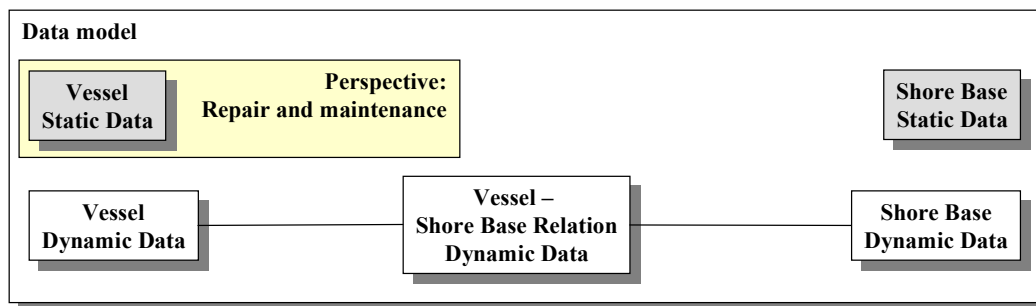


Figure 8 - Static vessel data in the data model for the *repair and maintenance* perspective

Information concerning repair and maintenance also include non-safety aspects due to the following reasons:

- It is sometimes difficult to analyse if repair and maintenance are safety related or not.
- Not-safety related repair and maintenance today could have safety related effects in the future.
- Complete repair and maintenance could be valuable when new perspectives (possibly not safety related) are added.

A perspective may include information that refers to items that no longer exist. In this case a special boolean flag is used for indication.

New perspectives can be added in future versions of MSML.

5.4 Administrative support

The administrative support is defined for handling MSML instances. This includes e.g. version, author data and date. Use of administrative support is mandatory.

5.5 Security support

Security addresses the following aspects:

- integrity – supported within the scope of MSML
- authorisation – not supported within the scope of MSML
- authentication - supported within the scope of MSML
- confidentiality - supported within the scope of MSML
- non-repudiation - supported within the scope of MSML

The supported aspects could be applied independently and individually for each piece of data. Security is supported by using the following W3C recommendations:

- XML Encryption Syntax and Processing, W3C Recommendation 10 December 2002

- XML-Signature Syntax and Processing, W3C Recommendation 12 February 2002

Security aspects are *only enabled* by MSML, the actual use of them is outside the scope of MSML. Use of security support is optional.

For assuring that complete and correct information has been given, a digital signature could be used covering a specific part of the information. That the issuer is allowed to authorise it is, however, outside the scope of MSML.

Other means could be used for security issues, e.g. at protocol level, but they are outside the scope of MSML.

5.6 Successive information build-up

Validation can be made even if the information is not complete. The following parts can be individually controlled at successive information build-up:

- Vessel Dynamic Data
- Vessel Static Data
- Shore Base Dynamic Data
- Shore Base Static Data
- Vessel - Shore Base Relation
- Each perspective (Repair and Maintenance, Inspection)

Within each part there are also possibilities e.g. when zero or more elements has been specified one can start without any element at all and add successively.

The rules given below govern the dependence on other parts at successive build-up. The following notation is used:

$\exists(x)$ x exists and can be validated
 \Rightarrow implies
 \wedge logical AND

$\exists(\text{Vessel Dynamic Data}) \Rightarrow \exists(\text{Vessel Static Data})$
 $\exists(\text{Shore Base Dynamic Data}) \Rightarrow \exists(\text{Shore Base Static Data})$
 $\exists(\text{Vessel - Shore Base Relation}) \Rightarrow (\exists(\text{Vessel Dynamic Data}) \wedge \exists(\text{Shore Base Dynamic Data}))$
 $\exists(\text{Perspective Repair and Maintenance}) \Rightarrow \exists(\text{Vessel Static Data})$
 $\exists(\text{Perspective Inspection}) \Rightarrow (\exists(\text{Vessel Static Data}) \wedge \exists(\text{Vessel Dynamic Data}))$

How checking of the rules is performed is outside the scope of MSML.

5.7 MSML vs. standardised maritime terminology

There are several different notations for describing the same item e.g. the type of vessel. Since it is not beneficial to define another standard in MSML the approach is instead to specify the source of standard terminology and refer to items according to this standard. The advantage is that the notation is uniquely identified and extensions can be made without affecting MSML. The disadvantage is that it is not possible to validate the information within MSML. Thus, the verification of e.g. correct vessel type has to be made by the system and outside the scope of MSML.

5.8 MSML processor

The picture shows the principal handling of an MSML instance at transmission.

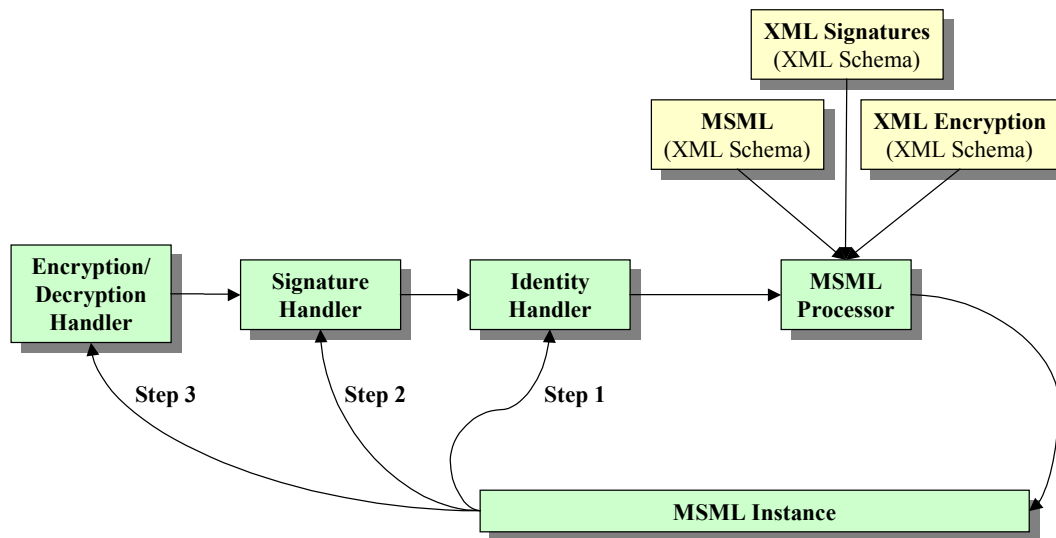


Figure 9 - Principal handling of an MSML instance at transmission

References and unique element identities are used internally within an MSML instance for associating different elements. The algorithm for creating unique identities is specified in Administrative Support. The advantage of using a specific algorithm is that it is easily extended and that different users can modify the structure in a consistent way. A consequence is that different MSML instances cannot be compared by using element identities (not even if they use the same algorithm).

Signatures are used for establishing that data is authenticated. Encryption is used for hiding sensitive information. Signature and encryption handling are optional but when used it must be checked that references and unique element identities are still handled in a consistent way. The processing associated with digital signatures and encryption is specified in [13] and [14] respectively.

The logical behaviour of the MSML processor at transmission of information is specified in the following steps:

- *Step 1* – the “Identity Handler” generates and matches identities between referencing elements and elements being referenced. The MSML processor can then check for syntax errors.
- *Step 2* – the “Signature Handler” adds signatures for important parts. The MSML processor can then check for syntax errors.
- *Step 3* - the “Encryption/Decryption Handler” makes encryption of important parts. The MSML processor can then check for syntax errors.

The logical behaviour of the MSML processor at reception of information is specified in the following steps:

- *Step 1* - the “Encryption/Decryption Handler” makes decryption of encrypted parts and indicates any errors. The MSML processor can then check for syntax errors.
- *Step 2* – the “Signature Handler” checks signatures for signed parts and indicates any errors. The MSML processor can then check for syntax errors.

6 MSML Specification (normative)

6.1 Introduction

The specification is based on XML Schema (see [10] and [11]). Lower case letters are used in names as far as possible. Underscore is used for separating words. Element information after colon is type used for the element. A type can be predefined in XML Schema or created specifically for MSML.

The MSML namespace is defined by “urn:oasis:names:tc:MSML:1.0” (*only valid if oasis is chosen*). Version handling is thus included in the namespace and 1.0 is the first official version.

6.2 Referencing

Referencing occurs between elements i.e. one element referencing another using unique identities. The following apply:

- the sets of references and identities are specific for each MSML instance
- each element *of the data model only* has an explicit unique identity if referenced. It could have a unique identity even if not referenced but this is not a requirement. The algorithm used for creating identities is specified in each MSML instance
- referencing is allowed within the data model and for any element having a relation with an element of the data model
- a unique identity cannot be reused i.e. once used it will be considered consumed forever. The reason is that perspectives might refer to old elements even if they do not exist anymore.

For both perspective and data model, elements are referencing data model elements using XML *attributes*. However, there are differences:

- A reference within the data model has a one-to-one relationship.
- A reference in a perspective (referencing an element of the data model) has a possibly one-to-many relationship. For example, a single repair and maintenance occasion may affect several constituents.
- More than one reference in a perspective can be referencing the same referenced element e.g. a single constituent may have been repaired at several repair and maintenance occasions.
- A reference within the data model includes at least one element that belongs to dynamic data (as defined earlier). This is not necessary for perspectives.
- For a reference within the data model the referenced element must exist.
- A reference can be removed in a perspective if the referenced element no longer exists. However, the corresponding information is kept. For example, a constituent could first be repaired and later on removed. The information will be kept, however, no attribute is used and the corresponding element *referenced_element_no_longer_exists* contains TRUE.
- There is no information from a referenced element back to the element referencing it. This is a consequence of letting the data model be unaffected when adding perspectives (a desirable property).
- Time information is used for synchronizing events within perspectives.

For referenced elements the attribute is declared as:

- element_identity (0,1): ID

i.e. optional and using the predefined type ID. *For clarity reasons, in the specification below, the declaration of this attribute is not shown.* The following items include the attribute declaration:

- Global complex types
- Local complex types within data model
- Data model elements

For an element referencing another element the corresponding attribute is declared as:

- `element_identity (0,1): IDREF`

i.e. optional and using the predefined type IDREF. This declaration is shown whenever applicable in the specification below. The attribute cannot be used when the referenced element no longer exists.

An element, within the data model only, that references another element can also be referenced. In this case a wrapper element is defined that *only* contains the element referencing the other element. The wrapper element includes the declaration:

- `element_identity (0,1): ID`

and could thus be referenced. The wrapper element is named `IDREF_wrapper_element_x` where x is an integer starting from one in its scope.

6.3 Security support

The following rules apply:

- Encryption can only be made for elements that *do not* in themselves contain elements (i.e. only for leaves of the MSML tree).
- Encryption is made for data only and does not include element markup tags. In this way references and identities are kept, preserving the validity of the MSML instance. Encrypting already encrypted data is not allowed (and of no real value).
- Data for encryption is copied from the original element and stored in a separate element which keeps a reference to the original element. After encryption, dummy data is placed in the original element. In the other direction, data is first decrypted and then stored in the original element. The encryption and decryption handling is made via an extra (logical) processor.
- Use of dummy data, i.e. after encryption, must be indicated as erroneous.
- Signatures are not allowed on encrypted information, signatures can only be made on human readable data.

A consequence of this is that there will be a relatively high overhead in textual size for each piece of encrypted data. However, it is likely that only small parts of information will be encrypted and so the total size overhead will in all normal cases be low.

For encryption the following apply:

- W3C standard “XML Encryption Syntax and Processing, W3C Recommendation 10 December 2002” is used, see [14].
- The XML Schema is found at <http://www.w3.org/TR/xmlsig-core/xmlsig-core-schema.xsd>
- The name space and prefix is defined by `xenc=http://www.w3.org/2001/04/xmlenc#`
- Copyright information is given in <http://www.w3.org/Consortium/Legal/IPR-FAQ-20000620.html#DTD>

For signature the following apply:

- W3C standard “XML-Signature Syntax and Processing, W3C Recommendation 12 February 2002” is used, see [13].
- The XML Schema is found <http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/xenc-schema.xsd>
- The name space and prefix is defined by `ds=http://www.w3.org/2000/09/xmlsig#`
- Copyright information is given in <http://www.w3.org/Consortium/Legal/IPR-FAQ-20000620.html#DTD>

Inclusion of `xenc.xsd` is made by using XML Schema item `import`, `ds.xsd` is then automatically included.

6.4 Relation to XML Schema

Below, values within parenthesis give minimum and maximum occurrence e.g. (1,1) denotes a single mandatory element in exact accordance with XML Schema notations minOccurs and maxOccurs.

Mixed content (see [1]) is not used.

Extension elements are added where applicable but validation of their contents is not possible since they are of XML Schema type *anyType* i.e. unspecified.

The XML Schema primitive datatypes are used.

Only period is allowed as a separator of whole-fraction float values e.g. 1.23 is allowed while 1,23 is not (will generate an error at validation).

For date and time the XML Schema primitive datatype *dateTime* is used which is defined according to § 5.4 of ISO 8601.

6.5 Special considerations

In many cases there is a choice between defining a new *element* or a new *attribute*. The approach in MSML is to use a new *element* whenever a new *attribute* is not explicitly required. The reasons for this are:

- Structure is improved and can be maintained
- It is easier to extend
- It improves human readability

Verification that complete information has been given is not directly supported by MSML. However, in some cases flags and counters have been added in order to lower the risk of inadvertent loss of information.

In several cases below an ordered list of elements can be created. One example is to describe events in time order. The actual order of these elements in the MSML instance *is not significant*. Instead each such element contains information that makes it possible to sort the elements in correct order, however, outside the scope of MSML. The total list size is in some cases, but not all, obvious but control is outside the scope of MSML.

Vessel administration is not affected by repair and maintenance and the corresponding history is kept within the *vessel administration* element.

Vessel constituents can only be affected at repair and maintenance and the corresponding history is kept within the *repair_and_maintenance* element i.e. within the perspective. The information kept in element *constituent* is the current state.

Vessel description can only be affected at repair and maintenance and the corresponding history is kept within the *repair_and_maintenance* element i.e. within the perspective. The information kept in element *description* is the current state.

Vessel certificates can be affected at repair and maintenance but also as a result of inspection and from other reasons (such as expiration). The corresponding history is kept within the *vessel certificate* element.

6.6simpleType: units_type

Value:

- SI_units
- US_units

The value of the attribute must be the same within an MSML instance. Presentation of data values must indicate the units used. The attribute is included for the element closest to the data. The table below shows the differences as applied in this document.

Item	SI_units	US_units
Basic length measure	meter	foot
Vessel, Helicopter, Plane speed	knot	knot
Time	According to XML Schema format	According to XML Schema format
Distance at sea	Nautical mile	Nautical mile
Weight	tonne	ton
Strength at pulling	tonne	ton
Turning rate	degrees (0-360) / second	degrees (0-360) / second
Course	degree (0-360)	degree (0-360)
Engine power	kW	hp
Longitude (according to ISO 6709:1983 and WGS 84)	+/- dddmmss.ss	+/- dddmmss.ss
Latitude (according to ISO 6709:1983 and WGS 84)	+/- dddmmss.ss	+/- dddmmss.ss
Engine consumption at service speed	tonne/24 hours	ton/24 hours
Towing capacity	tonne	ton
“identifier” that include units	according to identifier	according to identifier

For the last row “identifier” could be any name e.g. stopping_time_in_minutes.

According to the standard ISO 6709:1983 "Standard representation of latitude, longitude and altitude for geographic point locations" latitude can be specified using three possible formats:

- +/- dd.dd degrees and decimal degrees
- +/- ddmm.mmm degrees, minutes and decimal minutes
- +/- dddmmss.ss degrees, minutes, seconds and decimal seconds

In the same way, longitude can be specified using three possible formats:

- +/- ddd.dd degrees and decimal degrees
- +/- dddmm.mmm degrees, minutes and decimal minutes
- +/- dddmmss.ss degrees, minutes, seconds and decimal seconds

However, in MSML only the last format of latitude and longitude respectively is allowed. For latitude + denotes north and – south. For longitude + denotes east of Greenwich and – west up to 180th meridian. Leading zeros are required for latitude and longitude e.g. 6 minutes is represented as 06. For this reason MSML defines strings for latitude and longitude.

Information concerning WGS 84 can be found at <http://www.wgs84.com/>.

For SI units see ISO 31, Quantities and Units, ISO Standards Handbook, 3rd edition ISO, Geneva, 1993. Conversion factors are defined in Appendix B of:
NIST Special Publication 811 1995 Edition, Guide for the Use of the International System of Units (SI), Barry N.Taylor, <http://physics.nist.gov/Document/sp811.pdf>

Due to XML Schema rules only period is allowed as a separator of whole-fraction float values e.g. 1.23 is allowed while 1,23 is not.

6.7simpleType: element_identity_algorithm_type

Value:

- undefined
- integer_group

The following values exist:

- undefined: The algorithm is undefined i.e. any identity will do as long as it is unique.
- integer_group: The algorithm is described below

The integer_group algorithm has the advantage that changes can be handled in a controlled manner. An example of an element identity is r.2.17.1 which specifies the first child element of the 17th child element of the second element just below the root. The algorithm for creating unique identities is defined according to:

- the root (the MSML element) is marked as r
- the identity is created by a number of integers each separated by period
- the first possible integer value is one
- a period with a number represents a level in the data model structure
- elements on the same level have successive integer values
- if an element is removed and leaves a “hole” in the used integer values the value is not reused
- if an intermediate element is removed, its children are first removed and then restructured using new integer values.

The value of the attribute must be the same within an MSML instance.

6.8simpleType: vessel_type_reference_type

Enumeration value:

- lloyds
- equasis_data_base
- sirenac_data_base

These are the possible sources that define types of vessel.

6.9simpleType: waste_type_reference_type

Enumeration value:

- other

These are the possible sources that define types of waste.

6.10simpleType: dangerous_goods_type_reference_type

Enumeration value:

- IMDG
- HAZMAT
- UN_number

These are the possible sources that define types of dangerous goods. HAZMAT stands for HAZardous MATerials and IMDG stands for International Maritime Dangerous Goods. UN number is used for identifying hazardous chemicals.

6.11simpleType: non_dangerous_cargo_reference_type

Enumeration value:

- MSML

Only MSML own definition of cargo type is currently supported i.e. type *MSML_non_dangerous_cargo_type*.

6.12simpleType: MSML_non_dangerous_cargo_type

Enumeration value:

- animal_transport
- bulk_cargo_transport
- container_transport
- liquid_cargo_transport
- truck_transport
- car_transport
- wagon_transport
- bus_transport
- other

This type contains the MSML specific classification of non-dangerous cargo.

6.13simpleType: certificate_type

Enumeration value:

- BCH_code_certificate
- cargo_gear_compliance
- cargo_ship_safety_certificate
- cargo_ship_safety_construction_certificate
- cargo_ship_safety_equipment_certificate
- cargo_ship_safety_radio_certificate
- cargo_ship_safety_radio_telegraphy_certificate
- cargo_ship_safety_radio_telephony_certificate
- certificate_for_the_carriage_of_grain
- certificate_of_fitness_for_dangerous_chemicals_in_bulk
- certificate_of_fitness_for_INF_cargo
- certificate_of_fitness_for_liquefied_gases_in_bulk
- certificate_of_fitness_for_offshore_supply_vessels
- certificate_of_registry
- civil_liability_certificate_pollution
- document_of_compliance
- exemption_certificate
- fishing_vessel_safety_certificate

- high_speed_craft_safety_certificate
- hull_certificate
- IBC_code_certificate
- ice_class_certificate
- IGC_code_certificate
- INF_code_certificate
- international_load_line_certificate
- international_load_line_exemption_certificate
- international_oil_pollution_prevention_certificate
- international_pollution_prevention_certificate_for_noxious_liquid_in_bulk
- international_sewage_pollution_prevention_certificate
- international_suez_and_panama_tonnage_certificates
- international_tonnage_certificate
- ISM_code_certificate
- ISPS_certificate
- ISSC_certificate
- machinery_certificate
- mobile_offshore_drilling_unit_safety_certificate
- navigation_certificate
- nuclear_cargo_ship_safety_certificate
- nuclear_passenger_ship_safety_certificate
- passenger_ship_safety_certificate
- permit_to_operate_high_speed_craft
- quality_management_system_certificate
- safe_manning_certificate
- safety_management_certificate
- special_purpose_ship_safety_certificate
- special_trade_passenger_ships_certificate
- unattended_machinery_space_certificate
- other

The list contains the most important and the most commonly used certificates.

6.14simpleType: propulsion_power_type

Enumeration value:

- steam
- electric
- diesel
- nuclear
- gas_turbine
- steam_turbine
- other

These values define the possible power sources behind propulsion.

6.15simpleType: propulsion_principle_type

Enumeration value:

- propeller
- water_jet

- sail
- other

These values define the principle behind the propulsion.

6.16simpleType: network_power_source_type

Enumeration value:

- electric_DC
- electric_AC
- pneumatic
- hydraulic
- other

Different types of networks exist onboard. The electric_DC value could e.g. be used for computer network, emergency lighting etc. To understand the origin of power is necessary for understanding the consequences when e.g. this power is lost.

6.17simpleType: hull_material_type

Enumeration value:

- steel
- wood
- aluminium
- reinforced_plastic
- reinforced_concrete
- sandwich
- unknown
- other

This type is used for describing the material of the hull and makes it possible to better estimate e.g. extent of damage and its consequences.

6.18simpleType: damage_status_type

Enumeration value:

- increasing_under_control
- increasing_without_control
- not_increasing
- decreasing
- catastrophe
- non_dangerous

This type enables dynamic information concerning the status of damage. Depending on this value e.g. emergency actions could be prioritised.

6.19simpleType: supply_shortage_type

Enumeration value:

- fuel
- medical
- other

This type contains supply factors that could affect safety aspects and/or the capability of handling emergencies.

6.20simpleType: manoeuvrability_type

Enumeration value:

- limited_steering_capability
- limited_forward_capability
- limited_backward_capability
- limited_list
- without_steering_capability
- without_forward_capability
- without_backward_capability
- dangerous_list
- without_control
- normal_function

This type contains values that describe the capability of vessel manoeuvring.

6.21simpleType: manual_plan_type

Enumeration value:

- cargo_securing_manual
- damage_control_plan
- fire_control_plan
- waste_management_plan
- ICS_OCIMF_ship_to_ship_transfer_guide
- IMO_inert_gas_systems
- international_safety_guide_for_oil_tanker_and_terminals
- loading_unloading_plan
- master_decision_support_system
- personal_emergency_instruction
- pollution_emergency_plan
- procedures_and_arrangements_manual_MARPOL
- SAR_cooperation_plan
- ship_security_plan
- stowage_plan
- other

Document are separated into plans and record where plans are generally static and made in advance and records describe the status of changing data.

6.22simpleType: record_type

Enumeration value:

- A_A_maximum_ratio_ro_ro_passenger_ship_record
- bulk_carrier_booklet
- cargo_record_book
- condition_evaluation_reports
- damage_control_document

- dangerous_goods_list
- waste_record_book
- insurance_document
- list_of_operational_limitations
- loading_instrument_SOLAS
- log_book
- oil_discharge_last_ballast_voyage
- oil_record_book_part_1
- oil_record_book_part_2
- operational_limitations_passenger_ships
- radio_document
- results_of_enhanced_survey_guidelines_IMO
- stability_document
- survey_report_files
- thickness_measurement_reports
- other

Document are separated into plans and record where plans are generally static and made in advance and records describe the status of changing data.

6.23simpleType: shore_base_type

Enumeration value:

- AIS_base_station
- public_port
- VTS_radio_no_radar
- VTS_radio_and_radar
- private_port
- other

A shore base can be entered or passed by the vessel. The type of shore base reflects the capability of the shore base especially concerning the amount and quality of its services.

6.24simpleType: shore_base_arrival_passing_type

Enumeration value:

- emergency
- hijacked
- bunkering
- loading_or_unloading_cargo_passenger
- change_of_crew
- arrested
- repair
- maintenance
- drifting_object
- weather
- sea_condition
- ice_condition
- passing
- other

This type defines the reason for arriving to a shore base or just passing it. This is seen from the perspective of the shore base. Note that a vessel could be arrested and tugged to a shore base.

6.25simpleType: vessel_hindrance_reason_type

Enumeration value:

- banned_on_black_list
- not_acceptable_conditions
- missing_documents
- missing_information
- undeclared_goods
- arrested
- fines_not_payed
- not_seaworthy
- drifting_object
- weather
- sea_condition
- ice_condition
- tidal_stream
- not_handled_waste
- other

This type defines the reason for not allowing a vessel to enter or leave a shore base. This is seen from the perspective of the shore base.

6.26simpleType: deficiencies_rectified_limit_type

Enumeration value:

- no_limit
- time_limit
- before_departure
- at_next_shore_base
- other_condition

If there is a vessel deficiency a condition could be specified, e.g. by authorities, when it shall be rectified.

6.27simpleType: cargo_passenger_transfer_type

Enumeration value:

- delayed
- ongoing
- started
- not_started
- finished
- loading
- unloading
- other

These values give the status of handling cargo or passengers. The corresponding state is important for deciding the risks at an emergency.

6.28simpleType: repair_and_maintenance_reason_type

Enumeration value:

- pollution_incident
- grounding_incident
- hit_by_wave_incident
- collision_incident
- fire_incident
- explosion_incident
- sabotage_incident
- other_incident
- port_state_control_inspection
- classification_inspection
- certificate_inspection
- other_inspection
- special_periodical_survey
- annual_general_survey
- enhanced_survey_programme
- continuous_machinery_survey
- other_survey

These values define the reason why repair and/or maintenance will take place or has been taken place. The corresponding history creates a picture of what has happened to the vessel from this perspective.

6.29complexType: detailed_information_type

Element:

- information_reference (1,1): string
- information_in_place (1,1): boolean
- location (1,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type handles detailed information generally, both for vessel and shore base items. The amount of information could be extensive and only a reference is given here to more detailed information. An example could be to give title, document number and version. Note that the quality of this reference cannot be checked at validation (it is just a string). As a first improvement *information_in_place* has been added to indicate if vessel or shore base contains detailed information respectively. If vessel or shore base is addressed is clear from the context of the element using this type. As a second improvement there is the *location* element describing where to find the information.

6.30complexType: shore_base_identity_type

Element:

- UNLOCODE (0,1): string (5-11 upper case letters A-Z)
- name (1,1): string
- local_location_code (0,1): string
- longitude (1,1): string, attribute units (1,1): units_type
- latitude (1,1): string, attribute units (1,1): units_type
- extension (0,unbounded): anyType

Attribute:

- -

The UN code is generally accepted but there could also be a local identification when the UN code is too coarse or when the UN code is missing.

6.31complexType: vessel_type

Element:

- vessel_type_reference(1,1): vessel_type_reference_type
- vessel_type_according_to_reference(1,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type defines the type of vessel e.g. general cargo ship, passenger ship etc. according to an already established notation given by the reference. No restriction is placed on how to specify the vessel type; it could be a code or plain text. Thus information from the reference source might be needed in order to interpret the kind of vessel. This type is easily extended, by including a new reference, however validation of this information can not take place using an MSML instance in isolation.

6.32complexType: waste_type

Element:

- waste_type_reference(1,1): waste_type_reference_type
- waste_type_according_to_reference(1,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type defines the type of waste according to an already established notation given by the reference. No restriction is placed on how to specify the waste type; it could be a code or plain text. Thus information from the reference source might be needed in order to interpret the kind of waste. This type is easily extended, by including a new reference, however validation of this information can not take place using an MSML instance in isolation.

6.33complexType: dangerous_goods_type

Element:

- dangerous_goods_type_reference(1,1): dangerous_goods_type_reference_type
- dangerous_goods_type_according_to_reference(1,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type defines the type of dangerous goods according to an already established notation given by the reference. No restriction is placed on how to specify the dangerous goods type; it could be a code or plain text. Thus information from the reference source might be needed in order to interpret the kind of dangerous goods. This type is easily extended, by including a new reference, however validation of this information can not take place using an MSML instance in isolation.

6.34complexType: date_and_time_type

Element:

- GMT_date (0,1): dateTime
- local_date (1,1): dateTime
- country_for_local_date (1,1): string
- extension (0,unbounded): anyType

Attribute:

- -

At least local time must be specified. In this case also the country has to be specified for making a transfer to GMT possible. If there is no associated country GMT has to be used.

6.35complexType: address_information_type

Element:

- addressed_item (1,1): string
- item_role (1,1): string
- contact_person (1,1): string
- identity (0,1): string
- address (0,1): string
- telephone_number (0,5): string
- after_hours_telephone_number (0,1): string
- telex_number (0,1): string
- fax_number (0,1): string
- email_address (0,1): string
- web_address (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type is used anytime a person or computer contact information is needed. The item_role element is used for adding information concerning the role of the contact e.g. if there are two police offices and one is the main office.

6.36complexType: timed_address_information_type

Element:

- timed_addressed_item (1,1): address_information_type
- date (1,1): date_and_time_type
- valid (1,1): boolean
- extension (0,unbounded): anyType

Attribute:

- -

This type adds timing information to *address_information_type*. The *valid* flag is used for indicating if the address is valid. In some cases this type is used directly and then it is important to indicate if the address is still valid. In other cases the type is used in *address_history_type* and in that case only one address, the most recent one, can be valid.

6.37complexType: address_history_type

Element:

- address_history_field (1,unbounded): timed_address_information_type

- extension (0,unbounded): anyType

Attribute:

- -

This type makes it possible to create a history of contact information, e.g. the addresses of different owners of a vessel, and when changes took place.

6.38 complexType: **timed_item_type**

Element:

- timed_item_field (1,1): string
- item_role (1,1): string
- date (1,1): date_and_time_type
- valid (1,1): boolean
- associated_change (0,unbounded): string
- extension (0,unbounded): anyType

Attribute:

- -

This type adds timing information to an item defined by a description and a role. The *valid* flag is used for indicating if the information is valid. In some cases this type is used directly and then it is important to indicate if the item is still valid. In other cases the type is used in *item_history_type* and in that case only one item, the most recent one, can be valid.

6.39 complexType: **item_history_type**

Element:

- item_history_field (1,unbounded): timed_item_type
- extension (0,unbounded): anyType

Attribute:

- -

This type makes it possible to create a history for an item, e.g. the flag state of a vessel, and when changes took place.

6.40 complexType: **wire_rope_type**

Element:

- identity (1,1): string
- location (1,1): string
- strength (1,1): float, attribute units (1,1): units_type
- length (1,1): float, attribute units (1,1): units_type
- holding_strength (1,1): float, attribute units (1,1): units_type
- extension (0,unbounded): anyType

Attribute:

- -

Wire, rope, chain etc. are the means for holding the vessel in place e.g. at strong winds. The *identity* and *location* elements make it possible to identify the actual piece of wire, rope, chain etc. The *strength* and *length* elements give performance values. The *holding_strength* also relates to the strength of the vessel connection point. This type thus contains information defining the vessel holding capability related to a specific wire, rope, chain etc.

6.41 complexType: engine_type

Element:

- engine_role (1,1): string
- identity (1,1): string
- location (1,1): string
- manufacturer (1,1): string
- configuration_data (0,1): string
- detailed_information (1,1): detailed_information_type
- document_location_onboard (1,1): string
- backup_engine_identity (0, unbounded): string
- extension (0,unbounded): anyType

Attribute:

- -

This type contains information defining a specific engine and its role onboard e.g. the main engine. The *identity* contains a unique identity of the engine. The *configuration_data* contains extra information (if necessary) that describes special data needed for the actual use of the engine onboard i.e. a vessel specific configuration using an engine of a more general type. The *backup_engine_identity* specifies a unique identity of another engine (if possible) that could take over the role (wholly or partly) of this engine.

6.42 complexType: network_type

Element:

- detailed_information (1,1): detailed_information_type
- connected_constituent (1,unbounded): string
- network_power_source (1,1): network_power_source_type
- basic_power_supply_engine (1,1): engine_type
- power_supply_backup (1,1): string
- extension (0,unbounded): anyType

Attribute:

- -

There are different types of networks onboard as indicated by *network_power_source_type* e.g. hydraulic, electric etc. Since networks connect different constituents, networks are handled as a special kind of constituent in MSML. The elements containing power supply information are of high relevance since it is important to understand:

- the effects when a power supply is lost e.g. is there backup
- that a primary fault can cause several and different types of secondary faults (common mode fault)

6.43 complexType: room_type

Element:

- role (1,1): string
- identity (1,1): string
- location (1,1): string
- volume (1,1): float, attribute units (1,1): units_type
- type_of_protection (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type specifies a limited space onboard uniquely specified and with a specific role e.g. a cargo hold. Important is the *type_of_protection* that makes it possible to evaluate the state and the quality of protection e.g. in case of sabotage.

6.44 complexType: hull_mechanical_securing_type

Element:

- type (1,1): string
- identity (1,1): string
- location (1,1): string
- securing_mechanism (1,unbounded): string
- extension (0,unbounded): anyType

Attribute:

- -

This type contains mechanical securing information for mechanical devices connected to the hull. Enough information is included for uniquely identify the mechanism, however, the description of the mechanism as such is not detailed (only one or more strings). The constituents using this type are related to safety and maintenance and could e.g. be cargo lifting devices.

6.45 complexType: equipment_type

Element:

- type (1,1): string
- role (1,1): string
- identity (1,1): string
- location (1,1): string
- manufacturer (1,1): string
- emergency_equipment (1,1): boolean
- supervised_automatically (1,1): boolean
- detailed_information (1,1): detailed_information_type
- configuration_data (0,1): string
- dependent_on_other_item (0,unbounded): string
- network_powered (0,1): network_power_source_type
- engine_powered (0,1): engine_type
- capacity_type (0,unbounded): string
- capacity_value (0,unbounded): float, attribute units (1,1): units_type
- number_if_grouped (0,1): integer
- extension (0,unbounded): anyType

Attribute:

- -

This type is used generally and defines, in principle, any kind of equipment used both for vessel and for shore base and identifies uniquely the equipment. The *role* describes how the equipment is used e.g. two radars of the same type can be used for different purposes (roles). This type can be used also for emergency equipment as flagged by *emergency_equipment*. If this equipment is supervised automatically, in one way or another, there is a possibility of automatically generating alarms and warnings. The *configuration_data* contains extra information (if necessary) that describes special data needed for the actual use of the equipment e.g. used radio channels. The equipment can depend on other items for its function and this information is important for handling common mode faults (where

a primary fault generates secondary faults that could look independent but are actually dependent on the first one). For equipment that is powered the alternatives are network and separate engine, however, equipment could also be using no power at all. For some kinds of equipment capacity is of interest and this is described by *capacity_type* and *capacity_value*. This type also supports grouping of equipment as specified in *number_if_grouped*. This is practical if there are many identical items that is more meaningful to be handled as a group e.g. life rafts.

6.46 complexType: crew_group_capability_type

Element:

- *capability_type* (1,1): string
- *identity* (1,1): string
- *ratings_with_capability* (1,1): integer
- *officers_with_capability* (1,1): integer
- *ratings_with_valid_certificates* (1,1): integer
- *officers_with_valid_certificates* (1,1): integer
- *last_training* (0,1): date_and_time_type
- *last_training_responsible* (0,1): string
- *last_training_onboard* (0,1): date_and_time_type
- *group_common_language* (0,1): string
- *english_understood* (1,1): boolean
- *number_of_indisposed_ratings* (1,1): integer
- *number_of_indisposed_officers* (1,1): integer
- *reason_for_indisposed* (0,1): string
- *loss_of_capabilities_due_to_indisposed_persons* (0,unbounded): string
- *extension* (0,unbounded): anyType

Attribute:

- -

The overall question is: could the crew adequately handle an emergency? At the individual level there is the muster list. It is not meaningful to include the corresponding details in MSML since the scope is to handle safety aspects and considering the total capability and capacity of the crew. Doubling the information also gives a high risk of inconsistencies. The approach is instead to make it possible to define groups having unique identities. Each group contains crew members (without details of individuals) with specific capabilities e.g. one firefighting group, one pollution handling group etc. However, the groups do not have to have any relation with the organisation onboard. It is also possible to define different groups with the same capability e.g. several firefighting teams. The number of persons and if they have certificate for the capability are specified for officers and ratings. Last training information is of high interest and also if there is a common language spoken within the group; the effectiveness depends on these factors. The capability also depends on if there are indisposed persons within the group and the reason behind.

6.47 complexType: environmental_condition_type

Element:

- *aspect_type* (1,1): string
- *detailed_information* (1,1): detailed_information_type
- *valid_from_date* (1,1): date_and_time_type
- *valid_to_date* (1,1): date_and_time_type
- *issued_by* (1,1): address_information_type
- *valid_for_area* (1,1): string
- *extension* (0,unbounded): anyType

Attribute:

- -

There are environmental aspects, such as weather and ice conditions, that have to be considered from a vessel safety point of view. The *aspect_type* defines the environmental aspect and there is also added associated data concerning its scope and where detailed information can be found.

6.48 complexType: dangerous_goods_cargo_type

Element:

- dangerous_goods_type_reference(1,1): dangerous_goods_type_reference_type
- dangerous_goods_type_according_to_reference(1,1): string
- location (1,unbounded): string
- weight (1,1): float, attribute units (1,1): units_type
- weight_above_deck (1,1): float, attribute units (1,1): units_type
- securing_mechanism (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type is used for a specific kind of dangerous goods according to an already established notation given by the reference. No restriction is placed on how to specify the dangerous goods type; it could be a code or plain text. Thus information from the reference source might be needed in order to interpret the kind of dangerous goods. Details of cargo handling is outside the scope of MSML e.g. if parts of the cargo shall be delivered to different shore bases. However, if different types of dangerous goods exist each type must be specified separately using this type. The *securing_mechanism* describes how cargo is secured.

6.49 complexType: non_dangerous_goods_cargo_type

Element:

- cargo_type_reference(1,1): non_dangerous_cargo_reference_type
- cargo_type (1,1): MSML_non_dangerous_cargo_type
- location (1,unbounded): string
- weight (1,1): float, attribute units (1,1): units_type
- weight_above_deck (1,1): float, attribute units (1,1): units_type
- securing_mechanism (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type is used for a specific kind of non-dangerous goods according to an already established notation given by the reference. Currently the notation is according to MSML. Details of cargo handling is outside the scope of MSML e.g. if parts of the cargo shall be delivered to different shore bases. However, if different types of non-dangerous goods exist each type must be specified separately using this type. The *securing_mechanism* describes how cargo is secured.

6.50 complexType: damage_type

Element:

- type (1,1): string
- extent (1,1): string

- severity (1,1): string
- location (1,1): string
- reason_behind (0,1): string
- status (1,1): damage_status_type
- extension (0,unbounded): anyType

Attribute:

- -

This type is used for creating an overview of the damage. Note that the elements contain descriptive information and it is thus up to author to present adequate information. The type of damage e.g. oil pollution is specified in *type* and the extent, severity and location of the damage are contained in the corresponding element. The *reason_behind* might not always be clear but could be valuable for further limiting the consequences. The *status* element indicates the dynamic behaviour of the damage.

6.51 complexType: vessel_hindrance_type

Element:

- reason (1,unbounded): vessel_hindrance_reason_type
- date (1,1): date_and_time_type
- issued_by (1,1): address_information_type
- shore_base_identity (1,1): shore_base_identity_type
- country (1,1): string
- follow_up_activities (0,unbounded): string
- extension (0,unbounded): anyType

Attribute:

- -

This type is used for relating a reason why a vessel is not allowed to leave a shore base with additional information. Apart from information concerning decision authorisation an element *follow_up_activities* is defined. It could contain information, e.g. that fines shall be paid, but does not contain repair and maintenance information (this is handled by the corresponding perspective instead).

6.52 complexType: shore_base_service_type

Element:

- issued_by (1,1): address_information_type
- date (1,1): date_and_time_type
- type (1,1): string
- amount (1,1): string
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- element_identity (0,1): IDREF

This type is used for communicating “contract” information between vessel and shore base regarding a specific shore base service. Some service examples are firefighting, pilotage and pollution handling. Also loading and unloading are possible even though they are more indirectly related with safety aspects. The *type*, *amount* and *information* elements are used for describing the service. Reference is made to shore base service via attribute *element_identity* and a wrapper element *IDREF_wrapper_element_x* has to be defined for each element using this type.

6.53 complexType: vessel_id_type

Element:

- IMO_Number_exists (1,1): boolean
- IMO_Number (0,1): string
- name (1,1): item_history_type
- MMSI_number (0,1): string
- radio_call_sign (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This is the fundamental type for identification of the vessel. Since not all vessels have an IMO number a flag *IMO_Number_exists* is added. The name history is always included but *MMSI_number* and *radio_call_sign* are not strictly required, however, of high importance if existing.

6.54complexType: vessel_assistance_type

Element:

- other_vessel_id (0,1): vessel_id_type
- towing (1,1): boolean
- rescuing_people (1,1): boolean
- firefighting (1,1): boolean
- cargo_transfer (1,1): boolean
- pollution_support (1,1): boolean
- bilge_water_handling (1,1): boolean
- assistance_start (1,1): date_and_time_type
- assistance_end (1,1): date_and_time_type
- extension (0,unbounded): anyType

Attribute:

- -

This type is used for describing vessel to vessel assistance. The other vessel identity should be recorded, however, it might not be immediately possible e.g. due to a non-functioning radio. If vessel gets or gives assistance is given by the context of the corresponding element.

6.55complexType: route_type

6.55.1 complexType: anonymous_type_1

Element:

- start_waypoint_longitude (1,1): string, attribute units (1,1): units_type
- start_waypoint_latitude (1,1): string, attribute units (1,1): units_type
- start_waypoint_time (1,1): date_and_time_type
- end_waypoint_longitude (1,1): string, attribute units (1,1): units_type
- end_waypoint_latitude (1,1): string, attribute units (1,1): units_type
- end_waypoint_time (1,1): date_and_time_type
- extension (0,unbounded): anyType

Attribute:

- -

This type contains information for a leg i.e. start and end waypoints. If *start_waypoint_time* and *end_waypoint_time* represent passed times or estimated future times is given from the context of the corresponding element.

6.55.2 complexType: anonymous_type_2

Element:

- identity (1,1): shore_base_identity_type
- passage_time (1,1): date_and_time_type
- confirmed_passage (1,1): boolean
- confirmation_issued_by (0,1): string
- confirmation_time (0,1): date_and_time_type
- extension (0,unbounded): anyType

Attribute:

- -

This type contains information concerning the passage of a shore base, already passed or will pass in the future. An arrival-departure to a shore base is also considered a passage. If *confirmed_passage* contains TRUE this means that the shore base has recorded the passage and confirmed it to the vessel.

6.55.3 complexType

Element:

- plan_created (1,1): date_and_time_type
- plan_issued_by (1,1): string
- detailed_plan_information (0,1): detailed_information_type
- leg (0,unbounded): anonymous_type_1
- shore_bases_along_route (0,unbounded): anonymous_type_2
- extension (0,unbounded): anyType

Attribute:

- -

This type contains information regarding a planned route but is updated during the route. The synchronisation between shore bases and legs is made by matching time information.

6.56 complexType: derived_EncryptedType

A type *enc:EncryptedType* with attribute *abstract* set to TRUE is defined in [14]. A derived version *derived_EncryptedType* with attribute *abstract* set to FALSE is defined here in order to use the W3C encryption standard. The prefix *enc* is used to indicate the namespace of W3C encryption standard.

6.57 Element: MSML

Structure levels above: -

Element:

- administrative_support (1,1): element content
- security_support (1,1): element content
- data_model (1,1): element content
- perspective (1,1): element content
- extension (0,unbounded): anyType

Attribute:

- -

MSML element is the top element as required by [1] and is a wrapper element.

6.58 Element: administrative_support

6.58.1 complexType: anonymous_type_1

Element:

- MSML_instance_version (1,1): float
- MSML_instance_author (1,1): string
- MSML_instance_date (1,1): date_and_time_type
- reason_for_new_version (1,1): string
- description_of_change (1,unbounded): string
- extension (0,unbounded): anyType

Attribute:

- -

This type is used for revision history of the MSML instance, not MSML as such. The most recent version is the official version. The first version has version number 1.0 and successive versions have increasing values.

6.58.2 Element

Structure levels above: MSML

Element:

- MSML_version (1,1): float
- MSML_instance_identity (1,1): string
- instance_revision_history (1,unbounded): anonymous_type_1
- extension (0,unbounded): anyType

Attribute:

- element_identity_algorithm (1,1): element_identity_algorithm_type

The administrative support is used for setting the scope for handling MSML instances. A new revision of an MSML instance:

- *is not created* when encryption or decryption is used
- *is not created* when a digital signature is included or removed
- *is not created* at “successive information built up”. The guarantee that there are not two instances with the same revision and different contents is outside the definition of MSML
- *is normally not created* when a dynamic area of the data model changes (one area for vessel, one for shore base and one for relation between them).
- *can be created otherwise* but the actual decision is outside the definition of MSML

The revision history of MSML as such is handled outside the definition of MSML and information is not included in MSML instances.

6.59 Element: security_support

Structure levels above: MSML

Element:

- signed_element (0,unbounded): ds:SignatureType
- encrypted_element (0,unbounded): element content
- encrypted_element_exists (1,1): boolean

- `signed_element_exists` (1,1): boolean
- `extension` (0,unbounded): anyType

Attribute:

- -

The organisation W3C (see [12]) is responsible for the encryption and signature standards used for MSML. The prefix *ds* is used to indicate the namespace of W3C signature standard. The type *ds:SignatureType* is defined in [13]. The element *signed_element* makes it possible to put a signature on any other element within the MSML instance. The addressed element is specified via an attribute *URI*. The attribute is associated with element *ds:Reference* which is a subelement of *ds:SignedInfo* which is a subelement of *signed_element*. There are also other requirements and recommendations given in [13] in order to fulfill the intentions of the standard. These especially concern associated processing rules e.g. regarding the specification and interpretation of the *ds:Reference* element *URI* attribute. In *encrypted_element_exist* and *signed_element_exist* information is given that matches the encrypted and signed elements respectively. Thus it is possible to access this indication instead of actually searching for elements. If *encrypted_element_exists* contains TRUE it means that there exists dummy data and the real data is encrypted.

6.60Element: encrypted_element

Structure levels above: MSML, security_support

Element:

- `encrypted_data` (1,1): derived_EncryptedType

Attribute:

- `element_identity` (0,1): IDREF

Reference is made to data model element via attribute *element_identity*. There are also other requirements and recommendations given in [14] in order to fulfil the intentions of the standard. These especially concern associated processing rules and definition of encryption algorithms.

6.61Element: data_model

Structure levels above: MSML

Element:

- `vessel_static_data` (0,1): element content
- `vessel_dynamic_data` (0,1): element content
- `shore_base_static_data` (0,1): element content
- `shore_base_dynamic_data` (0,1): element content
- `vessel_shore_base_relation` (0,1): element content
- `extension` (0,unbounded): anyType

Attribute:

- -

This is a wrapper element. The reason for making these elements optional is that they then support successive information built-up.

6.62Element: vessel_static_data

Structure levels above: MSML, data_model

Element:

- `administration` (1,1): element content
- `certificate` (1,1): element content

- description (1,1): element content
- constituent (1,1): element content
- extension (0,unbounded): anyType

Attribute:

- -

This is a wrapper element.

6.63 Element: description

6.63.1 complexType: anonymous_type_1

Element:

- length_overall (1,1): float, attribute units (1,1): units_type
- length_between_pp (1,1): float, attribute units (1,1): units_type
- overall_breadth (1,1): float, attribute units (1,1): units_type
- depth (1,1): float, attribute units (1,1): units_type
- highest_fixed_point (1,1): float, attribute units (1,1): units_type
- extension (0,unbounded): anyType

Attribute:

- -

This type contains length measures that are of general interest when considering vessel movement.

6.63.2 complexType: anonymous_type_2

Element:

- deadweight_summer (1,1): float, attribute units (1,1): units_type
- deadweight_winter (1,1): float, attribute units (1,1): units_type
- lightweight (1,1): float, attribute units (1,1): units_type
- gross_register_tonnage (1,1): float, attribute units (1,1): units_type
- net_registered_tonnage (1,1): float, attribute units (1,1): units_type
- suez_tonnage (1,1): float, attribute units (1,1): units_type
- panama_tonnage (1,1): float, attribute units (1,1): units_type
- tons_per_centimeter (1,1): float, attribute units (1,1): units_type
- allowed_weight_of_bulk_cargo_above_deck (1,1): float, attribute units (1,1): units_type
- allowed_weight_of_containers_above_deck (1,1): float, attribute units (1,1): units_type
- extension (0,unbounded): anyType

Attribute:

- -

This type contains tonnage information that is of general interest when considering vessel cargo capacity.

6.63.3 complexType: anonymous_type_3_1

Element:

- initial_speed (1,1): float, attribute units (1,1): units_type
- forward_direction (1,1): boolean
- total_weight (1,1): float, attribute units (1,1): units_type
- stopping_time_in_minutes (1,1): float, attribute units (1,1): units_type
- stopping_distance (1,1): float, attribute units (1,1): units_type

- extension (0,unbounded): anyType

Attribute:

- -

This type makes it possible to record stopping time and distance when the vessel initially is moving with the speed given in *initial_speed* in forward or backward direction as given by the flag *forward_direction*. A factor that affects is total weight as given in *total_weight*.

6.63.4 complexType: anonymous_type_3

Element:

- maximum_speed_forward (1,1): float, attribute units (1,1): units_type
- maximum_speed_backward (1,1): float, attribute units (1,1): units_type
- service_speed (1,1): float, attribute units (1,1): units_type
- autonomy (1,1): float, attribute units (1,1): units_type
- turning_rate_at_service_speed (1,1): float, attribute units (1,1): units_type
- high_speed_craft (1,1): boolean
- maintain_control_below_5_Knots (1,1): boolean
- maintain_control_going_backwards (1,1): boolean
- stopping_performance (0,unbounded): anonymous_type_3_1
- extension (0,unbounded): anyType

Attribute:

- -

The element *autonomy* gives the maximum distance when going with service speed. The element *maintain_control_below_5_Knots* indicates if it is possible to have full control over the vessel at this speed or lower. The element *maintain_control_going_backwards* indicates if it is possible to have full control over the vessel when going backwards. By using *stopping_performance* it is possible to create a table of stopping time and distance as a function of initial speed and total weight.

6.63.5 complexType: anonymous_type_4

Element:

- main_engine_consumption (1,1): float, attribute units (1,1): units_type
- main_engine_fuel_type (1,1): string
- main_engine_power (1,1): float, attribute units (1,1): units_type
- propulsion_power (1,1): propulsion_power_type
- propulsion_principle (1,1): propulsion_principle_type
- unmanned_machinery_space_operation (1,1): boolean
- extension (0,unbounded): anyType

Attribute:

- -

This type contains information regarding engine performance and also if unmanned control is allowed as specified in *unmanned_machinery_space_operation*.

6.63.6 Element

Structure levels above: MSML, data_model, vessel_static_data

Element:

- vessel (1,1): vessel_type
- ro_ro (1,1): boolean
- length_measures (1,1): anonymous_type_1

- tonnage_measures (1,1): anonymous_type_2
- transport (1,1): anonymous_type_3
- engine (1,1): anonymous_type_4
- vessel_trading_limit (0,unbounded): string
- minimum_manning_officers (1,1): integer
- minimum_manning_ratings (1,1): integer
- extension (0,unbounded): anyType

Attribute:

- -

In some cases ro/ro can be used together with a general vessel type for specifying the kind of vessel. In other cases ro/ro is directly included in the specification of vessel type. In the latter case ro_ro could be used for consistency check. The element *vessel_trading_limit* informs if there are any restrictions where the vessel is allowed to go at sea. The minimum manning elements contain numbers not individuals and their roles. The purpose is just to get an overview and make it possible to compare with current manning.

6.64Element: administration

6.64.1 complexType: anonymous_type_1

Element:

- detailed_information (1,1): detailed_information_type
- date (1,1): date_and_time_type
- extension (0,unbounded): anyType

Attribute:

- -

This type contains information concerning registration of vessel.

6.64.2 complexType: anonymous_type_2

Element:

- shipyard (1,1): address_information_type
- hull_number (1,1): string
- date_launched (0,1): date_and_time_type
- date_delivered (0,1): date_and_time_type
- extension (0,unbounded): anyType

Attribute:

- -

This type contains information that makes it possible to find information concerning the building of the vessel. The shipyard could have detailed information concerning safety aspects onboard and also complete control over vessel constituents.

6.64.3 complexType: anonymous_type_3

Element:

- owner (1,1): address_history_type
- number_of_ships_owned (1,1): integer
- international_ship_managers_association_member (1,1): boolean
- extension (0,unbounded): anyType

Attribute:

- -

This type makes it possible to identify the owner of the vessel and to record a few basic facts concerning him. The purpose is to be able to contact him when needed e.g. at an emergency.

6.64.4 complexType: anonymous_type_4

Element:

- commercial_operator (1,1): address_history_type
- number_of_ships_operated (1,1): integer
- international_ship_managers_association_member (1,1): boolean
- bareboat_charterer (1,1): boolean
- voyage_charterer (1,1): boolean
- time_charterer (1,1): boolean
- extension (0,unbounded): anyType

Attribute:

- -

This type makes it possible to identify the commercial operator of the vessel and to record a few basic facts concerning him. Information for describing the type of charter is included. Note that this is considered as (relatively) static data since changes are assumed to occur more seldom than the different transports. The purpose of this type is to be able to contact the commercial operator when needed e.g. at an emergency.

6.64.5 complexType: anonymous_type_5

Element:

- technical_operator (1,1): address_history_type
- number_of_ships_operated (1,1): integer
- international_ship_managers_association_member (1,1): boolean
- emergency_contact (1,1): address_information_type
- extension (0,unbounded): anyType

Attribute:

- -

This type makes it possible to identify the technical operator of the vessel and to record a few basic facts concerning him. The purpose is to be able to contact the technical operator when needed e.g. at an emergency.

6.64.6 complexType: anonymous_type_6

Element:

- sister_vessel_id (1,1): vessel_id_type
- building_data (1,1): anonymous_type_2
- extension (0,unbounded): anyType

Attribute:

- -

This type is used for making it possible to identify sister vessels and thus making comparisons possible. Information concerning this vessel could be transferred to sister ships and vice versa. The main purpose is to detect possible weaknesses in advance (for the not affected vessels) regarding construction and/or repair and maintenance.

6.64.7 complexType: anonymous_type_7

Element:

- hull_insurance (1,1): address_history_type
- machinery_insurance (1,1): address_history_type
- protection_and_indemnity (1,1): address_history_type
- cargo_insurance (1,1): address_history_type
- passenger_insurance (1,1): address_history_type
- extension (0,unbounded): anyType

Attribute:

- -

This type contains contact information for insurance companies regarding different aspects of the vessel. Note that there could be (or normally is) an overlap between *protection_and_indemnity* and some of the other elements.

6.64.8 Element

Structure levels above: MSML, data_model, vessel_static_data

Element:

- vessel_id(1,1): vessel_id_type
- flag_state (1,1): item_history_type
- vessel_contact (0,1): address_information_type
- SATCOM_number (0,1): string
- INMARSAT_number (0,1): string
- registration (1,1): anonymous_type_1
- building_data (1,1): anonymous_type_2
- ship_operating_base (1,1): address_history_type
- owner_data (1,1): anonymous_type_3
- commercial_operator_data (1,1): anonymous_type_4
- technical_operator_data (1,1): anonymous_type_5
- sister_ship (0,unbounded): anonymous_type_6
- insurance (1,1): anonymous_type_7
- extension (0,unbounded): anyType

Attribute:

- -

The vessel as such could also contain contact information, e.g. email address, and this is described by *vessel_contact*. Satellite identification information is contained in *SATCOM_number* and *INMARSAT_number*.

6.65Element: certificate

6.65.1 complexType: anonymous_type_1

Element:

- issuing_society (1,1): address_history_type
- class (1,1): item_history_type
- IACS_member (1,1): boolean
- issue_date (1,1): date_and_time_type
- expiration_date (1,1): date_and_time_type

- `reason_if_not_valid (0,1): string`
- `extension (0,unbounded): anyType`

Attribute:

- -

This type is used exclusively for class. The membership of International Association of Classification Societies (IACS), flagged in *IACS_member* is a quality indication for the classification process as such. Note that *issuing_society* and *class* are not required to have synchronised histories. If the class certificate is no longer valid it is explained why in *reason_if_not_valid*. Since e.g. repair and maintenance can affect class it is necessary to keep the history of changes as shown by using types *address_history_type* and *item_history_type*.

6.65.2 complexType: anonymous_type_2

Element:

- `certificate (1,1): certificate_type`
- `issued_by (1,1): address_history_type`
- `certificate_validity (1,1): item_history_type`
- `issue_date (1,1): date_and_time_type`
- `expiration_date (1,1): date_and_time_type`
- `reason_if_not_valid (0,1): string`
- `class_for_which_certificate_is_valid (0,1): string`
- `extension (0,unbounded): anyType`

Attribute:

- -

This type is used for certificates onboard except class. From this type it is possible to check the status and the quality of work behind the certificate (by contacting the responsible as defined in *issued_by*). Note that *issued_by* and *certificate_validity* are not required to have synchronised histories. If certificate is related to a specific class it is specified in *class_for_which_certificate_is_valid*. If the certificate is no longer valid it is explained why in *reason_if_not_valid*. The type *item_history_type* is used for describing the history of valid and not valid status of the certificate i.e. only these two values are possible for the item in *item_history_type*. Since repair and maintenance and other reasons can affect the validity of the certificate it is necessary to keep the history of changes here as shown by using types *address_history_type* and *item_history_type*. Note that for a not valid certificate the corresponding element is *not* removed since it is important to record why this has happened.

6.65.3 Element

Structure levels above: MSML, data_model, vessel_static_data

Element:

- `class_exists (1,1): boolean`
- `classification_society (0,1): anonymous_type_1`
- `number_of_certificates (1,1): integer`
- `certificate (0,unbounded): anonymous_type_2`
- `extension (0,unbounded): anyType`

Attribute:

- -

The purpose is to specify class and list all certificates and this idea is implemented using the mandatory flags *class_exists* and *number_of_certificates*.

6.66Element: constituent

Structure levels above: MSML, data_model, vessel_static_data

Element:

- hull (1,1): element content
- safety_equipment (1,1): element content
- cargo_passenger_equipment (1,1): element content
- extension (0,unbounded): anyType

Attribute:

- -

This is a wrapper element.

6.67Element: hull

Structure levels above: MSML, data_model, vessel_static_data, constituent

Element:

- mooring (1,1): element content
- network (1,1): element content
- construction (1,1): element content
- propulsion (1,1): element content
- extension (0,unbounded): anyType

Attribute:

- -

This is a wrapper element.

6.68Element: mooring

Structure levels above: MSML, data_model, vessel_static_data, constituent, hull

Element:

- anchor (1,unbounded): wire_rope_type
- mooring_wire (1,unbounded): wire_rope_type
- extension (0,unbounded): anyType

Attribute:

- -

This element contains information concerning mooring capacity.

6.69Element: network

Structure levels above: MSML, data_model, vessel_static_data, constituent, hull

Element:

- computer_network (0,unbounded): network_type
- electrical_network (1,unbounded): network_type
- pipe_network (0,unbounded): network_type
- sensor_network (0,unbounded): network_type
- extension (0,unbounded): anyType

Attribute:

- -

At least some kind of electric network is assumed to exist. The partition principle of e.g. computer networks is outside the scope of MSML, however, common mode faults are considered via the type

network_type. The element *sensor_network* contains information regarding sensors used for a specific purpose e.g. fire detection.

6.70 Element: construction

6.70.1 complexType: anonymous_type_1_1

Element:

- condition (1,1): string
- quality_under_condition (1,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type describes evacuation route quality for a specific condition such as smoke, fire, bilge water, list etc.

6.70.2 complexType: anonymous_type_1

Element:

- identity (1,1): string
- route_description (1,1): string
- capacity (1,1): string
- route_quality (0,unbounded): anonymous_type_1_1
- alternative_route_identity (0,unbounded): string
- extension (0,unbounded): anyType

Attribute:

- -

The type is used for listing evacuation routes so that they could be referred to, via *identity*, and described. The element *alternative_route_identity* contains the identity of an alternative route with the same purpose but with possibly different qualities.

6.70.3 Element

Structure levels above: MSML, data_model, vessel_static_data, constituent, hull

Element:

- double_deck (1,1): boolean
- double_side (1,1): boolean
- double_bottom (1,1): boolean
- segregated_ballast_tank (1,1): boolean
- strengthened_for_heavy_cargo (1,1): boolean
- hull_material (1,1): hull_material_type
- ballast_tank (0,unbounded): room_type
- cargo_hold (0,unbounded): room_type
- bulkhead (0,unbounded): room_type
- hatch_cover (0,unbounded): hull_mechanical_securing_type
- deck_machinery_mounting (0,unbounded): hull_mechanical_securing_type
- restricted_access_areas (0,unbounded): room_type
- evacuation_route (0,unbounded): anonymous_type_1
- extension (0,unbounded): anyType

Attribute:

- -

If an element starting with “double_” contains the value FALSE it denotes “single”. The elements included are related to cargo capacity and handling and potential safety aspects. The purpose is not to partition the vessel into construction elements and characterise them individually since this will at a too detailed level of information and, further, the information should be available at the building shipyard. Also the status of such element is outside the scope since details could be found in inspection protocols and these are instead referenced in the MSML perspective *inspection*.

6.71Element: propulsion

Structure levels above: MSML, data_model, vessel_static_data, constituent, hull

Element:

- propulsion_engine (1,unbounded): engine_type
- side_thruster (0,unbounded): engine_type
- steering_gear (1,unbounded): equipment_type
- antiroll_stabiliser (0,unbounded): equipment_type
- extension (0,unbounded): anyType

Attribute:

- -

This element contains the means for moving the vessel in the desired direction and in a stable way.

6.72Element: safety_equipment

Structure levels above: MSML, data_model, vessel_static_data, constituent

Element:

- communication_equipment (1,1): element content
- navigation_equipment (1,1): element content
- supervision_equipment (1,1): element content
- emergency_equipment (1,1): element content
- extension (0,unbounded): anyType

Attribute:

- -

This is a wrapper element.

6.73Element: communication_equipment

Structure levels above: MSML, data_model, vessel_static_data, constituent, safety_equipment

Element:

- AIS (0,unbounded): equipment_type
- daylight_signalling_lamp (0,unbounded): equipment_type
- email (0,unbounded): equipment_type
- emergency_radio (0,unbounded): equipment_type
- radio_direction_finder (0,unbounded): equipment_type
- emergency_radio_telephones (0,unbounded): equipment_type
- fax (0,unbounded): equipment_type
- GMDSS_radio_equipment (0,unbounded): equipment_type
- helicopter_homing_signal (0,unbounded): equipment_type
- internal_telephone (0,unbounded): equipment_type
- main_radio (0,unbounded): equipment_type

- mobile_telephone (0,unbounded): equipment_type
- radar_transponders (0,unbounded): equipment_type
- radio_telephone (0,unbounded): equipment_type
- radiotelegraphy_equipment (0,unbounded): equipment_type
- SAR_Transponder (0,unbounded): equipment_type
- satellite_communication (0,unbounded): equipment_type
- telex (0,unbounded): equipment_type
- walkie_talkies (0,unbounded): equipment_type
- web (0,unbounded): equipment_type
- extension (0,unbounded): anyType

Attribute:

- -

This element defines communication means mainly to shore base but also internally for a vessel and between vessels.

6.74Element: navigation_equipment

Structure levels above: MSML, data_model, vessel_static_data, constituent, safety_equipment

Element:

- autopilot (0,unbounded): equipment_type
- course_monitor (0,unbounded): equipment_type
- echo_sounder (0,unbounded): equipment_type
- gyro_compass (0,unbounded): equipment_type
- gyro_repeater (0,unbounded): equipment_type
- magnetic_steering_compass (0,unbounded): equipment_type
- radar (0,unbounded): equipment_type
- rate_of_turn_indicator (0,unbounded): equipment_type
- satellite_navigation (0,unbounded): equipment_type
- speed_distance_log (0,unbounded): equipment_type
- extension (0,unbounded): anyType

Attribute:

- -

This element defines navigation means and means for supporting navigation. Note that this is from a safety point of view, e.g. for understanding the navigation capabilities in a crisis situation, and is not intended to replace information regarding normal navigation.

6.75Element: supervision_equipment

Structure levels above: MSML, data_model, vessel_static_data, constituent, safety_equipment

Element:

- cargo_supervision (0,unbounded): equipment_type
- fire_detection_equipment (0,unbounded): equipment_type
- gas_detection (0,unbounded): equipment_type
- intrusion_detection (0,unbounded): equipment_type
- rudder_angle_indicator (0,unbounded): equipment_type
- video_supervision (0,unbounded): equipment_type
- voyage_data_recorder (0,unbounded): equipment_type
- power_supply_supervision (0,unbounded): equipment_type
- emergency_power_supply_supervision (0,unbounded): equipment_type
- pollution_control (0,unbounded): equipment_type

- bilge_water_detection_equipment (0,unbounded): equipment_type
- extension (0,unbounded): anyType

Attribute:

- -

Supervision equipment is extra equipment used for supervising conditions onboard and the immediate surroundings of the vessel. For understanding supervision capability it is important to know:

- location
- purpose, extent and scope
- vulnerability and dependence on other items e.g. will supervision work in case of power loss
- quality and accuracy
- functional status
- detailed information

of the equipment and these are included in the type *equipment_type*.

6.76Element: emergency_equipment

Structure levels above: MSML, data_model, vessel_static_data, constituent, safety_equipment

Element:

- bilge_pump (0,unbounded): equipment_type
- embarkation_arrangement (0,unbounded): equipment_type
- emergency_electric_power (0,unbounded): equipment_type
- emergency_lighting (0,unbounded): equipment_type
- emergency_rudder (0,unbounded): equipment_type
- emergency_stop_of_fuel (0,unbounded): equipment_type
- emergency_towing (0,unbounded): equipment_type
- EPIRB (0,unbounded): equipment_type
- fire_dampers (0,unbounded): equipment_type
- fire_door (0,unbounded): equipment_type
- fire_extinguisher (0,unbounded): equipment_type
- firefighting_system (0,unbounded): equipment_type
- fire_pump (0,unbounded): equipment_type
- flare_and_rocket (0,unbounded): equipment_type
- life_raft (0,unbounded): equipment_type
- lifeboat (0,unbounded): equipment_type
- lifeboat_engine (0,unbounded):engine_type
- lifebuoy (0,unbounded): equipment_type
- lifejacket (0,unbounded): equipment_type
- navigation_and_search_light (0,unbounded): equipment_type
- sound_signal (0,unbounded): equipment_type
- medical_equipment (0,unbounded): equipment_type
- personal_protective_outfit (0,unbounded): equipment_type
- watertight_bulkhead_doors (0,unbounded): equipment_type
- extension (0,unbounded): anyType

Attribute:

- -

This element contains equipment that is used for:

- signalling an emergency and,
- limiting the criticality of an emergency and,
- limiting the extent of an emergency

The purpose is to have an immediately available picture of the total vessel capability. The actual resources will depend on the current status and not defined in this type.

6.77Element: cargo_passenger_equipment

Structure levels above: MSML, data_model, vessel_static_data, constituent, safety_equipment

Element:

- cargo_cooling_equipment (0,unbounded): equipment_type
- cargo_heating_equipment (0,unbounded): equipment_type
- cargo_hold (0,unbounded): equipment_type
- inert_gas_system (0,unbounded): equipment_type
- oil_water_separation (0,unbounded): equipment_type
- cargo_tank_venting_arrangement (0,unbounded): equipment_type
- crane (0,unbounded): equipment_type
- derrick (0,unbounded): equipment_type
- pump (0,unbounded): equipment_type
- tank (0,unbounded): equipment_type
- winch (0,unbounded): equipment_type
- extension (0,unbounded): anyType

Attribute:

- -

This element describes equipment that is needed for loading, unloading and maintaining cargo onboard.

6.78Element: vessel_dynamic_data

Structure levels above: MSML, data_model

Element:

- crew (1,1): element content
- route_at_sea (1,1): element content
- cargo_passenger (1,1): element content
- status (1,1): element content
- previous_tasks (1,1): element content
- extension (0,unbounded): anyType

Attribute:

- -

This is a wrapper element.

6.79Element: crew

Structure levels above: MSML, data_model, vessel_dynamic_data

Element:

- cargo_safety (1,unbounded): crew_group_capability_type
- firefighting (1,unbounded): crew_group_capability_type
- medical_care (1,unbounded): crew_group_capability_type
- rescue (1,unbounded): crew_group_capability_type
- passenger_safety (1,unbounded): crew_group_capability_type
- personal_safety (1,unbounded): crew_group_capability_type
- pollution_control (1,unbounded): crew_group_capability_type
- dangerous_goods (1,unbounded): crew_group_capability_type

- vessel_common_language (1,1): string
- officers_common_language (1,1): string
- ratings_common_language (1,1): string
- english_understood (1,1): boolean
- muster_list (1,1): detailed_information_type
- ratings_manning_agent (1,1): address_information_type
- officers_manning_agent (1,1): address_information_type
- master_manning_agent (1,1): address_information_type
- change_of_officers (0,1): item_history_type
- change_of_ratings (0,1): item_history_type
- extension (0,unbounded): anyType

Attribute:

- -

Handling individuals is at a too detailed level for MSML. Instead groups are handled with information where to find detailed individual information (via *muster_list*). Further information is also possible via manning agents given in *ratings_manning_agent*, *officers_manning_agent* and *master_manning_agent*. The elements *change_of_officers* and *change_of_ratings* are used for documenting how changes of crew have been made. It is important to document if whole group capabilities, as defined in this element, have been exchanged and if adequate replacements are onboard.

6.80Element: route_at_sea

6.80.1 complexType: anonymous_type_1

Element:

- date (1,1): date_and_time_type
- course (1,1): float, attribute units (1,1): units_type
- speed (1,1): float, attribute units (1,1): units_type
- longitude (1,1): string, attribute units (1,1): units_type
- latitude (1,1): string, attribute units (1,1): units_type
- extension (0,unbounded): anyType

Attribute:

- -

This type contains data for vessel position.

6.80.2 Element

Structure levels above: MSML, data_model, vessel_dynamic_data

Element:

- sea_condition_forecast (0,unbounded): environmental_condition_type
- ice_condition_forecast (0,unbounded): environmental_condition_type
- weather_forecast (0,unbounded): environmental_condition_type
- tidal_stream_forecast (0,unbounded): environmental_condition_type
- drifting_object (0,unbounded): environmental_condition_type
- current_position (1,1): anonymous_type_1
- previous_route (1,1): route_type
- coming_route (1,1): route_type
- extension (0,unbounded): anyType

Attribute:

- -

The purpose of this element is not to replace normal navigation support. The reason is instead to supply information needed for handling safety aspects. The information concerns areas outside shore base control. Drifting objects are treated as environmental condition since they cannot be controlled. The separation into *previous_route* and *coming_route* makes it possible to separate actual and estimated times even though both could refer to the same plan.

6.81Element: cargo_passenger

6.81.1 complexType: anonymous_type_1

Element:

- common_language (0,1): string
- english_understood (1,1): boolean
- nationality (1,1): string
- number_of_persons (1,1): integer
- number_of_indisposed_persons (1,1): integer
- reason_for_indisposed (0,unbounded): string
- consequences_due_to_indisposed_persons (0,unbounded): string
- extension (0,unbounded): anyType

Attribute:

- -

This type makes it possible to define groups of passengers mainly with respect to common language. A common language is important in an emergency situation since it is necessary to inform passengers in an effective way. An example of consequences due to indisposed persons could be that extra medical care is needed.

6.81.2 Element

Structure levels above: MSML, data_model, vessel_dynamic_data

Element:

- passenger_group (0,unbounded): anonymous_type_1
- passenger_list (0,1): detailed_information_type
- non_dangerous_cargo_transport (0,unbounded): non_dangerous_goods_cargo_type
- dangerous_cargo_transport (0,unbounded): dangerous_goods_cargo_type
- extension (0,unbounded): anyType

Attribute:

- -

Any combination of cargo and passengers can be defined even an “empty” vessel with no cargo and no passengers. The groups defined here are not official; there are no corresponding group definitions in the passenger list.

6.82Element: status

Structure levels above: MSML, data_model, vessel_dynamic_data

Element:

- constituent_status (1,1): element content
- emergency_status (1,1): element content
- document_status (1,1): element content

- overall_status (1,1): element content
- extension (0,unbounded): anyType

Attribute:

- -

This is a wrapper element.

6.83Element: constituent_status

6.83.1 complexType: anonymous_type_1_1

Element:

- tested (1,1): date_and_time_type
- tested_by (1,1): string
- tested_onboard (1,1): boolean
- test_passed (1,1): boolean
- test_documentation (1,1): detailed_information_type
- extension (0,unbounded): anyType

Attribute:

- -

This type is used for recording and referencing test information for a constituent.

6.83.2 complexType: anonymous_type_1

Element:

- function_ok (1,1): boolean
- function_loss (0,unbounded): string
- quality_ok (1,1): boolean
- degraded_quality (0,unbounded): string
- constituent_capable_of_generating_warning (1,1): boolean
- active_warning (0,unbounded): item_history_type
- constituent_capable_of_generating_alarm (1,1): boolean
- active_alarm (0,unbounded): item_history_type
- most_recent_test (0,1): anonymous_type_1_1
- extension (0,unbounded): anyType

Attribute:

- element_identity (0,1): IDREF

The element *function_ok* is used for indicating that the piece of equipment can be considered functioning while *quality_ok* indicates if properties are adequate. One example of correct function but inadequate property is a functioning pump but with degraded pumping capacity. Reference is made to vessel constituent via attribute *element_identity* and a wrapper element *IDREF_wrapper_element_x* has to be defined for each element using this type.

6.83.3 complexType: anonymous_type_2

Element:

- constituent (1,1): anonymous_type_1

Attribute:

- -

6.83.4 Element

Structure levels above: MSML, data_model, vessel_dynamic_data, status

Element:

- IDREF_wrapper_element_1 (0,unbounded): anonymous_type_2
- extension (0,unbounded): anyType

Attribute:

- -

It is not a requirement that each constituent shall have a corresponding *constituent* status element since it might not be applicable or significant for all kinds of equipment. However, the recommendation is to be as complete as reasonably possible since constituent status is of crucial importance for handling safety aspects.

6.84Element: emergency_status

Structure levels above: MSML, data_model, vessel_dynamic_data, status

Element:

- bilge_water (0,unbounded): damage_type
- fire (0,unbounded): damage_type
- explosion (0,unbounded): damage_type
- pollution (0,unbounded): damage_type
- collision (0,unbounded): damage_type
- grounding (0,unbounded): damage_type
- hit_by_wave (0,unbounded): damage_type
- other_reason (0,unbounded): damage_type
- vessel_under_attack (1,1): boolean
- vessel_hijacked (1,1): boolean
- vessel_gives_assistance (0,unbounded): vessel_assistance_type
- vessel_gets_assistance (0,unbounded): vessel_assistance_type
- supply_shortage (0,unbounded): supply_shortage_type
- manoeuvrability (0,unbounded): manoeuvrability_type
- extension (0,unbounded): anyType

Attribute:

- -

There is a possibility to include more than one emergency reason and also of different types e.g. two fire locations and one explosion. For element *other_reason* the actual reason is recorded using *damage_type*. More than two vessels can be involved in vessel-to-vessel assistance. The element *vessel_hijacked* contains TRUE when master is not in control.

6.85Element: document_status

6.85.1 complexType: anonymous_type_1

Element:

- title (1,1): manual_plan_type
- issued_by (1,1): address_information_type
- issue_date (1,1): date_and_time_type
- manual_plan (1,1): detailed_information_type
- extension (0,unbounded): anyType

Attribute:

- -

This type is used for manual and plan as defined by *manual_plan_type*.

6.85.2 complexType: anonymous_type_2

Element:

- title (1,1): record_type
- issued_by (1,1): address_information_type
- issue_date (1,1): date_and_time_type
- record (1,1): detailed_information_type
- extension (0,unbounded): anyType

Attribute:

- -

This type is used for record as defined by *record_type*.

6.85.3 Element

Structure levels above: MSML, data_model, vessel_dynamic_data, status

Element:

- manual_plan_status (0,unbounded): anonymous_type_1
- record_status (0,unbounded): anonymous_type_2
- extension (0,unbounded): anyType

Attribute:

- -

This element identifies plans, manuals and records an their availability onboard.

6.86Element: overall_status

Structure levels above: MSML, data_model, vessel_dynamic_data, status

Element:

- monitored_radio_channel (0,unbounded): string
- current_draught (1,1): float, attribute units (1,1): units_type
- helicopter_landing_enabled (1,1): boolean
- Paris_MOU_target_factor (0,1): float
- Paris_MOU_target_factor_algorithm_version (0,1): string
- weapon_onboard (0,unbounded): equipment_type
- medical_supply (0,unbounded): equipment_type
- ballast (0,1): item_history_type
- pollutive_substances_except_cargo (0, unbounded): item_history_type
- extension (0,unbounded): anyType

Attribute:

- -

This element contains information that has an overall scope and importance. The algorithm for computing target factor is not included in MSML but the algorithm version (if used) must be specified. For *weapon_onboard* the amount of ammunition is specified using *equipment_type*. For *ballast* and *pollutive_substances_except_cargo* history types are used since they could make it possible to detect not allowed disposals. This is done by going through the history and comparing with current values.

6.87Element: previous_tasks

6.87.1 complexType: anonymous_type_1

Element:

- dangerous_goods_type_reference(1,1): dangerous_goods_type_reference_type
- dangerous_goods_type_according_to_reference(1,1): string
- weight (1,1): float, attribute units (1,1): units_type
- cargo_loaded (1,1): date_and_time_type
- cargo_unloaded (1,1): date_and_time_type
- extension (0,unbounded): anyType

Attribute:

- -

This type contains information concerning previous dangerous goods of a specific type.

6.87.2 complexType: anonymous_type_2

Element:

- cargo_type_reference(1,1): non_dangerous_cargo_reference_type
- cargo_type (1,1): MSML_non_dangerous_cargo_type
- weight (1,1): float, attribute units (1,1): units_type
- cargo_loaded (1,1): date_and_time_type
- cargo_unloaded (1,1): date_and_time_type
- extension (0,unbounded): anyType

Attribute:

- -

This type contains information concerning previous non-dangerous goods of a specific type.

6.87.3 Element

Structure levels above: MSML, data_model, vessel_dynamic_data

Element:

- previous_dangerous_cargos (0,3): anonymous_type_1
- previous_non_dangerous_cargos (0,3): anonymous_type_2
- shore_base_access_denied (0,unbounded): vessel_hindrance_type
- detention_activated (0,unbounded): vessel_hindrance_type
- detention_cancelled (0,unbounded): vessel_hindrance_type
- extension (0,unbounded): anyType

Attribute:

- -

It is only required to store information concerning the last three cargo transports (they do not all exist for a new vessel) independent if they are dangerous or non-dangerous. A cargo transport is here defined as a *new loading of cargo* i.e. the previous loaded cargo does not have to be unloaded first. The reason is to see if any effects could remain or have their causes from previous transports. Important to record is also if shore base access has been denied and to record a list of previous detentions and their cancellations. The corresponding shore base identities are stored using *vessel_hindrance_type*.

6.88Element: shore_base_static_data

Structure levels above: MSML, data_model

Element:

- description (1,1): element content
- administration (1,1): element content
- service (1,1): element content
- extension (0,unbounded): anyType

Attribute:

- -

This is a wrapper element.

6.89Element: description

6.89.1 complexType: anonymous_type_1

Element:

- certificate (1,1): string
- issued_by (1,1): address_history_type
- certificate_validity (1,1): item_history_type
- issue_date (1,1): date_and_time_type
- expiration_date (1,1): date_and_time_type
- reason_if_not_valid (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

Even though certificates for shore bases are not common today they will probably be required under certain conditions in the future and this type is a preparation for this. From this type it is possible to check the status and the quality of work behind the certificate (by contacting the responsible as defined in *issued_by*). Note that *issued_by* and *certificate_validity* are not required to have synchronised histories. If the certificate is no longer valid it is explained why in *reason_if_not_valid*. The type *item_history_type* is used for describing the history of valid and not valid status of the certificate i.e. only these two values are possible for the item in *item_history_type*.

6.89.2 Element

Structure levels above: MSML, data_model, shore_base_static_data

Element:

- shore_base (1,1): shore_base_type
- maximum_tide_height_difference (1,1): float, attribute units (1,1): units_type
- certificate (0,unbounded): anonymous_type_1
- extension (0,unbounded): anyType

Attribute:

- -

This element contains general information not directly related to services.

6.90Element: administration

Structure levels above: MSML, data_model, shore_base_static_data

Element:

- identity (1,1): shore_base_identity_type
- radio_call_sign (0,1): string
- UNCTAD_code (0,1): string
- ambulance (0,1): address_information_type
- coast_guard (0,1): address_information_type
- fire_station (0,1): address_information_type
- harbour_master_office (0,1): address_information_type
- health_authority (0,1): address_information_type
- pilotage (0,1): address_information_type
- police (0,1): address_information_type
- pollution_control (0,1): address_information_type
- port_state_control_office (0,1): address_information_type
- repair_and_maintenance (0,1): address_information_type
- shore_base_main_contact (0,1): address_information_type
- stevedore_office (0,1): address_information_type
- surveyor_office (0,1): address_information_type
- tugboat_service (0,1): address_information_type
- waste_disposal (0,1): address_information_type
- watchmen_office (0,1): address_information_type
- extension (0,unbounded): anyType

Attribute:

- -

This element contains identification and contact information for the shore base.

6.91Element: service

6.91.1 complexType: anonymous_type_1

Element:

- number_of_firefighting_vessels (1,1): integer
- speed_of_firefighting_vessels (1,1): float, attribute units (1,1): units_type
- capacity_of_firefighting_vessels (1,1): string
- number_of_firefighting_vehicles (1,1): integer
- capacity_of_firefighting_vehicles (1,1): string
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

To identify individual firefighting vessels and vehicles is at a too low level for MSML, however, the total number and capacity are of high interest. The capacity is expressed as a string and could include amount of water/second, length of jet of water etc. The speed of the vessels makes it possible to calculate time to service start. The element *information* could be used for further data.

6.91.2 complexType: anonymous_type_2

Element:

- number_of_ambulances (1,1): integer
- capacity_of_ambulances (1,1): string
- medical_equipment (1,1): string
- information (0,1): string

- extension (0,unbounded): anyType

Attribute:

- -

To identify individual ambulances is at a too low level for MSML, however, the total number and capacity are of high interest. The capacity is expressed as a string and could include number of persons, specific equipment etc. The element *information* could be used for further data.

6.91.3 complexType: anonymous_type_3

Element:

- number_of_pilots (1,1): integer
- number_of_pilot_boats (1,1): integer
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

To identify individual pilots is at a too low level for MSML, however, the total number of pilots and pilot boats are of high interest. The element *information* could be used for further data.

6.91.4 complexType: anonymous_type_4

Element:

- number_of_tugboats (1,1): integer
- maximum_towing_capacity (1,1): float, attribute units (1,1): units_type
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

To identify individual tugboats is at a too low level for MSML, however, the total number and maximum capacity are of high interest. The maximum towing capacity is expressed according to the attribute *units_type*. The element *information* could be used for further data.

6.91.5 complexType: anonymous_type_5

Element:

- number_of_drydocks (1,1): integer
- maximum_drydock_capacity (1,1): string
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

To identify individual drydocks is at a too low level for MSML, however, the total number and maximum capacity are of high interest. The maximum capacity is expressed as a string and could include length, breadth, weight etc. The element *information* could be used for further data.

6.91.6 complexType: anonymous_type_6_1

Element:

- pollution_type (1,1): string

- pollution_capacity (1,1): string
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type is used for handling a specific kind of pollution. The capacity is expressed as a string and could include volume/second, weight/second etc. The element *information* could be used for further data.

6.91.7 complexType: anonymous_type_6

Element:

- number_of_pollution_control_vessels (1,1): integer
- speed_of_pollution_control_vessels (1,1): float, attribute units (1,1): units_type
- pollution_capability (0,unbounded): anonymous_type_6_1
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

Pollution is assumed to occur at sea and reachable from the shore base. The speed of the vessels makes it possible to calculate time to service start. Pollution occurring ashore at the shore base is outside the scope of MSML. To identify individual pollution control vessels is at a too low level for MSML, however, the total number and capability are of high interest. The element *information* could be used for further data.

6.91.8 complexType: anonymous_type_7_1

Element:

- waste (1,1): waste_type
- waste_capacity (1,1): string
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type is used for handling a specific kind of waste. The capacity is expressed as a string and could include volume/second, weight/second etc. The element *information* could be used for further data.

6.91.9 complexType: anonymous_type_7

Element:

- waste_capability (0,unbounded): anonymous_type_7_1
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

The handling of waste is assumed to occur ashore. The element *information* could be used for further data.

6.91.10 complexType: anonymous_type_8

Element:

- health_inspection_possible (1,1): boolean
- port_state_control_possible (1,1): boolean
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type contains the inspection capability of the shore base. The element *information* could be used for further data.

6.91.11 complexType: anonymous_type_9_1

Element:

- dangerous_goods(1,1): dangerous_goods_type
- dangerous_goods_capacity (1,1): string
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type is used for handling a specific kind of dangerous goods. The capacity is expressed as a string and could include volume/second, weight/second etc. The element *information* could be used for further data.

6.91.12 complexType: anonymous_type_9

Element:

- dangerous_goods_capability (0,unbounded): anonymous_type_9_1
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

The handling of dangerous goods is assumed to occur ashore. The element *information* could be used for further data.

6.91.13 complexType: anonymous_type_10

Element:

- SAR_equipment (0,unbounded): equipment_type
- number_of_rescue_boats (1,1): integer
- speed_of_rescue_boats (1,1): float, attribute units (1,1): units_type
- total_rescue_boat_capacity (1,1): string
- number_of_helicopters (1,1): integer
- speed_of_helicopters (1,1): float, attribute units (1,1): units_type
- total_helicopter_capacity (1,1): string
- number_of_planes (1,1): integer
- speed_of_planes (1,1): float, attribute units (1,1): units_type
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type contains the search and rescue capability of the shore base. The speed of vessels, helicopters and planes make it possible to calculate time to service start. The elements *total_rescue_boat_capacity* and *total_helicopter_capacity* define number of persons that could be rescued for *one transport* with all rescue boats and all helicopters respectively. The element *information* could be used for further data.

6.91.14 complexType: anonymous_type_11

Element:

- VTS (0,unbounded): equipment_type
- VTMIS (0,unbounded): equipment_type
- supported_radio_channel (0,unbounded): string
- radio_direction_finder (0,unbounded): equipment_type
- information (0,1): string
- extension (0,unbounded): anyType

Attribute:

- -

VTS and VTMIS contain information regarding equipment used for controlling vessel traffic. In this way the vessel could get a picture of the capability and quality of shore base navigation support. The element *information* could be used for further data.

6.91.15 Element

Structure levels above: MSML, data_model, shore_base_static_data

Element:

- firefighting_capability (1,1): anonymous_type_1
- medical_capability (1,1): anonymous_type_2
- pilotage_capability (1,1): anonymous_type_3
- tugging_capability (1,1): anonymous_type_4
- repair_and_maintenance_capability (1,1): anonymous_type_5
- pollution_control_capability (1,1): anonymous_type_6
- waste_disposal_capability (1,1): anonymous_type_7
- inspection_capability (1,1): anonymous_type_8
- dangerous_goods_capability (1,1): anonymous_type_9
- search_and_rescue_capability (1,1): anonymous_type_10
- navigation_and_communication_support (1,1): anonymous_type_11
- extension (0,unbounded): anyType

Attribute:

- -

The importance of the shore base is the capability and quality of service it can give. Since different types of services require different amount of information and also will develop differently each has a specific anonymous type as shown above. Handling not dangerous goods is not a safety issue and thus not included above.

6.92Element: shore_base_dynamic_data

Structure levels above: MSML, data_model

Element:

- `service_status` (1,1): element content
- `overall_status` (1,1): element content
- `extension` (0,unbounded): anyType

Attribute:

- -

This is a wrapper element.

6.93Element: `service_status`

6.93.1 complexType: `anonymous_type_1`

Element:

- `warning` (0,unbounded): `item_history_type`
- `alarm` (0,unbounded): `item_history_type`
- `function_ok` (1,1): boolean
- `function_loss` (0,unbounded): string
- `quality_ok` (1,1): boolean
- `degraded_quality` (0,unbounded): string
- `extension` (0,unbounded): anyType

Attribute:

- `element_identity` (0,1): IDREF

The element `function_ok` is used for indicating that the service can be considered functioning while `quality_ok` indicates if properties are adequate. The type `item_history_type` is used for making it possible to trace previous events that could be of importance. Reference is made to shore base service via attribute `element_identity` and a wrapper element `IDREF_wrapper_element_x` has to be defined for each element using this type.

6.93.2 complexType: `anonymous_type_2`

Element:

- `service` (1,1): `anonymous_type_1`

Attribute:

- -

6.93.3 Element

Structure levels above: MSML, `data_model`, `shore_base_dynamic_data`

Element:

- `IDREF_wrapper_element_1` (0,unbounded): `anonymous_type_2`
- `extension` (0,unbounded): anyType

Attribute:

- -

Each defined service of a shore base shall have a corresponding `service_status` element. This is important for understanding the current service capability of the shore base.

6.94Element: `overall_status`

Structure levels above: MSML, `data_model`, `shore_base_dynamic_data`

Element:

- fire (0,unbounded): damage_type
- explosion (0,unbounded): damage_type
- pollution (0,unbounded): damage_type
- shore_base_under_attack (0,unbounded): damage_type
- other_reason (0,unbounded): damage_type
- sea_condition_forecast (0,unbounded): environmental_condition_type
- ice_condition_forecast (0,unbounded): environmental_condition_type
- weather_forecast (0,unbounded): environmental_condition_type
- tidal_stream_forecast (0,unbounded): environmental_condition_type
- drifting_object (0,unbounded): environmental_condition_type
- extension (0,unbounded): anyType

Attribute:

- -

This element contains information that has an overall scope and importance and concerns the shore base as a whole. For element *other_reason* the actual reason is recorded using *damage_type*. The information concerns only the area within shore base control.

6.95Element: vessel_shore_base_relation

Structure levels above: MSML, data_model

Element:

- administration (1,1): element content
- status (1,1): element content
- vessel_shore_base_task (1,1): element content
- extension (0,unbounded): anyType

Attribute:

- -

This is a wrapper element.

6.96Element: administration

Structure levels above: MSML, data_model, vessel_shore_base_relation

Element:

- ships_agent (0,1): address_information_type
- cargo_passenger_responsible (0,1): address_information_type
- extension (0,unbounded): anyType

Attribute:

- -

The ships agent is normally the sole contact person between vessel and shore base when handling cargo. However, note that a shore base could be unmanned and just passed by the vessel.

6.97Element: status

Structure levels above: MSML, data_model, vessel_shore_base_relation

Element:

- estimated_time_of_arrival (1,1): date_and_time_type
- actual_time_of_arrival (0,1): date_and_time_type
- estimated_time_of_departure (1,1): date_and_time_type
- actual_time_of_departure (0,1): date_and_time_type

- shore_base_arrival_reason (1,1): shore_base_arrival_passing_type
- shore_base_access_denied (0,1): vessel_hindrance_type
- detention_activated (0,1): vessel_hindrance_type
- detention_cancelled (0,1): vessel_hindrance_type
- number_of_times_master_entered_shore_base (1,1): integer
- vessel_sent_request (0,1): item_history_type
- shore_base_sent_reply (0,1): item_history_type
- shore_base_sent_request (0,1): item_history_type
- vessel_sent_reply (0,1): item_history_type
- extension (0,unbounded): anyType

Attribute:

- -

A safety aspect is if it is not the first arriving to a port e.g. pilotage requirements can be lowered if waters are known. It will be a too detailed level for MSML if individuals or group of individuals shall be checked for this aspect. Thus only the experience of the master is considered as shown in *number_of_times_master_entered_shore_base* and this relates to a year back in time. Even though information logging is generally not included in MSML the communication between vessel and shore base can be recorded using *vessel_sent_request*, *shore_base_sent_reply*, *shore_base_sent_request* and *vessel_sent_reply*. The synchronisation is made via time information stored using *item_history_type*. The specification of vocabulary, amount, granularity and quality of this communication is not within the scope of MSML. When the vessel – shore base relation ceases the request/reply history could be removed in favour of new a vessel – shore base relation. If history shall be saved it has to be handled outside the scope of MSML.

6.98Element: vessel_shore_base_task

6.98.1 complexType: anonymous_type_1

Element:

- identity (1,1): string
- location (1,1): string
- wharf_terminal_identity (1,1): string
- length (1,1): float, attribute units (1,1): units_type
- depth (1,1): float, attribute units (1,1): units_type
- cyclon_safe (1,1): boolean
- electrical_supply (1,1): string
- restriction (0,unbounded): string
- extension (0,unbounded): anyType

Attribute:

- -

This type contains berth information. The elements *identity*, *location* and *wharf_terminal_identity* contain shore base local information i.e. they must be connected with shore base global information for unique identification. The element *restriction* contains extra information concerning the berth e.g. underwater cable disallowing anchorage.

6.98.2 complexType: anonymous_type_2

Element:

- identity (1,1): string
- number_of_persons (1,1): integer

- contact (1,1): address_information_type
- language (1, unbounded): string
- english_understood (1,1): boolean
- extension (0,unbounded): anyType

Attribute:

- -

This type contains stevedore information and handles the stevedores as a group. More than one language is only specified when there is no common language.

6.98.3 complexType: anonymous_type_3

Element:

- identity (1,1): string
- location (1,1): string
- depth (1,1): float, attribute units (1,1): units_type
- cyclon_safe (1,1): boolean
- restriction (0,unbounded): string
- extension (0,unbounded): anyType

Attribute:

- -

This type contains anchorage information. The elements *identity*, *location* and *wharf_terminal_identity* contain shore base local information i.e. they must be connected with shore base global information for unique identification. The element *restriction* contains extra information concerning the anchorage.

6.98.4 complexType: anonymous_type_4

Element:

- requested_shore_base_service (0,unbounded): shore_base_service_type

Attribute:

- -

6.98.5 complexType: anonymous_type_5

Element:

- acquired_shore_base_service (0,unbounded): shore_base_service_type

Attribute:

- -

6.98.6 Element

Structure levels above: MSML, data_model, vessel_shore_base_relation

Element:

- IDREF_wrapper_element_1 (1,1): anonymous_type_4
- IDREF_wrapper_element_2 (1,1): anonymous_type_5
- assigned_berth (0,1): anonymous_type_1
- assigned_new_crew (0,1): item_history_type
- assigned_stevedores (0,1): anonymous_type_2
- assigned_anchorage (0,1): anonymous_type_3
- passenger_transfer (0,1): cargo_passenger_transfer_type
- cargo_transfer (0,1): cargo_passenger_transfer_type

- extension (0,unbounded): anyType

Attribute:

- -

This element contains information concerning tasks that need a mutual relation between vessel and shore base. The stevedores assigned for the vessel are treated as a single group. The requested and acquired shore base services can be seen as a contract proposition between vessel and shore base and seen from the vessel point of view. If not acceptable the vessel could start searching for another shore base.

6.99Element: perspective

Structure levels above: MSML

Element:

- repair_and_maintenance (0,1): element content
- inspection (0,1): element content
- extension (0,unbounded): anyType

Attribute:

- -

This is a wrapper element. The reason for making these elements optional is that they support successive information built-up.

6.100Element: repair_and_maintenance

Structure levels above: MSML, perspective

Element:

- ashore (1,1): element content
- onboard (1,1): element content
- status (0,unbounded): element content
- extension (0,unbounded): anyType

Attribute:

- -

This is a wrapper element.

6.101Element: ashore

6.101.1 complexType: anonymous_type_1_1

Element:

- constituent_identity (1,1): string
- constituent_change (1,1): string
- status_afterwards (1,1): string
- referenced_element_no_longer_exists (1,1): boolean
- extension (0,unbounded): anyType

Attribute:

- element_identity (0,1): IDREF

Reference is made to vessel constituent via attribute *element_identity* if element still exists. In any case enough information must be provided in *constituent_identity* in order to uniquely identify the constituent. The quality of information is up to the author since the general type *string* is used.

6.101.2 complexType: anonymous_type_1_2

Element:

- certificate (1,1): certificate_type
- certificate_change_needed (1,1): boolean
- certificate_change_reason (1,1): string
- status_afterwards (1,1): string
- extension (0,unbounded): anyType

Attribute:

- element_identity (0,1): IDREF

Reference is made to vessel certificate via attribute *element_identity*. The quality of information is up to the author since the general type *string* is used.

6.101.3 complexType: anonymous_type_1_3

Element:

- description_change (1,1): string
- status_afterwards (1,1): string
- referenced_element_no_longer_exists (1,1): boolean
- extension (0,unbounded): anyType

Attribute:

- element_identity (0,1): IDREF

Reference is made to vessel description via attribute *element_identity* if element still exists. The quality of information is up to the author since the general type *string* is used.

6.101.4 complexType: anonymous_type_1

Element:

- decision_issued_by (1,1): address_information_type
- decision_date (1,1): date_and_time_type
- reason (1,1): repair_and_maintenance_reason_type
- work_responsible (1,1): address_information_type
- work_started (1,1): date_and_time_type
- work_finished (1,1): date_and_time_type
- type_of_work (1,1): string
- constituent_effect (0,unbounded): anonymous_type_1_1
- certificate_effect (0,unbounded): anonymous_type_1_2
- description_effect (0,unbounded): anonymous_type_1_3
- required_deficiencies_rectified (1,1): deficiencies_rectified_limit_type
- deficiencies_not_rectified_within_limit (0,unbounded): string
- outstanding_deficiency (0,unbounded): string
- not_handled_deficiency_reason (0,1): string
- not_handled_deficiency_decided_by (0,1): string
- not_handled_deficiency_decision_date (0,1): date_and_time_type
- status_afterwards (1,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type contains information concerning a specific repair and maintenance occasion ashore. This type contains administrative information concerning the reason for repair and maintenance, who has ordered it etc. To this, information concerning the work made and the quality of it are added. The type of work is specified using type *string* and thus the quality of information is up to the author. Repair and maintenance ashore can directly affect constituents, certificates and description but indirectly also the status of constituents (however outside the scope of perspectives). The last part of the type contains information regarding deficiencies with a further possible specification using *status_afterwards*.

6.101.5 Element

Structure levels above: MSML, perspective, repair_and_maintenance

Element:

- occasion (0,unbounded): anonymous_type_1
- extension (0,unbounded): anyType

Attribute:

- -

This element contains a list of repair and maintenance occasions ashore.

6.102Element: onboard

6.102.1 complexType: anonymous_type_1_1

Element:

- constituent_identity (1,1): string
- constituent_change (1,1): string
- status_afterwards (1,1): string
- referenced_element_no_longer_exists (1,1): boolean
- extension (0,unbounded): anyType

Attribute:

- element_identity (0,1): IDREF

Reference is made to vessel constituent via attribute *element_identity* if element still exists. In any case enough information must be provided in *constituent_identity* in order to uniquely identify the constituent. The quality of information is up to the author since the general type *string* is used.

6.102.2 complexType: anonymous_type_1_2

Element:

- certificate (1,1): certificate_type
- certificate_change_needed (1,1): boolean
- certificate_change_reason (1,1): string
- status_afterwards (1,1): string
- extension (0,unbounded): anyType

Attribute:

- element_identity (0,1): IDREF

Reference is made to vessel certificate via attribute *element_identity*. It might be necessary to change certificate later on when reaching a shore base and information is stored using *certificate_change_needed* and *certificate_change_reason*. The quality of information is up to the author since the general type *string* is used.

6.102.3 complexType: anonymous_type_1_3

Element:

- description_change (1,1): string
- status_afterwards (1,1): string
- referenced_element_no_longer_exists (1,1): boolean
- extension (0,unbounded): anyType

Attribute:

- element_identity (0,1): IDREF

Reference is made to vessel description via attribute *element_identity* if element still exists. The quality of information is up to the author since the general type *string* is used.

6.102.4 complexType: anonymous_type_1

Element:

- decision_issued_by (1,1): address_information_type
- decision_date (1,1): date_and_time_type
- reason (1,1): repair_and_maintenance_reason_type
- external_support (0,unbounded): address_information_type
- type_of_external_support (0,1): string
- work_responsible (1,1): address_information_type
- work_started (1,1): date_and_time_type
- work_finished (1,1): date_and_time_type
- type_of_work (1,1): string
- used_spare_part (0,unbounded): string
- constituent_effect (0,unbounded): anonymous_type_1_1
- certificate_effect (0,unbounded): anonymous_type_1_2
- description_effect (0,unbounded): anonymous_type_1_3
- outstanding_deficiency (0,unbounded): string
- not_handled_deficiency_reason (0,1): string
- not_handled_deficiency_decided_by (0,1): string
- not_handled_deficiency_decision_date (0,1): date_and_time_type
- status_afterwards (1,1): string
- extension (0,unbounded): anyType

Attribute:

- -

This type contains information concerning a specific repair and maintenance occasion onboard. This type contains administrative information concerning the reason for repair and maintenance, who has ordered it etc. To this, information concerning the work made and the quality of it are added. The type of work is specified using type *string* and thus the quality of information is up to the author. Since resources onboard are limited external support could be needed as given by *external_support* and *type_of_external_support*. In *used_spare_part* the spare parts used for this occasion are listed. Repair and maintenance onboard can directly affect constituents and description but indirectly also certificates and the status of constituents (however outside the scope of perspectives). The last part of the type contains information regarding deficiencies with a further possible specification using *status_afterwards*.

6.102.5 Element

Structure levels above: MSML, perspective, repair_and_maintenance

Element:

- occasion (0,unbounded): anonymous_type_1
- extension (0,unbounded): anyType

Attribute:

- -

This element contains a list of repair and maintenance occasions onboard.

6.103Element: status

6.103.1 complexType: anonymous_type_1

Element:

- identity (1,1): string
- type (1,1): string
- available_amount (1,1): integer
- status (1,1): string
- adequate_repair_equipment (1,1): boolean
- adequate_repair_competence (1,1): boolean
- adequate_repair_documentation (1,1): boolean
- external_support (0,1): address_information_type
- repair_information (1,1): detailed_information_type
- referenced_element_no_longer_exists (1,1): boolean
- extension (0,unbounded): anyType

Attribute:

- element_identity (0,1): IDREF

The purpose is to give information concerning a specific spare part carried onboard. Information includes identification information as well as capability onboard for handling the spare part. Also external support could be needed as specified in *external_support*. Reference is made to vessel constituent via attribute *element_identity* if element still exists. In any case enough information must be provided in *constituent_identity* in order to uniquely identify the constituent.

6.103.2 Element

Structure levels above: MSML, perspective, repair_and_maintenance

Element:

- previous_drydocking (0,1): item_history_type
- repair_and_maintenance_equipment_onboard (0,unbounded): equipment_type
- spare_part_onboard (0,unbounded): anonymous_type_1
- reference_to_maintenance_plan (0,1): detailed_information_type
- prevention_maintenance_programme (0,1): detailed_information_type
- next_planned_repair_and_maintenance_date (0,1): date_and_time_type
- next_planned_repair_and_maintenance_location (0,1): address_information_type
- extension (0,unbounded): anyType

Attribute:

- -

Previous drydockings are of interest since they imply more thorough surveys. In element *repair_and_maintenance_equipment_onboard* type and amount of equipment onboard are described. Information concerning next planned repair and maintenance makes it possible to optimise costs since vessel tasks and repair and maintenance can be synchronized.

6.104Element: inspection

6.104.1 complexType: anonymous_type_1

Element:

- type (1,1): string
- issued_by (1,1): address_information_type
- date (1,1): date_and_time_type
- rating (1,1): string
- extension (0,unbounded): anyType

Attribute:

- -

The element *type* contains further information for specifying condition assessment programme. A rating is defined for condition assessment programme and stored in *rating*.

6.104.2 complexType: anonymous_type_2

Element:

- inspection_type (1,1): string
- issued_by (1,1): address_information_type
- date (1,1): date_and_time_type
- reason_for_inspection (1,1): string
- result (1,1): detailed_information_type
- extension (0,unbounded): anyType

Attribute:

- -

Since *inspection* is a perspective, referencing should be made to different elements in the MSML data model. The current solution, however, does not use explicit references. Instead only a reference is given to detailed results of inspection, in element *result*. The reason is that it is duplicate work to map inspection results to the data model. Since MSML is more directed towards realtime information such as state and status, as opposed to the more static nature of inspection results, the importance of mapping inspection results to MSML data model is further decreased.

6.104.3 Element

Structure levels above: MSML, perspective

Element:

- condition_assessment_programme (0, unbounded): anonymous_type_1
- prevention_maintenance_programme (0,1): detailed_information_type
- next_planned_inspection_date (0,1): date_and_time_type
- next_planned_inspection_location (0,1): string
- next_planned_inspection_type (0,1): string
- inspection_program_in_place_for_void_spaces (1,1): boolean
- inspection_program_in_place_for_cargo (1,1): boolean
- inspection_program_in_place_for_ballast_tanks (1,1): boolean
- inspection (0,unbounded): anonymous_type_2
- extension (0,unbounded): anyType

Attribute:

- -

Information concerning planned inspections makes it possible to optimise costs since vessel tasks, repair and maintenance and inspections can be synchronized.

7 MSML instance processing (non-normative)

Even though actual use of MSML is outside the definition of MSML there are a number of checks that could be performed and are not included in validation. There are two types:

- Checks that could be performed considering an MSML instance in isolation.
- Checks that could be performed using an MSML instance and externally referenced information.

Below the most important ones are listed.

- Checking that standardised notation is correct according to the specified reference.
- Checking that attributes of type *units_type* and *element_identity_algorithm_type* are consistently handled.
- Checking that *address_history_type* and *item_history_type* contain a single element with the corresponding *valid* flag element containing TRUE and that it is the most recent one.
- Checking that references are made to allowed elements.
- Checking that references are one-to-one within the data model.
- For perspectives, checking that references are made whenever possible and matching with the value of *element_no_longer_exists*.
- Maritime reasonability checks e.g. breadth is shorter than length, draft compared to overall height, maximum speed, high speed craft flagged compared to maximum speed etc.
- Comparing different types of charter.
- Comparing MSML version in an MSML instance with the specified MSML namespace version.
- In some cases vessel type could be compared with information given in vessel description (e.g. see element *ro_ro*).
- Checking counting information with the items listed. This is possible for vessel certificates, vessel class, IMO number,
- Checking made at encryption and signatures, however, this is outside the scope of MSML.
- Checking, in element *security_support*, that *signed_element*, *encrypted_element*, *encrypted_element_exists* and *signed_element_exists* match.
- Checking that latitude and longitude are stored according to the specified format.

8 References (non-normative)

- [1] Extensible Markup Language (XML) 1.0 (Second Edition)
<http://www.w3.org/TR/2000/REC-xml-20001006>
- [2] Marine Trading Markup Language (MTML)
<http://www.meca.org.uk/standards.asp?standardsID=10>
- [3] SIRENAC database
<http://www.parismou.org/allships/inspdb-basic.html>
- [4] EQUASIS database
<http://www.equasis.org/>
- [5] SafeSeaNet
http://www.waterman-ts.net/proposals_fp6.pdf
- [6] ISO 10303, Product Data Representation and Exchange (STEP)
<http://www.iso.org/iso/en/ISOOnline.frontpage>
- [7] Condition Assessment Program (CAP),
http://www.classnk.or.jp/hp/nk_e/Activities/activitiessecap.htm
- [8] OPTIMISE – Optimal Maintenance Intervention of Ships in Europe
<http://www.bmtech.co.uk/optimise/index.html>
- [9] TELEMAS
http://www.issus.fh-hamburg.de/iss_web/projekte/telemas/index.html
- [10] XML Schema Part 1: Structures
W3C Recommendation 2 May 2001
<http://www.w3.org/TR/xmlschema-1/>
- [11] XML Schema Part 2: Datatypes
W3C Recommendation 02 May 2001
<http://www.w3.org/TR/xmlschema-2/>
- [12] W3C World Wide Web Consortium
<http://www.w3.org/>
- [13] XML-Signature Syntax and Processing
W3C Recommendation 12 February 2002
<http://www.w3.org/TR/xmlsig-core/>
- [14] XML Encryption Syntax and Processing
W3C Recommendation 10 December 2002
<http://www.w3.org/TR/xmlenc-core/>

9 Appendix A: Considered regulations and directives (non-normative)

The following EU directives and regulations have been considered.

Regulation 613/91	Directive 95/21/EC
Regulation 3051/95	Directive 94/57/EC
Regulation 2978/94	Directive 94/25/EC
Regulation 2158/93	Directive 93/103/EC
Regulation 179/98	Directive 92/29/EEC
Directive 99/95/EC	Directive 88/379/EEC
Directive 99/35/EC	Directive 79/115/EEC
Directive 98/85/EC	Directive 3/75/EEC
Directive 98/74/EC	Directive 2002/84/EC
Directive 98/55/EC	Directive 2002/59/EC
Directive 98/42/EC	Directive 2002/35/EC
Directive 98/41/EC	Directive 2001/96/EC
Directive 98/25/EC	Directive 2001/25/EC
Directive 98/18/EC	Directive 2001/106/EC
Directive 97/70/EC	Directive 2001/105/EC
Directive 97/58/EC	Directive 2000/59/EC
Directive 97/34/EC	Directive 1999/97/EC
Directive 96/98/EC	Directive 1999/35/EC
Directive 96/50/EC	Directive 1999/19/EC
Directive 96/39/EC	

10 Appendix B: Extensions (non-normative)

A natural extension is to replace the type *string* with new elements or enumeration types when more information has been gathered and the importance has become more clear. Enumeration types have whenever applicable included an alternative *other* that could be further specified in the future. This is also the case with the *extension* element of type *anyType*. For extracting information within *extension* element there are two main possibilities:

- to analyze the element structure and use the information as a subtree
- presenting the information as text in the same way it is written i.e. including element tags

A general advice is to not put too much effort on presentation since:

- extension information is temporary (until the next update is made)
- there will be much variation since there are no rules governing the internal structure of *extension* element

New types and elements could also be added. Since the new MSML version might not be backward compatible it is necessary to gather a number of changes before a significant update is made. In the meantime one then has to accept lowered validation quality of new aspects. Two examples are using *extension* element of type *anyType* and setting (0,1) for a mandatory element.

11 Appendix C: Standardisation of MSML

11.1 Background

ISO TC8 (International Standards Organisation – Technical Committee on Ships and marine technology) handles ship-related standardisation together with various other technical committees in ISO and IEC. Except for standards related to internal waterways, ISO and IEC are the de facto standardisation organisations in this area. CEN and CENELEC have little or no activity in international ship standards.

ISO TC8 held its advisory group (AG) meeting in Geneva from June 1st to June 3rd 2004. At this meeting the results from MANATEE and possible standardisation of MSML were discussed. The MANATEE results and possible standardisation strategies were presented to the group by its chairman.

ISO TC8 AG decided in the meeting to publish the MANATEE D2.2 deliverable as a committee draft (CD) for a Publicly Available Specification (PAS). It was also suggested that MANATEE inform EMSA (European Maritime Safety Agency)/SafeSeaNet and request that someone from either organisation/project joins the TC8 effort on standardisation.

The following explanation is derived from observations made by the chairman of TC8/SC10.

11.2 MSML and other initiatives – use of PAS

MSML was considered by the ISO advisory group to be also one of several initiatives to standardise *port clearance information* between ship and shore. While MSML may not be the best effort in this field, the specification of a mark-up language instead of explicit messages was deemed very advantageous. The advisory group concluded that MSML in its current version were not useful as an international standard, mainly because other initiatives exist and because it would be necessary to establish a working group to discuss the issue among the actors. However it was believed that it would be very useful to publish an intermediate draft, and hence the decision to publish it as a PAS was taken. Using a PAS means that the specification can be published as a “preliminary specification”, that is a basis for further work and/or statement from a technical committee on a certain subject. If the document passes the CD voting, it will immediately be published as a PAS. This takes minimum three months from the date of publish as a CD.

An ISO/PAS is reviewed after three years in order to decide whether it will be confirmed for a further three years, revised to become an International Standard, or withdrawn. If the ISO/PAS is confirmed, it is reviewed again after a further three years, at which time it must either be transformed into an International Standard or be withdrawn.

Concurrently with the PAS voting, a new work item (NWI) will also be taken to the vote. The details of the NWI are discussed in the next chapter. The idea of the NWI is to invite important actors to participate in the development of the actual electronic port clearance (EPC) standard, based on all available material. Already, from the AG meeting, it is clear that we can expect participation from Japan, Korea, China (including Hong Kong) and the US (US Coast Guard - USCG). It was believed that the concurrent publishing of the CD and the NWI would help to create a greater interest for the NWI.

Thus, the alternative of using MSML as a draft outline for the NWI and only publish one item was deemed to be less desirable.

11.3 Overview of the Electronic Port Clearance issue

Electronic Port Clearance (EPC) can be defined to cover all activities related to a ships clearance into or out of a port, port state area or any other related reporting. This can include:

- Navigation messages (notice to mariners, chart updates, weather).
- Entry into and exit from VTS/ship reporting areas.
- Mandatory report to next port of call (Dangerous goods, Notice of arrival, crew and passenger lists, incident lists etc.).
- ISPS related messages (safety level, safety certificates).
- Possibly port state inspection related reports (certificates).
- Port logistics (order of pilot, berth, supplies, cranes, garbage disposal etc.)
- Own supplies and logistics (cargo, crew, bunkers etc.)
- Reports to port on departure (next port, dangerous cargo etc.)

The term “one stop shopping” has been used to describe facilities that make it simpler to deliver and receive these reports. Issues in EPC are:

- Reports can be delivered or received by ship, owner, agent or other authorised source, using some kind of authentication.
- All reports can be delivered to one repository, possibly by forwarding components of messages to other destinations – again authentication and possibly encryption is necessary. This also requires multi-level acknowledgements.
- Establishment of standardised message format, including authentication, acknowledgement and encryption as necessary.
- Establishment of general addressing scheme including repository for encryption keys as necessary.

In conjunction with the consideration of MSML as an international standard, some research was done on the status of electronic port clearance. The short list of results is that:

- EDIFACT has a number of messages for this purpose, but these are only used to a certain degree and do not really support EPC.
- IMO and EU have mentioned XML as a desirable report format in a number of publications. SafeSeaNet (Directive 2002/59/EC) is the most commonly known in Europe.
- SafeSeaNet uses XML between administrations, but not to ships so far.
- USCG/US notice of arrival (based on Trade Act 2002) uses XML for many messages (not all).
- Port of Hong Kong accepts XML messages for some EPC tasks.
- Port@NET in Finland accepts some XML messages.
- Ports in many states are working on XML messages, but have no official support (Pacific rim, Spain, Singapore, Netherlands).

11.4 Rationale for a new ISO work item

If all ports and port states develop their own report formats, this will be a major hassle for ships in international trade. Thus, there is a clear benefit to the standardisation of these messages. Most of these messages are also covered by international legislation, so there is not a great need to allow national formats. Today, the development of XML based EPC has also not come very far, so there is a realistic chance that a new standard may succeed. Although some message formats are in place, there should be no large problem for the port or port state to implement the alternative standard formats.

Obviously a “one stop shopping” concept is highly desirable, particularly if implemented with sufficient flexibility. The ability to operate with authentication of ship or port as legitimate source of a message, although the message physically may have been sent by, e.g., an agent, should greatly enhance the EPC function without compromising security or safety.

One should also note the increasing focus on transport security that will most likely lead to even stricter reporting requirements in the future. These reports will have to be of a standard, which satisfies various security and authentication requirements that can be difficult to implement today. This means, for example, that a new system for validating sender identity may be required.

Thus, it seems as if many ports and port states have recognized that EPC is necessary in the near future, and also that very few of them have developed complete solutions. This means that there is a fair chance that a new standardisation item within this area may succeed.

11.5 Requirements for standardisation success

The NWI has to pass the voting procedure and also get enough positive votes as well as countries willing to participate in the work group. This requires some lobbying to make sure that enough national committees vote and also agrees to participate. This is the basic prerequisite for getting the NWI accepted.

For the standardisation process to succeed, i.e., to produce a standard that will actually be used, it is necessary to have a number of large port states represented in the work group. In the AG meeting this was emphasized and members of the AG meeting agreed to raise this item with their national governments. It will be necessary to follow up on this, once the NWI is put out for voting. Representatives from large port states that voiced their agreement were from:

- Japan
- Korea
- China (also including Hong Kong)
- US (represented by USCG)

It will also be necessary to have the European Union represented in the work group. This may be achieved through commitment in EU projects or directly towards the Commission. In addition we will certainly get support from Norway, Portugal and probably also Finland and Singapore. Other nations are of course welcome, but some of the nations represented in MANATEE are not P-members (voting members) in ISO TC8/SC10.

11.6 References

1. Resolutions of the 38th meeting of the ISO TC 8 Advisory Group (ISO/TC8 AG N604, June 2004)
2. Ships and marine technology — Computer applications — Specifications of MSML (Maritime Safety Markup Language), ISO TC 8/SC 10 N 141 of 11/08/2004, PAS Draft
3. Ships and marine technology – Computer applications – Specifications of MSML (Maritime Safety Markup Language) ISO TC 8/SC 10 N142 of 20/08/2004, NWI Proposal, submitted by the Norwegian Standards Organization
4. Rødseth, Ø. J., ISO TC8 new work item on electronic port clearance, MARINTEK Memo, 07/06/2004