



BSI Standards Publication

Ships and marine technology — Electronic port clearance (EPC)

Part 1: Message structures and application programming interfaces

National foreword

This British Standard is the UK implementation of ISO 28005-1:2024. It supersedes BS ISO 28005-1:2013, which is withdrawn.

The UK participation in its preparation was entrusted to Technical Committee SME/32, Ships and marine technology - Steering committee.

A list of organizations represented on this committee can be obtained on request to its committee manager.

Contractual and legal considerations

This publication has been prepared in good faith, however no representation, warranty, assurance or undertaking (express or implied) is or will be made, and no responsibility or liability is or will be accepted by BSI in relation to the adequacy, accuracy, completeness or reasonableness of this publication. All and any such responsibility and liability is expressly disclaimed to the full extent permitted by the law.

This publication is provided as is, and is to be used at the recipient's own risk.

The recipient is advised to consider seeking professional guidance with respect to its use of this publication.

This publication is not intended to constitute a contract. Users are responsible for its correct application.

© The British Standards Institution 2024
Published by BSI Standards Limited 2024

ISBN 978 0 539 19705 1
ICS 03.100.70; 35.240.60; 47.020.99

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 31 December 2024.

Amendments/corrigenda issued since publication

Date	Text affected
------	---------------



International
Standard

ISO 28005-1

**Ships and marine technology —
Electronic port clearance (EPC) —**

**Part 1:
Message structures and application
programming interfaces**

*Navires et technologie maritime — Opérations portuaires
assistées par systèmes électroniques —*

*Partie 1: Structures des messages et interfaces de programmation
des applications*

**Second edition
2024-12**

Reference number
ISO 28005-1:2024(en)

© ISO 2024  **Documents**
Document Sharing Hub



COPYRIGHT PROTECTED DOCUMENT

© ISO 2024

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword	ix
Introduction	xi
1 Scope	1
2 Normative references	1
3 Terms, definitions, and abbreviated terms	1
3.1 Terms and definitions	1
3.2 Abbreviated terms	5
4 Structure of XSD data type and object definitions	6
4.1 General	6
4.2 Principles for XML descriptions in the ISO 28005 series	7
4.2.1 No use of XML attributes	7
4.2.2 Defaults for minOccurs and maxOccurs	7
4.2.3 Signalling empty XML tags	7
4.2.4 Order of child elements in XSD files	7
4.2.5 Character set	7
4.2.6 Principles for defining types with code lists	7
4.2.7 XSD name space for general XSD data types	8
4.2.8 ISO 28005 name space	8
4.2.9 Use of Xpath expressions	8
4.3 Structure of clauses defining data types	8
4.3.1 Clause and data type name	8
4.3.2 Definition	8
4.3.3 Type defined as XSD code	9
4.3.4 Representation	9
4.4 Creating valid XSD schema files	9
4.4.1 File structure	9
4.4.2 Numbering of XSD files and message version code	10
4.4.3 Location of XSD files	11
4.5 Reference to data types defined in ISO 28005-2:2021	11
5 Adapted XSD data types for ISO 28005	12
5.1 General	12
5.2 epc:anyURI – Generalized URI	12
5.2.1 Definition	12
5.2.2 Type	12
5.2.3 Representation	12
5.3 epc:boolean – Boolean flag	12
5.3.1 Definition	12
5.3.2 Type	12
5.3.3 Representation	12
5.4 epc:date – General date	12
5.4.1 Definition	12
5.4.2 Type	12
5.4.3 Representation	13
5.5 epc:dateTime – Time and date, with time zone	13
5.5.1 Definition	13
5.5.2 Type	13
5.5.3 Representation	13
5.6 epc:decimal – decimal number	13
5.6.1 Definition	13
5.6.2 Type	13
5.6.3 Representation	13
5.7 epc:duration – Time duration	14
5.7.1 Definition	14

5.7.2	Type.....	14
5.7.3	Representation.....	14
5.8	epc:int – Integer number.....	14
5.8.1	Definition.....	14
5.8.2	Type.....	14
5.8.3	Representation.....	14
5.9	epc:string – General string.....	14
5.9.1	Definition.....	14
5.9.2	Type.....	14
5.9.3	Representation.....	15
5.10	epc:token – Computer-understandable string.....	15
5.10.1	Definition.....	15
5.10.2	Type.....	15
5.10.3	Representation.....	15
5.11	epc:xpath – Identification of an XML data item.....	15
5.11.1	Definition.....	15
5.11.2	Type.....	15
5.11.3	Representation.....	15
6	General ISO 28005 data types.....	15
6.1	General.....	15
6.2	epc:AuthenticatorType – Authenticator of information.....	15
6.2.1	Definition.....	15
6.2.2	Type.....	16
6.2.3	Representation.....	16
6.3	epc:AuthorizationTokenType – Authorization token.....	16
6.3.1	Definition.....	16
6.3.2	Type.....	16
6.3.3	Representation.....	16
6.4	epc:ContactInfoType – Contact information.....	16
6.4.1	Definition.....	16
6.4.2	Type.....	17
6.4.3	Representation.....	17
6.5	epc:CommunicationNumberType – Communication number information.....	17
6.5.1	Definition.....	17
6.5.2	Type.....	17
6.5.3	Representation.....	18
6.6	epc:CountryCodeContentType – Country identification.....	18
6.6.1	Definition.....	18
6.6.2	Type.....	18
6.6.3	Representation.....	18
6.7	epc:CountrySubdivisionCodeContentType – Country subdivision identification.....	18
6.7.1	Definition.....	18
6.7.2	Type.....	18
6.7.3	Representation.....	19
6.8	epc:CrewDutyType – Duty onboard or on shore.....	19
6.8.1	Definition.....	19
6.8.2	Type.....	19
6.8.3	Representation.....	19
6.9	epc:LocationType – Identification of a location.....	19
6.9.1	Definition.....	19
6.9.2	Type.....	19
6.9.3	Representation.....	21
6.10	epc:NameType – Name of person.....	21
6.10.1	Definition.....	21
6.10.2	Type.....	21
6.10.3	Representation.....	21
6.11	epc:OrganizationType – Description of an organization.....	22
6.11.1	Definition.....	22

6.11.2	Type	22
6.11.3	Representation	22
6.12	epc:PostalAddressType – A postal mail address	22
6.12.1	Definition	22
6.12.2	Type	22
6.12.3	Representation	23
6.13	epc:ShipIDType – Ship identity	23
6.13.1	Definition	23
6.13.2	Type	23
6.13.3	Representation	23
6.14	epc:ReportingSystemType – Name of a reporting system	24
6.14.1	Definition	24
6.14.2	Type	24
6.14.3	Representation	24
6.15	epc:AttachmentType – Reference to an attached document	24
6.15.1	Definition	24
6.15.2	Type	24
6.15.3	Representation	24
6.16	epc:ReferenceCodeType – General reference code	25
6.16.1	Definition	25
6.16.2	Type	25
6.16.3	Representation	25
6.17	epc:SystemIDType – Identity code for a software system	25
6.17.1	Definition	25
6.17.2	Type	25
6.17.3	Representation	25
6.18	epc:SignatureCertificateIDType – Name of digital signature holder	26
6.18.1	Definition	26
6.18.2	Type	26
6.18.3	Representation	26
6.19	epc:VersionType – Version code	26
6.19.1	Definition	26
6.19.2	Type	26
6.19.3	Representation	27
7	ISO 28005 design principles	27
7.1	Harmonization with the IMO reference data model	27
7.2	Fully automated machine to machine	27
7.3	Using carrier independent and internet-based protocols	28
7.4	General format of message sequence diagrams	28
7.5	Sender and receiver versus client and server — asynchronous message transfers	29
7.6	Generalization of service	30
7.7	Different levels of sessions	30
7.7.1	HTTP session	30
7.7.2	Session	31
7.7.3	Session context	32
7.8	One service per request and session	33
7.9	Linking receivers to service providers	33
7.10	Service request states	33
7.10.1	Message processing	33
7.10.2	State diagram for service requests	34
7.10.3	Message functions	36
7.10.4	Specification of request timeout	36
7.10.5	Message and service request return values	37
7.11	Send data once only	37
7.12	Message context	37
7.13	General message structure	39
7.14	Digital signatures	40
7.15	Secure data transfer	40

7.16	Additional authorization for accessing API.....	40
7.17	Message implementation guide.....	41
7.18	Other formats than XML for the message body.....	41
7.19	No explicit service discovery.....	41
8	Message exchange patterns.....	41
8.1	General rules.....	41
8.1.1	Application of this specification.....	41
8.1.2	Use of reference codes.....	42
8.1.3	Use of final flag in message header.....	42
8.1.4	Use of service timeout or session context end.....	43
8.1.5	Status and error codes.....	43
8.1.6	Multiple senders.....	44
8.1.7	Interleaving update requests with status messages.....	45
8.2	Sequence diagrams.....	45
8.2.1	Pattern 1: Service request and updates.....	45
8.2.2	Pattern 2: Status poll.....	47
8.2.3	Pattern 3: Simple report.....	47
8.2.4	Pattern 4: Request information.....	48
8.2.5	Pattern 5: Subscribe to service or information.....	48
9	Using HTTP multi-part message.....	49
9.1	General.....	49
9.2	Example of an ISO 28005-1 multi-part message.....	50
9.3	Content-Type: multipart/form-data.....	50
9.4	Content-Encoding: gzip.....	51
9.5	Prose text.....	51
9.6	Content-Type: application, image or other.....	51
9.7	Content-Disposition: form-data; name = name; filename = file.name;.....	51
10	Definitions related to the message header part.....	52
10.1	General.....	52
10.2	epc:MessageFunctionCodeContentType – Message function code.....	52
10.2.1	Definition.....	52
10.2.2	Type.....	52
10.2.3	Representation.....	52
10.3	epc:ReplyInformationType – Type of sender response code.....	52
10.3.1	Definition.....	52
10.3.2	Type.....	52
10.3.3	Representation.....	53
10.4	epc:MessageBodyFormatContentType – Format of body data.....	53
10.4.1	Definition.....	53
10.4.2	Type.....	53
10.4.3	Representation.....	53
10.5	epc:ServiceTypeCodeContentType – Code for identification of service type.....	53
10.5.1	Definition.....	53
10.5.2	Type.....	54
10.5.3	Representation.....	54
10.6	epc:ServiceCodeContentType – Code for identification of a service in a group.....	54
10.6.1	Definition.....	54
10.6.2	Type.....	54
10.6.3	Representation.....	54
10.7	epc:StatusType – General message and service request status and error codes.....	54
10.7.1	Definition.....	54
10.7.2	Type.....	54
10.7.3	Representation.....	55
10.8	epc:SpecialAttachmentType – Description of special attachment.....	55
10.8.1	Definition.....	55
10.8.2	Type.....	55
10.8.3	Representation.....	56

10.9	epc:MessageManifestType – Number of message parts	56
10.9.1	Definition	56
10.9.2	Type	56
10.9.3	Representation	56
10.10	epc:EPCMessageHeaderType – Standard header for an EPC message	56
10.10.1	Definition	56
10.10.2	Type	56
10.10.3	Representation	59
11	Definitions related to the message body part	61
11.1	General	61
11.2	XML message body	61
11.2.1	epc:EPCMessageBodyType – the XML body data type	61
11.2.2	Structure of message body definition table	62
11.3	Encryption of selected content	62
11.4	UN/EDIFACT message body	63
11.5	UN/EDIFACT status message	63
11.6	JSON message body	63
12	Definitions related to attachment message parts	63
12.1	General	63
12.2	Reference to an attached document in an XML body	63
13	Definitions related to X.509 certificate message parts	64
14	Definitions related to the digital signature message part	64
14.1	General	64
14.2	Signers	64
14.3	epc:EPCMessageSignatureType – Digital signatures of message parts	65
14.3.1	Definition	65
14.3.2	Type	65
14.3.3	Representation	65
15	General definitions related to the use of HTTP	65
15.1	Conceptual structure of a receiver	65
15.2	Conceptual structure of a sender	66
15.3	Transmission protocol	67
15.4	Avoid use of HTTP redirect and similar mechanisms	67
15.5	Optional use of HTTP keep-alive	67
15.6	API access point URL	67
15.7	HTTP methods	67
15.8	Different types of synchronous return values	68
15.8.1	General	68
15.8.2	Connection error	68
15.8.3	HTTP error codes	68
15.8.4	Message status	68
15.8.5	Service request status	69
16	API access points for asynchronous HTTP communication	69
16.1	General	69
16.2	Message patterns to use	69
16.3	No authorization on the sender's URL	69
17	API access point for synchronous HTTP communication	69
17.1	General	69
17.2	Message patterns to use	69
18	Authorization to access API access point	70
18.1	General	70
18.2	The message pattern	70
18.3	epc:ServiceAuthorizationType – Type of service authorization	71
18.3.1	Definition	71
18.3.2	Type	71

18.3.3	Representation.....	71
18.4	The message body.....	72
19	Specifications for the message implementation guide (MIG).....	72
19.1	General structure of MIG.....	72
19.2	MIG Introduction.....	72
19.3	High level description of use case.....	72
19.4	Prerequisites.....	72
19.5	Message sequence diagrams.....	72
Annex A (normative) EPC request body.....		74
Annex B (informative) Message implementation guide for access authorization.....		75
Annex C (informative) Message implementation guide for maritime single window and mandatory ship reporting system.....		80
Annex D (normative) Code list for special attachments.....		87
Annex E (normative) Message function codes for XML messages.....		88
Annex F (normative) Message and service request status codes.....		89
Annex G (normative) Service codes.....		90
Annex H (normative) Software system type codes.....		94
Annex I (normative) Code list for authenticator and contact point roles.....		95
Annex J (normative) Codes for digital signatures.....		96
Annex K (informative) IMO FAL mapping.....		97
Bibliography.....		99

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

ISO draws attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO takes no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents. ISO shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 8, *Ships and marine technology*, Subcommittee SC 11, *Intermodal and short sea shipping*.

This second edition cancels and replaces the first edition (ISO 28005-1:2013), which has been technically revised.

The main changes are as follows:

- a general introduction to all documents in the ISO 28005 series, including the structure of XML and XSD files and general XML type definitions, has been added in [Clauses 4 to 6](#);
- [Clause 7](#) defining the general design principles for the ISO 28005 series has been added;
- [Clause 8](#) has added general message exchange patterns that can be referenced in message implementation guides;
- [Clause 9](#) has added a more general multi-part message structure based on the HTTP multi-part form structure. This includes a formalization of how attachments to the XML based message body can be added to the message. It also includes possibilities for using EDIFACT or JSON as a message body or as attachments to the standardized XML message body or adding encrypted message parts as attachments (detailed in [Clauses 11](#) and [12](#));
- the message header has been updated so that it contains sufficient information to do frontend message processing before forwarding the message to the service specific software modules (see [Clause 10](#));
- support for digital signatures has been added in [Clause 13](#). Digital signature certificates can be attached to a message (see [Clause 14](#));
- a definition of a HTTP-based transport protocol has been added in [Clauses 15 to 17](#);
- a possibility for access authorization has been added in [Clause 18](#);
- a definition for the structure of message implementation guides has been added in [Clause 19](#);

ISO 28005-1:2024(en)

- message implementation guides for access authorization have been added in [Annex B](#), and a maritime single window and mandatory reporting systems have been added in [Annex C](#).

A list of all parts in the ISO 28005 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

This document contains technical specifications that facilitate an efficient exchange of electronic information between ships and shore parties for coastal transit or port calls. It defines the general message format, general message exchange patterns and a message transfer protocol. Other parts of the ISO 28005 series define data models for various types of message transfers.

The message transfer protocol specified in this document uses HTTP over TLS. While many HTTP type application program interfaces (API) are built on the representational state transfer (REST) principle, this document does not use REST. APIs developed according to this document can support physical services such as ordering tugs or pilots, where state changes cannot be guaranteed to be compliant with REST.

This document can be used as a specification of a message format that can be transmitted over other transport protocols than the one defined in this document.

This document is aligned with the IMO FAL guidelines on authentication, integrity and confidentiality in information exchanges via maritime single windows and related services (IMO FAL.5/Circ.46)^[1] and the IMO FAL guidelines for harmonized communication and electronic exchange of operational data for port calls, 31 March 2023 (IMO FAL.5/Circ.52).^[2] The specifications in this document are aligned with the requirements in ISO 23807.^[3]

Ships and marine technology — Electronic port clearance (EPC) —

Part 1: Message structures and application programming interfaces

1 Scope

This document defines the principles, methods and requirements for message exchanges between ships, ship representatives, and other shore parties via a peer-to-peer communication system. This document defines the message structure, including how the data content is assembled from other parts of the ISO 28005 series, and how digital signatures for authentication, integrity, and confidentiality of the message can be used. It also specifies a transport protocol, the basic message exchange patterns, and the protocol related roles of each party in the message exchange. Furthermore, it specifies how more specific message implementation guides (MIGs) are provided for each type of communication application.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 3166-1, *Codes for the representation of names of countries and their subdivisions — Part 1: Country code*

ISO 3166-2, *Codes for the representation of names of countries and their subdivisions — Part 2: Country subdivision code*

ISO 6709, *Standard representation of geographic point location by coordinates*

RFC 1952, *GZIP file format specification version 4.3*

RFC 3986, *Uniform resource identifier (URI): Generic syntax*.

RFC 5246, *The transport layer security (TLS) protocol version 1.2*

RFC 7578, *Returning values from forms: multipart/form-data*

RFC 8446, *The transport layer security (TLS) Protocol Version 1.3*

UN/EDIFACT code list 3035 – Party function code qualifier, Release D.00A, <https://service.unece.org/trade/untdid/d00a/tred/tred3035.htm>

UN/EDIFACT code list 3139 - Contact function code, Release D.23A, <https://service.unece.org/trade/untdid/latest/tred/tred3139.htm>

3 Terms, definitions, and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

- ISO Online browsing platform: available at <https://www.iso.org/obp>
- IEC Electropedia: available at <https://www.electropedia.org/>

3.1.1

HTTP

hypertext transfer protocol

client/server communication protocol used to transfer information in the World Wide Web

Note 1 to entry: HTTP is defined in RFC 2068^[4] as a generic and stateless application-level protocol for distributed and collaborative, information systems.

Note 2 to entry: In this document, the term HTTP is used to mean HTTP implemented over *transport layer security* (3.1.2), i.e. *HTTPS* (3.1.3), unless otherwise specified.

3.1.2

TLS

transport layer security

protocol for secure communication over the internet

Note 1 to entry: TLS is defined in RFC 5246 (Version 1.2) and RFC 8446 (Version 1.3).

3.1.3

HTTPS

hypertext transfer protocol secure

HTTP (3.1.1) over *transport layer security* (TLS) (3.1.2)

Note 1 to entry: HTTPS is an extension of HTTP used for secure communication and is defined in RFC 2818.^[6] It uses TLS to encrypt the communication.

Note 2 to entry: In this document, the term HTTP is used to mean HTTP implemented over TLS, i.e. HTTPS, unless otherwise specified.

3.1.4

PKI

public key infrastructure

set of hardware, software, people, policies and procedures needed to create, manage, distribute, use, store and revoke digital certificates and manage public-key encryption

[SOURCE: ISO 20415:2019, 3.14]

3.1.5

digital signature

data appended to, or cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the source and integrity of the unit and protect against forgery by, for example, the recipient

[SOURCE: ISO 20415:2019, 3.5]

3.1.6

API

application programming interface

collection of communication methods and associated parameters used by a *client* (3.1.11) to exchange information with a *server* (3.1.12)

Note 1 to entry: In this document, the term API is used in the meaning of one specific arrow in the sequence diagram (see [Figure 2](#)). Thus, a session will normally require the use of several APIs. This document describes one general *API access point* (3.1.17) that can be used by all APIs by changing the number of message components as well as the contents of one or more message parts.

3.1.7

API access point

application programming interface access point

collection of a specific communication method, one or more general message parts, and related semantics for formatting the message parts that can be used to realize *application programming interfaces (APIs)* (3.1.6)

3.1.8

authorization

granting a *sender* (3.1.9) the right to use one or more *application programming interface access points* (3.1.7)

3.1.9

sender

party that initiates a *session* (3.1.18) by sending a *request* (3.1.15) to a receiver

[SOURCE: IMO FAL.5/Circ.46:2022, modified — the definition has been shortened.]

3.1.10

receiver

party that receives a *request* (3.1.15) from a *sender* (3.1.9), executes a *service* (3.1.14), and returns a status message

[SOURCE: IMO FAL.5/Circ.46:2022, modified — the definition has been shortened.]

3.1.11

client

unit that sends a message to a *server* (3.1.12) and expects a response

Note 1 to entry: The sender and receiver can be both a server and a client.

3.1.12

server

unit that listens for incoming messages from one or more *clients* (3.1.11) and responds to them

Note 1 to entry: The sender and receiver can be both a server and a client.

3.1.13

message

information payload of an *HTTP* (3.1.1) request or response between the *sender* (3.1.9) and *receiver* (3.1.10)

Note 1 to entry: Examples of messages are *request* (3.1.15), *message status* (3.1.16) and *service request status* (3.1.17).

Note 2 to entry: When the full HTTP request or response itself is referred to, the term HTTP message will be used.

3.1.14

service

result of a process performed by a *receiver* (3.1.10) after a *request* (3.1.15) has been delivered by a *sender* (3.1.9)

Note 1 to entry: In this document, service is a general term and includes that the receiver unconditionally accepts an incoming report without doing any processing of the content of the report.

3.1.15

request

message sent by a *sender* (3.1.9) to request a *service* (3.1.14)

[SOURCE: IMO FAL.5/Circ.46:2022, modified — the definition has been shortened.]

3.1.16

message status

message (3.1.13) sent from a *server* (3.1.12) to a *client* (3.1.11) to inform about the status of a received message

Note 1 to entry: A message status can be combined with a *service request status* (3.1.17) in one message.

Note 2 to entry: A message status will normally only consist of a message header. If the message status is combined with a service request status, the message can also contain other message parts.

3.1.17

service request status

message (3.1.13) sent from a *receiver* (3.1.10) to a *sender* (3.1.9) to inform about the status of a *service* (3.1.14)

Note 1 to entry: A service request status can be combined with a *message status* (3.1.16) in one message.

3.1.18

session

sequence of *messages* (3.1.13) that is related to one and the same initial *request* (3.1.15)

[SOURCE: IMO FAL.5/Circ.46:2022]

3.1.19

session context

one or more *sessions* (3.1.18) related to the same overarching operation or event

Note 1 to entry: Session contexts can be nested. As an example, several berth calls can be individual session contexts nested inside a port call session context, which may also be nested inside an entry into national waters session context.

3.1.20

HTTP session

hypertext transfer protocol session

message (3.1.13) exchanges that take place between establishment of an *HTTP* (3.1.1) connection and its closing

Note 1 to entry: One HTTP session can include sending more than one *request* (3.1.15).

3.1.21

MIG

message implementation guide

specific requirements to *message* (3.1.13) exchanges in conjunction with one or more concrete *services* (3.1.14)

3.1.22

EPC

electronic port clearance

services (3.1.14) a ship may request before, during, or after a port call

Note 1 to entry: EPC is used as a general term for all services a ship may request before, during, or after a port call. It is not limited to maritime single windows and other authority operated systems.

3.1.23

GLN

global location number

13-digit decimal number uniquely identifying a geographic position

Note 1 to entry: The list of registered numbers is maintained by GS1.^[17]

3.1.24

UN/LOCODE

United Nations code for trade and transport locations

two or three letter code identifying a country and a trade location within it

Note 1 to entry: The countries have a two-letter code as listed in ISO 3166-1. The locations in the respective countries have a three-letter code as maintained by UNECE.^[15]

3.1.25

MMSI

maritime mobile service identity

nine-digit decimal number uniquely identifying radiocommunication equipment

Note 1 to entry: MMSI is normally assigned by the national telecommunication authorities.

3.1.26

US-ASCII

7-bit character encoding defined in ISO/IEC 646

3.1.27

X.509

digital signature public certificate as defined in RFC 2459^[33]

3.2 Abbreviated terms

For the purposes of this document, the following abbreviated terms apply.

API	application programming interface
DER	Distinguished encoding rules: Binary X.509 certificate format.
EDIFACT	United Nations electronic data interchange for administration, commerce and transport (UN/EDIFACT)
EPC	Electronic port clearance
GISIS	Global integrated shipping information system ^[16]
GLN	Global location number ^[17]
GS1	International industry organization maintaining code lists for international business operations
	NOTE 1 This includes universal product codes (“bar codes”) and GLN (see https://www.gs1.org/).
HTTP	Hypertext transfer protocol
HTTPS	Hypertext transfer protocol secure
IMO	International Maritime Organization
ISPS	International ship and port facility security
	NOTE 2 A list of ISPS facility codes is available in GISIS ^[16] under the link to Maritime Security.
JSON	JavaScript ^a object notation
MIG	Message implementation guide
MIME	Multipurpose internet mail extensions (see RFC 2045 ^[7])
MMSI	Maritime mobile service identity
MRS	Mandatory ship reporting system (see IMO A.851(20) ^[8])
MSW	Maritime single window
PCS	Port community system

PKI	Public key infrastructure
PMIS	Port management information system
REST	Representational state transfer
TCP/IP	Transfer control protocol over internet protocol (transport protocol for HTTP)
URI	Uniform resource identifier
	NOTE 3 In this document, URI is sometimes used interchangeably with URL. URL normally refers to a specific URI while an URI is meant as a more generic term for any URL.
URL	Uniform resource locator
SMDG	Formerly ship messaging design group, now user group for electronic data interchange (EDI) in the maritime container business. Maintaining the SMDG terminal list. ^[29]
TOS	Terminal operating system
TLS	Transport layer security
UNECE	United Nations Economic Commission for Europe
UN/LOCODE	United Nations code for trade and transport locations (see UNECE Recommendation 16 ^[10] and list of codes ^[15])
VTIS	VTIS information system
VTS	Vessel traffic services
XML	Extensible markup language (see W3C Recommendation XML 1.0 ^[11])
XSD	XML schema definition language (defined in W3C recommendations XML schema structures, ^[12] and data types ^[13]).

^a JavaScript is a registered trademark of Oracle Corporation. This information is given for the convenience of users of this document and does not constitute an endorsement by ISO or IEC.

4 Structure of XSD data type and object definitions

4.1 General

Each message specified in the ISO 28005 series consists of different parts as described in 7.16. Some of these parts are in XML^[11] format. Each of the XML-formatted parts consist of several XML data objects with a corresponding XML data type definition.

The principles used in the ISO 28005 series for describing XML data types in XSD code are described in 4.2. The structure of the clauses in the ISO 28005 series that contain definitions of XML data types are described in 4.3. In addition to defining the syntax of the XML data objects in XSD code, 4.3 also describes how these clauses define the semantics of the data contained in the XML objects.

The XSD code is assembled into an XSD file that can be used for syntactic verification of XML message parts specified in the ISO 28005 series. In 4.4, the individual XSD code definitions are converted into a valid XSD file.

Data objects defined in ISO 28005-2:2021, which have been reused by definitions in this document are specified in 4.5.

4.2 Principles for XML descriptions in the ISO 28005 series

4.2.1 No use of XML attributes

The data types defined by this document do not use XML attributes. All information is contained within XML start and stop tags.

NOTE This makes it easy to create JSON versions from the XSD file defined in [4.4](#) by replacing XML tag names with the corresponding JSON object or attribute name.

4.2.2 Defaults for minOccurs and maxOccurs

The XSD specification^[13] defines the default values for minOccurs and maxOccurs equal to one. This definition is used in the XSD code described in [4.3.3](#) to shorten the type specifications in the data type definition clauses by sometimes omitting 'minOccurs = "1"' and 'maxOccurs = "1"' where these otherwise would be expected to occur.

4.2.3 Signalling empty XML tags

A client or server can signal that an XML tag is empty (not in use) by not including any content between the start and end XML tags, or by omitting the tags themselves, if the latter is allowed by the corresponding schema definition i.e. that "minOccurs=0" is defined for the tag.

4.2.4 Order of child elements in XSD files

The XSD templates use "<xs:sequence>" to list child elements. This means that the XML file shall have child elements in exactly the same order as the XSD code. Optional child elements (minOccurs = "0") can be omitted.

NOTE The XSD code is automatically generated. The order of child elements shown in the XSD parts of the data type definitions are not always logical.

4.2.5 Character set

The character set used in the XML-formatted parts of the messages specified in the ISO 28005 series (see [7.16](#)) is UTF-8.

4.2.6 Principles for defining types with code lists

Enumerated types, i.e. types that are associated with a fixed set of code values, are defined in one of the following ways.

- a) When the code set is small, defined in this document, and not likely to be extended, the permitted code values are listed in the definition of the data type as a token with constraints. This means that the permitted code values will be included in the final XSD file.
- b) If the code set is larger, defined in this document, and likely to be expanded in later editions of the relevant part of the ISO 28005 series, the data type is defined as an unconstrained token or number and the code set provided in tabular form. The permitted code values will not be included in the final XSD file.
- c) When the code set is not defined in the document but maintained by an organization, the data type is defined as an unconstrained token or number. There will be a reference to where the code set is defined, and if necessary, constraints as to how the code set shall be used in the context of the document. The allowed code values will not be included in the final XSD file.

4.2.7 XSD name space for general XSD data types

All basic data types defined in the XSD data type standard,^[14] and which are used in this document, use the name space “xs”. Thus, the data type name are prefixed with “xs”: This corresponds to the header of the XSD file defined in 4.4 including the following attribute:

```
<xs:schema ...  
  xmlns:xs="https://www.w3.org/2001/XMLSchema" ...
```

4.2.8 ISO 28005 name space

All data types defined in the ISO 28005 series are defined in the namespace “epc”. Thus, the data type name is prefixed with “epc”:

This corresponds to the XSD file header including the following attribute:

```
<xs:schema ...  
  xmlns:epc="https://standards.iso.org/iso/28005/" ...
```

4.2.9 Use of Xpath expressions

The ISO 28005 series uses Xpath expressions to identify individual data objects in XML message parts. The Xpath expression syntax is defined in the W3C Recommendation XML Path Language (Xpath).^[23] The data type `epc:xpath` (see 5.11) has been defined to hold such expressions when used in XML message parts.

EXAMPLE Identification of a company name in the message header `SenderId` object is `/EPCMessageHeader/SenderId/Company`. The full sender identity is `/EPCMessageHeader/SenderId`.

4.3 Structure of clauses defining data types

4.3.1 Clause and data type name

All data types defined in the ISO 28005 series are given a name that is also included as the first part of the clause heading where the data type is defined.

The data type name follows the specifications for XML tag names, with the following additional constraints.

- a) The data type name shall always end with the string “Type”.
- NOTE** Some data object names can also have the ending “Type”. In that case, the corresponding data type name will have a trailing “TypeType”.
- b) Enumerated data type names shall end with “ContentType” instead of only “Type”.
 - c) This document uses the “Upper Camel Case” in all core data types, i.e. the first letter is upper case and, when the tag name consists of a number of concatenated words, each of the words starts with an upper-case letter.
 - d) The name consists only of characters from the sets (A-Z), (a-z) and, exceptionally, (0-9). The names have been selected to be generally understandable in the context of ship-to-shore communication. The names use British English spelling without any special characters. Names are in singular form except where the data element contains a list of items, in which case the tag name is in plural form.

4.3.2 Definition

Each data type has a definition that is intended to give an unambiguous description of what the data element shall contain and in what context it is valid. The definition appears in the first subclause after the clause heading.

4.3.3 Type defined as XSD code

Each data type is defined as a section of XSD code.^[14] This section only covers the actual data type definition and is not a valid XML or XSD document in itself.

See [4.4](#) for details on how a complete XSD file shall be constructed from the type definitions' XSD code.

4.3.4 Representation

The representation paragraph consists of additional requirements that define the semantics of the data type.

This can include how the information is coded, e.g. if distances are in km or nautical miles, or references to code lists that shall be used for enumerated types.

4.4 Creating valid XSD schema files

4.4.1 File structure

The minimum XSD format for defining the XSD schema file is shown in the below example, including comments starting with “<!--” that separate the different parts of the file from each other.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="https://www.iso.org/28005-1"
           xmlns:xs="https://www.w3.org/2001/XMLSchema"
           xmlns:epc="https://www.iso.org/28005">

<!--The definition of individual data types>

    <xs:complexType name="DataElement1Type"> ... </xs:complexType>
    <xs:simpleType name="DataElement2Type"> ... </xs:simpleType>
    ...
<!--The definition of the message part types>

    <xs:complexType name="EPCMessageHeaderType">
        ...
    </xs:complexType>
    <xs:complexType name="EPCMessageBodyType">
        ...
    </xs:complexType>
    <xs:complexType name="EPCMessageSignatureType">
        ...
    </xs:complexType>

<!--The instantiation of the message parts>

    <xs:element name="EPCMessageHeader"
               type="EPCMessageHeaderType"/>
    <xs:element name="EPCMessageBody" type="EPCMessageBodyType"/>
    <xs:element name="EPCMessageSignature"
               type="EPCMessageSignatureType"/>

</xs:schema>
```

The first four lines is the general preamble in an XSD file. This specifies encoding and any name-spaces being used. UTF-8 encoding shall be used (see also [9.5](#)).

The subsequent lines define the different individual data types. All data types which are defined in this document, as well as in the data definition clauses of other parts of the ISO 28005 series, are expected to be included here.

The next lines define the three different message parts. Currently, these are the ones listed in [Table 1](#).

Table 1 — Message parts defined in the XSD file

Part	Clause	Message part
EPCMessageHeader	10.10	The message header
EPCMessageBody	11.2	The message body part
EPCMessageSignature	14.3	The message signature part

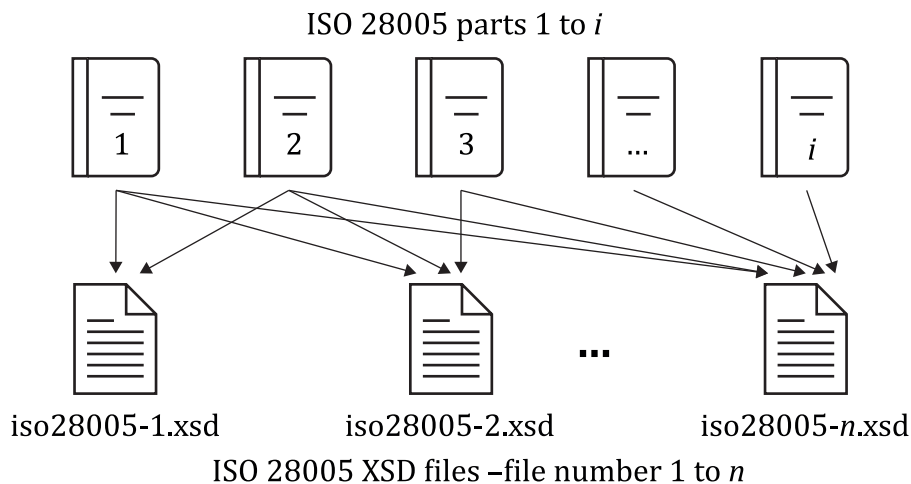
The final part is the definitions related to the instantiation of the message parts.

4.4.2 Numbering of XSD files and message version code

Each part of the ISO 28005 series can contain definitions that change the structure of the XSD file. In this case, the relevant part of the ISO 28005 series is expected to be associated with a new XSD file number as illustrated in [Figure 1](#).

Each XSD file contains definitions for all message parts listed in [Table 1](#) as well as all necessary data type that are used by those message parts. Definitions from one part of the ISO 28005 series will be included in all following XSD file numbers. The first XSD file number that each part in the ISO 28005 series is associated with is specified in [Annex A](#) of that part. See [11.2.2](#) for more details.

NOTE [Figure 1](#) is for illustration only. The actual assignment of XSD file numbers is determined by the parts that together define the related XSD file.

**Figure 1 — Structure of XSD files related to the ISO 28005 series**

The message version code is a mandatory part of the message header (see [10.10](#)). It is a string consisting of two or three integer numbers separated by a full stop (see [6.19](#)). These numbers, from left to right shall have the following values:

1. The first number shall be '2'. This corresponds to the protocol and message specifications contained in this document.
2. The second number shall correspond to the XSD file number to which the XML message is designed to be conformant. The data elements listed in ISO 28005-2:2021, Annex A as well as the elements listed in [Annex A](#) shall all be included in file number 1.
3. The optional third number can be used for local variants of the XSD file that are not defined by the ISO 28005 series. See [11.2](#) for a description of this mechanism.

The structure of the message header and body is defined so that clients that are designed to understand a specific XSD file number, safely can validate XML messages with the same or lower file number by using the XSD file. Lower numbered XSD files cannot be used to validate higher numbered XML messages.

If the client's parser is constructed to only look for the tags in one version of an XSD file, a client can safely parse also higher numbered XML files if unknown tags are discarded. A client that is designed to understand a higher number XSD file than indicated by an incoming message version code can parse the incoming message, but shall in that case parse the message in accordance with the XSD file number that the incoming message version code specifies, i.e. it shall not look for tags that are only defined in higher numbered XSD files.

NOTE This capability is ensured by only adding new data objects or attributes. Data objects or attributes will not be deleted from definitions, but some objects or attributes can be depreciated in other parts of the ISO 28005 series.

It is always possible to validate local variants with the correspondingly numbered XSD file. However, this will not check the syntax of any additional elements in the local variant. To do this, the XSD file corresponding to the local variant is required.

4.4.3 Location of XSD files

The XSD files can be found an address patterned after the following: <https://standards.iso.org/iso/28005/-n/ed-m/en/>. This applies to all parts of the ISO 28005 series, where the desired part number ("n") and edition ("m") is replaced with the relevant numbers in the URL.

EXAMPLE The XSD file corresponding to the definitions in ISO 28005-2:2021 is available on <https://standards.iso.org/iso/28005/-2/ed-2/en/>.

4.5 Reference to data types defined in ISO 28005-2:2021

All adapted XSD data types defined in ISO 28005-2:2021, Clause 5 have been reproduced in [Clause 5](#). Several general data types in ISO 28005-2:2021.

Some of the definitions from ISO 28005-2:2021 have been extended in this document. A summary of the changes is listed in [Table 2](#). Definitions in ISO 28005-2:2011¹⁾ remain valid but the old data types listed in [Table 2](#) may have been depreciated by newer definitions in this document, which should be used for new implementations.

Table 2 — Data types modified from ISO 28005-2:2021

Data type	Definition in ISO 28005-2:2021	Definition in this document	Changes made
epc:AuthenticatorType	7.2.6	6.2	Changed authenticator role code list.
epc:ContactInfoType	6.3	6.4	Added URL attribute
epc:LocationType	7.10	6.9	Added GLN and MRN attributes.
epc:PostalAddressType	6.11	6.12	Changed PostCodeCode to PostCode
epc:EPCMessageHeaderType	7.2.2	10	Redefined (including all sub-types)
epc:EPCRequestBodyType	Annex A	11.2.1	Redefined and changed to EPCMessageBodyType
epc:JournalNumberType	7.9.22	6.16	Changed type name to epc:ReferenceCodeType (Ship stay reference)
epc:MessageTypeContentType epc:OtherServiceRequestType	7.7.2, 7.7.3	10	Replaced by epc:MessageFunctionCodeContentType epc:ServiceTypeCodeContentType and epc:ServiceCodeContentType New code values have been defined in Annex G .
epc:RequestStatusType	7.7.4	10	Replaced by epc:StatusType

1) Withdrawn.

5 Adapted XSD data types for ISO 28005

5.1 General

The adapted XSD data types used in the ISO 28005 series are based on the standard XML types defined in XML Schema Part 2.^[13]

5.2 epc:anyURI – Generalized URI

5.2.1 Definition

This data type contains a valid generalized URI. This may be a URL, prefixed by “http”: or “https”:, or a file name, prefixed by “file”:

5.2.2 Type

```
<xs:simpleType name="anyURI">
  <xs:restriction base="xs:anyURI"/>
</xs:simpleType>
```

5.2.3 Representation

All generalized URIs are permitted in this document. To ensure compatibility with older systems, URLs are restricted to use only the ASCII character set.

NOTE The “file”: type URI is used to refer to message attachments as described in [12.2](#).

5.3 epc:boolean – Boolean flag

5.3.1 Definition

This data type contains a flag that can have the logical values true or false.

5.3.2 Type

```
<xs:simpleType name="boolean">
  <xs:restriction base="xs:boolean"/>
</xs:simpleType>
```

5.3.3 Representation

All flag values allowed in the XSD standard are also allowed in this document. When the boolean type is used in contexts where the value represents an answer to a yes/no question, the true value shall represent “yes” while the false value shall represent “no”.

5.4 epc:date – General date

5.4.1 Definition

This data type contains a date without additional time of day or time zone information.

5.4.2 Type

```
<xs:simpleType name="date">
  <xs:restriction base="xs:date"/>
</xs:simpleType>
```

5.4.3 Representation

This is a date in the standard XSD format, without any time zone code in the value.

EXAMPLE “2002-09-10” (10th of September 2002).

Senders of date information should not include time zone information. Receivers should be prepared to accept a time zone code, but it shall be disregarded in further processing of the data.

5.5 epc:dateTime – Time and date, with time zone

5.5.1 Definition

This data type contains a date with additional time of day and time zone information.

5.5.2 Type

```
<xs:simpleType name="dateTime">
  <xs:restriction base="xs:dateTime"/>
</xs:simpleType>
```

5.5.3 Representation

This is a date and time in the standard XSD format, with a time zone code in the value.

Senders of date and time information shall include time zone information. Receivers should be prepared to accept values without time zone codes. In this case, the time zone is undefined and proper actions should be taken by the systems processing this data to ensure that this does not cause problems for the intended operation.

EXAMPLE 1 “2012-02-03T23:00:00+02:00” (3 February 2012, local time 23:00 hours in time zone UTC plus two hours).

EXAMPLE 2 “2012-02-03T21:00Z” (same time as Example 1 but transmitted as being in UTC).

NOTE The hour field can be 24 if minutes and seconds are zero. The seconds field can sometimes contain the value 60 (when leap seconds occur). Seconds can contain a decimal part followed by a period (“.”).

5.6 epc:decimal – decimal number

5.6.1 Definition

This data type is used to specify a quantity.

5.6.2 Type

```
<xs:simpleType name="decimal">
  <xs:restriction base="xs:decimal"/>
</xs:simpleType>
```

5.6.3 Representation

The decimal type represents a subset of the real numbers, which can be represented by decimal numerals. The value space of the decimal is the set of numbers that can be obtained by multiplying an integer i by a non-positive power n of 10, i.e. expressible as $i \times 10^{-n}$ where i and n are integers and $n \geq 0$. Precision is not reflected in this value space; the number 2.0 is not distinct from the number 2.00.

The decimal has a lexical representation consisting of a finite-length sequence of decimal digits separated by a period (“.”) as a decimal indicator. An optional leading sign is allowed. If the sign is omitted, “+” is assumed.

Leading and trailing zeros are optional. If the fractional part is zero, the period and following zeros can be omitted.

EXAMPLE -1.23, 12678967.543233, +100000.00, and 210 are all valid decimal numbers

NOTE The XSD format used in this document requires the use of the period (".") as the decimal sign.

5.7 epc:duration – Time duration

5.7.1 Definition

This data type is used to specify a duration in time.

5.7.2 Type

```
<xs:simpleType name="duration">
  <xs:restriction base="xs:duration"/>
</xs:simpleType>
```

5.7.3 Representation

This type can specify a time period. The general format is "[–]P n Y n M n D n H n M n [. n]S", where [–] is an optional minus sign, n is a positive integer number, and [n] is an optional decimal field for seconds. The number n ahead of Y, M, D, H and M, respectively, indicates the length of the duration in years, months, days, hours and minutes. If n is zero, it can be omitted together with the letter it precedes.

EXAMPLE P3DT10H30M is three days, 10 hours and 30 minutes; -P120D is a negative duration of 120 days.

5.8 epc:int – Integer number

5.8.1 Definition

This data type is used to specify an integer quantity.

5.8.2 Type

```
<xs:simpleType name="int">
  <xs:restriction base="xs:int"/>
</xs:simpleType>
```

5.8.3 Representation

The integer type is an integer in the range from -2147483648 to 2147483647 (inclusive).

5.9 epc:string – General string

5.9.1 Definition

This data type contains a general string that is mainly intended to be read by humans. There are no restrictions on the format of the string. See `epc:token` for a string type that is intended for computer use.

5.9.2 Type

```
<xs:simpleType name="string">
  <xs:restriction base="xs:string"/>
</xs:simpleType>
```

5.9.3 Representation

This is an UTF-8 character string where white space (space, tab, line feed etc.) is preserved.

Applications can put length constraints on the string. The server should be prepared to accept longer strings than what is specified by the application, but longer strings can be truncated. Thus, the client cannot rely on longer strings being fully processed or displayed.

5.10 epc:token – Computer-understandable string

5.10.1 Definition

The token data type is normally used for a text string that is mainly meant to be interpreted by a computer. This includes enumerated codes as well as other structured strings.

5.10.2 Type

```
<xs:simpleType name="token">
  <xs:restriction base="xs:token"/>
</xs:simpleType>
```

5.10.3 Representation

This is represented as a text string that can consist of letters or numbers or both, no leading or trailing spaces and only single spaces inside the string. The use of this data type will also require the definition of the legal tokens or code values.

5.11 epc:xpath – Identification of an XML data item

5.11.1 Definition

The xpath data type is used to identify an XML object in one of the message parts.

5.11.2 Type

```
<xs:simpleType name="xpath">
  <xs:restriction base="xs:token"/>
</xs:simpleType>
```

5.11.3 Representation

This data type is used to identify an XML data object by use of Xpath expressions (see [4.2.9](#) for the definition of the Xpath format).

6 General ISO 28005 data types

6.1 General

This clause defines a number of general data types that is used both in the message header and message body.

NOTE Some of these definitions can also be found in ISO 28005-2:2021. See [4.5](#) for an overview.

6.2 epc:AuthenticatorType – Authenticator of information

6.2.1 Definition

This type contains information about the party attesting to the validity of the transmitted information.

6.2.2 Type

```
<xs:complexType name="AuthenticatorType">
  <xs:complexContent>
    <xs:extension base="epc:ContactInfoType">
      <xs:sequence>
        <xs:element name="AuthenticationDate" type="epc:dateTime"
          minOccurs="0"/>
        <xs:element name="AuthenticatorLocation"
          type="epc:LocationType" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

6.2.3 Representation

The element is an extension of the general contact information type and shall contain the additional data elements:

- **AuthenticationDate:** This is the date and time of the authentication.
- **AuthenticatorLocation:** This is the location of the person when submitting the information.

The “Authenticator party identification code” (IMO 0017 in [14]) use the `CompanyId` attribute in the `ContactInfoType` part of the definition. The code values that shall be used are defined in [Annex I](#).

The “Authenticator role, coded” (IMO 0128 in [14]) use the `ContactType` attribute in the `ContactInfoType` part of the definition. The code values that shall be used are defined in [Annex I](#).

6.3 epc:AuthorizationTokenType – Authorization token

6.3.1 Definition

This type is used to send an authorization token between the sender and receiver.

6.3.2 Type

```
<xs:simpleType name="AuthorizationTokenType">
  <xs:restriction base="epc:token"/>
</xs:simpleType>
```

6.3.3 Representation

The token is used to authorize access to the service API. It is retrieved through a special API as described in [Clause 18](#). The token is a text string that only has meaning for the receiver. It should be long and randomly scrambled to make it highly unlikely for other parties to replicate it.

NOTE The API that is used to retrieve and send the authorization codes use secure HTTP and is assumed to be very difficult to eavesdrop on for hostile third parties. However, the sender and receiver are expected to take measures as necessary to protect the authorization code from third parties.

6.4 epc:ContactInfoType – Contact information

6.4.1 Definition

This data type contains contact information for either a person or a company. A person's name can be included if the contact information is for a company.

6.4.2 Type

```
<xs:simpleType name="ContactTypeContentCode">
  <xs:restriction base="xs:token"/>
</xs:simpleType>
<xs:complexType name="ContactInfoType">
  <xs:sequence>
    <xs:element name="Company" type="epc:string" minOccurs="0"/>
    <xs:element name="CompanyId" type="epc:string" minOccurs="0"/>
    <xs:element name="ContactNumbers"
      type="epc:CommunicationNumberType" minOccurs="0"/>
    <xs:element name="ContactType" type="epc:ContactTypeContentCode"
      minOccurs="0"/>
    <xs:element name="Person" type="epc:NameType" minOccurs="0"/>
    <xs:element name="Address" type="epc:PostalAddressType"
      minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

6.4.3 Representation

This is information about a company or person that can be physically contacted. Common information for both persons and companies are addresses and contact numbers. Companies and persons differ in that companies have one name in a string, while persons can have a given name, family name and middle name. A person's name can be submitted for a company contact point, but not vice versa.

The attributes are:

- **Company:** This is the official name of a company.
- **CompanyId:** The recognized identification number of the company, if defined.

NOTE *CompanyId* is normally a business-related. It can be a tax identification number or a local identification number e.g. used in a specific port.
- **ContactNumbers:** Telephone, telefax or other communication channel information.
- **ContactType:** The role of the contact point. This attribute may be required in conjunction with dangerous goods reporting or when the contact point is used as authenticator (see 6.2). The code shall be taken from UN/EDIFACT code lists 3035 or 3139. Codes with a defined meaning in this document are listed in [Table I.1](#).
- **Person:** The name of the person.
- **Address:** The postal address of person or company.

6.5 epc:CommunicationNumberType – Communication number information

6.5.1 Definition

This data type specifies a contact point via telephone or other means.

6.5.2 Type

```
<xs:complexType name="CommunicationNumberType">
  <xs:sequence>
    <xs:element name="BusinessTelephone" type="epc:string"
      minOccurs="0"/>
    <xs:element name="ContactURL" type="epc:anyURI" minOccurs="0"/>
    <xs:element name="EMail" type="epc:anyURI" minOccurs="0"/>
    <xs:element name="HomeTelephone" type="epc:string" minOccurs="0"/>
    <xs:element name="MobileTelephone" type="epc:string"
      minOccurs="0"/>
    <xs:element name="Telefax" type="epc:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

```
</xs:sequence>
</xs:complexType>
```

6.5.3 Representation

This element contains a list of contact points for a person or organization. Telephone numbers shall be specified with an international prefix code in the format “+*it*” where *i* is the international prefix and *t* the telephone number without any spaces between or inside either.

The attributes are:

- **BusinessTelephone**: Daytime telephone number.
- **ContactURL**: A web page where information about the party can be obtained.
- **Email**: Email address as URI, i.e. prefixed with “mailto”:
- **HomeTelephone**: Home telephone number.
- **MobileTelephone**: Telephone number for access outside office.
- **Telefax**: Telefax number.

6.6 epc:CountryCodeContentType – Country identification

6.6.1 Definition

This data type gives a unique and coded representation of a country identity.

6.6.2 Type

```
<xs:simpleType name="CountryCodeContentType">
  <xs:restriction base="xs:token">
    <xs:length value="2"/>
  </xs:restriction>
</xs:simpleType>
```

6.6.3 Representation

The content of the country code element shall be the two-letter country code defined in ISO 3166-1.

6.7 epc:CountrySubdivisionCodeContentType – Country subdivision identification

6.7.1 Definition

This data type gives a unique and coded representation of a country subdivision identity.

6.7.2 Type

```
<xs:simpleType name="CountrySubdivisionCodeContentType">
  <xs:restriction base="xs:token"/>
</xs:simpleType>
```

6.7.3 Representation

The country subdivision code is the final characters of the ISO 3166-2 country subdivision code, following the initial two-letter country code and the hyphen-minus.

NOTE The definition of the subdivision code is a length of maximum six characters where the two first are the country code followed by a hyphen-minus (see ISO 3166-2).

If no code is defined in ISO 3166-2 for the subdivision in question, it is permitted to use a commonly agreed code between the sender and receiver.

6.8 epc:CrewDutyType – Duty onboard or on shore

6.8.1 Definition

This data type specifies the duty of a person.

6.8.2 Type

```
<xs:simpleType name="CrewDutyCodeContentType">
  <xs:restriction base="epc:token">
    </xs:restriction>
  </xs:simpleType>

<xs:complexType name="CrewDutyType">
  <xs:sequence>
    <xs:element name="Code" type="epc:CrewDutyCodeContentType"
      minOccurs="0" />
    <xs:element name="Text" type="epc:string" minOccurs="0" />
  </xs:sequence>
</xs:complexType>
```

6.8.3 Representation

The two fields shall be used as follows:

- Code: This is the crew duty type in coded representation. The code list is maintained in the IMO Compendium^[14] under IMO object code IM00043.
- Text: This is the crew duty in free text. It is mandatory only if a valid coded value is not supplied.

If the text-field is included and if the crew duty has a code corresponding to a code in the IMO Compendium, the text field should contain the same textual description as in the IMO Compendium.

6.9 epc:LocationType – Identification of a location

6.9.1 Definition

This data type identifies any location, which may also be given a name. This data type is normally used in a specific context to define what type of location this is.

6.9.2 Type

```
<xs:simpleType name="FacilityNameType">
  <xs:list itemType="epc:token"/>
</xs:simpleType>

<xs:simpleType name="GLNCodeContentType">
  <xs:restriction base="epc:token"/>
</xs:simpleType>

<xs:simpleType name="GISISCodeContentType">
```

```

    <xs:restriction base="xs:token"/>
</xs:simpleType>

<xs:complexType name="PositionType">
  <xs:sequence>
    <xs:element name="Latitude" type="epc:LatitudeType"/>
    <xs:element name="Longitude" type="epc:LongitudeType"/>
  </xs:sequence>
</xs:complexType>

<xs:simpleType name="LatitudeType">
  <xs:restriction base="epc:decimal">
    <xs:minInclusive value="-90"/>
    <xs:maxInclusive value="90"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="LongitudeType">
  <xs:restriction base="epc:decimal">
    <xs:minInclusive value="-180"/>
    <xs:maxInclusive value="180"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="UNLoCodeContentType">
  <xs:restriction base="epc:token">
    <xs:length value="3"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="MRNCodeContentType">
  <xs:restriction base="epc:token">
    </xs:restriction>
</xs:simpleType>

<xs:simpleType name="SMDGTerminalCodeContentType">
  <xs:restriction base="epc:token">
    </xs:restriction>
</xs:simpleType>

<xs:complexType name="LocationType">
  <xs:sequence>
    <xs:element name="CountryCode" type="epc:CountryCodeContentType"
      minOccurs="0"/>
    <xs:element name="CountrySubdivisionCode"
      type="epc:CountrySubdivisionCodeContentType"
      minOccurs="0"/>
    <xs:element name="FacilityCode" type="epc:GISISCodeContentType"
      minOccurs="0"/>
    <xs:element name="FacilityName" type="epc:FacilityNameType"
      minOccurs="0"/>
    <xs:element name="GLN" type="epc:GLNCodeContentType"
      minOccurs="0"/>
    <xs:element name="MRN" type="epc:MRNCodeContentType"
      minOccurs="0"/>
    <xs:element name="Name" type="xs:string" minOccurs="0"/>
    <xs:element name="NauticalMilesToDestination" type="epc:decimal"
      minOccurs="0"/>
    <xs:element name="Position" type="epc:PositionType" minOccurs="0"/>
    <xs:element name="SMDGTerminalCode"
      type="epc:SMDGTerminalCodeContentType" minOccurs="0"/>
    <xs:element name="UNLoCode" type="epc:UNLoCodeContentType"
      minOccurs="0"/>
    <xs:element name="VisualPosition" type="epc:VisualPositionType"
      minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

```

6.9.3 Representation

The location type allows the user to specify a location by one or more of the following attributes:

- **CountryCode**: Two-letter country code defined in ISO 3166-1.
- **CountrySubdivisionCode**: Country subdivision code defined in ISO 3166-2 (see [6.7](#) for code values).
- **FacilityCode**: The port facility 4-digit code defined in GISIS.^[16] Not all facilities are listed in GISIS. In these cases, the facility name field shall be used instead.
- **FacilityName**: The name of an ISPS facility in a port or terminal. This field is mandatory if the **FacilityCode** is not provided.
- **GLN**: Global location number of the position.^[17]
- **MRN**: Marine Resource Name - a text string that identifies a specific marine resource, including positions.^[28]
- **Name**: This is a free text name that can be used either as an alternative to the other code types or as an optional name on the location.
- **NauticalMilesToDestination**: As given by name.
- **Position**: Geographic position specifying the latitude and longitude. The latitude and longitude shall be represented using the degrees and decimal degree format specified by ISO 6709, i.e. { ± }dd.dd for latitude and { ± }ddd.dd for longitude, where "+" (plus) indicates north or east and "-" (minus) indicates south or west. The "+" (plus) sign can be omitted.
- **SMDGTerminalCode**: A terminal code defined by SMDG and maintained in their code list.^[29]
- **UNLoCode**: Three letter national location code for a port as defined in UN/LOCODE.^[15]
- **VisualPosition**: A free text description of location relative to landmarks that can be seen from the reported position.

6.10 epc:NameType – Name of person

6.10.1 Definition

This data type contains the full name of a person.

6.10.2 Type

```
<xs:complexType name="NameType">
  <xs:sequence>
    <xs:element name="FamilyName" type="epc:string" minOccurs="0" />
    <xs:element name="GivenName" type="epc:string" minOccurs="0" />
    <xs:element name="MiddleName" type="epc:string" minOccurs="0" />
  </xs:sequence>
</xs:complexType>
```

6.10.3 Representation

This shall be represented as three text strings. Strings are in free text and possibly not suitable for computer-based comparisons.

NOTE Users of this data element can impose additional restrictions on the data type, e.g. by requiring the spelling to be identical to that used in an electronically readable identification document.