Open Architecture Radar Interface Standard,

v2.0 - change bar version

OMG Document Number: formal/23-09-02
Standard document URL: http://www.omg.org/spec/OARIS/2.0
https://www.omg.org/spec/OARIS/3.0
Machine Consumable File(s): https://www.omg.org/OARIS/20230201
Associated Normative Machine Consumable Files:
https://www.omg.org/spec/OARIS/20230501/oaris.xml/
https://www.omg.org/spec/OARIS/20230501/oaris.graphqls/
https://www.omg.org/spec/OARIS/20230501/IDL/TimeBase.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Common_Types.idl
https://www.omg.org/spec/OARIS/20230501/IDL/System_Track.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Coordinates_and_Positions.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Shape_Model.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Requests.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Encyclopaedic_Support.idl
https://www.omg.org/spec/OARIS/20230501/IDL/ExtendedSubsystem_Control.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Recording_and_Replay.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Simulation_Support.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Subsystem_Control.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Clutter_Reporting.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Media_Streaming.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Sensor_Assessment.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Supplementary_Measurement.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Search.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Plot_Reporting.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Subsystem_Control.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Track_Reporting.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Tracking_Control.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Air_Engagement_Support.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Engagement_Support.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Missile_Guidance.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Surface_Engagement_Support.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Manage_Network_Participation.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Manage_Physical_Configuration.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Perform_Offline_Test.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Provide_Networking_Statistics.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Restart.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Shutdown.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Startup.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Control_Recording.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Control_Replay.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Define_Simulation_Scenario.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Control_Simulation.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Define_Fault_Scripts.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Control_Fault_Scripts.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Manage_Technical_State.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Provide_Subsystem_Identification.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Provide_Health_State.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Manage_Operational_Mode.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Control_Battle_Override.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Manage_Subsystem_Parameters.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Provide_Subsystem_Services.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Manage_Mastership.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Manage_Mastership.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Register_Interest.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Provide_Area_with_Plot_Concentration.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Provide_Clutter_Assessment.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Allocate_Tracks_To_Stream.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Perform_Cued_Search.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Assess_Sensor_Track.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Filter_Plots.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Provide_Sensor_Characteristics.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Manage_Frequency_Usage.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Manage_Transmission_Sectors.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Control_Emissions.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Define_Test_Target_Scenario.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Test_Target_Facility.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Provide_Interference_Reports.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Provide_Nominal_Performance.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Provide_Performance_Assessment.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Provide_Jammer_Assessment.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Filter_Tracks.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Label_Tracks.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Provide_Sensor_Tracks.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Delete_Sensor_Track.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Receive_Track_Information.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Initiate_Track.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Manage_Tracking_Zones.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Provide_Projectile_Positional_Information.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Process_Target_Designation.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Support_Kill_Assessment.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Support_Surface_Target_Engagement.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Perform_Missile_Downlink.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Perform_Missile_Uplink.idl
https://www.omg.org/spec/OARIS/20230501/IDL/Perform_Splash_Spotting.idl
This OMG document replaces the submission document (c4i/23-05-01). It is an OMG Adopted Beta Specification and is currently in the finalization phase. Comments on the content of this document are welcome and should be directed to issues@omg.org by February 28, 2024.

You may view the pending issues for this specification from the OMG revision issues web page https://issues.omg.org/issues/lists.

The FTF Recommendation and Report for this specification will be published in September 2024. If you are reading this after that date, please download the available specification from the OMG Specifications Catalog.
USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

The material in this document details an Object Management Group specification in accordance with the terms, conditions and notices set forth below. This document does not represent a commitment to implement any portion of this specification in any company's products. The information contained in this document is subject to change without notice.

LICENSES

The company listed above have granted to the Object Management Group, Inc. (OMG) a nonexclusive, royalty-free, paid up, worldwide license to copy and distribute this document and to modify this document and distribute copies of the modified version. Each of the copyright holders listed above has agreed that no person shall be deemed to have infringed the copyright in the included material of any such copyright holder by reason of having used the specification set forth herein or having conformed any computer software to the specification.

Subject to all of the terms and conditions below, the owners of the copyright in this specification hereby grant you a fully-paid up, non-exclusive, nontransferable, perpetual, worldwide license (without the right to sublicense), to use this specification to create and distribute software and special purpose specifications that are based upon this specification, and to use, copy, and distribute this specification as provided under the Copyright Act; provided that:

(1) both the copyright notice identified above and this permission notice appear on any copies of this specification;
(2) the use of the specifications is for informational purposes and will not be copied or posted on any network computer or broadcast in any media and will not be otherwise resold or transferred for commercial purposes; and (3) no modifications are made to this specification. This limited permission automatically terminates without notice if you breach any of these terms or conditions. Upon termination, you will destroy immediately any copies of the specifications in your possession or control.

PATENTS

The attention of adopters is directed to the possibility that compliance with or adoption of OMG specifications may require use of an invention covered by patent rights. OMG shall not be responsible for identifying patents for which a license may be required by any OMG specification, or for conducting legal inquiries into the legal validity or scope of those patents that are brought to its attention. OMG specifications are prospective and advisory only. Prospective users are responsible for protecting themselves against liability for infringement of patents.
GENERAL USE RESTRICTIONS

Any unauthorized use of this specification may violate copyright laws, trademark laws, and communications regulations and statutes. This document contains information, which is protected by copyright. All Rights Reserved. No part of this work covered by copyright herein may be reproduced or used in any form or by any means--graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems--without permission of the copyright owner.
DISCLAIMER OF WARRANTY

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE OBJECT MANAGEMENT GROUP AND THE COMPANIES LISTED ABOVE MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE. IN NO EVENT SHALL THE OBJECT MANAGEMENT GROUP OR ANY OF THE COMPANIES LISTED ABOVE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The entire risk as to the quality and performance of software developed using this specification is borne by you. This disclaimer of warranty constitutes an essential part of the license granted to you to use this specification.

RESTRICTED RIGHTS LEGEND

Use, duplication or disclosure by the U.S. Government is subject to the restrictions set forth in subparagraph (c) (1) (ii) of The Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013 or in subparagraph (c)(1) and (2) of the Commercial Computer Software - Restricted Rights clauses at 48 C.F.R. 52.227-19 or as specified in 48 C.F.R. 227-7202-2 of the DoD F.A.R. Supplement and its successors, or as specified in 48 C.F.R. 12.212 of the Federal Acquisition Regulations and its successors, as applicable. The specification copyright owners are as indicated above and may be contacted through the Object Management Group, 9C Medway Road, PMB 274, Milford, MA 01757, U.S.A.

TRADEMARKS

IMM®, MDA®, Model Driven Architecture®, UML®, UML Cube logo®, OMG Logo®, CORBA® and XMI® are registered trademarks of the Object Management Group, Inc., and Object Management Group™, OMG™, Unified Modeling Language™, Model Driven Architecture Logo™, Model Driven Architecture Diagram™, CORBA logos™, XMI Logo™, CWM™, CWM Logo™, IIOP™, MOF™, OMG Interface Definition Language (IDL)™, and OMG SysML™ are trademarks of the Object Management Group. All other products or company names mentioned are used for identification purposes only, and may be trademarks of their respective owners.

COMPLIANCE

The copyright holders listed above acknowledge that the Object Management Group (acting itself or through its designees) is and shall at all times be the sole entity that may authorize developers, suppliers and sellers of computer software to use certification marks, trademarks or other special designations to indicate compliance with these materials.

Software developed under the terms of this license may claim compliance or conformance with this specification if and only if the software compliance is of a nature fully matching the applicable compliance points as stated in the specification. Software developed only partially matching the applicable compliance points may claim only that the software was based on this specification, but may not claim compliance or conformance with this specification. In the event that testing suites are implemented or approved by Object Management Group, Inc., software developed using this specification may claim compliance or conformance with the specification only if the software satisfactorily completes the testing suites.
OMG’s Issue Reporting Procedure

All OMG specifications are subject to continuous review and improvement. As part of this process we encourage readers to report any ambiguities, inconsistencies, or inaccuracies they may find by completing the Issue Reporting Form listed on the main web page http://www.omg.org, under Documents, Report a Bug/Issue (http://www.omg.org/report_issue).
## Table of Contents

1. Scope ............................................................................................................... 29  
2. Conformance .................................................................................................. 29  
3. Normative References ................................................................................... 30  
4. Terms and Definitions ................................................................................... 31  
5. Symbols ........................................................................................................... 32  
6. Additional Information ................................................................................. 32  
   6.1 Acknowledgements ..................................................................................................... 32  
   6.2 Specification Generation ................................................................................................. 33  
7. Open Architecture Radar Information Specification ................................ 34  
   7.1 Introduction ..................................................................................................................... 34  
   7.1.1 Background.................................................................................................................. 34  
   7.1.2 Section Structure ........................................................................................................ 35  
   7.2 Document Structure ........................................................................................................ 36  
   7.3 Usage Overview................................................................................................................ 36  
   7.3.1 Compliance Level 1 ..................................................................................................... 37  
   7.3.1.1 Basic Picture Compiler .......................................................................................... 38  
   7.3.1.2 Basic Sensor .......................................................................................................... 38  
   7.3.2 Compliance Level 2 ..................................................................................................... 38  
   7.3.2.1 General CMS Sensor Manager .............................................................................. 43  
   7.3.2.2 General Sensor Configuration Manager ................................................................ 43  
   7.3.3 Compliance Level 3A .................................................................................................. 43  
   7.3.3.1 CMS Analysis Data Manager ................................................................................ 48  
   7.3.3.2 CMS Diagnostics Manager .................................................................................... 48  
   7.3.3.3 CMS On Board Training Manager ....................................................................... 48  
   7.3.3.4 Subsystem Diagnostics Manager ........................................................................... 49  
   7.3.3.5 Subsystem Record and Replay Manager ............................................................... 49  
   7.3.3.6 Subsystem Simulation Manager ............................................................................ 49  
   7.3.4 Compliance Level 3B .................................................................................................. 49  
   7.3.4.1 CMS Combat System Configuration Manager ...................................................... 53  
   7.3.4.2 CMS Subsystem Manager ..................................................................................... 53  
   7.3.4.3 Subsystem Configuration Manager ....................................................................... 54  
   7.3.4.4 Subsystem Physical Configuration Manager ......................................................... 54  
   7.3.5 Compliance Level 3C .................................................................................................. 54  
   7.3.5.1 CMS Sensor Picture Manager ............................................................................... 57  
   7.3.5.2 Sensor Track Reporter ........................................................................................... 57  
   7.3.6 Compliance Level 3D .................................................................................................. 57  
   7.3.6.1 CMS AAW Engagement Coordinator ................................................................. 61  
   7.3.6.2 CMS ASuW Engagement Coordinator ................................................................. 62  
   7.3.6.3 Radar AAW Engagement Support Manager ......................................................... 62  
   7.3.6.4 Surface Fire Control Radar Manager ..................................................................... 62  
   7.3.6.5 Surface Surveillance Radar Engagement Support Manager .................................. 62  
   7.3.7 Compliance Level 3E .................................................................................................. 62
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.3.7.1 CMS Combat System Performance Optimizer</td>
<td>65</td>
</tr>
<tr>
<td>7.3.7.2 CMS Interference Mitigation Coordinator</td>
<td>65</td>
</tr>
<tr>
<td>7.3.7.3 Sensor Functional Performance Manager</td>
<td>65</td>
</tr>
<tr>
<td>7.3.7.4 Sensor Interference Reporter</td>
<td>65</td>
</tr>
<tr>
<td>7.3.8 Compliance Level 3H</td>
<td>65</td>
</tr>
<tr>
<td>7.3.8.1 CMS Media Manager</td>
<td>68</td>
</tr>
<tr>
<td>7.3.8.2 CMS Picture Compilation</td>
<td>68</td>
</tr>
<tr>
<td>7.3.8.3 Sensor Media Manager</td>
<td>68</td>
</tr>
<tr>
<td>7.3.8.4 Sensor Parameter Assessment</td>
<td>68</td>
</tr>
<tr>
<td>7.3.8.5 Sensor Plot Detector</td>
<td>68</td>
</tr>
<tr>
<td>7.3.8.6 Sensor Track Reporter</td>
<td>68</td>
</tr>
<tr>
<td>7.3.8.7 Track Extractor</td>
<td>69</td>
</tr>
<tr>
<td>7.3.9 Compliance Level 3I</td>
<td>69</td>
</tr>
<tr>
<td>7.3.9.1 CMS Plot Sharing Manager</td>
<td>72</td>
</tr>
<tr>
<td>7.3.9.2 CMS Tactical Picture Manager</td>
<td>72</td>
</tr>
<tr>
<td>7.3.9.3 OARIS External Interface</td>
<td>72</td>
</tr>
<tr>
<td>7.3.9.4 Platform A</td>
<td>73</td>
</tr>
<tr>
<td>7.3.9.5 Platform B</td>
<td>73</td>
</tr>
<tr>
<td>7.3.9.6 Plot Fuser</td>
<td>73</td>
</tr>
<tr>
<td>7.3.9.7 Sensor Plot Detector</td>
<td>73</td>
</tr>
<tr>
<td>7.4 Common Types</td>
<td>73</td>
</tr>
<tr>
<td>7.4.1 anonymous blob type</td>
<td>74</td>
</tr>
<tr>
<td>7.4.2 confidence interval type</td>
<td>75</td>
</tr>
<tr>
<td>7.4.3 confidence type</td>
<td>75</td>
</tr>
<tr>
<td>7.4.4 filter id type</td>
<td>75</td>
</tr>
<tr>
<td>7.4.5 filter mode type</td>
<td>75</td>
</tr>
<tr>
<td>7.4.6 identity type</td>
<td>75</td>
</tr>
<tr>
<td>7.4.7 percentage type</td>
<td>76</td>
</tr>
<tr>
<td>7.4.8 quality interval type</td>
<td>76</td>
</tr>
<tr>
<td>7.4.9 strength type</td>
<td>76</td>
</tr>
<tr>
<td>7.4.10 subsystem id type</td>
<td>76</td>
</tr>
<tr>
<td>7.4.11 system track id type</td>
<td>76</td>
</tr>
<tr>
<td>7.4.12 time type</td>
<td>77</td>
</tr>
<tr>
<td>7.4.13 track priority type</td>
<td>77</td>
</tr>
<tr>
<td>7.4.14 track quality type</td>
<td>77</td>
</tr>
<tr>
<td>7.4.15 frequency band type</td>
<td>77</td>
</tr>
<tr>
<td>7.4.16 System Track</td>
<td>77</td>
</tr>
<tr>
<td>7.4.16.1 system track type</td>
<td>78</td>
</tr>
<tr>
<td>7.4.17 Coordinates and Positions</td>
<td>79</td>
</tr>
<tr>
<td>7.4.17.1 absolute duration type</td>
<td>83</td>
</tr>
<tr>
<td>7.4.17.2 angle of climb type</td>
<td>83</td>
</tr>
<tr>
<td>7.4.17.3 azimuth coordinate type</td>
<td>83</td>
</tr>
<tr>
<td>7.4.17.4 azimuth interval type</td>
<td>84</td>
</tr>
<tr>
<td>7.4.17.5 azimuth qualification type</td>
<td>84</td>
</tr>
<tr>
<td>7.4.17.6 azimuth rate type</td>
<td>84</td>
</tr>
<tr>
<td>7.4.17.7 cartesian coordinate type</td>
<td>84</td>
</tr>
</tbody>
</table>
7.4.17.8 cartesian interval type ........................................................................... 84
7.4.17.9 cartesian position type ........................................................................... 85
7.4.17.10 cartesian velocity component type ..................................................... 85
7.4.17.11 cartesian velocity type ......................................................................... 85
7.4.17.12 coordinate kind type ........................................................................... 85
7.4.17.13 coordinate orientation type ................................................................. 86
7.4.17.14 coordinate origin type .......................................................................... 87
7.4.17.15 coordinate specification type ............................................................... 88
7.4.17.16 course type .......................................................................................... 88
7.4.17.17 covariance matrix type ....................................................................... 88
7.4.17.18 diagonal covariance matrix type ......................................................... 89
7.4.17.19 duration type ....................................................................................... 89
7.4.17.20 elevation coordinate type .................................................................... 89
7.4.17.21 elevation interval type .......................................................................... 89
7.4.17.22 elevation qualification type .................................................................. 90
7.4.17.23 elevation rate type .............................................................................. 90
7.4.17.24 full 2d covariance matrix type ............................................................ 90
7.4.17.25 full covariance matrix type .................................................................. 91
7.4.17.26 height coordinate type ........................................................................ 92
7.4.17.27 height interval type ............................................................................. 92
7.4.17.28 latitude coordinate type ..................................................................... 92
7.4.17.29 latitude interval type ........................................................................... 92
7.4.17.30 longitude coordinate type ................................................................. 93
7.4.17.31 longitude interval type ........................................................................ 93
7.4.17.32 polar position type ............................................................................... 93
7.4.17.33 polar velocity type ............................................................................... 94
7.4.17.34 position accuracy coordinate type .................................................... 94
7.4.17.35 position coordinate type ..................................................................... 94
7.4.17.36 range coordinate type ......................................................................... 95
7.4.17.37 range interval type ............................................................................... 95
7.4.17.38 range qualification type ...................................................................... 95
7.4.17.39 range rate type ................................................................................... 95
7.4.17.40 speed interval type ............................................................................. 95
7.4.17.41 speed type .......................................................................................... 96
7.4.17.42 velocity accuracy coordinate type ..................................................... 96
7.4.17.43 velocity coordinate type ...................................................................... 96
7.4.17.44 wgs84 position type .......................................................................... 97
7.4.17.45 wgs84 velocity type ........................................................................... 97
7.4.17.46 cartesian position accuracy type ....................................................... 97
7.4.17.47 cartesian velocity accuracy type ......................................................... 98
7.4.17.48 polar position accuracy type ............................................................... 98
7.4.17.49 polar velocity accuracy type ............................................................... 98
7.4.17.50 wgs84 position accuracy type ............................................................ 99
7.4.17.51 wgs84 velocity accuracy type ............................................................. 99
7.4.18 Shape Model .......................................................................................... 99
7.4.18.1 area 2d type ....................................................................................... 101
7.5.4.6 start stop sim mode request type................................................................. 116
7.5.4.7 stop freeze session request type................................................................. 116
7.5.5 Subsystem Control...................................................................................................... 116
7.5.5.1 equipment category type.................................................................................. 119
7.5.5.2 function id type............................................................................................... 119
7.5.5.3 function type.................................................................................................... 119
7.5.5.4 platform category type.................................................................................... 119
7.5.5.5 product category type...................................................................................... 119
7.5.5.6 service name type........................................................................................... 119
7.5.5.7 battle override state type................................................................................ 121
7.5.5.8 descriptor type............................................................................................... 121
7.5.5.9 descriptor sequence type................................................................................ 122
7.5.5.10 device identification type.............................................................................. 122
7.5.5.11 device name type.......................................................................................... 123
7.5.5.12 event type..................................................................................................... 123
7.5.5.13 fault type....................................................................................................... 124
7.5.5.14 fault list type............................................................................................... 124
7.5.5.15 health state reason type.............................................................................. 124
7.5.5.16 health state type.......................................................................................... 124
7.5.5.17 information name type................................................................................ 125
7.5.5.18 interest type.................................................................................................. 126
7.5.5.19 interest list type........................................................................................... 126
7.5.5.20 mastership state type................................................................................. 126
7.5.5.21 parameter name type.................................................................................... 126
7.5.5.22 name error pair type.................................................................................... 127
7.5.5.23 name error sequence type.......................................................................... 127
7.5.5.24 parameter name sequence type................................................................. 127
7.5.5.25 name value pair type................................................................................ 127
7.5.5.26 name value sequence type.......................................................................... 128
7.5.5.27 operational mode type............................................................................... 128
7.5.5.28 parameter value response type................................................................. 128
7.5.5.29 registration type.......................................................................................... 128
7.5.5.30 service type.................................................................................................. 129
7.5.5.31 service health type...................................................................................... 129
7.5.5.32 service indication list type.......................................................................... 129
7.5.5.33 service indication type................................................................................ 129
7.5.5.34 service information type............................................................................. 130
7.5.5.35 service list type........................................................................................... 130
7.5.5.36 subsystem health type................................................................................ 130
7.5.5.37 technical state type..................................................................................... 131
7.5.5.38 version type............................................................................................... 132

7.6 Sensor Domain ...................................................................................................... 132
7.6.1 Clutter Reporting............................................................................................... 132
7.6.1.1 clutter assessment request type................................................................. 133
7.6.1.2 clutter indication type................................................................................... 133
7.6.1.3 clutter map cell type.................................................................................... 133
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.6.1.4</td>
<td>clutter report type</td>
<td>134</td>
</tr>
<tr>
<td>7.6.1.5</td>
<td>concentration plot cell type</td>
<td>134</td>
</tr>
<tr>
<td>7.6.1.6</td>
<td>intensity units type</td>
<td>134</td>
</tr>
<tr>
<td>7.6.1.7</td>
<td>plot concentration report type</td>
<td>135</td>
</tr>
<tr>
<td>7.6.1.8</td>
<td>plot concentration request data type</td>
<td>135</td>
</tr>
<tr>
<td>7.6.2</td>
<td>Media Streaming</td>
<td>135</td>
</tr>
<tr>
<td>7.6.2.1</td>
<td>codec type</td>
<td>136</td>
</tr>
<tr>
<td>7.6.2.2</td>
<td>media allocation type</td>
<td>136</td>
</tr>
<tr>
<td>7.6.2.3</td>
<td>media kind type</td>
<td>137</td>
</tr>
<tr>
<td>7.6.2.4</td>
<td>media name type</td>
<td>137</td>
</tr>
<tr>
<td>7.6.2.5</td>
<td>media stream id type</td>
<td>137</td>
</tr>
<tr>
<td>7.6.2.6</td>
<td>media stream metadata type</td>
<td>137</td>
</tr>
<tr>
<td>7.6.3</td>
<td>Search</td>
<td>138</td>
</tr>
<tr>
<td>7.6.3.1</td>
<td>cued search cue type</td>
<td>138</td>
</tr>
<tr>
<td>7.6.3.2</td>
<td>cued search report type</td>
<td>139</td>
</tr>
<tr>
<td>7.6.3.3</td>
<td>search pattern type</td>
<td>139</td>
</tr>
<tr>
<td>7.6.3.4</td>
<td>search repeat type</td>
<td>140</td>
</tr>
<tr>
<td>7.6.3.5</td>
<td>surveillance area type</td>
<td>140</td>
</tr>
<tr>
<td>7.6.3.6</td>
<td>surveillance search type</td>
<td>140</td>
</tr>
<tr>
<td>7.6.3.7</td>
<td>surveillance task type</td>
<td>141</td>
</tr>
<tr>
<td>7.6.4</td>
<td>Sensor Assessment</td>
<td>141</td>
</tr>
<tr>
<td>7.6.4.1</td>
<td>country code type</td>
<td>147</td>
</tr>
<tr>
<td>7.6.4.2</td>
<td>descriptor list type</td>
<td>147</td>
</tr>
<tr>
<td>7.6.4.3</td>
<td>descriptor name type</td>
<td>147</td>
</tr>
<tr>
<td>7.6.4.4</td>
<td>descriptor single value type</td>
<td>147</td>
</tr>
<tr>
<td>7.6.4.5</td>
<td>descriptor value type</td>
<td>148</td>
</tr>
<tr>
<td>7.6.4.6</td>
<td>equipment match type</td>
<td>148</td>
</tr>
<tr>
<td>7.6.4.7</td>
<td>equipment mode match type</td>
<td>148</td>
</tr>
<tr>
<td>7.6.4.8</td>
<td>equipment name type</td>
<td>149</td>
</tr>
<tr>
<td>7.6.4.9</td>
<td>equipment parameter match type</td>
<td>149</td>
</tr>
<tr>
<td>7.6.4.10</td>
<td>equipment type</td>
<td>149</td>
</tr>
<tr>
<td>7.6.4.11</td>
<td>function name type</td>
<td>149</td>
</tr>
<tr>
<td>7.6.4.12</td>
<td>match id type</td>
<td>149</td>
</tr>
<tr>
<td>7.6.4.13</td>
<td>match link type</td>
<td>150</td>
</tr>
<tr>
<td>7.6.4.14</td>
<td>match type</td>
<td>150</td>
</tr>
<tr>
<td>7.6.4.15</td>
<td>measurement element match type</td>
<td>150</td>
</tr>
<tr>
<td>7.6.4.16</td>
<td>mode name type</td>
<td>150</td>
</tr>
<tr>
<td>7.6.4.17</td>
<td>multipath set type</td>
<td>151</td>
</tr>
<tr>
<td>7.6.4.18</td>
<td>observable function type</td>
<td>151</td>
</tr>
<tr>
<td>7.6.4.19</td>
<td>parametric mode match type</td>
<td>151</td>
</tr>
<tr>
<td>7.6.4.20</td>
<td>platform activity name type</td>
<td>152</td>
</tr>
<tr>
<td>7.6.4.21</td>
<td>platform activity type</td>
<td>152</td>
</tr>
<tr>
<td>7.6.4.22</td>
<td>platform class type</td>
<td>152</td>
</tr>
<tr>
<td>7.6.4.23</td>
<td>platform equipment match type</td>
<td>153</td>
</tr>
<tr>
<td>7.6.4.24</td>
<td>platform match type</td>
<td>153</td>
</tr>
<tr>
<td>7.6.4.25</td>
<td>platform mode match type</td>
<td>153</td>
</tr>
</tbody>
</table>

Open Architecture Radar Interface Standard (OARIS), v2.0
7.6.4.26 platform name type .................................................................................. 153
7.6.4.27 platform type ............................................................................................ 153
7.6.4.28 reference descriptor type ......................................................................... 154
7.6.4.29 reference id type ...................................................................................... 154
7.6.4.30 reference mode type ................................................................................ 154
7.6.4.31 reference parameter match type ............................................................... 154
7.6.4.32 reference parameter type .......................................................................... 155
7.6.4.33 reference type .......................................................................................... 155
7.6.4.34 sensor plot equipment assessment type .................................................... 155
7.6.4.35 sensor plot mode assessment type ............................................................. 155
7.6.4.36 sensor plot platform assessment type ....................................................... 156
7.6.4.37 sensor track equipment assessment type .................................................. 156
7.6.4.38 sensor track mode assessment type ......................................................... 156
7.6.4.39 sensor track platform assessment type ..................................................... 157
7.6.4.40 subplatform match type ............................................................................ 157
7.6.5 Supplementary Measurement ........................................................................ 157
7.6.5.1 continuous measurement type .................................................................... 163
7.6.5.2 discrete measurement type ......................................................................... 163
7.6.5.3 discrete order type ...................................................................................... 163
7.6.5.4 discrete set measurement type ................................................................... 164
7.6.5.5 distribution mode type ............................................................................... 164
7.6.5.6 distribution parameter measurement type .................................................. 165
7.6.5.7 distribution parameter name type ............................................................... 165
7.6.5.8 measurement drift type .............................................................................. 165
7.6.5.9 measurement element type ....................................................................... 166
7.6.5.10 measurement interval type ....................................................................... 166
7.6.5.11 measurement kind id type ........................................................................ 167
7.6.5.12 measurement name type .......................................................................... 167
7.6.5.13 measurement parameter kind type ............................................................ 167
7.6.5.14 measurement parameter set name type ..................................................... 167
7.6.5.15 measurement parameter status type .......................................................... 167
7.6.5.16 measurement parameter type .................................................................... 168
7.6.5.17 measurement unit type .............................................................................. 168
7.6.5.18 measurement variation kind type .............................................................. 169
7.6.5.19 modulation type ......................................................................................... 170
7.6.5.20 multi modal measurement type ................................................................ 170
7.6.5.21 normal measurement type ....................................................................... 171
7.6.5.22 parameter distribution type ...................................................................... 171
7.6.5.23 parameter id type ...................................................................................... 171
7.6.5.24 pdf measurement type .............................................................................. 171
7.6.5.25 pdf name type .......................................................................................... 171
7.6.5.26 plot measurement parameter set type ....................................................... 171
7.6.5.27 poisson measurement type ....................................................................... 172
7.6.5.28 qualitative measurement type ................................................................... 172
7.6.5.29 sample range type .................................................................................... 172
7.6.5.30 scalar measurement type ......................................................................... 172
### 7.6.5.31 sensor defined pdf measurement type

### 7.6.5.32 sequence name type

### 7.6.5.33 single measurement type

### 7.6.5.34 track measurement parameter set type

### 7.6.5.35 track measurement parameter type

### 7.6.5.36 vector measurement type

#### 7.6.6 Plot Reporting

- **error distribution kind type**
- **plot distribution type**
- **plot filter parameters type**
- **plot id type**
- **plot level filter attributes type**
- **plot summary type**
- **processing capability type**
- **sensor calibration model type**
- **sensor plot set type**
- **sensor plot type**
- **sensor precision model type**
- **sensor processing model type**
- **sensor stabilization model type**
- **track level filter attributes type**
- **track measurement parameter type**
- **track measurement parameter set type**
- **sequence name type**
- **sensor defined pdf measurement type**

---

#### 7.6.7 Sensor Control

- **selected frequency list type**
- **transmission frequency state type**
- **all frequencies state type**
- **reported frequency state type**
- **transmission frequency mode type**
- **transmission sector set type**
- **transmission sector type**
- **transmission sector power level type**
- **sector reference type**
- **control emission state type**
- **test target scenario type**
- **test target scenario independent target type**
- **test target scenario common parameter target type**
- **test target type**
- **test target plus scenario type**
- **test target scenario id type**
- **test target scenario state type**

#### 7.6.8 Sensor Performance

- **absolute performance type**
- **interference report type**
- **interference kind**
- **interferer type**
- **jamming magnitude type**
7.6.8.6 performance bin type ................................................................. 195
7.6.8.7 performance assessment parameters type ........................................... 195
7.6.8.8 performance assessment report type ..................................................... 196
7.6.8.9 performance band type ................................................................. 197
7.6.10 performance beam type ............................................................... 197
7.6.11 performance sector type ............................................................... 197
7.6.12 relative performance type ............................................................ 198
7.6.9 Track Reporting ................................................................................. 198
7.6.9.1 assessment objective id type ......................................................... 200
7.6.9.2 external track number type .............................................................. 200
7.6.9.3 plot association type ........................................................... 201
7.6.9.4 protocol name type ................................................................. 201
7.6.9.5 sensor track id type ............................................................... 201
7.6.9.6 track filter parameters type .......................................................... 201
7.6.9.7 environment type ................................................................. 202
7.6.9.8 initiation mode type ............................................................. 202
7.6.9.9 recognition type ................................................................. 203
7.6.9.10 sensor track type ............................................................... 203
7.6.9.11 sensor track set type ............................................................ 205
7.6.9.12 track phase type .............................................................. 205
7.6.10 Tracking Control .............................................................................. 205
7.6.10.1 track info type ............................................................. 206
7.6.10.2 tracking zone set type .......................................................... 207
7.6.10.3 tracking zone type ............................................................. 207
7.6.10.4 tracking zone kind type ......................................................... 207
7.6.10.5 tracking zone id type ............................................................ 208
7.7 Radar Domain ...................................................................................... 208
7.7.1 Air Engagement Support .................................................................... 208
7.7.1.1 expected hit data type .............................................................. 209
7.7.1.2 miss indication data type .......................................................... 209
7.7.1.3 projectile kinematics type ........................................................ 209
7.7.2 Engagement Support ......................................................................... 210
7.7.2.1 available fire control channels type ........................................... 210
7.7.2.2 fire control channel id type ....................................................... 210
7.7.2.3 kill assessment result type ........................................................ 211
7.7.2.4 kinematics type ................................................................. 211
7.7.3 Missile Guidance ............................................................................... 211
7.7.3.1 downlink report type ............................................................... 213
7.7.3.2 downlink request type ......................................................... 213
7.7.3.3 frequency channel type ............................................................. 214
7.7.3.4 illumination request type ........................................................ 214
7.7.3.5 track id type ................................................................. 214
7.7.3.6 uplink report type ............................................................... 214
7.7.3.7 uplink request type ............................................................... 215
7.7.4 Surface Engagement Support .......................................................... 215
7.7.4.1 splash spotting area id type ..................................................... 216
7.8 Subsystem Services

7.8.1 Encyclopaedic Support ................................................................. 217
7.8.1.1 Receive Encyclopaedic Data .................................................. 217
7.8.2 Extended Subsystem Control ...................................................... 219
7.8.2.1 Manage Physical Configuration .......................................... 219
7.8.2.2 Perform Offline Test .............................................................. 222
7.8.2.3 Manage Network Participation ............................................ 224
7.8.2.4 Startup ................................................................................. 228
7.8.2.5 Provide Networking Statistics .............................................. 230
7.8.2.6 Shutdown ............................................................................. 231
7.8.2.7 Restart ................................................................................ 233
7.8.3 Subsystem Control ...................................................................... 235
7.8.3.1 Manage Technical State ...................................................... 235
7.8.3.2 Heartbeat Signal ................................................................. 238
7.8.3.3 Provide Subsystem Identification ....................................... 240
7.8.3.4 Provide Health State ............................................................ 241
7.8.3.5 Manage Operational Mode .................................................. 245
7.8.3.6 Control Battle Override ....................................................... 248
7.8.3.7 Manage Subsystem Parameters ........................................ 250
7.8.3.8 Provide Subsystem Services ............................................... 254
7.8.3.9 Manage Mastership ............................................................ 257
7.8.3.10 Register Interest ................................................................. 262
7.8.4 Recording and Replay ............................................................... 263
7.8.4.1 Control Recording .............................................................. 263
7.8.4.2 Control Replay ................................................................. 265
7.8.5 Simulation Support .................................................................... 268
7.8.5.1 Define Simulation Scenario ................................................. 268
7.8.5.2 Control Simulation .............................................................. 271
7.8.5.3 Define Fault Scripts ............................................................. 274
7.8.5.4 Control Fault Scripts ......................................................... 276

7.9 Sensor Services ........................................................................... 278

7.9.1 Clutter Reporting ................................................................. 278
7.9.1.1 Provide Area with Plot Concentration ............................... 278
7.9.1.2 Provide Clutter Assessment ................................................. 281
7.9.2 Media Streaming ............................................................... 283
7.9.2.1 Allocate Tracks To Stream ............................................... 283
7.9.2.2 Configure Media Streams ............................................... 286
7.9.3 Search ................................................................................ 287
7.9.3.1 Perform Cued Search ....................................................... 287
7.9.4 Sensor Assessment ............................................................... 292
7.9.4.1 Assess Sensor Plot .......................................................... 292
7.9.4.2 Assess Sensor Track ......................................................... 293
7.9.5 Supplementary Measurement ............................................... 299
7.9.5.1 Configure Measurement Parameters ................................................................. 299
7.9.5.2 Provide Sensor Plot Parameters ........................................................................ 300
7.9.5.3 Provide Sensor Track Parameters ..................................................................... 301
7.9.6 Plot Reporting ........................................................................................................... 303
7.9.6.1 Filter Plots ........................................................................................................... 303
7.9.6.2 Provide Sensor Characteristics .......................................................................... 305
7.9.6.3 Provide Plots ....................................................................................................... 307
7.9.6.4 Provide Sensor Orientation ................................................................................ 308
7.9.7 Sensor Control ......................................................................................................... 310
7.9.7.1 Manage Frequency Usage ................................................................................. 310
7.9.7.2 Manage Transmission Sectors ........................................................................... 314
7.9.7.3 Control Emissions ............................................................................................... 317
7.9.7.4 Define Test Target Scenario ............................................................................ 319
7.9.7.5 Test Target Facility ............................................................................................ 322
7.9.8 Sensor Performance .................................................................................................. 325
7.9.8.1 Provide Interference Reports ............................................................................. 325
7.9.8.2 Provide Nominal Performance .......................................................................... 327
7.9.8.3 Provide Performance Assessment ..................................................................... 329
7.9.8.4 Provide Jammer Assessment ............................................................................. 331
7.9.9 Track Reporting ........................................................................................................ 333
7.9.9.1 Filter Tracks ........................................................................................................ 333
7.9.9.2 Label Tracks ....................................................................................................... 335
7.9.9.3 Provide Sensor Tracks ....................................................................................... 338
7.9.10 Tracking Control .................................................................................................... 342
7.9.10.1 Delete Sensor Track ......................................................................................... 343
7.9.10.2 Receive Track Information .............................................................................. 344
7.9.10.3 Initiate Track ..................................................................................................... 346
7.9.10.4 Manage Tracking Zones .................................................................................. 348
7.10 Radar Services ........................................................................................................... 351
7.10.1 Air Engagement Support ...................................................................................... 351
7.10.1.1 Provide Projectile Positional Information ....................................................... 351
7.10.2 Engagement Support ............................................................................................ 353
7.10.2.1 Process Target Designation ........................................................................... 353
7.10.2.2 Support Kill Assessment ................................................................................ 357
7.10.2.3 Support Surface Target Engagement .............................................................. 358
7.10.3 Missile Guidance ................................................................................................... 362
7.10.3.1 Perform Illumination ..................................................................................... 362
7.10.3.2 Perform Missile Downlink ............................................................................ 365
7.10.3.3 Perform Missile Uplink ................................................................................ 368
7.10.4 Surface Engagement Support ................................................................................ 370
7.10.4.1 Perform Splash Spotting .............................................................................. 370
<table>
<thead>
<tr>
<th>OMG’s Issue Reporting Procedure</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Issues To Be Discussed</td>
<td>17</td>
</tr>
<tr>
<td>0.1 Unclassified Specification</td>
<td>17</td>
</tr>
<tr>
<td>0.2 Impact on Existing Users</td>
<td>17</td>
</tr>
<tr>
<td>0.3 Architectural Extensibility</td>
<td>17</td>
</tr>
<tr>
<td>0.4 Sensor Specific Aspects</td>
<td>17</td>
</tr>
<tr>
<td>0.5 Information from all Sensor Types</td>
<td>17</td>
</tr>
<tr>
<td>0.6 Additional PSM</td>
<td>17</td>
</tr>
<tr>
<td>0.7 Compatibility with OARIS v1.1</td>
<td>18</td>
</tr>
<tr>
<td>Scope</td>
<td>18</td>
</tr>
<tr>
<td>Conformance</td>
<td>19</td>
</tr>
<tr>
<td>Normative References</td>
<td>21</td>
</tr>
<tr>
<td>Terms and Definitions</td>
<td>22</td>
</tr>
<tr>
<td>Symbols</td>
<td>24</td>
</tr>
<tr>
<td>Additional Information</td>
<td>24</td>
</tr>
<tr>
<td>6.1 Acknowledgements</td>
<td>24</td>
</tr>
<tr>
<td>Open Architecture Radar Information Specification</td>
<td>25</td>
</tr>
<tr>
<td>7.1 Introduction</td>
<td>25</td>
</tr>
<tr>
<td>7.2 Usage Overview</td>
<td>25</td>
</tr>
<tr>
<td>7.3 Common Types</td>
<td>40</td>
</tr>
<tr>
<td>7.3.1 anonymous_blob_type</td>
<td>41</td>
</tr>
<tr>
<td>7.3.2 confidence_type</td>
<td>42</td>
</tr>
<tr>
<td>7.3.3 identity_type</td>
<td>42</td>
</tr>
<tr>
<td>7.3.4 strength_type</td>
<td>42</td>
</tr>
<tr>
<td>7.3.5 subsystem_id_type</td>
<td>43</td>
</tr>
<tr>
<td>7.3.6 system_track_id_type</td>
<td>43</td>
</tr>
<tr>
<td>7.3.7 time_type</td>
<td>43</td>
</tr>
<tr>
<td>7.3.8 frequency_band_type</td>
<td>43</td>
</tr>
<tr>
<td>7.3.9 System Track</td>
<td>43</td>
</tr>
<tr>
<td>7.3.9.1 system_track_type</td>
<td>44</td>
</tr>
<tr>
<td>7.3.10 Coordinates and Positions</td>
<td>44</td>
</tr>
<tr>
<td>7.3.10.1 absolute_duration_type</td>
<td>49</td>
</tr>
<tr>
<td>7.3.10.2 angle_of_climb_type</td>
<td>49</td>
</tr>
<tr>
<td>7.3.10.3 azimuth_coordinate_type</td>
<td>49</td>
</tr>
<tr>
<td>7.3.10.4 azimuth_interval_type</td>
<td>50</td>
</tr>
<tr>
<td>7.3.10.5 azimuth_qualification_type</td>
<td>50</td>
</tr>
<tr>
<td>7.3.10.6 azimuth_rate_type</td>
<td>50</td>
</tr>
<tr>
<td>7.3.10.7 cartesian_coordinate_type</td>
<td>50</td>
</tr>
<tr>
<td>7.3.10.8 cartesian_interval_type</td>
<td>51</td>
</tr>
<tr>
<td>7.3.10.9 cartesian_position_type</td>
<td>51</td>
</tr>
<tr>
<td>7.3.10.10 cartesian_velocity_component_type</td>
<td>51</td>
</tr>
<tr>
<td>7.3.10.11 cartesian_velocity_type</td>
<td>52</td>
</tr>
<tr>
<td>Open Architecture Radar Interface Standard (OARIS), v2.0</td>
<td>xxiii</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>7.3.10.12</td>
<td>coordinate_kind_type</td>
</tr>
<tr>
<td>7.3.10.13</td>
<td>coordinate_orientation_type</td>
</tr>
<tr>
<td>7.3.10.14</td>
<td>coordinate_origin_type</td>
</tr>
<tr>
<td>7.3.10.15</td>
<td>coordinate_specification_type</td>
</tr>
<tr>
<td>7.3.10.16</td>
<td>course_type</td>
</tr>
<tr>
<td>7.3.10.17</td>
<td>covariance_matrix_type</td>
</tr>
<tr>
<td>7.3.10.18</td>
<td>diagonal_covariance_matrix_type</td>
</tr>
<tr>
<td>7.3.10.19</td>
<td>duration_type</td>
</tr>
<tr>
<td>7.3.10.20</td>
<td>elevation_coordinate_type</td>
</tr>
<tr>
<td>7.3.10.21</td>
<td>elevation_interval_type</td>
</tr>
<tr>
<td>7.3.10.22</td>
<td>elevation_qualification_type</td>
</tr>
<tr>
<td>7.3.10.23</td>
<td>elevation_rate_type</td>
</tr>
<tr>
<td>7.3.10.24</td>
<td>full_2d_covariance_matrix_type</td>
</tr>
<tr>
<td>7.3.10.25</td>
<td>full_covariance_matrix_type</td>
</tr>
<tr>
<td>7.3.10.26</td>
<td>height_coordinate_type</td>
</tr>
<tr>
<td>7.3.10.27</td>
<td>height_interval_type</td>
</tr>
<tr>
<td>7.3.10.28</td>
<td>latitude_coordinate_type</td>
</tr>
<tr>
<td>7.3.10.29</td>
<td>latitude_interval_type</td>
</tr>
<tr>
<td>7.3.10.30</td>
<td>longitude_coordinate_type</td>
</tr>
<tr>
<td>7.3.10.31</td>
<td>longitude_interval_type</td>
</tr>
<tr>
<td>7.3.10.32</td>
<td>polar_position_type</td>
</tr>
<tr>
<td>7.3.10.33</td>
<td>polar_velocity_type</td>
</tr>
<tr>
<td>7.3.10.34</td>
<td>position_accuracy_coordinate_type</td>
</tr>
<tr>
<td>7.3.10.35</td>
<td>position_coordinate_type</td>
</tr>
<tr>
<td>7.3.10.36</td>
<td>range_coordinate_type</td>
</tr>
<tr>
<td>7.3.10.37</td>
<td>range_interval_type</td>
</tr>
<tr>
<td>7.3.10.38</td>
<td>range_qualification_type</td>
</tr>
<tr>
<td>7.3.10.39</td>
<td>range_rate_type</td>
</tr>
<tr>
<td>7.3.10.40</td>
<td>speed_coordinate_type</td>
</tr>
<tr>
<td>7.3.10.41</td>
<td>speed_type</td>
</tr>
<tr>
<td>7.3.10.42</td>
<td>velocity_accuracy_coordinate_type</td>
</tr>
<tr>
<td>7.3.10.43</td>
<td>velocity_coordinate_type</td>
</tr>
<tr>
<td>7.3.10.44</td>
<td>wgs84_position_type</td>
</tr>
<tr>
<td>7.3.10.45</td>
<td>wgs84_velocity_type</td>
</tr>
<tr>
<td>7.3.10.46</td>
<td>cartesian_position_accuracy_type</td>
</tr>
<tr>
<td>7.3.10.47</td>
<td>cartesian_velocity_accuracy_type</td>
</tr>
<tr>
<td>7.3.10.48</td>
<td>polar_position_accuracy_type</td>
</tr>
<tr>
<td>7.3.10.49</td>
<td>polar_velocity_accuracy_type</td>
</tr>
<tr>
<td>7.3.10.50</td>
<td>wgs84_position_accuracy_type</td>
</tr>
<tr>
<td>7.3.10.51</td>
<td>wgs84_velocity_accuracy_type</td>
</tr>
<tr>
<td>7.3.11</td>
<td>Shape Model</td>
</tr>
<tr>
<td>7.3.11.1</td>
<td>area_2d_type</td>
</tr>
<tr>
<td>7.3.11.2</td>
<td>figure_ref_point_type</td>
</tr>
<tr>
<td>7.3.11.3</td>
<td>general_polar_volume_type</td>
</tr>
<tr>
<td>7.3.11.4</td>
<td>polar_volume_type</td>
</tr>
<tr>
<td>7.3.11.5</td>
<td>polygon_type</td>
</tr>
<tr>
<td>7.4 Subsystem_Domain</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td></td>
</tr>
<tr>
<td>7.4.1 Encyclopaedic Support</td>
<td></td>
</tr>
<tr>
<td>7.4.1.1 data_descriptor_type</td>
<td></td>
</tr>
<tr>
<td>7.4.1.2 url_type</td>
<td></td>
</tr>
<tr>
<td>7.4.2 Extended Subsystem Control</td>
<td></td>
</tr>
<tr>
<td>7.4.2.1 configuration_url_type</td>
<td></td>
</tr>
<tr>
<td>7.4.2.2 offline_test_result_details_type</td>
<td></td>
</tr>
<tr>
<td>7.4.2.3 offline_test_result_type</td>
<td></td>
</tr>
<tr>
<td>7.4.2.4 offline_test_type</td>
<td></td>
</tr>
<tr>
<td>7.4.3 Recording and Replay</td>
<td></td>
</tr>
<tr>
<td>7.4.3.1 actual_time_type</td>
<td></td>
</tr>
<tr>
<td>7.4.3.2 change_threshold_type</td>
<td></td>
</tr>
<tr>
<td>7.4.3.3 parameter_type</td>
<td></td>
</tr>
<tr>
<td>7.4.3.4 rate_type</td>
<td></td>
</tr>
<tr>
<td>7.4.3.5 record_on_change_type</td>
<td></td>
</tr>
<tr>
<td>7.4.3.6 recorded_data_type</td>
<td></td>
</tr>
<tr>
<td>7.4.3.7 recorded_time_type</td>
<td></td>
</tr>
<tr>
<td>7.4.3.8 recording_descriptor_type</td>
<td></td>
</tr>
<tr>
<td>7.4.3.9 recording_id_type</td>
<td></td>
</tr>
<tr>
<td>7.4.3.10 recording_set_type</td>
<td></td>
</tr>
<tr>
<td>7.4.3.11 recording_type</td>
<td></td>
</tr>
<tr>
<td>7.4.3.12 replay_set_type</td>
<td></td>
</tr>
<tr>
<td>7.4.3.13 replay_speed_type</td>
<td></td>
</tr>
<tr>
<td>7.4.4 Simulation Support</td>
<td></td>
</tr>
<tr>
<td>7.4.4.1 fault_script_id_type</td>
<td></td>
</tr>
<tr>
<td>7.4.4.2 fault_script_ids_type</td>
<td></td>
</tr>
<tr>
<td>7.4.4.3 fault_script_type</td>
<td></td>
</tr>
<tr>
<td>7.4.4.4 fault_scripts_type</td>
<td></td>
</tr>
<tr>
<td>7.4.4.5 sim_mode_status_type</td>
<td></td>
</tr>
<tr>
<td>7.4.4.6 start_stop_sim_mode_request_type</td>
<td></td>
</tr>
<tr>
<td>7.4.4.7 stop_freeze_session_request_type</td>
<td></td>
</tr>
<tr>
<td>7.4.5 Subsystem Control</td>
<td></td>
</tr>
<tr>
<td>7.4.5.1 function_id_type</td>
<td></td>
</tr>
<tr>
<td>7.4.5.2 function_type</td>
<td></td>
</tr>
</tbody>
</table>

Open Architecture Radar Interface Standard (OARIS), v2.0
7.5.2.1 service_name_type
7.5.2.2 battle_override_state_type
7.5.2.3 descriptor_type
7.5.2.4 descriptor_sequence_type
7.5.2.5 device_identification_type
7.5.2.6 device_name_type
7.5.2.7 event_type
7.5.2.8 fault_type
7.5.2.9 fault_list_type
7.5.2.10 health_state_reason_type
7.5.2.11 health_state_type
7.5.2.12 information_name_type
7.5.2.13 interest_type
7.5.2.14 interest_list_type
7.5.2.15 mastership_state_type
7.5.2.16 parameter_name_type
7.5.2.17 parameter_name_sequence_type
7.5.2.18 parameter_name_value_pair_type
7.5.2.19 name_error_pair_type
7.5.2.20 name_error_sequence_type
7.5.2.21 name_value_pair_type
7.5.2.22 name_value_sequence_type
7.5.2.23 operational_mode_type
7.5.2.24 parameter_value_response_type
7.5.2.25 registration_type
7.5.2.26 service_type
7.5.2.27 service_health_type
7.5.2.28 service_information_list_type
7.5.2.29 service_indication_type
7.5.2.30 service_list_type
7.5.2.31 service_type
7.5.2.32 subsystem_health_type
7.5.2.33 technical_state_type
7.5.2.34 version_type
7.5.2.35 initial

7.5. Sensor Domain

7.5.1 Clutter Reporting
7.5.1.1 clutter_assessment_request_type
7.5.1.2 clutter_indication_type
7.5.1.3 clutter_map_cell_type
7.5.1.4 clutter_report_type
7.5.1.5 concentration_plot_cell_type
7.5.1.6 intensity_units_type
7.5.1.7 plot_concentration_report_type
7.5.1.8 plot_concentration_request_data_type

7.5.2 Media Streaming
7.5.2.1 codec_type
| 2.5.2.2 media_allocation_type | 100 |
| 2.5.2.3 media_kind_type | 101 |
| 2.5.2.4 media_name_type | 101 |
| 2.5.2.5 media_stream_id_type | 101 |
| 2.5.2.6 media_stream_metadata_type | 101 |
| 7.5.3 Search | 101 |
| 7.5.3.1 cued_search_cue_type | 102 |
| 7.5.3.2 cued_search_report_type | 103 |
| 7.5.3.3 search_pattern_type | 103 |
| 7.5.3.4 search_repeat_type | 103 |
| 7.5.3.5 surveillance_area_type | 104 |
| 7.5.3.6 surveillance_search_type | 104 |
| 7.5.3.7 surveillance_task_type | 104 |
| 7.5.4 Sensor Assessment | 105 |
| 7.5.4.1 assessment_objective_id_type | 111 |
| 7.5.4.2 country_code_type | 111 |
| 7.5.4.3 descriptor_list_type | 111 |
| 7.5.4.4 descriptor_name_type | 111 |
| 7.5.4.5 descriptor_single_value_type | 112 |
| 7.5.4.6 descriptor_value_type | 112 |
| 7.5.4.7 equipment_match_type | 112 |
| 7.5.4.8 equipment_mode_match_type | 112 |
| 7.5.4.9 equipment_name_type | 112 |
| 7.5.4.10 equipment_parameter_match_type | 113 |
| 7.5.4.11 equipment_type | 113 |
| 7.5.4.12 function_name_type | 113 |
| 7.5.4.13 match_id_type | 113 |
| 7.5.4.14 match_link_type | 113 |
| 7.5.4.15 match_type | 113 |
| 7.5.4.16 measurement_element_match_type | 114 |
| 7.5.4.17 mode_name_type | 114 |
| 7.5.4.18 multipath_set_type | 114 |
| 7.5.4.19 observable_function_type | 114 |
| 7.5.4.20 parametric_mode_match_type | 115 |
| 7.5.4.21 platform_activity_name_type | 115 |
| 7.5.4.22 platform_activity_type | 115 |
| 7.5.4.23 platform_class_type | 115 |
| 7.5.4.24 platform_equipment_match_type | 116 |
| 7.5.4.25 platform_match_type | 116 |
| 7.5.4.26 platform_mode_match_type | 116 |
| 7.5.4.27 platform_name_type | 116 |
| 7.5.4.28 platform_type | 116 |
| 7.5.4.29 reference_descriptor_type | 116 |
| 7.5.4.30 reference_id_type | 117 |
| 7.5.4.31 reference_mode_type | 117 |
| 7.5.4.32 reference_parameter_match_type | 117 |
7.5.5.32  reference parameter type ................................................................. 117
7.5.5.34  reference type .................................................................................. 117
7.5.5.35  sensor plot equipment assessment type ........................................... 118
7.5.5.36  sensor plot mode assessment type .................................................. 118
7.5.5.37  sensor plot platform assessment type .............................................. 118
7.5.5.38  sensor track equipment assessment type ........................................... 118
7.5.5.39  sensor track mode assessment type .................................................. 118
7.5.5.40  sensor track platform assessment type .............................................. 118
7.5.5.41  subplatform match type .................................................................. 118
7.5.5 Supplementary Measurement ................................................................ 119
  7.5.5.1  continuous measurement type ........................................................... 124
  7.5.5.2  discrete measurement type ................................................................. 124
  7.5.5.3  discrete order type ............................................................................. 124
  7.5.5.4  discrete set measurement type ............................................................ 125
  7.5.5.5  distribution mode type ......................................................................... 125
  7.5.5.6  distribution parameter measurement type ........................................... 125
  7.5.5.7  distribution parameter name type ....................................................... 126
  7.5.5.8  measurement drift type ...................................................................... 126
  7.5.5.9  measurement element type ................................................................. 126
  7.5.5.10 measurement interval type ................................................................. 126
  7.5.5.11 measurement kind id type .................................................................. 127
  7.5.5.12 measurement name type .................................................................... 127
  7.5.5.13 measurement parameter kind type .................................................... 127
  7.5.5.14 measurement parameter set name type .............................................. 127
  7.5.5.15 measurement parameter status type .................................................. 127
  7.5.5.16 measurement parameter type .............................................................. 128
  7.5.5.17 measurement unit type ...................................................................... 128
  7.5.5.18 measurement variation kind type ...................................................... 129
  7.5.5.19 modulation type ................................................................................. 130
  7.5.5.20 multi modal measurement type ......................................................... 130
  7.5.5.21 normal measurement type ................................................................. 130
  7.5.5.22 parameter distribution type ............................................................... 130
  7.5.5.23 parameter id type ................................................................................ 131
  7.5.5.24 pdf measurement type ....................................................................... 131
  7.5.5.25 pdf name type ................................................................................... 131
  7.5.5.26 plot measurement parameter set type ............................................... 131
  7.5.5.27 poisson measurement type ................................................................. 131
  7.5.5.28 qualitative measurement type ............................................................ 131
  7.5.5.29 sample range type .............................................................................. 132
  7.5.5.30 scalar measurement type ................................................................... 132
  7.5.5.31 sensor defined pdf measurement type .............................................. 132
  7.5.5.32 sequence name type .......................................................................... 132
  7.5.5.33 single measurement type ................................................................. 132
  7.5.5.34 track measurement parameter set type .............................................. 133
  7.5.5.35 track measurement parameter type .................................................... 133
  7.5.5.36 vector measurement type ................................................................... 133

 xxviii
7.5.6 Plot Reporting ........................................................................................................ 123
  7.5.6.1 plot_id_type .................................................................................................... 124
  7.5.6.2 sensor_plot_set_type .................................................................................... 124
  7.5.6.3 sensor_plot_type ........................................................................................ 124
  7.5.6.4 sensor_orientation_type .............................................................................. 125
7.5.7 Sensor Control ..................................................................................................... 126
  7.5.7.1 selected_frequency_list_type ...................................................................... 130
  7.5.7.2 transmission_frequency_state_type ............................................................ 130
  7.5.7.3 all_frequencies_state_type ......................................................................... 130
  7.5.7.4 reported_frequency_state_type .................................................................. 130
  7.5.7.5 transmission_frequency_mode_type ............................................................ 130
  7.5.7.6 transmission_sector_set_type ...................................................................... 130
  7.5.7.7 transmission_sector_type ........................................................................... 130
  7.5.7.8 transmission_sector_power_level_type ....................................................... 130
  7.5.7.9 sector_reference_type ............................................................................... 130
  7.5.7.10 control_emission_state_type ..................................................................... 130
  7.5.7.11 test_target_scenario_type ......................................................................... 130
  7.5.7.12 test_target_scenario_independent_target_type .......................................... 130
  7.5.7.13 test_target_scenario_common_parameter_target_type ................................ 130
  7.5.7.14 test_target_type ....................................................................................... 130
  7.5.7.15 test_target_plus_scenario_type ............................................................... 130
  7.5.7.16 test_target_scenario_id_type .................................................................... 130
  7.5.7.17 test_target_scenario_state_type ............................................................... 130
7.5.8 Sensor Performance ............................................................................................. 130
  7.5.8.1 interference_report_type ............................................................................. 130
  7.5.8.2 interferer_kind ........................................................................................... 130
  7.5.8.3 interferer_type ............................................................................................ 130
  7.5.8.4 jamming_magnitude_type .......................................................................... 130
  7.5.8.5 performance_bin_type .............................................................................. 130
  7.5.8.6 performance_assessment_report_type ........................................................ 130
  7.5.8.7 performance_assessment_request_type ...................................................... 130
  7.5.8.8 performance_band_type ............................................................................ 130
  7.5.8.9 performance_beam_type ............................................................................ 130
  7.5.8.10 performance_sector_type ......................................................................... 130
  7.5.8.11 performance_type .................................................................................... 130
7.5.9 Track Reporting .................................................................................................... 130
  7.5.9.1 sensor_track_id_type .................................................................................. 130
  7.5.9.2 track_priority_type .................................................................................... 130
  7.5.9.3 track_quality_type .................................................................................... 130
  7.5.9.4 environment_type ...................................................................................... 130
  7.5.9.5 initiation_mode_type ................................................................................ 130
  7.5.9.6 recognition_type ....................................................................................... 130
  7.5.9.7 sensor_track_type ...................................................................................... 130
  7.5.9.8 sensor_track_set_type ............................................................................... 130
  7.5.9.9 track_phase_type ....................................................................................... 130
  7.5.10 Tracking Control ............................................................................................. 130
2.5.10.1 track_info_type ................................................................. 155
2.5.10.2 track_priority_type ......................................................... 156
2.5.10.3 tracking_zone_set_type ................................................... 156
2.5.10.4 tracking_zone_type ......................................................... 156
2.5.10.5 tracking_zone_kind_type ................................................ 156
2.5.10.6 tracking_zone_id_type ..................................................... 157

7.6 Radar Domain ................................................................. 157

7.6.1 Air Engagement Support .................................................. 157
7.6.1.1 expected hit data type ...................................................... 158
7.6.1.2 miss indication data type ............................................... 158
7.6.1.3 projectile kinematics type ............................................... 158

7.6.2 Engagement Support ......................................................... 159
7.6.2.1 available fire control channel_type ................................. 159
7.6.2.2 fire control channel id_type ............................................ 159
7.6.2.3 kill assessment result_type ............................................. 159
7.6.2.4 kinematics type ............................................................. 160

7.6.3 Missile Guidance ............................................................ 160
7.6.3.1 downlink report_type ...................................................... 162
7.6.3.2 downlink request_type .................................................... 162
7.6.3.3 frequency channel type .................................................. 163
7.6.3.4 illumination request_type .............................................. 163
7.6.3.5 track_id_type ............................................................... 163
7.6.3.6 uplink report_type ......................................................... 163
7.6.3.7 uplink request_type ........................................................ 164

7.6.4 Surface Engagement Support ........................................... 164
7.6.4.1 splash spotting area id_type ............................................ 164
7.6.4.2 splash spotting area position_type ................................ 165
7.6.4.3 splash spotting area set_type ......................................... 165
7.6.4.4 splash spotting area_type ............................................... 165

7.7 Subsystem Services .......................................................... 166

7.7.1 Encyclopaedic Support ...................................................... 166
7.7.1.1 Receive Encyclopaedic Data ........................................... 166
7.7.2 Extended Subsystem Control ............................................ 168
7.7.2.1 Manage Physical Configuration .................................... 168
7.7.2.2 Perform Offline Test ....................................................... 170
7.7.2.3 Restart ...................................................................... 172
7.7.2.4 Shutdown .................................................................. 173
7.7.2.5 Startup ..................................................................... 175

7.7.3 Recording and Replay ...................................................... 177
7.7.3.1 Control Recording ......................................................... 177
7.7.3.2 Control Replay .............................................................. 179

7.7.4 Simulation Support ........................................................ 182
7.7.4.1 Define Simulation Scenario .......................................... 182
7.7.4.2 Control Simulation ....................................................... 185
7.7.4.3 Define Fault Script ......................................................... 188
7.7.4.4 Control Fault Scripts ................................................................. 190
7.7.5 Subsystem Control ................................................................. 192
7.7.5.1 Manage Technical State .................................................... 192
7.7.5.2 Heartbeat Signal ............................................................... 195
7.7.5.3 Provide Subsystem Identification ...................................... 196
7.7.5.4 Provide Health State ........................................................ 198
7.7.5.5 Manage Operational Mode .............................................. 202
7.7.5.6 Control Battle Override ................................................... 205
7.7.5.7 Manage Subsystem Parameters ...................................... 207
7.7.5.8 Provide Subsystem Services ............................................ 211
7.7.5.9 Manage Mastership ........................................................ 214
7.7.5.10 Register Interest ............................................................. 219

7.8 Sensor Services ........................................................................ 220
7.8.1 Clutter Reporting ................................................................. 221
7.8.1.1 Provide Area with Plot Concentration ......................... 221
7.8.1.2 Provide Clutter Assessment .......................................... 223
7.8.2 Media Streaming ................................................................. 225
7.8.2.1 Allocate Tracks To Stream ............................................ 225
7.8.2.2 Configure Media Streams ............................................ 228
7.8.3 Search ................................................................................ 229
7.8.3.1 Perform Cued Search ................................................... 229
7.8.4 Sensor Assessment ............................................................. 234
7.8.4.1 Assess Sensor Plot ....................................................... 234
7.8.4.2 Assess Sensor Track .................................................... 236
7.8.5 Supplementary Measurement ............................................ 242
7.8.5.1 Configure Measurement Parameters ......................... 242
7.8.5.2 Provide Sensor Plot Parameters ................................... 243
7.8.5.3 Provide Sensor Track Parameters ................................. 244
7.8.6 Plot Reporting ................................................................. 246
7.8.6.1 Provide Plots ............................................................. 246
7.8.6.2 Provide Sensor Orientation ......................................... 247
7.8.7 Sensor Control ................................................................. 249
7.8.7.1 Manage Frequency Usage ........................................... 249
7.8.7.2 Manage Transmission Sectors ..................................... 253
7.8.7.3 Control Emissions ........................................................ 256
7.8.7.4 Define Test Target Scenario ........................................ 258
7.8.7.5 Test Target Facility ..................................................... 261
7.8.8 Sensor Performance .......................................................... 264
7.8.8.1 Provide Interference Reports ....................................... 264
7.8.8.2 Provide Nominal Performance ..................................... 266
7.8.8.3 Provide Performance Assessment ............................... 268
7.8.8.4 Provide Jammer Assessment ....................................... 270
7.8.9 Track Reporting .............................................................. 272
7.8.9.1 Provide Sensor Tracks ................................................ 272
7.8.10 Tracking Control .......................................................... 276
7.8.10.1 Delete Sensor Track .................................................... 277
Preface
OMG

Founded in 1989, the Object Management Group, Inc. (OMG) is an open membership, not-for-profit computer industry standards consortium that produces and maintains computer industry specifications for interoperable, portable, and reusable enterprise applications in distributed, heterogeneous environments. Membership includes Information Technology vendors, end users, government agencies, and academia.

OMG member companies write, adopt, and maintain its specifications following a mature, open process. OMG’s specifications implement the Model Driven Architecture® (MDA®), maximizing ROI through a full-lifecycle approach to enterprise integration that covers multiple operating systems, programming languages, middleware and networking infrastructures, and software development environments. OMG’s specifications include: UML® (Unified Modeling Language™); CORBA® (Common Object Request Broker Architecture); CWM™ (Common Warehouse Metamodel); and industry-specific standards for dozens of vertical markets.

More information on the OMG is available at http://www.omg.org/.

OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. All OMG Specifications are available from the OMG website at:

http://www.omg.org/spec

Specifications are organized by the following categories:

Business Modeling Specifications

Middleware Specifications

- CORBA/IIOP
- Data Distribution Services
- Specialized CORBA

1. IDL/Language Mapping Specifications Modeling and Metadata Specifications

2. Modeling and Metadata Specifications

- UML, MOF, CWM, XMI
- UML Profile
3. Modernization Specifications

4. Platform Independent Model (PIM), Platform Specific Model (PSM), Interface Specifications
   · CORBAServices
   · CORBAFacilities
OMG Domain Specifications

CORBA Embedded Intelligence Specifications

CORBA Security Specifications

Signal and Image Processing

All of OMG’s formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. at:

OMG Headquarters
9C Medway Road

PMB 274

Milford, MA 01757
USA

Tel: +1-781-444-0404

Fax: +1-781-444-0320

Email: pubs@omg.org

Certain OMG specifications are also available as ISO standards. Please consult http://www.iso.org
Typographical Conventions

The type styles shown below are used in this document to distinguish programming statements from ordinary English. However, these conventions are not used in tables or section headings where no distinction is necessary.

- **Times/Times New Roman - 10 pt.:** Standard body text
- **Helvetica/Arial - 10 pt. Bold:** OMG Interface Definition Language (OMG IDL) and syntax elements.
- **Courier/Courier New - 10 pt. Bold:** Programming language elements
- **Helvetica/Arial - 10 pt:** Exceptions
1.1 Scope

This specification defines the interface between the Combat Management System (CMS) and a Sensor system (especially a Radar system) within a modular combat system architecture for naval platforms. It is structured to align with the objective of dividing the interface into three categories, namely subsystem services (interfaces applicable to any module within a combat system), sensor services (interfaces applicable to any sensor component within a combat system) and radar services (interfaces applicable to complex radar components within a combat system), as illustrated below.

Version 1.0 of the specification addressed the scope required for radar integration specifically.

Version 2.0 of the specification expanded the scope of the sensor services such that it provides the necessary interfaces for the integration of other combat system sensors and subsystems.
Figure 1.1 - The OARIS specification exploits specialisation and generalisation to promote modularity and extensibility.

Version 3.0 of the specification expanded the scope of the sensor services such that plots and other measurements can be shared between cooperating platform units within a joint operation.

2.2 Conformance

In order to support utilization by a range of sensors from simple navigation radars and electro-optic systems to complex multi-function radars, sonars, or electronic warfare systems the RFP defines the following compliance levels:

- **Level 1**
  - The simplest sensor operation providing just plots and tracks

- **Level 2**
  - Basic sensor operation, but a complete interface supporting control and essential system configuration for a combat system context

- **Level 3A**
  - In addition to basic operation (level 2), interfaces for training support

- **Level 3B**
  - In addition to basic operation (level 2), full system configuration interfaces

- **Level 3C**
  - In addition to basic operation (level 2), the full track and plot reporting interfaces

- **Level 3D**
  - In addition to basic operation (level 2), the engagement support interface

- **Level 3E**
  - In addition to basic operation (level 2), the advanced radar interfaces

- **Level 3F**
  - In addition to basic operation (level 2), compliance with C2INav

- **Level 3G**
In addition to basic operation (level 2), compliance with METOC (Not supported in thisa future version of the response) OARIS.

- **Level 3H**
  - In addition to basic operation (level 2), the full parameter measurement and identification assessment interfaces

- **Level 3I**
  - In addition to basic operation (level 2), the interfaces to support cooperative plot sharing

Radars conforming to this specification shall indicate which compliance levels are supported. The following options are possible:

- **Level 1**
- **Level 2**
- **Any combination of levels 3A to 3E (in addition to level 2)**

In order to comply with the specification levels the following respective interfaces shall be supported in full, with the exception of level 3C where at least one of the environment types (Space/Air/Land/Surface) shall be supported and appropriately qualified, e.g., level 3C Air and Surface.

<table>
<thead>
<tr>
<th>Compliance Level</th>
<th>Required Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Register Interest</td>
</tr>
<tr>
<td></td>
<td>Track Reporting</td>
</tr>
<tr>
<td></td>
<td>Plot Reporting</td>
</tr>
<tr>
<td>2</td>
<td>Control Interface Connection</td>
</tr>
<tr>
<td></td>
<td>Provide Subsystem Identification</td>
</tr>
<tr>
<td></td>
<td>Provide Subsystem Services</td>
</tr>
<tr>
<td></td>
<td>Manage Subsystem Parameters</td>
</tr>
<tr>
<td></td>
<td>Provide Health State</td>
</tr>
<tr>
<td></td>
<td>Manage Mastership</td>
</tr>
<tr>
<td></td>
<td>Manage Technical State</td>
</tr>
<tr>
<td></td>
<td>Exchange Heartbeat</td>
</tr>
<tr>
<td></td>
<td>Register Interest</td>
</tr>
<tr>
<td></td>
<td>Track Reporting</td>
</tr>
<tr>
<td></td>
<td>Plot Reporting</td>
</tr>
<tr>
<td></td>
<td>Manage Operational Mode</td>
</tr>
<tr>
<td></td>
<td>Manage Tracking Zones</td>
</tr>
<tr>
<td></td>
<td>Manage Frequency Usage</td>
</tr>
<tr>
<td></td>
<td>Manage Transmission Sectors</td>
</tr>
</tbody>
</table>
| 3A | Control Battle Override  
Control Emissions  
Define Test Target Scenario  
Define Fault Script  
Control Simulation  
Control Fault Script  
Control Test Target Facility  
Control Recording  
Control Replay  
Provide Simulation Data  |
|---|---|
| 3B | Shutdown  
Restart  
Startup  
Manage Physical Configuration  
Perform Offline Test  
Receive Encyclopedic Data  |
| 3C | Receive Track Information  
Delete Sensor Track  
Initiate Track  
Perform Cued Search  
Provide Space Plots  
Provide Land Plots  
Provide Surface Plots  
Provide Air Plots  
Provide Sensor Space Tracks  
Provide Sensor Land Tracks  
Provide Sensor Surface Tracks  
Provide Sensor Air Tracks  |
| 3D | Process Target Designation  
Provide Projectile Positional Information  
Perform Missile Downlink  
Perform Missile Uplink  
Kill Assessment  
Support Surface Engagement  
Perform Splash Plotting  |
| 3E | Provide Interference Reports |
Further detail on service interfaces contained within each conformance level is presented in section 7.2.

3.3 Normative References

The following normative documents contain provisions which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply.

- ALMAS (formal/2009-11-01)
- AMSM (formal/2010-11-02)
- CORBA (formal/2011-11-01,02,03)
- DDS (formal/2007-01-01)
- EVOT (formal/2009-05-01)
- HLA (IEEE 1516 2000-series and RPR DOM 2.0)
- ISO 19111 (www.iso.org)
- ISO 19115 (www.iso.org)
- METOC RFP (c4i/08-12-02)
- C2Nav (doi/19-07-03)
- Network Time Protocol (www.ntp.org)
- Precision Time Protocol (IEEE 1588 – http://www.ieee1588.com)
- SoaML (www.omg.org/spec/SoaML)
4.4 Terms and Definitions

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

- **AAW** (Anti-Air Warfare)
- AB (Architecture Board)
  - ALMAS (Alert Management Service)
  - AMSM (Application Management and Status Monitoring)
- API (Application Programming Interface)
- APP (Allied Procedural Publication)
- ASuW (Anti-Surface Warfare)
- ATC (Air Traffic Control)
- BC (Business Committee)
- BCQ (Business Committee Questionnaire)
- BoD (Board of Directors)
  - CCM (CORBA Component Model)
- CMS (Combat Management System)
  - CORBA (Common Object Request Broker Architecture)
  - CSIV2 (Common Secure Interoperability Protocol Version 2)
  - CWM (Common Warehouse Metamodel)
- DAIS (Data Acquisition from Industrial Systems)
- DDS (Data Distribution Service)
- EDOC (Enterprise Distributed Object Computing)
  - EJB (Enterprise Java Bean)
- EVOT (Enhanced View of Time)
- FTF (Finalization Task Force)
  - GE (Gene Expression)
  - GIOP (General Inter-Orb Protocol)
  - GLS (General Ledger Specification)
- IDL (Interface Definition Language)
- IEC (International Electrotechnical Commission)
- IFF (Interrogation, Friend or Foe)
  - IIOP (Internet Inter-Orb Protocol)
- IPR (Intellectual Property Right)
- ISO (International Organization for Standardization)
- LOI (Letter of Intent)
- MDA (Model Driven Architecture)
- METOC (Meteorological and Oceanographic)
5.5 Symbols
No special symbols are introduced in this specification.

6.6 Additional Information

6.1 Acknowledgements
The following companies submitted this specification:

- BAE Systems
- THALES Group
7.1 Open Architecture Radar Information Specification

7.1.1 Introduction

6.2 Specification Generation

The specification is captured as an Enterprise Architect (EA) UML version 2.1 model, with this document being automatically generated as a report from the model.
7 Open Architecture Radar Information Specification

7.1 Introduction

7.1.1 Background
A Combat System on a naval warship (or platform) typically consists of, amongst other things, a Combat Management System (CMS) interfacing with a number of sensors and communication systems (e.g. DataLink network), together providing the user with a tactical picture of all the real world entities that have been detected. These are then passed to other ship systems (e.g. comms and weapon systems) to support ongoing warfighting activities.

In OARIS, sensors and these other ship systems are generalized as subsystems. OARIS partitions its data model and services into abstraction layers that are applicable to subsystems (most general), sensor and radars (most specific).

Figure 1.1 - The OARIS specification is applicable to Sensor systems within the Combat System Architecture

Sensors typically operate by recording detections of whatever physical property they are sensing (e.g. acoustic or electromagnetic events). These detections are called plots. A sensor may analyze the plots it is detecting over a period of time and make decisions about whether each plot has come from a real world object of interest, or whether the plot has been received as clutter from the environment (e.g. returns from the crests of waves). If the sensor has confidence that a number of received plots correspond to a real world object then the sensor will form a track based on those plots which is then sent to the CMS; a track being a sensor-view representation of a real-world object over time. The CMS maintains a track list which in general has contributions from all the sensor on the platform.
Where a platform is working as part of a task group then historically the platform is able to share tracks from its track list with other platforms via a DataLink network. Version 3.0 of OARIS expands the scope such that it provides the necessary interfaces for coherent sharing of plots and other measurements between cooperating platform units in a joint operation. Version 2.0 of OARIS extended the scope of services and the data model to cover the functionality and capabilities of sensors in addition to radars, whilst version 1.0 of OARIS focused on radars more specifically.

---

**Figure 1.2 - The OARIS 3.0 specification in context with other C4I specifications and other platforms (e.g. ships) implementing the standard**

### 7.1.2 Section Structure

This section of the document is organized as follows:

7.1 (this section) Introduces the OARIS specification and gives some context to the use of OARIS within a naval environment.

7.2 Provides an overview of how the various interfaces (later described in sections 7.7 to 7.9) are used by nominal components to achieve a particular level of compliance with the OARIS specification.

7.3 Identifies all the common data types used within the specification.

7.4 Identifies all the data types that are applicable to the Subsystem domain interfaces (described in 7.7).

7.5 Identifies all the data types that are applicable to the Sensor domain interfaces (described in 7.8).

7.6 Identifies all the data types that are applicable to the Radar domain interfaces (described in 7.9).

7.7 Identifies all the interfaces that are applicable to the Subsystem domain.

7.8 Identifies all the interfaces that are applicable to the Sensor domain.

7.9 Identifies all the interfaces that are applicable to the Radar domain.

### Document Structure
Open Architecture Radar Interface Standard (OARIS), v2.0

- **Overview**: Package
  - Usage Parameters: Package

- **Common**: Package
  - Common Types: Package

- **Domain**: Package
  - Subsystem Domain: Package
  - Sensor Domain: Package
  - Radar Domain: Package

- **Service**: Package
  - Subsystem Services: Package
  - Sensor Services: Package
  - Radar Services: Package
This specification is presented as:

- An overview of how the services are used to achieve levels of conformance to the standard
- Common data types used throughout
- Domain specific data types for the three domains (Subsystem, Sensor and Radar)
- Service interfaces for the three domains (Subsystem, Sensor and Radar)

**7.27.3 Usage Overview**

Parent Package: Analysis Model (PIM)
OARIS defines compliance levels as follows:

- **Level 1**: A simple radar which provides just plots and tracks
- **Level 2**: Basic radar operation, but a complete interface supporting control and essential system configuration for a combat system context
- **Level 3A**: In addition to basic operation (level 2), interfaces for training support
- **Level 3B**: In addition to basic operation (level 2), full system configuration interfaces
- **Level 3C**: In addition to basic operation (level 2), the full track and plot reporting interfaces
- **Level 3D**: In addition to basic operation (level 2), the engagement support interface
- **Level 3E**: In addition to basic operation (level 2), the advanced radar interfaces
- **Level 3F** (compliance with C2Nav) and **Level 3G** (compliance with METOC) are outside the scope of this response
- **Level 3H**: In addition to basic operation (level 2), the full parameter measurement and identification assessment interfaces.

Sensors conforming to this specification shall indicate which compliance levels are supported. The following options are possible:

- **Level 1**
- **Level 2**
- Any combination of levels 3A to 3E or 3H (in addition to level 2)

**The activity diagrams and**

In order to comply with the associated notes specification levels the following respective interfaces shall be supported in full, with the exception of level 3C where at least one of the environment types (Space/Air/Land/Surface) shall be supported and appropriately qualified, e.g. level 3C Air and Surface. This section continues below with activity and component diagrams that show how the interfaces relate to achieve the different compliance levels. The activity diagrams capture pre-requisites for interface usage, whilst the component diagrams illustrate non-normative functionality enabled by the interfaces within a compliance level. The component diagrams contain non-normative components representing subsets of a typical functional decomposition of the Subsystem and CMS interface abstractions used by the normative sections of this specification. The interfaces entailed by each conformance level are defined in sections 7.7 to 7.9 interact in order to support these compliance levels describing the subsystem, sensor and radar services.

### 7.3.1 Compliance Level 1

**Parent Package:** Usage Overview

The Compliance Level 1 required interfaces are:

- Register Interest
- Track Reporting
CMS and Subsystem partitions indicate the initiator of the service only. For example, a service initiated by the CMS may include a response from the Subsystem even though the service is in the CMS partition.
• Plot Reporting
This compliance level is aimed at the simplest integration use cases and provides an entry-point for initial adoption of the standard by implementers.

Figure 7.2.2 Compliance Level 1 (Component diagram)
This component diagram shows the interfaces realized and used by CMS and Subsystem components integrated at compliance level 1.
For compliance level 1, the radar powers up and commences track and plot reporting either without intervention or using an out of scope facility, such as a maintainer interface. The CMS detects the presence of the interface, registers interest then processes the incoming track and plot streams.
7.3.1.1 Figure 7.3 Basic Picture Compiler
A non-normative minimal example of the picture compilation function realizing the abstraction of a CMS.

7.3.1.2 Basic Sensor
A non-normative minimal example of a sensor realizing the abstraction of a Subsystem.

7.3.2 Compliance Level 2
Parent Package: Usage Overview
The Compliance Level 2 required interfaces are:
• Control Interface Connection
- Provide Subsystem Identification
- Provide Subsystem Services
- Manage Subsystem Parameters
- Provide Health State
- Manage Mastership
- Manage Technical State
- Exchange Heartbeat
- Register Interest
- Track Reporting
- Plot Reporting
- Manage Operational Mode
- Manage Tracking Zones
- Manage Frequency Usage
- Manage Transmission Sectors
- Control Battle Override
- Control Emissions

This compliance level supports core functionality required for operational usage in a fully integrated combat system.

Figure 7.4 Compliance Level 2 - Initialization (Component diagram)
This component diagram shows the interfaces realized and used by CMS and Subsystem components, to achieve initialization, integrated at compliance level 2.
Open Architecture Radar Interface Standard (OARIS), v2.0

Subsystem

CMS

{Power Applied}

Provide Subsystem Identification

Provide Subsystem Services

Exchange Heart Beat

Manage Subsystem Parameters

Register Interest

[from Compliance Level 1]

Provide Health State

Manage Mastership

Manage Technical State

ActivityFinal
For compliance level 2 a more versatile startup sequence is supported, with the subsystem and CMS going through a negotiation and configuration stage followed by more detailed interface control and reporting, including management of reversionary modes.
This component diagram shows the interfaces realized and used by CMS.

**Manage Operational Mode**

**Activity Final**
Figure 7.4.7 Compliance Level 2 - Operational Mode (Activity diagram)

Level 2 continues to manage the operational mode while the CMS has mastership.

CMS

Manage Operational Mode

ActivityFinal

-Manage Subsystem Parameters has completed successfully and has identified the currently available operational modes and CMS has mastership

Manage Operational Mode

Provide Subsystem Services has successfully executed and CMS has mastership

-Manage Subsystem Parameters is not ONLINE

-Manage Subsystem Parameters is ONLINE

-Manage Technical State

-Manage Tracking Zones

-Manage Frequency Usage

-Manage Transmission Sectors

-Control Battle Override

-Control Emissions
This component diagram shows the interfaces realized and used by CMS and Subsystem components to achieve status monitoring of the subsystem, integrated at compliance level 2.

**Figure 7.9 Compliance Level 2 - Subsystem Setup (Activity diagram)**

Level 2 caters for continuous management of sensor configuration when the CMS has mastership.
CMS and Subsystem partitions indicate the initiator of the service only. For example a service initiated by the CMS may include a response from the subsystem even though the service is not in the Subsystem swimlane.

Define Test Target
Define Fault Scripts
Manage Technical State
[CMS has mastership]
[CMS does not have mastership]
[CMS has mastership]
[Subsystem is not in a READY or ONLINE state]
Manage Mastership
[Subsystem is in a READY or ONLINE state]
Control Simulation
[Simulation mode is OFF]
[Simulation mode is ON]
Control Fault Script
[CMS decides to activate a fault script that has been previously defined]
ActivityFinal
Control Test Target Facility
[CMS decides to activate a test target scenario that has been previously defined]
Figure 7.6.10 Compliance Level 2 - Subsystem Setup (Component diagram)

This component diagram shows the interfaces realized and used by CMS and Sensor Subsystem components, to achieve subsystem setup, integrated at compliance level 2. The CMS sets up the sensor subsystem such that it has the required configuration to perform the necessary operational role for the current task or mission assigned to the platform.

7.3.2.1 General CMS Sensor Manager
A non-normative example of sensor management function within a CMS; contains functionality to support system users in configuring combat system sensors to support their tasking and mission objectives.

7.3.2.2 General Sensor Configuration Manager
A non-normative example of a sensor configuration management function that allows the sensor to be configured to best perform the tasks to which it is allocated.

7.3.3 Compliance Level 3A
Parent Package: Usage Overview
The Compliance Level 3A required interfaces are:

- Define Test Target Scenario
- Define Fault Scripts
- Control Simulation
- Control Fault Script
- Control Test Target Facility
- Control Recording
- Control Replay
- Provide Simulation Data

This compliance level supports specialized functionality relating to simulation, online test and analysis. This level is applicable to all types of subsystem.

[Diagram of Compliance Level 3A - Fault Scripts and Test Targets (Activity diagram)]

Figure 7.11 Compliance Level 3A - Fault Scripts and Test Targets (Activity diagram)

Level 3 provides for the simulation of faults and targets for test and training purposes.
CMS and Subsystem partitions indicate the initiator of the service only. For example, a service initiated by the CMS may include a response from the subsystem even though the service is not in the Subsystem partition.
This component diagram shows the interfaces realized and used by CMS and Subsystem components, to support online diagnostic analysis, integrated at compliance level 3A.
Recording and replay facilities support recording and replay of subsystem parameters for the purposes of training and/or post exercise review.

**Figure 7.13 Compliance Level 3A - Recording/Replay (Activity diagram)**

Recording and replay facilities support recording and replay of subsystem parameters for the purposes of training and/or post exercise review.
This component diagram shows the interfaces realized and used by CMS and Subsystem components, to support record and replay for analysis, integrated at compliance level 3A.
This component diagram shows the interfaces realized and used by CMS and Subsystem components, to support simulation, integrated at compliance level 3A.
Figure 7.16 Compliance Level 3A - Simulation (Activity diagram)
The simulation interfaces are used to support training.

7.3.3.1 CMS Analysis Data Manager
A non-normative example of an analysis data management function within a CMS. This component would support CMS users in learning from tasks recently undertaken.

7.3.3.2 CMS Diagnostics Manager
A non-normative example of a diagnostics management function within a CMS. This component would enable CMS users to investigate potential faults in the combat system.

7.3.3.3 CMS On Board Training Manager
A non-normative example of a simulation management function within a CMS. A plan rehearsal function would be another example.

7.3.3.4 Subsystem Diagnostic
A non-normative example of a diagnostic function within a subsystem. Such a function enables the generation of diagnostic tests on the subsystem's other components.

7.3.3.5 Subsystem Record and Replay Manager
A non-normative example of a combined record and replay function within a subsystem to manage recording and later replay of the data the subsystem generates.

7.3.3.6 Subsystem Simulation Manager
A non-normative example of a simulation function within a subsystem enabling the subsystem to take part in federated simulated operations such as on-board training.

7.3.4 Compliance Level 3B
Parent Package: Usage Overview
The Compliance Level 3B required interfaces are:
- Shutdown
- Restart
- Startup
- Manage Physical Configuration
- Perform Offline Test
- Receive Encyclopedic Data

This compliance level supports specialized configuration and state management of the subsystem (and applies to subsystems in general).
This component diagram shows the interfaces realized and used by CMS and Subsystem components, to achieve macro state management, integrated at compliance level 3B.

These interfaces provide for more finely grained control of startup and shutdown.
Provide Subsystem Services has successfully executed

Manage Membership

CMS does not have mastership
CMS has mastership
CMS has mastership

Request Current Configuration

Manage Technical State

Subsystem is not in STANDBY
Subsystem is in STANDBY
Subsystem is in STANDBY
Subsystem is in STANDBY

Manage Physical Configuration

Activity Final
Figure 7.10.19 Compliance Level 3B - Manage Physical Configuration (Activity diagram)

These interfaces support more detailed control of the subsystem configuration.
Figure 7.11.20 Compliance Level 3B - Manage Physical Configuration (Component diagram)

This component diagram shows the interfaces realized and used by CMS and Subsystem components, to manage the subsystem's physical configuration, integrated at compliance level 3B.
Offline test provides a mechanism for diagnosing subsystem failures, after which the subsystem's technical state is adjusted accordingly.

Figure 7.21 Compliance Level 3B - Perform Offline Test (Activity diagram)

Offline test provides a mechanism for diagnosing subsystem failures, after which the subsystem’s technical state is adjusted accordingly.
This component diagram shows the interfaces realized and used by CMS and Subsystem components, to perform offline tests, integrated at compliance level 3B.
The subsystem is able to receive relevant encyclopaedic data from the CMS.
The sensor supports detailed track management.

Figure 7.13 Compliance Level 3C - Advanced Track Management (Activity diagram)

The sensor supports detailed track management.
7.3.4.1 CMS Combat System Configuration Manager
A non-normative example of a CMS function to manage the configuration of the Combat System.

7.3.4.2 CMS Subsystem Manager
A non-normative example of a CMS function to manage the state of subsystems in the combat system.

Subsystem
7.3.4.3 Configuration Manager
A non-normative example of a subsystem function to manage its configuration and state.

7.3.4.4 Subsystem Physical Configuration Manager
A non-normative example of a subsystem function to manage its physical configuration (i.e., state of hardware and associated mechanical aspects and devices).

### 7.3.5 Compliance Level 3C

**Parent Package:** Usage Overview

The Compliance Level 3C required interfaces are:

- Receive Track Information
- Delete Sensor Track
- Initiate Track
- Perform Cued Search
- Provide Space Plots
- Provide Land Plots
- Provide Surface Plots
- Provide Air Plots
- Provide Sensor Space Tracks
- Provide Sensor Land Tracks
- Provide Sensor Surface Tracks
- Provide Sensor Air Tracks

This compliance level supports specialized provision and management of tracks and plots; it applies to sensors in general.
The sensor supports reporting tracks and plots selectively based on the operational environment (space/air/land/surface).
CMS and Subsystem partitions indicate the initiator of the service only. For example, a service initiated by the CMS may include a response from the subsystem even though the service is not in the Subsystem swimlane.
The sensor supports detailed track management.

**Figure 7.26 Compliance Level 3C - Advanced Track Management (Activity diagram)**

The sensor supports detailed track management.
7.3.5.1 CMS Sensor Picture Manager
A non-normative example of a CMS picture management function.

7.3.5.2 Sensor Track Reporter
A non-normative example of a sensor function to manage track reporting.

7.3.6 Compliance Level 3D

Parent Package: Usage Overview
The Compliance Level 3D required interfaces are:

- Process Target Designation
- Provide Projectile Positional Information
- Perform Missile Downlink
- Perform Missile Uplink
- Kill Assessment
- Support Surface Engagement
- Perform Splash Plotting

This compliance level supports specialized engagement related radar functionality; it is specific to radar sensors.

Figure 7.28 Compliance Level 3D - Air Engagement Support (Activity diagram)

Level 3D provides additional information to support air engagements, including missile links and kill assessment.
CMS has determined an engagement is required against a surface track, CMS has mastership. Subsystem is ONLINE (simulated engagements may be performed in READY as well).

- Process Target Designation
- Track Reporting

CMS and Subsystem partitions indicate the initiator of the service only. For example, a service initiated by the CMS may include a response from the subsystem, even though the service is not in the subsystem's swimlane.
This component diagram shows the interfaces realized and used by CMS and Radar Subsystem components, to support air engagements, integrated at compliance level 3D.
Figure 7.30 Compliance Level 3D - Surface Engagement Support - Fire Control Radar (Activity diagram)

This provides additional surface engagement support for fire control.

Figure 7.31 Compliance Level 3D - Surface Engagement Support - Fire Control Radar (Component diagram)
This component diagram shows the interfaces realized and used by CMS and Fire Control Radar Subsystem components, to support surface engagements integrated at compliance level 3D.

Figure 7.12.32 Compliance Level 3D - Surface Engagement Support - Surveillance Radar (Activity diagram)
This provides additional surface engagement support for surveillance purposes.
7.3.6.1 CMS AAW Engagement Coordinator
A non-normative example of CMS functionality to coordinate anti-air warfare engagements.

7.3.6.2 CMS ASuW Engagement Coordinator
A non-normative example of CMS functionality to coordinate anti-surface warfare engagements.

7.3.6.3 Radar AAW Engagement Support Manager
A non-normative example of Radar Sensor functionality providing anti-air warfare engagement support.

7.3.6.4 Surface Fire Control Radar Manager
A non-normative example of Fire-Control Radar Sensor functionality providing anti-surface warfare engagement support.

7.3.6.5 Surface Surveillance Radar Engagement Support Manager
A non-normative example of Surveillance Radar Sensor functionality providing anti-surface warfare engagement support.

7.3.7 Compliance Level 3E
Parent Package: Usage Overview
The Compliance Level 3E required interfaces are:
- Provide Interference Reports
- Provide Jammer Strobes
• Provide Jammer Tracks
• Provide Area with Plot Concentration
• Provide Clutter Assessment
• Provide Jamming Effect Assessment
• Provide Performance Assessment
• Provide Nominal Performance

This compliance level is for the provision and management of specialized services to manage sensor functional performance and mitigate jamming; it is applicable to sensors in general.

Figure 7.34 Compliance Level 3E (Component diagram)

This component diagram shows the interfaces realized and used by CMS and Subsystem components integrated at compliance level 3E.
Level 3E provides for detailed interference reporting, including jammers.
These interfaces provide for reporting sensor specified and actual performance in addition to interference related information.
7.3.7.1 CMS Combat System Performance Optimizer
A non-normative example of CMS functionality to understand and hence optimize the performance of the combat system.

7.3.7.2 CMS Interference Mitigation Coordinator
A non-normative example of CMS functionality to coordinate mitigation with respect to active interference in the environment - e.g. jamming.

7.3.7.3 Sensor Functional Performance Manager
A non-normative example of sensor functionality to manage, interrogate and publish its own functional performance.
7.3.7.4 Sensor Interference Reporter
A non-normative example of sensor functionality to report interference detected in the external environment.

7.3.8 Compliance Level 3H
Parent Package: Usage Overview
The Compliance Level 3H required interfaces are:
• Allocate_Tracks_To_Stream
• Configure_Media_Streams
• Assess_Sensor_Plot
• Assess_Sensor_Track
• Configure_Measurement_Parameters
• Provide_Sensor_Plot_Parameters
• Provide_Sensor_Track_Parameters
This compliance level is for the integration of sensors other than radars and in particular the publication of parametric data, assessment of identity and classification, and to relate media streams to tracks.
These interfaces support the processing and assessment of information derived from the sensor's processing chain (especially detailed parametric data) to aid the identification and classification processes within the CMS.

Figure 7.20.37 Compliance Level 3H - Measurement and Identification Assessment (Activity diagram)

CMS and Subsystem partitions indicate the initiator of the service only. For example a service initiated by the CMS may include a response from the subsystem even though the service is not in the Subsystem swimlane.
Figure 7.38 Compliance Level 3H - Media Streaming (Component diagram)
This component diagram shows the interfaces realized and used by CMS and Subsystem components, to relate media streams to the tactical picture, integrated at compliance level 3H.

Figure 7.39 Compliance Level 3H - Picture Compilation From Plots (Component diagram)
This component diagram shows the interfaces realized and used by CMS and Subsystem components, to utilize plot-level parametric data and assessment functions, integrated at compliance level 3H.
7.3.8.1 CMS Media Manager
A non-normative example of CMS functionality to manage media streams derived from or otherwise available to the combat system.

7.3.8.2 CMS Picture Compilation
A non-normative example of CMS tactical picture management functionality.

7.3.8.3 Sensor Media Manager
A non-normative example of sensor functionality for providing a media stream.

7.3.8.4 Sensor Parameter Assessment
A non-normative example of functionality for the assessment of sensor parametric data.

7.3.8.5 Sensor Plot Detector
A non-normative example of sensor plot detection and reporting functionality.
7.3.8.6 Sensor Track Reporter
A non-normative example of functionality to report sensor tracks to the combat system.

7.3.8.7 Track Extractor
A non-normative example of functionality to extract tracks from a stream of plots.

7.3.9 Compliance Level 3I
Parent Package: Usage Overview
The Compliance Level 3I required interfaces are:
• Manage Network Participation
• Provide Networking Statistics
• Filter Plots
• Provide Sensor Characteristics
• Filter Tracks
• Label Tracks
This compliance level is for the sharing and exploitation of plots from multiple sensors across multiple platforms (e.g. ships) cooperating within a task group.

Figure 7.41 Compliance Level 3I (Activity diagram)
Figure 7.42 Compliance Level 3I - Plots (Component diagram)

This component diagram shows the interfaces realized and used by CMS and Subsystem components, to share plots within a platform, integrated at compliance level 3I.
This component diagram shows the interfaces realized and used by CMS and Subsystem components, to manage the sharing of tactical picture information, integrated at compliance level 3I.
Figure 7.44 Compliance Level 3I - External Information Exchange (Component diagram)
This component diagram shows the interfaces realized and used by CMS and Subsystem components, to exchange information between platforms, integrated at compliance level 3I.

7.3.9.1 CMS Plot Sharing Manager
A non-normative example of CMS functionality to manage the sharing of plot-level information.

7.3.9.2 CMS Tactical Picture Manager
A non-normative example of CMS functionality to produce and manage the tactical picture.

7.3.9.3 OARIS External Interface
A non-normative example of functionality to provide the external-to-platform interface for distributing plots and other information to enable the exploitation of plots across multiple platforms. This component expects to interface peer-to-peer with other equivalent components using a symmetric, bidirectional interface.

7.3.9.4 Platform A
A non-normative instance of the component, notionally resident on a nominal platform A.

7.3.9.5 Platform B
A non-normative instance of the component, notionally resident on a nominal platform B.

7.3.9.6 Plot Fuser
A non-normative example of functionality to fuse plots from multiple sensors (and platforms) into continuous tracks.

7.3.9.7 Sensor Plot Detector
A non-normative example of sensor plot detection and reporting functionality.

7.3.4 Common_Types
Parent Package: Domain_Model

This package contains the types that are common to several areas of the model. Most of the content is in three sub-packages: Coordinates_and_Positions, Shape_Model and Requests. General types are captured at the top level.
cartesian_position_type
cartesian_velocity_component_type
cartesian_velocity_type
coordinate_kind_type
coordinate_orientation_type
coordinate_origin_type
coordinate_specification_type
course_type
covariance_matrix_type
diagonal_covariance_matrix_type
duration_type
elevation_coordinate_type
elevation_interval_type
elevation_qualification_type
elevation_rate_type
full_covariance_matrix_type
height_coordinate_type
height_interval_type
latitude_coordinate_type
latitude_interval_type
longitude_coordinate_type
longitude_interval_type
polar_position_type
polar_velocity_type
position_accuracy_coordinate_type
position_coordinate_type
range_coordinate_type
range_interval_type
range_qualification_type
range_rate_type
speed_interval_type
speed_type
velocity_accuracy_coordinate_type
velocity_coordinate_type
wgs84_position_type
wgs84_velocity_type
cartesian_position_accuracy_type
cartesian_velocity_accuracy_type
polar_position_accuracy_type
polar_velocity_accuracy_type
wgs84_position_accuracy_type
wgs84_velocity_accuracy_type
Figure 7.21.45 Domain Model (Class diagram)
Type: Class octet
Package: Common_Types
Representation for a general binary type
ElementTag: Length = 1024

7.3.27.4.2 confidence_interval_type
Type: IDLStruct
Package: Common_Types
An abstraction for a range of confidence values.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimum</td>
<td>confidence_type</td>
</tr>
<tr>
<td>maximum</td>
<td>confidence_type</td>
</tr>
</tbody>
</table>

Table 7.1 - Attributes of IDLStruct confidence_interval_type

7.4.3 confidence_type
Type: Class float
Package: Common_Types
The confidence in the measurement or assessment expressed as a probability. This is the result of a
hypothesis test that the data is a measurement of real-world phenomenon corresponding to its label. For an
assessment it is the hypothesis that the assessment describes the real-world.
ElementTag: Range = 0 .. 1

7.3.37.4.4 identityfilter_id_type
Type: IDLEnum IDLTypeDef
Package: Common_Types Identity according to STANAG 5516.

7.4.5 filter_mode_type
Type: IDLEnum
Package: Common_Types
This class encapsulates the possible modes in which a filter can operate.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PENDING</td>
<td>idlEnum FILTER_TRANSMISSION</td>
</tr>
<tr>
<td>UNKNOWN</td>
<td>idlEnum FILTER_RECEPTION</td>
</tr>
<tr>
<td>ASSUMED_FRIEND</td>
<td>idlEnum FILTER_BOTH</td>
</tr>
<tr>
<td>FRIEND</td>
<td></td>
</tr>
</tbody>
</table>

Table 7.12 - Attributes of IDLEnum identityfilter_mode_type
### 7.3.4.1.1 strength_type

#### 7.4.6 identity_type

**Type:** IDLEnum  
**Package:** Common_Types  
Identity according to STANAG 5516.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PENDING</td>
<td>Value pending the completion of the initial identification process</td>
</tr>
<tr>
<td>UNKNOWN</td>
<td>Initial identification complete but identity is unknown.</td>
</tr>
<tr>
<td>ASSUMED_FRIEND</td>
<td>Assumed to be a friend</td>
</tr>
<tr>
<td>FRIEND</td>
<td>Known to be a friend</td>
</tr>
<tr>
<td>NEUTRAL</td>
<td>Known to be neutral</td>
</tr>
<tr>
<td>SUSPECT</td>
<td>Suspected to be hostile</td>
</tr>
<tr>
<td>HOSTILE</td>
<td>Known to be hostile</td>
</tr>
</tbody>
</table>

#### 7.4.7 percentage_type

**Type:** IDLTypeDef  
**Package:** Common_Types

#### 7.4.8 quality_interval_type

**Type:** IDLStruct  
**Package:** Common_Types  
An abstraction for a range of track quality values.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimum</td>
<td>The minimum inclusive value for the interval.</td>
</tr>
<tr>
<td>maximum</td>
<td>The maximum inclusive value for the interval.</td>
</tr>
</tbody>
</table>

### 7.4.9 strength_type
Strength of the measurement (for a track or plot). The precise semantics of this type are sensor subsystem specific, but a typical interpretation is as a signal to noise ratio in dB.

7.3.67.4.10 subsystem_id_type

This type provides a unique id for different subsystems. Subsystem ids shall be allocated by the platform integrator. Subsystem id equal to zero is reserved to imply applicability to all and any subsystem.

The lowest two bytes are used for designating subsystems within local control of a particular platform (e.g. a ship including any off-board sensors that it controls). The highest two bytes designate a platform within a cooperating task-force or group. (e.g. highest byte may designate country and the next highest, one of a country's platforms).

7.3.67.4.11 system_track_id_type

System Track Identification
7.3.7.4.12  time_type

Type:        Class TimeT
Package:     Common_Types

based on start of Gregorian calendar (1582-10-15T 00:00UTC)
unit: 100 nano seconds
i.a.w CORBA Time Service Time T

7.3.8.4.14  frequency_band_type

7.4.13  track_priority_type

Type:        Class unsigned short
Package:     Common_Types

The representation of the track's priority with respect to the allocation of the sensor's resources.
The meaning of track_priority_type is to assign a priority among a set of tracks based on some criteria (i.e.,
subsystem's time dedicated to a track analysis). Higher values indicate higher priority and importance and
hence that more resources should be extended.
Example of values:
1 Track While Scan (TWS)
2 Low Priority Target (LPT)
3 High Priority Target (HPT)

ElementTag: Range = 0 .. 100

7.4.14  track_quality_type

Type:        Class
Package:     Common_Types

The representation of the quality of a track for the purposes of comparison according to system specific
criteria.
ElementTag: Range = 0.0 .. 1.0

7.4.15  frequency_band_type

Type:        Class
Package:     Common_Types

An index indicating a particular frequency channel or band. The actual frequency is typically not of concern to
the command team. A band refers to a discrete frequency or a range of frequencies such bands may
overlap.

Package:     Common_Types

An index indicating a particular frequency channel or band. The actual frequency is typically not of concern to
the command team. A band refers to a discrete frequency or a range of frequencies such bands may
overlap.

7.3.9.4.16  System_Track

Parent Package:     Common_Types
A package to contain the system track class.

### Table 7.25 - Attributes of IDLStruct `system_track_type`

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>system_track_number</code></td>
<td>The identifier for the system track</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>simulated boolean</td>
<td>Whether the system track is part of a simulation</td>
</tr>
<tr>
<td>time_of_information time_type</td>
<td>The absolute time at which the information in the attributes of the system track is valid.</td>
</tr>
<tr>
<td>position_coordinate_system</td>
<td>The coordinate system used for the system track’s position.</td>
</tr>
<tr>
<td>position position_coordinate_type</td>
<td>The position of the system track.</td>
</tr>
<tr>
<td>velocity_coordinate_system</td>
<td>The coordinate system used for the system track’s velocity.</td>
</tr>
<tr>
<td>velocity velocity_coordinate_type</td>
<td>The velocity of the system track.</td>
</tr>
<tr>
<td>position_accuracy_coordinate_system</td>
<td>The coordinate system used for the system track’s position accuracy.</td>
</tr>
<tr>
<td>position_accuracy position_accuracy Coordinate_type</td>
<td>The position accuracy of the system track.</td>
</tr>
<tr>
<td>velocity_accuracy_coordinate_system</td>
<td>The coordinate system used for the system track’s velocity accuracy.</td>
</tr>
<tr>
<td>velocity_accuracy velocity_accuracy Coordinate_type</td>
<td>The velocity accuracy of the system track.</td>
</tr>
<tr>
<td>max_range_limit range_coordinate_type [0..1]</td>
<td>The estimated maximum range of the system track (for cases where the position coordinate does not specify range - i.e. bearing only).</td>
</tr>
</tbody>
</table>

### 7.4.17 Coordinates and Positions

**Parent Package:** Common Types

Definitions of types to describe positions in accordance with the ISO 19111 abstract model.
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>velocity_accuracy_coordinate_system</td>
<td></td>
</tr>
<tr>
<td>coordinate_specification_type [0..1]</td>
<td></td>
</tr>
<tr>
<td>velocity_accuracy_velocity_accuracy_coordinate_type [0..1]</td>
<td></td>
</tr>
<tr>
<td>max_range_limit_range_coordinate_type [0..1]</td>
<td></td>
</tr>
</tbody>
</table>

7.3.101.1.1 Coordinates and Positions

Parent Package: Common Types

Definitions of types to describe positions, in accordance with the ISO 19111 abstract model.
To offer flexibility, three variants of coordinate system representation are supported—corresponding to the coordinate kind type enumerated. An implementation should support one kind for each relevant interface as defined by the coordinate specification type value, and it should only send data of that variant and should check that it only receives data of that variant. It should not implement conversion of data in an unexpected variant. Receipt of such data constitutes an error in the operation of the interface.
To offer flexibility, three variants of coordinate system representation are supported—corresponding to the coordinate_kind_type enumerated. An implementation should support one kind for each relevant interface as defined by the coordinate_specification_type value, and it should only send data of that variant and it should check that all data received is of that variant. It should not implement conversion of data in an unexpected variant. Receipt of such data constitutes an error in the operation of the interface.
To offer flexibility, three variants of coordinate system representation are supported - corresponding to the coordinate_kind_type enumerate. An implementation should support on kind for each relevant interface as defined by the coordinate_specification_type value, and it should only send data of that variant and it should check that all data received is of that variant. It should not implement conversion of data in an unexpected variant. Receipt of such data constitutes an error in the operation of the interface.

```idlUnion```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
Choice of SI units and double base type reflects the use of broadest international standard and a flexible representation for many represent any large and very small distances with equal precision. It is noted that there are other military international standards (e.g. STANAGs) which sometimes make different choices. However, these often reflect pressures to represent data in the most compact format, e.g. legacy systems or secure wireless communications.

tags
Range = -90 .. 90
Resolution = 1 e-6
Unit = deg

double «idlTypedef»
latitude_coordinate_type
 tags
Range = -180 .. 180
Resolution = 1 e-6
Unit = deg
double «idlTypedef»
longitude_coordinate_type

tags
Issue = Range = -1 e4 .. 1 e6
Resolution = 1 e-3
Unit = m
double «idlTypedef»
height_coordinate_type
To offer flexibility, three variants of coordinate system representation are supported - corresponding to the coordinate_kind_type enumerate. An implementation should support one kind for each relevant interface as defined by the coordinate_specification_type value, and it should only send data of that variant and it should check that all data received is of that variant. It should not implement conversions of data in an unexpected variant. Receipt of such data constitutes an error in the operation of the interface.

Choice of SI units and double base type reflects the use of broadest international standard and a flexible representation (it may represent very large and very small distances with equal precision). It is noted that there are other military international standards (e.g. STANAGs), which sometimes make different choices. However, these often reflect pressures to represent data in the most compact format - e.g. legacy systems or secure wireless communication.
Open Architecture Radar Interface Standard (OARIS), v2.0

spread: range_coordinate_type [0..1]
accuracy: range_coordinate_type

«idlStruct»
range_qualification_type

spread: elevation_coordinate_type [0..1]
accuracy: elevation_coordinate_type

«idlStruct»
elevation_qualification_type

spread: azimuth_coordinate_type [0..1]
accuracy: azimuth_coordinate_type

«idlStruct»
azimuth_qualification_type

«idlUnion»
covariance_matrix_type
«idlCase»
diagonal_covariance_matrix: diagonal_covariance_matrix_type
full_covariance_matrix: full_covariance_matrix_type
full_2d_covariance_matrix: full_2d_covariance_matrix_type
«idlStruct»
diagonal_covariance_matrix_type
xx_variance: float
yy_variance: float
zz_variance: float
vxvx_variance: float
vyvy_variance: float
vzvz_variance: float
«idlStruct»
full_covariance_matrix_type
xx_variance: float
xy_variance: float
xz_variance: float
xvx_variance: float
xvy_variance: float
xvz_variance: float
yy_variance: float
yz_variance: float
yvx_variance: float
yvy_variance: float
yvz_variance: float
zz_variance: float
zvx_variance: float
zvy_variance: float
zvz_variance: float
vxvx_variance: float
vxvy_variance: float
vxvz_variance: float
vyvy_variance: float
vyvz_variance: float
vzvz_variance: float
«idlStruct»
full_2d_covariance_matrix_type
xx_variance: float
xy_variance: float
xvx_variance: float
xvy_variance: float
yy_variance: float
yvx_variance: float
yvy_variance: float
vxvx_variance: float
vxvy_variance: float
vyvy_variance: float
Figure 7.25.49 Covariance and Qualification (Class diagram)
<table>
<thead>
<tr>
<th>IDL Struct</th>
<th>Start</th>
<th>Stop</th>
</tr>
</thead>
<tbody>
<tr>
<td>latitude_interval_type</td>
<td>start: latitude_coordinate_type</td>
<td>stop: latitude_coordinate_type</td>
</tr>
<tr>
<td>longitude_interval_type</td>
<td>start: longitude_coordinate_type</td>
<td>stop: longitude_coordinate_type</td>
</tr>
<tr>
<td>height_interval_type</td>
<td>start: height_coordinate_type</td>
<td>stop: height_coordinate_type</td>
</tr>
</tbody>
</table>
«idlStruct»
azimuth_interval_type
+ start: azimuth_coordinate_type
+ stop: azimuth_coordinate_type

«idlStruct»
elevation_interval_type
+ start: elevation_coordinate_type
+ stop: elevation_coordinate_type

«idlStruct»
range_interval_type
+ start: range_coordinate_type
+ stop: range_coordinate_type

«idlStruct»
latitude_interval_type
+ start: latitude_coordinate_type
+ stop: latitude_coordinate_type

«idlStruct»
longitude_interval_type
+ start: longitude_coordinate_type
+ stop: longitude_coordinate_type

«idlStruct»
height_interval_type
+ start: height_coordinate_type
+ stop: height_coordinate_type

«idlStruct»
cartesian_interval_type
+ start: cartesian_coordinate_type
+ stop: cartesian_coordinate_type

unsigned long long
«idlTypedef»
duration_type

double
«idlTypedef»
cartesian_velocity_component_type

double
cartesian_velocity_type
+ x_dot: cartesian_velocity_component_type
+ y_dot: cartesian_velocity_component_type
+ z_dot: cartesian_velocity_component_type [0..1]

double
cartesian_velocity_rate_type

double
azimuth_velocity_rate_type

double
elevation_velocity_rate_type

double
range_velocity_rate_type

double
polar_velocity_type
+ azimuth_rate: azimuth_rate_type
+ elevation_rate: elevation_rate_type [0..1]
+ range_rate: range_rate_type [0..1]

double
azimuth_rate_type

double
elevation_rate_type

double
range_rate_type

«idlStruct»
absolute_duration_type
+ start: time_type
+ stop: time_type

«idlStruct»
cartesian_interval_type
+ start: cartesian_coordinate_type
+ stop: cartesian_coordinate_type

Figure 7.26.50 Intervals (Class diagram)
Open Architecture Radar Interface Standard (OARIS), v2.0

To offer flexibility, three variants of coordinate system representation are supported—corresponding to the coordinate_kind_type enumerant. An implementation should support one kind for each relevant service as defined by the coordinate_specification_type value, and it should only send data of that variant and it should check that all data received is of that variant. It should not implement conversion of data to an unexpected variant. Receipt of such data constitutes an error in the operation of the interface. These representations are supported:

- Cartesian coordinate system (time derivatives within a Cartesian coordinate system; range rate, bearing rate, etc.).
- Polar coordinate system (range rate, bearing rate, etc.).
- Course and speed relative to the WGS84 spheroid.

Notes:

An implementation should support one variant for each relevant service as defined by the coordinate_specification_type value, and it should only send data of that variant. It should check that all data received is of that variant. It should not implement conversion of data to an unexpected variant. Receipt of such data constitutes an error in the operation of the interface. These representations are supported:

- Cartesian coordinate system (time derivatives within a Cartesian coordinate system; range rate, bearing rate, etc.).
- Polar coordinate system (range rate, bearing rate, etc.).
- Course and speed relative to the WGS84 spheroid.
To offer flexibility, three variants of coordinate system representation are supported - corresponding to the coordinate_kind_type enumerant. An implementation should support one kind for each relevant service as defined by the coordinate_specification_type value, and it should only send data of that variant and check that all data received is of that variant. It should not implement conversion of data in an unexpected variant. Receipt of such data constitutes an error in the operation of the interface. Three representations are supported: time derivatives within a Cartesian coordinate system; time derivatives of a polar coordinate system (range rate, bearing rate etc.); course and speed relative to the WGS84 spheroid.
7.3.10.17.4.17.1 absolute_duration_type

Type: Class

Package: Coordinates_and_Positions

This class represents a duration fixed to an absolute point in time.

Package: Coordinates_and_Positions

This class represents a duration fixed to an absolute point in time.
### Attributes of Class `absolute_duration_type`

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>start_time_type</code></td>
<td>The earliest value at the beginning of the duration</td>
</tr>
<tr>
<td><code>stop_time_type</code></td>
<td>The latest value at the end of the duration</td>
</tr>
</tbody>
</table>

#### 7.4.17.2 `angle_of_climb_type`

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>start_time_type</code></td>
<td></td>
</tr>
<tr>
<td><code>stop_time_type</code></td>
<td></td>
</tr>
</tbody>
</table>

#### 7.3.10.2 1.1.1.1 `angle_of_climb_type`

Type: `Class double`  
Package: `Coordinates_and_Positions`  
The angle representing the direction of travel relative to the horizontal. Up is positive.  
ElementTag: Range = \(-\pi/2 \ldots \pi/2\)  
ElementTag: Resolution = 1 e-3  
ElementTag: Unit = Rad

#### 7.4.17.3 `azimuth_coordinate_type`  
Type: `Class double`  
Package: `Coordinates_and_Positions`  
Axis in the azimuth direction, i.e. horizontal angle from the associated coordinate system reference. Radians, positive clockwise from above.  
See diagram note on choice of SI units  
ElementTag: Range = 0 .. 2 pi  
ElementTag: Resolution = 1 e-4  
ElementTag: Unit = rad

#### 7.4.17.4 `azimuth_interval_type`
7.4.17.5 azimuth_qualification_type

**Type:** IDLStruct  
**Package:** Coordinates_and_Positions

Qualifies a measurement of azimuth with attributes of accuracy and, if possible, variability.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>spread azimuth_coordinate_type [0..1]</td>
<td>The spread of the measurement. The combined measures of spread should encompass the full extent of the plot. This attribute is optional. Not all sensors are capable of measuring it.</td>
</tr>
<tr>
<td>accuracy azimuth_coordinate_type</td>
<td>The accuracy of the measurement, equal to one standard deviation of uncertainty.</td>
</tr>
</tbody>
</table>
spread azimuth_coordinate_type [0..1]
The spread of the measurement. The combined measures of spread should encompass the full extent of the plot. This attribute is optional. Not all sensors are capable of measuring it.

accuracy azimuth_coordinate_type
The accuracy of the measurement, equal to one standard deviation of uncertainty.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>start cartesian_coordinate_type</td>
<td></td>
</tr>
<tr>
<td>start cartesian_coordinate_type</td>
<td>Lower valued, starting coordinate.</td>
</tr>
<tr>
<td>stop cartesian_coordinate_type</td>
<td>Higher valued, ending coordinate.</td>
</tr>
<tr>
<td>stop cartesian_coordinate_type</td>
<td></td>
</tr>
</tbody>
</table>

**Table 7.69 - Attributes of IDLStruct cartesian_interval_type**

7.4.17 cartesian_position_type
Coordinates in a Cartesian reference frame as described by a coordinate specification object

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>start cartesian_coordinate_type</td>
<td></td>
</tr>
<tr>
<td>stop cartesian_coordinate_type</td>
<td></td>
</tr>
</tbody>
</table>
Coordinates in a Cartesian reference frame as described by a coordinate specification object

### Table 7.210 - Attributes of IDLStruct cartesian_position_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_coordinate cartesian_coordinate_type</td>
<td></td>
</tr>
<tr>
<td>x_coordinate cartesian_coordinate_type [0..1]</td>
<td>Optional as some sensors are 2D (horizontal plane or no elevation information)</td>
</tr>
<tr>
<td>y_coordinate cartesian_coordinate_type</td>
<td>The coordinate of the position on the x-axis</td>
</tr>
<tr>
<td>y_coordinate cartesian_coordinate_type</td>
<td>The coordinate of the position on the y-axis</td>
</tr>
<tr>
<td>z_coordinate cartesian_coordinate_type [0..1]</td>
<td>The coordinate of the position on the z-axis. Optional as some sensors are 2D (horizontal plane or no elevation information)</td>
</tr>
<tr>
<td>z_coordinate cartesian_coordinate_type</td>
<td>The coordinate of the position on the z-axis. Optional as some sensors are 2D (horizontal plane or no elevation information)</td>
</tr>
</tbody>
</table>

### 7.4.17.10 cartesian_velocity_component_type

**Type:** IDLTypeDef double  
**Package:** Coordinates_and_Positions

ElementTag: Range = -1 e5 .. 1 e5  
ElementTag: Resolution = 0.01  
ElementTag: Unit = m/s

### 7.4.17.11 cartesian_velocity_type

**Type:** IDLStruct  
**Package:** Coordinates_and_Positions

### Table 7.311 - Attributes of IDLStruct cartesian_velocity_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_dot cartesian_velocity_component_type</td>
<td></td>
</tr>
<tr>
<td>y_dot cartesian_velocity_component_type</td>
<td></td>
</tr>
<tr>
<td>z_dot cartesian_velocity_component_type [0..1]</td>
<td></td>
</tr>
</tbody>
</table>
The rate of change in the x-axis coordinate corresponding to the velocity

The rate of change in the y-axis coordinate corresponding to the velocity

The rate of change in the z-axis coordinate corresponding to the velocity

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>enum CARTESIAN</td>
<td></td>
</tr>
<tr>
<td>enum POLAR</td>
<td></td>
</tr>
<tr>
<td>enum WGS84</td>
<td></td>
</tr>
</tbody>
</table>
**enum** CARTESIAN
Indicates a Cartesian Coordinate System

**enum** POLAR
Indicates a polar coordinate system

**enum** WGS84
Indicates a coordinate system based on the WGS-84 spheroid

### 7.4.17.13 coordinate_orientation_type

#### 7.3.10.13 Type: **IDLEnum**

**Package:** Coordinates_and_Positions

This enumeration defines the set of coordinate systems, which compliant implementations may use. A compliant implementation may not fully support all of these coordinate systems.

**Package:** Coordinates_and_Positions

This enumeration defines the set of coordinate systems, which compliant implementations may use. A compliant implementation may not fully support all of these coordinate systems.

#### Table 7.4013 - Attributes of IDLEnum coordinate_orientation_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>enum</strong> NORTH_HORIZONTAL</td>
<td>Valid for Polar Coordinate Kind&lt;br&gt;Azimuth has origin (0.0) at North, positive clockwise, measured in the horizontal plane&lt;br&gt;Elevation has origin (0.0) at the Horizontal, positive up, measured in the vertical plane.</td>
</tr>
<tr>
<td><strong>enum</strong> NORTH_DOWN</td>
<td>Valid for Polar Coordinate Kind&lt;br&gt;Azimuth has origin (0.0) at North, clockwise positive, measured in the horizontal plane&lt;br&gt;Elevation has origin (0.0) when pointing directly down, and 180.0 degrees when pointing directly up, measured in the vertical plane.</td>
</tr>
<tr>
<td><strong>enum</strong> EAST_NORTH_UP</td>
<td>Valid for Cartesian coordinate type&lt;br&gt;x is positive to the East&lt;br&gt;y is positive to the North&lt;br&gt;z is positive up</td>
</tr>
<tr>
<td><strong>enum</strong> EAST_NORTH_DOWN</td>
<td>Valid for Cartesian coordinate type&lt;br&gt;x is positive to the East&lt;br&gt;y is positive to the North&lt;br&gt;z is positive down</td>
</tr>
<tr>
<td><strong>enum</strong> NORTH_EAST_UP</td>
<td>Valid for Cartesian coordinate type&lt;br&gt;x is positive to the North&lt;br&gt;y is positive to the East&lt;br&gt;z is positive up</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------</td>
</tr>
</tbody>
</table>
| «enum» NORTH_EAST_DOWN | Valid for Cartesian coordinate type  
x is positive to the North  
y is positive to the East  
z is positive down |
| «enum» EARTH_CENTRED   | Cartesian system with origin at centre of the Earth  
absolute reference point  
x positive through Greenwich meridian  
y positive through 90 degrees east (of Greenwich meridian)  
z positive through north pole  
x & y are in the equatorial plane |
| «enum» LAT_LONG_HEIGHT | WGS84 has unique well-defined orientation (NIMA  
Technical Report TR8350.2) |
| «enum» NORTH_HORIZONTAL| Valid for Polar Coordinate Kind  
Azimuth has origin (0.0) at North, positive clockwise,  
measured in the horizontal plane  
Elevation has origin (0.0) at the Horizontal, positive up,  
measured in the vertical plane |
| «enum» NORTH_DOWN      | Valid for Polar Coordinate Kind  
Azimuth has origin (0.0) at North, clockwise positive,  
measured in the horizontal plane  
Elevation has origin (0.0) when pointing directly down,  
and 180.0 degrees when pointing directly up, measured  
in the vertical plane |
| «enum» EAST_NORTH_UP   | Valid for Cartesian coordinate type  
x is positive to the East  
y is positive to the North  
z is positive up |
| «enum» EAST_NORTH_DOWN | Valid for Cartesian coordinate type  
x is positive to the East  
y is positive to the North  
z is positive down |
| «enum» NORTH_EAST_UP   | Valid for Cartesian coordinate type  
x is positive to the North  
y is positive to the East  
z is positive up |
| «enum» NORTH_EAST_DOWN | Valid for Cartesian coordinate type  
x is positive to the North  
y is positive to the East  
z is positive down |
<table>
<thead>
<tr>
<th>Enumerated Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| **EARTH CENTRED**    | Cartesian system with origin at centre of the Earth (absolute reference point)  
|                      | x positive through Greenwich meridian  
|                      | y positive through 90 degrees east (of Greenwich meridian)  
|                      | z positive through north pole  
|                      | x & y are in the equatorial plane                                                                                                          |
| **LAT LONG HEIGHT**  | WGS84 has unique well-defined orientation (NIMA Technical Report TR8350.2)                                                                  |
| **STERN KEEL**       | Valid for Polar Coordinate Kind  
|                      | This is a platform orientation relative frame  
|                      | Azimuth has origin (0.0) in line with the ship's stern (heading), measured anti-clockwise  
|                      | Elevation has origin (0.0) when pointing directly down to the keel (perpendicular to the current inclination of the deck-level, not necessarily to the Earth's surface) |
| **STERN DECK LEVEL** | Valid for Polar Coordinate Kind  
|                      | This is a platform orientation relative frame  
|                      | Azimuth has origin (0.0) in line with the ship's stern (heading), measured anti-clockwise  
|                      | Elevation has origin (0.0) when pointing parallel to the deck-level (not necessarily parallel to the Earth's surface)                      |
| **STERN STARBOARD MAST** | Valid for Cartesian coordinate type  
|                      | This is a platform orientation relative frame  
|                      | x is positive towards the stern (negative to bow)  
|                      | y is positive to starboard (negative to port)  
|                      | z is positive towards the mast (negative to keel)                                                                                           |
| **STERN STARBOARD KEEL** | Valid for Cartesian coordinate type  
|                      | This is a platform orientation relative frame  
|                      | x is positive towards the stern (negative to bow)  
|                      | y is positive to starboard (negative to port)  
|                      | z is positive towards the keel (negative to mast)                                                                                           |
STERN_DECK_LEVEL
Valid for Polar Coordinate Kind
This is a platform orientation relative frame
Azimuth has origin (0.0) in line with the ship's stern (heading), measured anti-clockwise
Elevation has origin (0.0) when pointing parallel to the deck level (not necessarily parallel to the Earth's surface)

STERN_STARBOARD_MAST
Valid for Cartesian coordinate type
This is a platform orientation relative frame
x is positive towards the stern (negative to bow)
y is positive to starboard (negative to port)
z is positive towards the mast (negative to keel)

STERN_STARBOARD_KEEL
Valid for Cartesian coordinate type
This is a platform orientation relative frame
x is positive towards the stern (negative to bow)
y is positive to starboard (negative to port)
z is positive towards the keel (negative to mast)

7.4.17.14 coordinate_origin_type
7.3.10.14 Type: IDLEnum
Package: Coordinates_and_Positions

Table 7.414 - Attributes of IDLEnum coordinate_origin_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLATFORM_REFERENCE_POINT</td>
<td>The origin of the coordinate system is 'well known' reference point for the platform (on which the CMS and subsystem reside)</td>
</tr>
<tr>
<td>SENSOR_REFERENCE_POINT</td>
<td>The origin for the coordinate system is the 'well known' reference/datum point for the sensor, which is interacting using the interface.</td>
</tr>
<tr>
<td>ABSOLUTE_REFERENCE_POINT</td>
<td>The origin for the coordinate system is a fixed point in Earth (WGS84) coordinates. This point is known to the CMS and Subsystems using the interface by means beyond the scope of the interface.</td>
</tr>
<tr>
<td>PLATFORM_REFERENCE_POINT</td>
<td>The origin of the coordinate system is 'well known' reference point for the platform (on which the CMS and subsystem reside)</td>
</tr>
</tbody>
</table>
### coordinate_specification_type

7.4.17.15  
**Type:** IDLStruct  
**Package:** Coordinates_and_Positions  
Specifies the interpretation of position coordinate_type and velocity coordinate_type.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>kind coordinate_kind_type</td>
<td>The kind of coordinate system used.</td>
</tr>
<tr>
<td>orientation coordinate_orientation_type</td>
<td>The orientation convention used by the coordinates</td>
</tr>
<tr>
<td>origin coordinate_origin_type</td>
<td>The meaning of the coordinate origin.</td>
</tr>
</tbody>
</table>

7.14.17.16  
**course_type**

<table>
<thead>
<tr>
<th>Attribute</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>kind coordinate_kind_type</td>
<td></td>
</tr>
<tr>
<td>orientation coordinate_orientation_type</td>
<td></td>
</tr>
</tbody>
</table>
7.3.10.16.1.1.1 course_type
Type: Class double
Package: Coordinates_and_Positions
The angle representing the direction of travel relative to North in the horizontal plane. Clockwise (facing down) is positive.
ElementTag: Range = 0 .. 2 pi
ElementTag: Resolution = 1 e-3
ElementTag: Unit = rad

7.3.10.17.4.17.17 covariance_matrix_type
Type: Class
Package: Coordinates_and_Positions
This class represents a covariance matrix for coordinate estimates and their time derivatives through a choice of formats.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlCase» diagonal_covariance_matrix</td>
<td>diagonal covariance matrix type</td>
</tr>
<tr>
<td>full_2d_covariance_matrix</td>
<td>full 2d covariance matrix type</td>
</tr>
<tr>
<td>full_covariance_matrix</td>
<td>full covariance matrix type</td>
</tr>
<tr>
<td>diagonal_covariance_matrix</td>
<td>The diagonal matrix option</td>
</tr>
<tr>
<td>full_2d_covariance_matrix</td>
<td>the full 2d covariance option</td>
</tr>
<tr>
<td>full_covariance_matrix</td>
<td>the full covariance option</td>
</tr>
</tbody>
</table>

7.4.17.18 diagonal_covariance_matrix_type
7.3.10.18 Type: Class
Package: Coordinates_and_Positions
Covariance of just the diagonal elements (i.e., the variance of the coordinate estimates).
Package: Coordinates_and_Positions
Covariance of just the diagonal elements (i.e., the variance of the coordinate estimates).
### Table 7.1417 - Attributes of Class diagonal_covariance_matrix_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>xx_variance</td>
<td>The variance of the x coordinate value</td>
</tr>
<tr>
<td>yy_variance</td>
<td>The variance of the y coordinate value</td>
</tr>
<tr>
<td>zz_variance</td>
<td>The variance of the z coordinate value</td>
</tr>
<tr>
<td>vxvx_variance</td>
<td>The variance of the x component of velocity</td>
</tr>
<tr>
<td>vyvy_variance</td>
<td>The variance of the y component of velocity</td>
</tr>
<tr>
<td>vzvz_variance</td>
<td>The variance of the z component of velocity</td>
</tr>
</tbody>
</table>

#### 7.3.10.19 7.4.17.19 duration_type
**Type:** Class unsigned long long
**Package:** Coordinates_and_Positions
The length of a time interval (not fixed to an absolute point in time). unit: 100 nano seconds

#### 7.3.10.20 7.4.17.20 elevation_coordinate_type
**Type:** Class double
**Package:** Coordinates_and_Positions
Axis in the direction of elevation, i.e. vertical angle from the associated coordinate system datum, radians, positive up.
See diagram note on choice of SI units.
ElementTag: Range = -pi / 2 .. pi / 2
ElementTag: Resolution = 1 e-4
ElementTag: Unit = rad

#### 7.3.10.21 7.4.17.21 elevation_interval_type
**Type:** Class
**Package:** Coordinates_and_Positions
**Package:** Coordinates_and_Positions

### Table 7.1518 - Attributes of Class elevation_interval_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>start elevation_coordinate_type</td>
<td></td>
</tr>
<tr>
<td>stop elevation_coordinate_type</td>
<td></td>
</tr>
<tr>
<td>start elevation_coordinate_type</td>
<td>The lower starting elevation value.</td>
</tr>
<tr>
<td>stop elevation_coordinate_type</td>
<td>The higher ending elevation value.</td>
</tr>
</tbody>
</table>

### 7.4.17.22 elevation_qualification_type

**Type:** IDLStruct

**Package:** Coordinates_and_Positions

Qualifies a measurement of elevation with attributes of accuracy and, if possible, variability.
### Table 7.19 - Attributes of IDLStruct elevation_qualification_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>spread elevation_coordinate_type [0..1]</td>
<td>The spread of the measurement. The combined measures of spread should encompass the full extent of the plot. This attribute is optional. Not all sensors are capable of measuring it.</td>
</tr>
<tr>
<td>accuracy elevation_coordinate_type</td>
<td>The accuracy of the measurement, equal to one standard deviation of uncertainty.</td>
</tr>
</tbody>
</table>

### 7.4.17.23 elevation_rate_type

<table>
<thead>
<tr>
<th>Type:</th>
<th>Class double</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package:</td>
<td>Coordinates_and_Positions radians per second</td>
</tr>
<tr>
<td>ElementTag:</td>
<td>Range = -100 .. 100</td>
</tr>
<tr>
<td>ElementTag:</td>
<td>Resolution = 1 e-4</td>
</tr>
<tr>
<td>ElementTag:</td>
<td>Unit = rad/s</td>
</tr>
</tbody>
</table>

### 7.4.17.24 full_2d_covariance_matrix_type

<table>
<thead>
<tr>
<th>Type:</th>
<th>IDLStruct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package:</td>
<td>Coordinates_and_Positions</td>
</tr>
<tr>
<td>The full covariance terms (in triangular form as necessarily a symmetric matrix) for reports in just the x and y dimensions.</td>
<td></td>
</tr>
<tr>
<td>Package:</td>
<td>Coordinates_and_Positions</td>
</tr>
<tr>
<td>The full covariance terms (in triangular form as necessarily a symmetric matrix) for reports in just the x and y dimensions.</td>
<td></td>
</tr>
</tbody>
</table>

### Table 7.1620 - Attributes of IDLStruct full_2d_covariance_matrix_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>xx_variance</td>
<td></td>
</tr>
<tr>
<td>xy_variance</td>
<td></td>
</tr>
<tr>
<td>xvy_variance</td>
<td></td>
</tr>
<tr>
<td>yy_variance</td>
<td></td>
</tr>
<tr>
<td>yvx_variance</td>
<td></td>
</tr>
<tr>
<td>yyyy_variance</td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>xx_variance float</td>
<td>The variance of the x coordinate value</td>
</tr>
<tr>
<td>xy_variance float</td>
<td>The covariance of the x coordinate with the y coordinate.</td>
</tr>
<tr>
<td>xxs_variance float</td>
<td>The covariance of the x coordinate with the x velocity coordinate.</td>
</tr>
<tr>
<td>xys_variance float</td>
<td>The covariance of the x coordinate with the y velocity coordinate.</td>
</tr>
<tr>
<td>yy_variance float</td>
<td>The variance of the y coordinate value</td>
</tr>
<tr>
<td>vxx_variance float</td>
<td>The covariance of the y coordinate with the x velocity coordinate.</td>
</tr>
<tr>
<td>vyx_variance float</td>
<td>The covariance of the v coordinate with the x velocity coordinate.</td>
</tr>
<tr>
<td>vyv_variance float</td>
<td>The variance of the y component of velocity</td>
</tr>
<tr>
<td>vyx_variance float</td>
<td>The covariance of the v coordinate with the y velocity coordinate.</td>
</tr>
<tr>
<td>vyy_variance float</td>
<td>The variance of the y component of velocity</td>
</tr>
</tbody>
</table>

7.4.17.25  **full_covariance_matrix_type**

<table>
<thead>
<tr>
<th>Type: Class</th>
<th>Package: Coordinates_and_Positions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Triangular representation of a full covariance matrix (which is by definition symmetric).</strong></td>
<td></td>
</tr>
</tbody>
</table>

7.3.10.25  **Coordinates_and_Positions**

<table>
<thead>
<tr>
<th>Type: Class</th>
<th>Package: Coordinates_and_Positions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Triangular representation of a full covariance matrix (which is by definition symmetric).</strong></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>xx_variance float</td>
<td>The variance of the x coordinate value</td>
</tr>
<tr>
<td>xy_variance float</td>
<td>The covariance of the x coordinate with the y coordinate.</td>
</tr>
<tr>
<td>xz_variance float</td>
<td>The covariance of the x coordinate with the z coordinate.</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>xvx_variance</code></td>
<td>The covariance of the x coordinate with the x velocity coordinate.</td>
</tr>
<tr>
<td><code>xvy_variance</code></td>
<td>The covariance of the x coordinate with the y velocity coordinate.</td>
</tr>
<tr>
<td><code>xvz_variance</code></td>
<td>The covariance of the x coordinate with the z velocity coordinate.</td>
</tr>
<tr>
<td><code>yy_variance</code></td>
<td>The variance of the y coordinate value</td>
</tr>
<tr>
<td><code>yz_variance</code></td>
<td>The covariance of the y coordinate with the z coordinate.</td>
</tr>
<tr>
<td><code>yvx_variance</code></td>
<td>The covariance of the y coordinate with the x velocity coordinate.</td>
</tr>
<tr>
<td><code>yvy_variance</code></td>
<td>The covariance of the y coordinate with the y velocity coordinate.</td>
</tr>
<tr>
<td><code>yvz_variance</code></td>
<td>The covariance of the y coordinate with the z velocity coordinate.</td>
</tr>
<tr>
<td><code>zz_variance</code></td>
<td>The variance of the z coordinate value</td>
</tr>
<tr>
<td><code>zvx_variance</code></td>
<td>The covariance of the z coordinate with the x velocity coordinate.</td>
</tr>
<tr>
<td><code>zvy_variance</code></td>
<td>The covariance of the z coordinate with the y velocity coordinate.</td>
</tr>
<tr>
<td><code>zvz_variance</code></td>
<td>The covariance of the z coordinate with the z velocity coordinate.</td>
</tr>
<tr>
<td><code>vxvx_variance</code></td>
<td>The variance of the x component of velocity</td>
</tr>
<tr>
<td><code>vxvy_variance</code></td>
<td>The covariance of the x velocity coordinate with the y velocity coordinate.</td>
</tr>
<tr>
<td><code>vxvz_variance</code></td>
<td>The covariance of the x velocity coordinate with the z velocity coordinate.</td>
</tr>
<tr>
<td><code>vyvy_variance</code></td>
<td>The variance of the y component of velocity</td>
</tr>
<tr>
<td><code>vyvz_variance</code></td>
<td>The covariance of the y velocity coordinate with the z velocity coordinate.</td>
</tr>
<tr>
<td><code>vzvz_variance</code></td>
<td>The variance of the z component of velocity</td>
</tr>
</tbody>
</table>

7.3.10.267.4.17.26
height_coordinate_type
Type: Class double

Type: Class
Package: Coordinates_and_Positions
For positive values, height above coordinate system ellipsoid, for negative values, depth below; measured in meters. This quantity is height as a measured distance rather than an inference from (for instance) barometric pressure.

See diagram note on choice of SI units:

- Range: $-1 \times 10^4 \ldots 1 \times 10^6$
- Resolution: $1 \times 10^{-3}$
- Unit: m
7.4.17.27 **height_interval_type**

**Type:** Class

**Package:** Coordinates_and_Positions

**Notes:**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>start height_coordinate_type</td>
<td></td>
</tr>
<tr>
<td>stop height_coordinate_type</td>
<td></td>
</tr>
</tbody>
</table>

7.4.17.28 **latitude_coordinate_type**

**Type:** Class double

**Package:** Coordinates_and_Positions

Degrees north (positive), south (negative) relative to coordinate system datum. See diagram note on choice of SI units.

- See diagram note on choice of SI units
- ElementTag: Range = -90 .. 90
- ElementTag: Resolution = 1 e-6
- ElementTag: Unit = deg

7.4.17.29 **latitude_interval_type**

**Type:** Class

**Package:** Coordinates_and_Positions

**Notes:**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>start latitude_coordinate_type</td>
<td></td>
</tr>
<tr>
<td>stop latitude_coordinate_type</td>
<td></td>
</tr>
</tbody>
</table>

7.3.10.28 **longitude_coordinate_type**

**Type:** Class double

**Package:** Coordinates_and_Positions

**Notes:**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>start longitude_coordinate_type</td>
<td></td>
</tr>
<tr>
<td>stop longitude_coordinate_type</td>
<td></td>
</tr>
</tbody>
</table>
### Table 7.4.17.30 - Attributes of Class `longitude_coordinate_type`

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>start latitude_coordinate_type</code></td>
<td>Lower valued starting latitude</td>
</tr>
<tr>
<td><code>stop latitude_coordinate_type</code></td>
<td>Higher valued ending latitude</td>
</tr>
</tbody>
</table>

### 7.4.17.30 `longitude_coordinate_type`

**Type:** Class  
**Package:** Coordinates and Positions  
Degrees east (positive), west (negative) relative to coordinate system datum. See diagram note on choice of SI units  
Element Tag: Range = -180 .. 180  
Element Tag: Resolution = 1 e-6  
Element Tag: Unit = deg

### 7.4.17.31 `longitude_interval_type`

#### 7.3.10.31

**Type:** Class  
**Package:** Coordinates and Positions

**A range of longitude values**

#### Table 7.4.1924 - Attributes of Class `longitude_interval_type`

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>start longitude_coordinate_type</code></td>
<td>The lowest longitude value at the beginning of the interval</td>
</tr>
<tr>
<td><code>stop longitude_coordinate_type</code></td>
<td>The highest longitude value at the end of the interval</td>
</tr>
</tbody>
</table>
7.4.17.32 polar_position_type

Type: IDLStruct

Package: Coordinates_and_Positions

Coordinates in a polar reference frame as described by a coordinate specification object

Table 7.2025 - Attributes of IDLStruct polar_position_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>azimuth_coordinate</td>
<td>The coordinate in the azimuth plane.</td>
</tr>
<tr>
<td>elevation_coordinate</td>
<td>elevation_coordinate_type [0..1] Optional as some sensors provide no elevation information</td>
</tr>
<tr>
<td>range_coordinate</td>
<td>range_coordinate_type [0..1] Optional as some sensor sensors provide no range information (e.g. most passive sensors)</td>
</tr>
<tr>
<td>origin</td>
<td>wgs84_position_type [0..1] Specifies the origin from which to interpret the polar position. This attribute is optional so that the originator of the data unambiguously specifies the origin can be implicitly specified according to the value of the applicable coordinate specification enumeration. (this is prioritizing accuracy and integration). AttributeTag: Issue =</td>
</tr>
</tbody>
</table>

7.4.17.33 polar_velocity_type

Type: IDLStruct

Package: Coordinates_and_Positions

Velocity defined in a polar reference frame as described by a coordinate specification object

Table 7.2126 - Attributes of IDLStruct polar_velocity_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>azimuth_rate</td>
<td>azimuth_rate_type The rate of change in azimuth corresponding to the velocity</td>
</tr>
<tr>
<td>elevation_rate</td>
<td>elevation_rate_type [0..1] The rate of change in elevation corresponding to the velocity. Optional as some sensors provide no elevation information</td>
</tr>
<tr>
<td>range_rate</td>
<td>range_rate_type [0..1] The rate of change in range corresponding to the velocity. Optional as some sensor sensors provide no range information (e.g. most passive sensors)</td>
</tr>
</tbody>
</table>

7.4.17.34 position_accuracy_coordinate_type

Type: Class

Package: Coordinates_and_Positions
To offer flexibility, three variants of coordinate system representation are supported - corresponding to the coordinate_kind_type enumerate. An implementation should support one kind for each relevant interface as defined by the coordinate_specification_type value, and it should only send data of that variant and it should check that all data received is of that variant. It should not implement conversion of data in an unexpected variant. Receipt of such data constitutes an error in the operation of the interface.

Table 7.227 - Attributes of Class position_accuracy_coordinate_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlCase» cartesian_position_accuracy</td>
<td>The Cartesian accuracy option.</td>
</tr>
<tr>
<td>cartesian_position_accuracy_type</td>
<td></td>
</tr>
<tr>
<td>«idlCase» polar_position_accuracy</td>
<td>The polar accuracy option.</td>
</tr>
<tr>
<td>polar_position_accuracy_type</td>
<td></td>
</tr>
<tr>
<td>«idlCase» wgs84_position_accuracy</td>
<td>The accuracy option using the WGS-84 spheroid.</td>
</tr>
<tr>
<td>wgs84_position_accuracy_type</td>
<td></td>
</tr>
</tbody>
</table>
To offer flexibility, three variants of coordinate system representation are supported - corresponding to the coordinate_kind_type enumerate. An implementation should support one kind for each relevant interface as defined by the coordinate_specification_type value, and it should only send data of that variant and it should check that all data received is of that variant. It should not implement conversion of data in an unexpected variant. Receipt of such data constitutes an error in the operation of the interface.

ElementTag: case type = coordinate_kind_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>@idlCase cartesian_position</code> cartesian_position_type</td>
<td>The Cartesian coordinate position option &lt;br&gt;AttributeTag: case value = CARTESIAN</td>
</tr>
<tr>
<td><code>@idlCase polar_position</code> polar_position_type</td>
<td>The polar coordinates position option &lt;br&gt;AttributeTag: case value = POLAR</td>
</tr>
<tr>
<td><code>@idlCase wgs84_position</code> wgs84_position_type</td>
<td>The position option using the WGS-84 spheroid &lt;br&gt;AttributeTag: case value = WGS84</td>
</tr>
</tbody>
</table>

Axis in range, i.e. linear distance from the coordinate system datum. Metres. See diagram note on choice of SI units.

ElementTag: Range = 0 .. 1 e7 <br>ElementTag: Resolution = 1 e-3 <br>ElementTag: Unit = m

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>start range_coordinate_type</code></td>
<td></td>
</tr>
<tr>
<td><code>stop range_coordinate_type</code></td>
<td></td>
</tr>
</tbody>
</table>

The position option using the WGS-84 spheroid.
A continuous set of range values.

### Table 7.29 - Attributes of Class range_interval_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>start range_coordinate_type</td>
<td>The nearest value at the beginning of the interval</td>
</tr>
<tr>
<td>stop range_coordinate_type</td>
<td>The furthest value at the end of the interval</td>
</tr>
</tbody>
</table>

#### 7.4.17.38 range_qualification_type

**Type:** IDLStruct  
**Package:** Coordinates_and_Positions  
Qualifies a measurement of range with attributes of accuracy and, if possible, variability.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>spread range_coordinate_type [0..1]</td>
<td>The spread of the measurement. The combined measures of spread should encompass the full extent of the plot. This attribute is optional. Not all sensors are capable of measuring it.</td>
</tr>
<tr>
<td>accuracy range_coordinate_type</td>
<td>The accuracy of the measurement; equal to one standard deviation of uncertainty.</td>
</tr>
</tbody>
</table>

#### 7.4.17.39 range_rate_type

**Type:** Class  
**Package:** Coordinates_and_Positions  
metres per second  
ElementTag: Range = 0.0 .. 1e5  
ElementTag: Resolution = 0.01  
ElementTag: Unit = m/s

#### 7.4.17.40 speed_interval_type

**Type:** Class  
**Package:** Coordinates_and_Positions  
This class represents a range of speeds.
Table 7.31 - Attributes of Class speed_interval_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>spread range_coordinate</td>
<td>The spread of the measurement. The combined measures of spread should encompass the full extent of the plot. This attribute is optional. Not all sensors are capable of measuring it. The minimum speed.</td>
</tr>
<tr>
<td>min speed_type</td>
<td>The minimum speed.</td>
</tr>
<tr>
<td>max speed_type</td>
<td>The maximum speed.</td>
</tr>
<tr>
<td>accuracy range_coordinate</td>
<td>The accuracy of the measurement; equal to one standard deviation of uncertainty. The maximum speed.</td>
</tr>
<tr>
<td>min speed_type</td>
<td>The minimum speed.</td>
</tr>
<tr>
<td>max speed_type</td>
<td>The maximum speed.</td>
</tr>
</tbody>
</table>

7.3.10.39 range_rate_type
Type: Class double
Package: Coordinates_and_Positions metres per second
ElementTag: Range = 0.0 .. 1 e5 ElementTag: Resolution = 0.01 ElementTag: Unit = m/s

7.3.10.40 speed_interval_type
Type: Class
Package: Coordinates_and_Positions This class represents a range of speeds.

Table 7.25 - Attributes of Class speed_interval_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>min speed_type</td>
<td>The minimum speed.</td>
</tr>
<tr>
<td>max speed_type</td>
<td>The maximum speed.</td>
</tr>
</tbody>
</table>

7.3.10.41 speed_type
Type: Class double
Package: Coordinates_and_Positions metres per second
ElementTag: Range = 0.0 .. 1 e5 ElementTag: Resolution = 0.01 ElementTag: Unit = m/s

7.4.17.42 velocity_accuracy_coordinate_type
Type: Class
Package: Coordinates_and_Positions
metres per second
ElementTag: Range = 0.0 .. 1 e5
ElementTag: Resolution = 0.01
ElementTag: Unit = m/s

7.4.17.42 velocity_accuracy_coordinate_type
Type: Class
Package: Coordinates_and_Positions
To offer flexibility, three variants of coordinate system representation are supported - corresponding to the coordinate_kind_type enumerate. An implementation should support one kind for each relevant interface as defined by the coordinate_specification_type value, and it should only send data of that variant and it should
check that all data received is of that variant. It should not implement conversion of data in an unexpected variant. Receipt of such data constitutes an error in the operation of the interface.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlCase» cartesian_velocity_accuracy cartesian_velocity_accuracy_type</td>
<td>The Cartesian velocity accuracy option.</td>
</tr>
<tr>
<td>«idlCase» polar_velocity_accuracy polar_velocity_accuracy_type</td>
<td>The polar velocity accuracy option.</td>
</tr>
<tr>
<td>«idlCase» wgs84_velocity_accuracy wgs84_velocity_accuracy_type</td>
<td>The velocity accuracy option using the WGS-84 spheroid.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlCase» wgs84_velocity_accuracy wgs84_velocity_accuracy_type</td>
<td></td>
</tr>
</tbody>
</table>

**Table 7.2632 - Attributes of Class velocity_accuracy_coordinate_type**

**Table 7.2733 - Attributes of IDLUnion velocity_coordinate_type**

**7.4.17.44 velocity_coordinate_type**

**7.3.10.43 Type:** _____ IDLUnion

**Package:** Coordinates_and_Positions

To offer flexibility, three variants of coordinate system representation are supported - corresponding to the coordinate_kind_type enumerate. An implementation should support one kind for each relevant service as defined by the coordinate_specification_type value, and it should only send data of that variant and it should check that all data received is of that variant. It should not implement conversion of data in an unexpected variant. Receipt of such data constitutes an error in the operation of the interface. Three representations are supported: time derivatives within a Cartesian coordinate system; time derivatives of a polar coordinate system (range rate, bearing rate etc.); course and speed relative to the WGS84 spheroid.

**ElementTag:** case type = coordinate_kind_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlCase» cartesian_velocity cartesian_velocity_type</td>
<td>The Cartesian velocity option. AttributeTag: case value = CARTESIAN</td>
</tr>
<tr>
<td>«idlCase» polar_velocity polar_velocity_type</td>
<td>The polar velocity option. AttributeTag: case value = POLAR</td>
</tr>
<tr>
<td>«idlCase» wgs84_velocity wgs84_velocity_type</td>
<td>The option of velocity specified with reference to the WGS-84 spheroid. AttributeTag: case value = WGS84</td>
</tr>
</tbody>
</table>

**7.4.17.44 wgs84_position_type**

**7.3.10.44 Type:** _____ Class
Table 7.2834 - Attributes of Class wgs84_position_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>height_coordinate</td>
<td>height_coordinate_type [0..1] Optional as some sensors as 2D (work in horizontal plane) and some other functions do not supply or require this information either.</td>
</tr>
<tr>
<td>latitude_coordinate</td>
<td>latitude_coordinate_type The latitude of the position on the WGS-84 spheroid.</td>
</tr>
<tr>
<td>longitude_coordinate</td>
<td>longitude_coordinate_type The longitude of the position on the WGS-84 spheroid.</td>
</tr>
</tbody>
</table>

7.4.17.45 wgs84_velocity_type

7.3.10.45 Type: IDLStruct
Package: Coordinates_and_Positions
Velocity defined in the WGS84 grid system from the viewpoint of the object in terms of course and speed with optional angle of climb for changes in height.

Table 7.2935 - Attributes of IDLStruct wgs84_velocity_type
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>course course_type</td>
<td>Relative to North in the WGS84 spheroid.</td>
</tr>
<tr>
<td>angle_of_climb angle_of_climb_type [0..1]</td>
<td>Optional as some sensors as 2D (work in horizontal plane) and some other functions do not supply or require this information either.</td>
</tr>
<tr>
<td>speed speed_type</td>
<td>The total speed within the WGS84 spheroid (not speed over ground) in the direction of travel including angle of climb when present.</td>
</tr>
</tbody>
</table>

Table 7. Methods of IDLStruct wgs84_velocity_type

7.4.17.46 cartesian_position_accuracy_type

<table>
<thead>
<tr>
<th>Type:</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package:</td>
<td>Coordinates_and_Positions</td>
</tr>
</tbody>
</table>

The accuracy of the components of Cartesian position

Table 7. Attributes of Class cartesian_position_accuracy_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_coordinate_accuracy cartesian_coordinate_type</td>
<td></td>
</tr>
<tr>
<td>y_coordinate_accuracy cartesian_coordinate_type</td>
<td></td>
</tr>
<tr>
<td>z_coordinate_accuracy cartesian_coordinate_type [0..1]</td>
<td>Optional as some sensors are 2D (horizontal plane or no elevation information)</td>
</tr>
</tbody>
</table>

Table 7. Methods of Class cartesian_position_accuracy_type

7.3.10.46 cartesian_velocity_accuracy_type

<table>
<thead>
<tr>
<th>Type:</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package:</td>
<td>Coordinates_and_Positions</td>
</tr>
</tbody>
</table>

The accuracy of the components of Cartesian velocity

Table 7. Attributes of Class cartesian_velocity_accuracy_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_dotcoordinate_accuracy cartesian_velocity_component_coordinate_type</td>
<td>The accuracy of the x-axis coordinate</td>
</tr>
<tr>
<td>y_dotcoordinate_accuracy cartesian_velocity_component_coordinate_type</td>
<td>The accuracy of the y-axis coordinate</td>
</tr>
<tr>
<td>z_dotcoordinate_accuracy cartesian_velocity_component_coordinate_type [0..1]</td>
<td>The accuracy of the z-axis coordinate. Optional as some sensors are 2D (horizontal plane or no elevation information)</td>
</tr>
</tbody>
</table>
Table 7.4 - Methods of Class cartesian_velocity_accuracy_type

7.3.10.48 polar_position_accuracy_type
Type: Class
Package: Coordinates_and_Positions
The accuracy of the components of polar_position Cartesian velocity

Table 7.37 - Attributes of Class polar_position cartesian_velocity_accuracy_type
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>azimuth_accuracy azimuth_coordinate_type</strong></td>
<td>Optional as some sensors provide no azimuth information. The accuracy of the azimuth coordinate.</td>
</tr>
<tr>
<td><strong>elevation_x_dot_accuracy</strong></td>
<td>Accuracy of the x_dot velocity attribute. Optional as some sensors provide no elevation information.</td>
</tr>
<tr>
<td><strong>range_y_dot_accuracy</strong></td>
<td>Accuracy of the y_dot velocity attribute. Optional as some sensors provide no range information (e.g. most passive sensors).</td>
</tr>
<tr>
<td><strong>origin_wgs84_position_z_dot_accuracy cartesian_velocity_component_type [0..1]</strong></td>
<td>Accuracy of the z_dot velocity attribute. Optional as some sensors are 2D (horizontal plane or no elevation information). Specifies the accuracy of the origin from which to interpret the polar position. The attribute is optional as the origin can be implicitly specified according to the value of the applicable coordinate specification enumeration.</td>
</tr>
</tbody>
</table>

### Table 7.3038 - Attributes of Class polar_velocityposition_accuracy_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>azimuth_rate_accuracy azimuth_rate_coordinate_type</strong></td>
<td>The accuracy of the azimuth coordinate.</td>
</tr>
<tr>
<td><strong>elevation_rate_accuracy</strong></td>
<td>Optional as some sensors provide no elevation information.</td>
</tr>
<tr>
<td><strong>range_rate_accuracy range_rate_coordinate_type [0..1]</strong></td>
<td>Optional as some sensors provide no range information (e.g. most passive sensors).</td>
</tr>
<tr>
<td><strong>origin_wgs84_position_accuracy_type [0..1]</strong></td>
<td>Specifies the accuracy of the origin from which to interpret the polar position. This attribute is optional as the origin can be implicitly specified according to the value of the applicable coordinate specification enumeration.</td>
</tr>
</tbody>
</table>

### Table 7.39 - Attributes of Class polar_velocity_accuracy_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>azimuth_rate_accuracy azimuth_rate_type</strong></td>
<td>The accuracy of the azimuth rate.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>elevation_rate_accuracy</td>
<td>elevation_rate_type [0..1] The accuracy of the elevation rate. Optional as some sensors provide no elevation information</td>
</tr>
<tr>
<td>range_rate_accuracy</td>
<td>range_rate_type [0..1] The accuracy of the range rate. Optional as some sensors provide no range information (e.g. most passive sensors)</td>
</tr>
</tbody>
</table>

**7.3.10.50 wgs84_position_accuracy_type**  
_Type: Class_  
.Package: Coordinates_and_Positions  
The accuracy of the components of a WGS84 position

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>height_accuracy</td>
<td>height_coordinate_type [0..1] The accuracy of the height coordinate. Optional as some sensors as 2D (work in horizontal plane) and some other functions do not supply or require this information either.</td>
</tr>
<tr>
<td>latitude_accuracy</td>
<td>latitude_coordinate_type The accuracy of the latitude coordinate.</td>
</tr>
<tr>
<td>longitude_accuracy</td>
<td>longitude_coordinate_type The accuracy of the longitude coordinate.</td>
</tr>
</tbody>
</table>

**7.4.17.51 wgs84_velocity_accuracy_type**  
_Type: Class_  
.Package: Coordinates_and_Positions  
The accuracy of the components of a WGS84 velocity

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td><em>course_accuracy</em></td>
<td>The accuracy of the course attribute of the velocity</td>
</tr>
<tr>
<td><em>angle_of_climb_accuracy</em></td>
<td>The accuracy of the angle of climb attribute of the velocity. Optional as some sensors as 2D (work in horizontal plane) and some other functions do not supply or require this information either.</td>
</tr>
<tr>
<td><em>speed_accuracy</em></td>
<td>The accuracy of the speed attribute of the velocity</td>
</tr>
</tbody>
</table>

### 7.3.4.17.4.18 Shape_Model

**Parent Package:** Common_Types

```
figure_ref_point_type
+ ref_point
  + position: position_coordinate_type
  + length: range_coordinate_type
  + width: range_coordinate_type
  + orientation: azimuth_coordinate_type

rectangle_type
+ point: position_coordinate_type [3..*]

polygon_type

area_2d_type
```

```
sector_type
+ sector: sector_type
+ rectangle: rectangle_type
+ polygon: polygon_type
+ truncated_sector: truncated_sector_type
```

```
polygon_polygon_type
```
Figure 7.29.53 Domain Model - non polar (Class diagram)
«idlStruct»
truncated_polar_volume_type
+ centre_bearing: azimuth_coordinate_type
+ delta_bearing: azimuth_coordinate_type
+ centre_elevation: elevation_coordinate_type
+ delta_elevation: elevation_coordinate_type
+ inner_range: range_coordinate_type
+ outer_range: range_coordinate_type

«idlStruct»
polar_volume_type
+ centre_bearing: azimuth_coordinate_type
+ delta_bearing: azimuth_coordinate_type
+ centre_elevation: elevation_coordinate_type
+ delta_elevation: elevation_coordinate_type
+ origin: 0..1

«idlStruct»
figure_ref_point_type
+ position: position_coordinate_type

«idlCase»
sector_type
+ position: position_coordinate_type

«idlUnion»
general_polar_volume_type
+ centre_bearing: azimuth_coordinate_type
+ delta_bearing: azimuth_coordinate_type

«idlStruct»
truncated_sector_type
+ centre_bearing: azimuth_coordinate_type
+ delta_bearing: azimuth_coordinate_type
+ inner_range: range_coordinate_type
+ outer_range: range_coordinate_type
Figure 7.30.54 Domain Model - polar (Class diagram)
Z.3.11.17.4.18.1 area_2d_type
Type: IDLUnion
Package: Shape_Model
An area for the sensor to keep under surveillance

ElementTag: **Issue** = \_switchType = long

### Table 7.33 - Attributes of IDLUnion area\_2d\_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlCase» sector sector_type</td>
<td>The sector option for a 2d area</td>
</tr>
<tr>
<td>«idlCase» rectangle rectangle_type</td>
<td>The rectangle option for a 2d area</td>
</tr>
<tr>
<td>«idlCase» polygon polygon_type</td>
<td>The polygon option for a 2d area</td>
</tr>
<tr>
<td>«idlCase» truncated_sector truncated_sector_type</td>
<td>The truncated sector option for a 2d area</td>
</tr>
</tbody>
</table>

### 7.3.11.27.4.18.2 figure\_ref\_point\_type

Type: Class

Package: Shape_Model

A figure\_ref\_point specifies a reference point for a figure.

This reference point is a mathematically meaningful point of the figure. For a circle it is the centre of the circle, for a polygon it is the centre of gravity of the polygon, etc.

When rotating the figure, the figure\_ref\_point acts as the rotation point.

When a figure is not slaved to a track its figure\_ref\_point shall be mapped on a (moving) geo point.

When the figure is slaved to an object (track, point) its figure\_ref\_point shall be mapped on an offset position which is relative to the master object.

### Table 7.34 - Attributes of Class figure\_ref\_point\_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>position position_coordinate_type</td>
<td>The position of the reference point.</td>
</tr>
</tbody>
</table>

### 7.4.18.3 general\_polar\_volume\_type

Type: IDLUnion

Package: Shape_Model

This class allow definition of a volume in space, bounded by standard polar coordinates (azimuth, elevation and range). The different options allow the dimension of either range, elevation or both to be omitted.

### Table 7.35 - Attributes of IDLUnion general\_polar\_volume\_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlCase» sector sector_type</td>
<td>The general polar volume is a sector</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>«idlCase» polar_volume polar_volume_type</td>
<td>The general polar volume is a polar volume</td>
</tr>
<tr>
<td>«idlCase» truncated_sector truncated_sector_type</td>
<td>The general polar volume is a truncated sector</td>
</tr>
<tr>
<td>«idlCase» truncated_polar_volume truncated_polar_volume_type</td>
<td>The general polar volume is a truncated polar volume.</td>
</tr>
</tbody>
</table>
7.3.11.4 polar_volume_type
Type: Class
Package: Shape_Model
A polar_volume specifies a 3D volume based on a horizontal plane by means of its origin, its centre bearing and centre elevation, its bearing delta and elevation delta.
The origin is the figure reference point of the Polar Volume.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>.centre_bearing azimuth_coordinate_type</td>
<td>This attribute specifies the horizontal angle measured clockwise between the Y-axis of the relevant coordinate system (true north, heading/course) and the centre bearing line of the volume.</td>
</tr>
<tr>
<td>.delta_bearing azimuth_coordinate_type</td>
<td>This attribute specifies the bearing delta on each side of a specified centre bearing line.</td>
</tr>
<tr>
<td>.centre_elevation elevation_coordinate_type</td>
<td>This attribute specifies the vertical angle measured counterclockwise between the horizontal plane and the centre elevation line of the volume.</td>
</tr>
<tr>
<td>.delta_elevation elevation_coordinate_type</td>
<td>This attribute specifies the elevation delta on each side of a specified centre elevation line.</td>
</tr>
</tbody>
</table>

7.3.11.5 polygon_type
Type: IDLStruct
Package: Shape_Model
A geographically defined general area

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
</table>
| .point position_coordinate_type [3..*] | The set of points for the polygon; there must be at least 3.
AttributeTag: Length = 12 |

7.3.11.6 rectangle_type
Type: IDLStruct
Package: Shape_Model
A geographically defined rectangle in the environment

Table 7.4#Table#47 - Attributes of IDLStruct rectangle_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>length range_coordinate_type</td>
<td>distance along of orientation from the ref point to the next corner</td>
</tr>
<tr>
<td>width range_coordinate_type</td>
<td>distance perpendicular to angle of orientation (clockwise) from ref point to the next corner</td>
</tr>
<tr>
<td>orientation azimuth_coordinate_type</td>
<td>angle of azimuth of the length sides of the rectangle</td>
</tr>
</tbody>
</table>

7.3.11.77.4.18.7 sector_type

Type: Class
Package: Shape_Model
A sector specifies a 2D area in a horizontal plane by means of its origin, its centre bearing with its bearing delta, that together define the sector.
The origin is the figure reference point of the sector.
In case the sector is north oriented, the centre bearing is specified with respect to true north; otherwise it is specified with respect to the object's (own ship/other track, point) heading/course.

Table 7.3748 - Attributes of Class sector_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>centre_bearing azimuth_coordinate_type</td>
<td>This attribute specifies the horizontal angle measured clockwise between the Y-axis of the relevant coordinate system (true north, heading/course) and the centre bearing line of the sector.</td>
</tr>
<tr>
<td>delta_bearing azimuth_coordinate_type</td>
<td>This attribute specifies the bearing delta on each side of a specified centre bearing line.</td>
</tr>
</tbody>
</table>

7.4.18.8 truncated_polar_volume_type

7.3.11.8 Type: Class
Package: Shape_Model
A truncated_polar_volume specifies a 3D volume based on a horizontal plane by means of its origin, its centre bearing and centre elevation, its bearing delta and elevation delta, its inner range and outer range

Table 7.3849 - Attributes of Class truncated_polar_volume_type
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
</table>
| `centre_bearing`      | azimuth_coordinate_type  
This attribute specifies the horizontal angle measured clockwise between the Y-axis of the relevant coordinate system (true north, heading/course) and the centre bearing line of the volume. |
| `delta_bearing`       | azimuth_coordinate_type  
This attribute specifies the bearing delta on each side of a specified centre bearing line.                                             |
| `centre_elevation`    | elevation_coordinate_type  
This attribute specifies the vertical angle measured counterclockwise between the horizontal plane and the centre elevation line of the volume. |
| `delta_elevation`     | elevation_coordinate_type  
This attribute specifies the elevation delta on each side of a specified centre elevation line.                                         |
| `inner_range`         | range_coordinate_type     
This attribute specifies the range that limits a volume; i.e. the minimum distance from the volume's origin.                           |
| `outer_range`         | range_coordinate_type     
This attribute specifies the range that limits a volume; i.e. the maximum distance from the volume's origin.                           |
### 7.3.1.11.97.4.18.9 truncated_sector_type

**Type:** Class

**Package:** Shape_Model

A truncated_sector specifies a 2D area in a horizontal plane by means of its origin, its centre bearing with its bearing delta, and its inner range and outer range, that together define the truncated sector.

The origin is the figure reference point of the truncated sector.

In case the truncated sector is north oriented, the centre bearing is specified with respect to true north; otherwise (object oriented) it is specified with respect to the object's (own ship/other track, point) heading/course.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>centre_bearing</td>
<td>This attribute specifies the horizontal angle measured clockwise between the Y-axis of the relevant coordinate system (true north, heading/course) and the centre bearing line of the truncated sector.</td>
</tr>
<tr>
<td>delta_bearing</td>
<td>This attribute specifies the bearing delta on each side of a centre bearing line.</td>
</tr>
<tr>
<td>inner_range</td>
<td>This attribute specifies the range that limits a truncated sector; i.e. the minimum distance from the truncated sector's origin.</td>
</tr>
<tr>
<td>outer_range</td>
<td>This attribute specifies the range that limits a truncated sector; i.e. the maximum distance from the truncated sector's origin.</td>
</tr>
</tbody>
</table>

### 7.3.4.27.4.19 Requests

**Parent Package:** Common_Types

This package contains common operations and associated parameters which are used by multiple interfaces. This includes the operation to acknowledge a CMS request as accepted or denied, as well as an operation to report errors while processing an accepted CMS request.
receive_acknowledgement(request_id_type, request_ack_type): void
receive_error(request_id_type, error_reason_type): void

«idlInterface»
common_use_case_interface

unsigned long long request_id_type

«idlTypedef»

string error_reason_type

«idlStruct»
request_ack_type
+ accepted: boolean
+ rejection: 0..1

«idlStruct»
denial_type
+ reason: denial_reason_type
+ related_parameter: parameter_reference_type[0..*]

«idlTypedef»
denial_reason_type
Length = 64

string parameter_reference_type
Length = 40
### 7.3.12.4.19.1 denial_reason_type

**Type:** Class **string**  
**Package:** Requests  

String which indicates rationale for rejection of the request. Is not valid when the request has been accepted.  

*ElementTag: Length = 40*

### 7.3.12.4.19.2 denial_type

**Type:** Class **string**  
**Package:** Requests  

Struct used within the receive_acknowledgement operation to provide information on (one of the reasons) why a request has been rejected.

Table 7.4051 - Attributes of Class denial_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>reason denial_reason_type</td>
<td>Textual explanation of (one of) the reasons for rejection</td>
</tr>
<tr>
<td>related_parameter parameter_reference_type [0..*]</td>
<td>A reference to the parameter or parameters that relate to the reason for rejection. If no related parameters are supplied the rejection relates to the whole request.</td>
</tr>
</tbody>
</table>
7.2.12.37.4.19.3 error_reason_type
Type: Class string
Package: Requests
A string which gives an indication of the error associated with processing of the request.
7.4.19.4 parameter_reference_type

Type: IDLTypeDef string
Package: Requests

A string which refers to a parameter in a request using an implementation specific notation.

ElementTag: Length = 40
ElementTag: Length = 64

7.3.12.5 request_ack_type

Type: Class
Package: Requests

Struct used within the receive_acknowledgement operation to indicate acceptance or rejection (which includes rationale).

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>accepted</td>
<td>boolean</td>
</tr>
<tr>
<td></td>
<td>Attribute to indicate whether a request has been accepted (1) or rejected (0).</td>
</tr>
</tbody>
</table>

7.4.19.6 request_id_type

Type: IDLTypeDef unsigned long-long
Package: Requests

The purpose of the request_id is to uniquely relate responses of the subsystem (server) to requests of the CMS (client). The request_id is set by the client. It is the responsibility of the client to specify a system-wide unique request_id (e.g. based on a combination of client id and a sequence number / time of request).

7.4.19.7 common_use_case_interface

Type: Interface
Package: Requests

Interface which includes operations common to all CMS interfaces.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_acknowledgement()</td>
<td>This operation is used by the subsystem to indicate whether it has accepted or rejected a request from the CMS. MethodTag: ea_guid = {C15FF90A-E3EE-4c87-AFD6-2234A76786B2}</td>
<td>request_id_type request_id request_ack_type request_ack</td>
</tr>
<tr>
<td>receive_error()</td>
<td>This operation is used by the subsystem to indicate an error in processing a request.</td>
<td>request_id_type request_id error_reason_type error_reason</td>
</tr>
</tbody>
</table>
7.47.5 Subsystem_Domain
**Parent Package:** Domain_Model
This package contains the Domain Models for the Encyclopaedic Support, Extended Subsystem Control, Subsystem Control, Recording and Replay, and Simulation Support services.

### 7.4.47.5.1 Encyclopaedic_Support

**Parent Package:** Subsystem_Domain

Domain classes for Encyclopaedic Support

![Domain Model Class Diagram]

**Figure 7.32.56 Domain Model (Class diagram)**
7.5.1.1 data_descriptor_type

7.4.1.1 Type: Class string
Package: Encyclopaedic_Support
Standard description of the encyclopaedic data set
ElementTag: Length = 60

7.4.1.2 7.5.1.2 url_type

Type: Class string
Package: Encyclopaedic_Support
Representation of a Uniform Resource Locator see www.w3.org www.w3.org
ElementTag: Length = 255

7.4.2 7.5.2 Extended_Subsystem_Control

Parent Package: Subsystem_Domain
Contains Structs used within the Extended Subsystem Control service.
7.5.2.1  configuration_url_type

7.4.2.1  Type:  IDLTypeDef string

Package:  Extended_Subsystem_Control

String which provides a url location for configuration data.
7.5.2.2 network name type
Type: IDLTypeDef
Package: Extended_Subsystem_Control
The name identifying an external network.

7.4.2.3 offline test result details type
Type: IDLTypeDef string
Package: Extended_Subsystem_Control
Subsystem specific detailed test results
ElementTag: Length = 4096

7.5.2.4 offline test result type
7.4.2.3 Type: Class
Package: Extended_Subsystem_Control
Used to return the test results: failed, partial_pass or passed

Table 7.4354 - Attributes of Class offline test result type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>FAILED</td>
<td>A number of tests were not successful, such that the subsystem exceeded its failure threshold. Detailed information is available upon request.</td>
</tr>
<tr>
<td>PARTIAL_PASS</td>
<td>A number of tests were not successful, but the subsystem did not exceed its failure threshold. Detailed information is available upon request.</td>
</tr>
<tr>
<td>PASSED</td>
<td>All tests were successful.</td>
</tr>
</tbody>
</table>
7.4.2.47.5.2.5 offline_test_type

Type: IDLTypeDef string

Package: Extended_Subsystem_Control

A subsystem specific string identifying the required test type.
ElementTag: Length = 255

7.4.37.5.3 Recording_and_Replay

Parent Package: Subsystem_Domain

Defines the domain model for the Recording and Replay interfaces.

This contains the classes associated with Recording and Replay.
7.5.3.1 **actual_time_type**  

7.4.3.1 Type: Class time_type  
Package: Recording_and_Replay  
The current time (time of day). Used to indicate when playback should start. This allows synchronisation of playback from different subsystems.
of playback from different subsystems.

7.5.3.2 change_threshold_type

7.4.3.2 Type: IDLTypeDef float
Package: Recording_and_Replay
The amount by which a parameter shall change in order to be recorded, when recording on change

7.5.3.3 parameter_type

7.4.3.3 Type: Class
Package: Recording_and_Replay
Identified the parameter to be recorded

Table 7.4455 - Attributes of Class parameter_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameter</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>The parameter value.</td>
</tr>
<tr>
<td></td>
<td>AttributeTag: StringLength = 32</td>
</tr>
</tbody>
</table>

7.4.3.4 rate_type

7.5.3.4 Type: IDLTypeDef float
Package: Recording_and_Replay
Defined the rate at which the parameter is to be recorded for periodic recording

7.5.3.5 record_on_change_type

7.4.3.5 Type: IDLTypeDef boolean
Package: Recording_and_Replay
Boolean specifying record on change (true) or periodic (false)

7.5.3.6 recorded_data_type

7.4.3.6 Type: Class
Package: Recording_and_Replay
Data recorded against the specified parameter

Table 7.4556 - Attributes of Class recorded_data_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>recorded_value</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>This needs to reference allowable values defined by the possible recording parameters - see 'recording parameters'. AttributeTag: StringLength = 20</td>
</tr>
</tbody>
</table>
### Attribute

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_stamp</td>
<td>The absolute time at which the value was recorded</td>
</tr>
</tbody>
</table>

**Table 7.57 - Relations of Class recorded_data_type**

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation:</td>
<td>parameter</td>
</tr>
<tr>
<td>parameter</td>
<td>parameter_type [1]</td>
</tr>
</tbody>
</table>

#### 7.5.3.7 recorded_time_type

**7.4.3.7 Type:** Class time_type  
**Package:** Recording_and_Replay  
The time in a recording. This is used to indicate the position in the recording at which playback should start.

#### 7.5.3.8 recording_descriptor_type

**7.4.3.8 Type:** Class  
**Package:** Recording_and_Replay  
Specifies the recording characteristics required for each parameter
### Table 7.4658 - Attributes of Class recording_descriptor_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>change_threshold</td>
<td>When record_on_change is true, any change greater than the change_threshold from the last recorded value shall be recorded. This only applies for numeric quantities i.e. not enumerated types, and is ignored otherwise.</td>
</tr>
<tr>
<td>rate</td>
<td>Specifies recording rate when record_on_change is false. AttributeTag: Unit = Hz</td>
</tr>
<tr>
<td>record_on_change</td>
<td>Indicates whether to record all changes greater than the change threshold or record at the specified rate.</td>
</tr>
</tbody>
</table>

### Table 7.59 - Relations of Class recording_descriptor_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: parameter parameter_type [1]</td>
<td></td>
</tr>
</tbody>
</table>

#### 7.5.3.9 recording_id_type

**Type:** Class long  
**Package:** Recording_and_Replay  
Used to identify a specific recording. The subsystem shall manage a number of recordings and associate recording ids with them in a subsystem dependent way. Once associated, it passes that reference through the parameter recording_id to the CMS so that the CMS may ask for a specific recording later on. Again, the CMS manages the relationship between the recording_id and the recording it requested to be made in a system dependent way.

There is no intention to model the method either the subsystem or the CMS uses to manage the relationship between recording_id and the recordings as this is transparent to the interface and would unnecessarily restrict the choices available to the designers.

#### 7.5.3.10 recording_set_type

**Type:** Class  
**Package:** Recording_and_Replay  
A set of recording descriptors specifying what is to be recorded

#### 7.5.3.11 recording_type

**Type:** Class  
**Package:** Recording_and_Replay  
A recording: a set of recorded data
Table 7.61 - Relations of Class recording_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation:</td>
<td></td>
</tr>
<tr>
<td>recorded_data</td>
<td></td>
</tr>
<tr>
<td>recorded_data_type</td>
<td>[1..*]</td>
</tr>
<tr>
<td>Association:</td>
<td></td>
</tr>
<tr>
<td>recording_id</td>
<td></td>
</tr>
<tr>
<td>recording_id_type</td>
<td>reference</td>
</tr>
<tr>
<td>[1]</td>
<td></td>
</tr>
</tbody>
</table>

7.4.3.12 7.5.3.12 replay_set_type

Type: Class
Package: Recording_and_Replay
A set of parameters required to be replayed. These must exist in the associated recording set to be of any use.

Table 7.62 - Relations of Class replay_set_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation:</td>
<td></td>
</tr>
<tr>
<td>parameter</td>
<td></td>
</tr>
<tr>
<td>parameter_type</td>
<td>[1..*]</td>
</tr>
</tbody>
</table>

7.4.3.13 7.5.3.13 replay_speed_type

Type: Class float
Package: Recording_and_Replay
Controls the replay speed. 1.0 represents real time.

7.4.4 7.5.4 Simulation_Support

Parent Package: Subsystem_Domain
Figure 7.35.59 Domain Model (Class diagram)
7.5.4.1 fault_script_id_type

Type: Class string
Package: Simulation_Support
Identifies a single fault script.
ElementTag: Length = 6

7.5.4.2 fault_script_ids_type

Type: Class
Package: Simulation_Support
This class represents a set of references to fault scripts

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: <code>script_id</code></td>
<td>fault_script_id_type</td>
</tr>
</tbody>
</table>

7.5.4.3 fault_script_type

Type: Class
Package: Simulation_Support
Definition of a fault script. The exact form of this is not yet defined, this class represents the essential attributes. It would probably be some form of string, perhaps an XML document.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>details_of_fault</code> string</td>
<td>A description of the fault, such as is interpretable during the simulation AttributeTag: <code>StringLength</code> = 200</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: <code>script_id</code></td>
<td>fault_script_id_type</td>
</tr>
</tbody>
</table>

7.5.4.4 fault_scripts_type

Type: Class
Package: Simulation_Support
This class represents a set of fault scripts

**7.4.4.5 sim_mode_status_type**

*Type:* __Class__  
*Package:* __Simulation_Support__

Whether simulated mode is in operation

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>sim_mode_active</td>
<td>boolean</td>
</tr>
<tr>
<td>Aggregation: script</td>
<td></td>
</tr>
<tr>
<td>fault_script_type [0..*]</td>
<td></td>
</tr>
</tbody>
</table>

**7.5.4.5 sim_mode_requeststatus_type**

*Type:* __Class__  
*Package:* __Simulation_Support__

A request to change the simulation mode

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>start_simulation sim_mode_active</td>
<td>boolean</td>
</tr>
<tr>
<td>Flag to indicate if the simulation mode shall be started or stopped is active.</td>
<td></td>
</tr>
</tbody>
</table>

**7.5.4.6 start_stop_sim_mode_request_type**

*Type:* __Class__  
*Package:* __Simulation_Support__

A request to change the simulation mode

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>start_simulation_mode</td>
<td>boolean</td>
</tr>
<tr>
<td>Flag to indicate if the simulation mode shall be started or stopped.</td>
<td></td>
</tr>
</tbody>
</table>

**7.4.4.7 stop_freeze_session_request_type**

*Type:* __Class__  
*Package:* __Simulation_Support__

A Simulation Management (SIMAN) request, sent from a Simulation Manager to request that one or more entities either  

a) pause their simulation session  
b) stop their simulation session.
### Table 7.5069 - Attributes of Class stop_freeze_session_request_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>reflect_values</em> boolean</td>
<td>Whether the entity or entities being stopped/frozen should continue to reflect values when stopped/frozen.</td>
</tr>
<tr>
<td><em>run_internal_simulation_clock</em> boolean</td>
<td>Whether the entity or entities being stopped/frozen should continue to run their internal simulation clock when stopped/frozen.</td>
</tr>
<tr>
<td><em>update_attributes</em> boolean</td>
<td>Whether the entity or entities being stopped/frozen should continue to update attributes when stopped/frozen.</td>
</tr>
</tbody>
</table>

### 7.4.67.5.5 Subsystem_Control

**Parent Package:** Subsystem_Domain  
Contains Structs used within the Subsystem Control service and a state diagram corresponding with the Manage Technical State interface.
Figure 7.36.60 Domain Model - 1 (Class diagram)
Figure 7.37.61 Domain Model - 2 (Class diagram)
7.5.5.1 equipment_category_type
Type: Class
Package: Subsystem_Control
Categorization of equipment. Values correspond to items in an externally defined list.

7.4.5.1 7.5.5.2 function_id_type
Type: Class unsigned short
Package: Subsystem_Control
Unique identifier for a function within a subsystem.

7.5.5.3 function_type
7.4.5.2 Type: Class
Package: Subsystem_Control
One of the functions of a subsystem

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«key» function_id</td>
<td>The functions unique identifier</td>
</tr>
<tr>
<td>function_name</td>
<td>The name of function as understood by an operator</td>
</tr>
<tr>
<td></td>
<td>AttributeTag: StringLength = 32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association</td>
<td>The operational modes in which the function is available</td>
</tr>
<tr>
<td>applicable_mode</td>
<td>reference [0..*]</td>
</tr>
<tr>
<td>operational_mode_type</td>
<td></td>
</tr>
</tbody>
</table>

7.5.5.4 platform_category_type
Type: Class
Package: Subsystem_Control
Categorization of platforms (i.e. structures such as ships and planes) that host CMS, sensors and other subsystems. Values correspond to items in an externally defined list.

7.5.5.5 product_category_type
Type: Class
Package: Subsystem_Control
Categorization of a product. Values correspond to items in an externally defined list.

7.5.5.6 service_name_type
7.4.5.3 Type: IDLEnum
Package: Subsystem_Control
Enumeration of possible service names. Where a service may be offered at different compliance levels, multiple names are introduced with _LEVEL_x postfix to indicate different parts.

Table 7.5272 - Attributes of IDLEnum service_name_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlEnum» AIR ENGAGEMENT SUPPORT</td>
<td>air engagement support service</td>
</tr>
<tr>
<td>«idlEnum» CLUTTER_REPORTING</td>
<td>clutter reporting service</td>
</tr>
<tr>
<td>«idlEnum» ENCYCLOPAEDIC SUPPORT</td>
<td>encyclopaedic support service</td>
</tr>
<tr>
<td>«idlEnum» ENGAGEMENT_SUPPORT</td>
<td>engagement support service</td>
</tr>
<tr>
<td>«idlEnum» ENVIRONMENT AND STABILIZATION NAVIGATION INFORMATION LEVEL 3F</td>
<td>C2I nav service support as per compliance level 3F</td>
</tr>
<tr>
<td>«idlEnum» ENVIRONMENT AND STABILIZATION METOC LEVEL 3G</td>
<td>Meteorological and Oceanographic support service as per compliance level 3G</td>
</tr>
<tr>
<td>«idlEnum» EXTENDED SUBSYSTEM CONTROL</td>
<td>Extensions to the subsystem control service</td>
</tr>
<tr>
<td>«idlEnum» JAMMER REPORTING</td>
<td>jammer reporting service</td>
</tr>
<tr>
<td>«idlEnum» MISSILE GUIDANCE</td>
<td>missile guidance service</td>
</tr>
<tr>
<td>«idlEnum» PLOT REPORTING LEVEL 1</td>
<td>plot reporting service to compliance level 1</td>
</tr>
<tr>
<td>«idlEnum» PLOT REPORTING LEVEL 3C</td>
<td>plot reporting service to compliance level 3C</td>
</tr>
<tr>
<td>«idlEnum» PLOT REPORTING LEVEL 3E</td>
<td>plot reporting service to compliance level 3E</td>
</tr>
<tr>
<td>«idlEnum» RECORDING AND REPLAY</td>
<td>recording and replay service</td>
</tr>
<tr>
<td>«idlEnum» SEARCH</td>
<td>search service</td>
</tr>
<tr>
<td>«idlEnum» SENSOR_CONTROLLEVEL 2</td>
<td>sensor control service to compliance level 2</td>
</tr>
<tr>
<td>«idlEnum» SENSOR_PERFORMANCE</td>
<td>sensor performance service</td>
</tr>
<tr>
<td>«idlEnum» SIMULATION SUPPORT</td>
<td>simulation support service</td>
</tr>
<tr>
<td>«idlEnum» SUBSYSTEM_CONTROLLEVEL 1</td>
<td>subsystem control service to compliance level 1</td>
</tr>
<tr>
<td>«idlEnum» SUBSYSTEM_CONTROLLEVEL 2</td>
<td>subsystem control service to compliance level 2</td>
</tr>
<tr>
<td>«idlEnum» SURFACE ENGAGEMENT_SUPPORT</td>
<td>surface engagement support service</td>
</tr>
<tr>
<td>«idlEnum» TRACK REPORTING LEVEL 1</td>
<td>track reporting service to compliance level 1</td>
</tr>
<tr>
<td>«idlEnum» TRACK_REPORTING LEVEL 3C</td>
<td>track reporting service to compliance level 3C</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>------------------------------------------------------------</td>
</tr>
<tr>
<td><code>«idlEnum» TRACK_REPORTING_LEVEL_3E</code></td>
<td>track reporting service to compliance level 3E</td>
</tr>
<tr>
<td><code>«idlEnum» TRACKING_CONTROL_LEVEL_2</code></td>
<td>tracking control service to compliance level 2</td>
</tr>
<tr>
<td><code>«idlEnum» TRACKING_CONTROL_LEVEL_3C</code></td>
<td>tracking control service to compliance level 3C</td>
</tr>
<tr>
<td><code>«idlEnum» SENSOR_CONTROL_LEVEL_3A</code></td>
<td>sensor control service to compliance level 3A</td>
</tr>
<tr>
<td><code>«idlEnum» CONTACT_REPORTING</code></td>
<td>contact reporting service</td>
</tr>
<tr>
<td><code>«idlEnum» PARAMETRIC_REPORTING</code></td>
<td>parametric reporting</td>
</tr>
<tr>
<td><code>«idlEnum» TRACK_ASSESSMENT</code></td>
<td>track assessment service</td>
</tr>
<tr>
<td><code>«idlEnum» MEDIA_STREAMING</code></td>
<td>media streaming service</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>«idlEnum» SEARCH</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» SENSOR_CONTROL_LEVEL_3</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» SENSOR_PERFORMANCE</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» SIMULATION_SUPPORT</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» SUBSYSTEM_CONTROL_LEVEL_1</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» SUBSYSTEM_CONTROL_LEVEL_2</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» SURFACE_ENGAGEMENT_SUPPORT</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» TRACK_REPORTING_LEVEL_1</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» TRACK_REPORTING_LEVEL_3C</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» TRACK_REPORTING_LEVEL_3E</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» TRACKING_CONTROL_LEVEL_2</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» TRACKING_CONTROL_LEVEL_3C</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» SENSOR_CONTROL_LEVEL_3A</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» CONTACT_REPORTING</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» PARAMETRIC_REPORTING</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» TRACK_ASSESSMENT</code></td>
<td></td>
</tr>
<tr>
<td><code>«idlEnum» MEDIA_STREAMING</code></td>
<td></td>
</tr>
</tbody>
</table>

**7.5.5.7 battle_override_state_type**
7.4.5.4 Type: Class
Package: Subsystem_Control
If the boolean is true the battle override is applied.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>battle_override_applied</td>
<td>boolean</td>
</tr>
<tr>
<td></td>
<td>Indicates if the battle override is applied or not.</td>
</tr>
</tbody>
</table>

7.5.5.8 descriptor_type
7.4.5.5 Type: Class
Package: Subsystem_Control
Type for parameter descriptors.
### Table 7.74 - Attributes of Class descriptor_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameter_name</td>
<td>parameter_name values are unique within the scope of a subsystem.</td>
</tr>
<tr>
<td>parameter_type</td>
<td>The type of the information parameter.</td>
</tr>
<tr>
<td>parameter_unit</td>
<td>The units in which the value of the parameter is expressed.</td>
</tr>
<tr>
<td>typical_value</td>
<td>A typical value of the information parameter so as to assist in providing a suitable value.</td>
</tr>
<tr>
<td>parameter_range</td>
<td>The valid range of the information parameter.</td>
</tr>
<tr>
<td>technical_state</td>
<td>Technical state(s) in which this parameter may be modified.</td>
</tr>
<tr>
<td>applicable_operational_mode</td>
<td>Operational modes to which the information value applies.</td>
</tr>
</tbody>
</table>

#### 7.5.5.9 descriptor_sequence_type

**Type:** Class

**Package:** Subsystem_Control

Sequence of parameter descriptors, used in retrieving parameter descriptors.

#### 7.4.5.6 device_identification_type

**Type:** IDLStruct

**Package:** Subsystem_Control

Identification data of the equipment.

### Table 7.54 - Attributes of IDLStruct device_identification_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>product</td>
<td>Name of the product. Example: TRS3D</td>
</tr>
<tr>
<td>serial_number</td>
<td>Serial number identifying the individual device.</td>
</tr>
</tbody>
</table>

### Table 7.75 - Relations of Class descriptor_sequence_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>product_device_name</td>
<td>Name of the product. Example: TRS3D</td>
</tr>
<tr>
<td>serial_number_device</td>
<td>Serial number identifying the individual device.</td>
</tr>
</tbody>
</table>
### 7.5.5.10 `device_name` identification type

**Type:** `IDLStruct`  
**Package:** `Subsystem_Control`  

Identification data of the equipment.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>product</code></td>
<td>Name of the product. Example TRS3D</td>
</tr>
</tbody>
</table>
| `serial_number` | Serial number identifying the individual device.  
AttributeTag: Length = 64 |
| `equipment_type` | This describes the general type of the equipment. Example: Air Surveillance Radar |
| `version` | Version of the device. |
| `product_category` | Categorization of the product implementing the interface. This is the unique identification of the product given a particular external schema.  
AttributeTag: Issue = |
| `equipment_category` | Categorization of the kind of equipment implementing the interface. This is the specific identification of the product’s equipment category given a particular external schema.  
AttributeTag: Issue = |
| `platform_name` | The name of the platform hosting the interface participant.  
AttributeTag: Issue = |
| `platform_category` | Categorization of platform hosting the product implementing the interface. This is the unique identification of the platform given a particular external schema.  
AttributeTag: Issue = |

#### 7.5.5.11 `device_name` type

**7.4.5.8 Type:** `IDLTypeDef string`  
**Package:** `Subsystem_Control`  

Name of an entry in the device identification.  
ElementTag: Length = 64
<table>
<thead>
<tr>
<th><strong>1.1.1.1 device_name_type</strong></th>
</tr>
</thead>
</table>

**Table 7.55 - Attributes of IDITypeDef**
### 7.5.5.12 event_type

**Type:** IDLEnum  
**Package:** Subsystem_Control  
Type of event

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlEnum» OCCURRENCE</td>
<td>The event corresponds to the occurrence of some phenomena</td>
</tr>
<tr>
<td>«idlEnum» DISAPPEARANCE</td>
<td>The event corresponds to the disappearance of some phenomena</td>
</tr>
</tbody>
</table>

### 7.4.5.10 7.5.5.13 fault_type

**Type:** IDLStruct  
**Package:** Subsystem_Control  
Class to represent a subsystem fault

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
</table>
| fault_name string  | The name of the fault. Distinct instances of the same fault condition have the same name  
AttributeTag: StringLength = 32 |
| event event_type   | The categorization of the fault as an event; whether it is an occurrence or the disappearance of some phenomenon |
| simulated boolean  | Indicates whether this fault is real or simulated/inserted. |
| overridden boolean  | Indicates whether this fault is overridden by Battle Override when determining the health state. |
| fault_isolation_data string  | For instance cabinet id and rack id.  
AttributeTag: StringLength = 32 |

### 7.4.5.11 7.5.5.14 fault_list_type

**Type:** Class  
**Package:** Subsystem_Control  
A list of faults

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregate: element fault_type [0..*]</td>
<td></td>
</tr>
</tbody>
</table>
### 7.5.5.15 health_state_reason_type

**Type:** IDLStruct  
**Package:** Subsystem_Control

Reason for the health state

Table 7.5.5.15 - Attributes of IDLStruct health_state_reason_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>.caused_by_fault boolean</td>
<td>The health state has been caused by a fault</td>
</tr>
<tr>
<td>.caused_by_technical_state boolean</td>
<td>The health state is due to the subsystem being in a particular technical state</td>
</tr>
<tr>
<td>.caused_by_simulation_mode boolean</td>
<td>The health state is due to the subsystem being in a particular simulation mode</td>
</tr>
<tr>
<td>.caused_by_operational_mode boolean</td>
<td>The health state is due to the subsystem being in a particular operational mode</td>
</tr>
</tbody>
</table>

### 7.5.5.16 health_state_type

**Type:** IDLEnum  
**Package:** Subsystem_Control

Encapsulation of health state

Table 7.5.5.16 - Attributes of IDLEnum health_state_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
</table>
| «idlEnum» AVAILABLE     | Service: Indicates that the service is available with specified performance.  
                          | Subsystem: Indicates that all implemented services of the subsystem have health state AVAILABLE. |
| «idlEnum» DEGRADED      | Service: Indicates that the service may perform its operational task, but possibly with less than specified performance.  
                          | Subsystem: Indicates that at least one of the implemented services of the subsystem have health state other than AVAILABLE. |
| «idlEnum» NOT_AVAILABLE | Service: Indicates that the service is not available.  
                          | Subsystem: Indicates that all implemented services of the subsystem have health state NOT_AVAILABLE. |
| «idlEnum» UNKNOWN_HEALTH | Indicates that the subsystem may not determine the health state of the service or subsystem (e.g. because BIT is not running). |
### Table 7.5.5.17 - Attributes of IDLEnum information_name_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlEnum» AIR_PLOTS</td>
<td>Air plots information service</td>
</tr>
<tr>
<td>«idlEnum» SURFACE_PLOTS</td>
<td>Surface plots information service</td>
</tr>
<tr>
<td>«idlEnum» LAND_PLOTS</td>
<td>Land plots information service</td>
</tr>
<tr>
<td>«idlEnum» SPACE_PLOTS</td>
<td>Space plots information service</td>
</tr>
<tr>
<td>«idlEnum» SUBSURFACE_PLOTS</td>
<td>Subsurface plots information service</td>
</tr>
<tr>
<td>«idlEnum» SENSOR_AIR_TRACKS</td>
<td>Air tracks information service</td>
</tr>
<tr>
<td>«idlEnum» SENSOR_SURFACE_TRACKS</td>
<td>Name of information service</td>
</tr>
<tr>
<td>«idlEnum»</td>
<td>SENSOR_SURFACE_TRACKS</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>«idlEnum»</td>
<td>SENSOR_LAND_TRACKS</td>
</tr>
<tr>
<td>«idlEnum»</td>
<td>SENSOR_SPACE_TRACKS</td>
</tr>
<tr>
<td>«idlEnum»</td>
<td>SENSOR_SUBSURFACE_TRACKS</td>
</tr>
<tr>
<td>«idlEnum»</td>
<td>JAMMER_STROBES</td>
</tr>
<tr>
<td>«idlEnum»</td>
<td>JAMMER_TRACKS</td>
</tr>
<tr>
<td>«idlEnum»</td>
<td>JAMMING_EFFECT_ASSESSMENTS</td>
</tr>
<tr>
<td>«idlEnum»</td>
<td>INTERFERENCE_REPORTS</td>
</tr>
</tbody>
</table>

### 7.4.5.157.5.5.18 interest_type

**Type:** IDLStruct  
**Package:** Subsystem_Control

**Encapsulation of interest in service**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>registration registration_type</td>
<td>Whether adding or removing interest in an information service.</td>
</tr>
</tbody>
</table>
| quality_of_service string | The quality of service being requested of the information service.  
AttributeTag: StringLength = 32 |
| recipient string | Identification of the recipient of the information service.  
AttributeTag: StringLength = 32 |

### 7.4.5.167.5.5.19 interest_list_type

**Type:** Class  
**Package:** Subsystem_Control

A list of interest
### Table 7.84 - Relations of Class interest_list_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: element interest type [1..*]</td>
<td></td>
</tr>
</tbody>
</table>

7.5.5.20 mastership_state_type

**Type:** Class  
**Package:** Subsystem_Control

This enumeration represents the state of the mastership. The subsystem Mastership may be either “free”, that is assigned to none and then available to anybody asks for it, or assigned to somebody: CMS or not.

### Table 7.5985 - Attributes of Class mastership_state_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>enum MASTERSHIP_FREE</td>
<td>Mastership state is “free”, the first received Mastership request shall be satisfied.</td>
</tr>
<tr>
<td>enum MASTERSHIP_OTHER</td>
<td>The Mastership is assigned to somebody other than CMS.</td>
</tr>
<tr>
<td>enum MASTERSHIP_TO_CMS</td>
<td>The Mastership is assigned to CMS.</td>
</tr>
</tbody>
</table>
7.4.5.18 7.5.21 parameter_name_type
Type: IDLStruct
Package: Subsystem_Control
Typedef for strings representing names of parameters.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameter_name string</td>
<td>parameter_name values are unique within the scope of a subsystem. AttributeTag: StringLength $\equiv$ 128</td>
</tr>
</tbody>
</table>

7.4.5.19 7.5.22 name_error_pair_type
Type: Class
Package: Subsystem_Control
Combination of name of parameter (for which a request could not be processed) and an indication of the error.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameter_name string</td>
<td>parameter_name values are unique within the scope of a subsystem. AttributeTag: StringLength $\equiv$ 128</td>
</tr>
<tr>
<td>error_indication string</td>
<td>A description of or reference for the error condition. AttributeTag: StringLength $\equiv$ 32</td>
</tr>
</tbody>
</table>

7.5.23 name_error_sequence_type
7.4.20 Type: Class
Package: Subsystem_Control
Sequence of error reports identifying the parameter names for which the request could not be processed, including an indication of the error (e.g. unknown parameter, illegal value).

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: element name_error_pair_type $[0,*]$</td>
<td></td>
</tr>
</tbody>
</table>

7.5.24 parameter_name_sequence_type
7.4.24 Type: Class
Package: Subsystem_Control
A sequence of strings (names). Used in request for parameters and parameter descriptors. If the sequence is empty, the request is for all parameters.
Table 7.89 - Relations of Class parameter_name_sequence_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: element parameter_name_sequence_type [0..*]</td>
<td></td>
</tr>
</tbody>
</table>

### 7.4.5.22/5.5.25 name_value_pair_type

**Type:** Class  
**Package:** Subsystem_Control  
A generic struct for (name, value) pairs. Used in multiple situations.

Table 7.6290 - Attributes of Class name_value_pair_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameter_name</td>
<td>parameter_name values are unique within the scope of a subsystem. AttributeTag: StringLength = 128</td>
</tr>
<tr>
<td>value</td>
<td>The value of the parameter</td>
</tr>
<tr>
<td>value_string</td>
<td>AttributeTag: StringLength = 32</td>
</tr>
</tbody>
</table>

### 7.5.5.26 name_value_sequence_type

**7.4.5.23 Type:** Class  
**Package:** Subsystem_Control  
Sequence of (name, value) pairs used in retrieving and modifying parameters.

Table 7.91 - Relations of Class name_value_sequence_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: element name_value_sequence_type [0..*]</td>
<td></td>
</tr>
</tbody>
</table>

### 7.5.5.27 operational_mode_type

**7.4.5.24 Type:** IDLTypeDef unsigned short  
**Package:** Subsystem_Control  
The value should be mapped to the corresponding operational mode. This mapping is retrieved through the service 'Manage Subsystem Parameters'.

### 7.5.5.28 parameter_value_response_type

**7.4.5.25 Type:** Class  
**Package:** Subsystem_Control  
Response type for retrieving and modifying sequences of parameters.
Table 7.6392 - Attributes of Class parameter_value_response_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_id</td>
<td>The identifier for the request.</td>
</tr>
</tbody>
</table>

**7.4.5.267.5.5.29 registration_type**

Type: IDLEnum
Package: Subsystem_Control
Type of registration

Table 7.6493 - Attributes of IDLEnum registration_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlEnum» REGISTER</td>
<td>Registering for a service</td>
</tr>
<tr>
<td>«idlEnum» DEREGISTER</td>
<td>Deregistering for a service</td>
</tr>
</tbody>
</table>

**7.4.5.277.5.5.30 service_type**

Type: IDLStruct
Package: Subsystem_Control
Type of service

Table 7.6594 - Attributes of IDLStruct service_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>service_name</td>
<td>service_name_type Only registrable services are allowed</td>
</tr>
</tbody>
</table>

**7.4.5.287.5.5.31 service_health_type**

Type: IDLStruct
Package: Subsystem_Control
Health of service
### Table 7.6695 - Attributes of IDLStruct service_health_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>service_name</td>
<td>The name of the service being reported on</td>
</tr>
<tr>
<td>health_state</td>
<td>The state of health of the service</td>
</tr>
<tr>
<td>health_state_reason</td>
<td>The reason for the health state</td>
</tr>
<tr>
<td>time_of_information</td>
<td>The absolute time at which the information was known to be valid</td>
</tr>
</tbody>
</table>

### Table 7.96 - Relations of IDLStruct service_health_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: influences</td>
<td>fault_type reference [0..*]</td>
</tr>
</tbody>
</table>

### 7.5.5.32 service_indication_list_type

- **Type:** IDLStruct
- **Package:** Subsystem_Control
- A list of service indications as used by Provide_Subsystem_Services.

### Table 7.97 - Relations of IDLStruct service_indication_list_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: service_indication</td>
<td>service_indication_type [0..*]</td>
</tr>
</tbody>
</table>

### 7.5.5.33 service_indication_type

- **Type:** IDLStruct
- **Package:** Subsystem_Control
- Indication of a service provided by the subsystem.

### Table 7.6798 - Attributes of IDLStruct service_indication_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>service_name</td>
<td>Name of the service.</td>
</tr>
<tr>
<td>registration_indicator</td>
<td>Indication whether the service is registered.</td>
</tr>
</tbody>
</table>

### 7.5.5.34 service_information_type

- **Type:** IDLStruct
- **Package:** Subsystem_Control
- Information about a service

### Table 7.6899 - Attributes of IDLStruct service_information_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
</table>
### 7.4.5.327 7.5.5.35 service_list_type

**Type:** IDLStruct

**Package:** Subsystem_Control

A list of service names as used by Provide_Subsystem_Services.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>information_name</td>
<td>The name of the information in the service</td>
</tr>
</tbody>
</table>

#### Table 7.100 - Relations of IDLStruct service_list_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation:</td>
<td>service_indication service_name_type</td>
</tr>
<tr>
<td>[0..*]</td>
<td></td>
</tr>
</tbody>
</table>

### 7.5.5.36 subsystem_health_type

**7.4.5.33 Type:** IDLStruct

**Package:** Subsystem_Control

Type describing the health state of a subsystem

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>health_state</td>
<td>Current health state</td>
</tr>
</tbody>
</table>

#### Table 7.101 - Attributes of IDLStruct subsystem_health_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>health_state_reason</td>
<td>Reason for last change of health state</td>
<td></td>
</tr>
<tr>
<td>subsystem_identification</td>
<td>The subsystem being reported upon</td>
<td></td>
</tr>
<tr>
<td>time_of_information</td>
<td>The absolute time at which the information provided in the report was known to be valid</td>
<td></td>
</tr>
</tbody>
</table>

#### Table 7.102 - Relations of IDLStruct subsystem_health_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association:</td>
<td>influences service_health_type reference</td>
</tr>
<tr>
<td>[1..*]</td>
<td></td>
</tr>
</tbody>
</table>

### 7.4.5.347 7.5.5.37 technical_state_type

**Type:** Class

**Package:** Subsystem_Control
Package: **Subsystem_Control**  
Type which is used to indicate a technical state.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIT</td>
<td>Subsystem is running Built-In-Test procedure. CMS may communicate with subsystem, but subsystem shall only respond affirmatively to a limited set of commands. From this state the subsystem may transition to READY, FAILED, CALIBRATE, STANDBY (transition may be ordered before completion of BIT if Battle Override is enabled), or OFFLINE.</td>
</tr>
<tr>
<td>CALIBRATE</td>
<td>Subsystem is running calibration procedure. Subsystem shall only respond to a limited set of commands from CMS. From this state the subsystem may transition to READY, FAILED, BIT, STANDBY (transition may be ordered before completion of calibration if Battle Override is enabled), or OFFLINE.</td>
</tr>
<tr>
<td>DORMANT</td>
<td>Interface between CMS and subsystem may or may not exist. Some power is applied to the subsystem and temperature control (e.g. cooling) is active. From this state, the sub-system may transition to FAILED, STANDBY, or OFFLINE.</td>
</tr>
<tr>
<td>FAILED</td>
<td>Subsystem is non-operational due to a critical fault such as a primary power supply failure. CMS is able to communicate with subsystem to perform diagnostics. In the FAILED state, the health state of the sub-system and nearly all associated services is NOT AVAILABLE or UNKNOWN (provided via Health State). If the health state of the sub-system or some services is DEGRADED, the sub-system is not required to enter into this state. From this state the sub-system may transition to BIT, STANDBY, READY, CALIBRATE, DORMANT or OFFLINE.</td>
</tr>
<tr>
<td>OFFLINE</td>
<td>No connection between CMS and Subsystem is open. Main power is usually not applied to subsystem. From OFFLINE, subsystem transitions to FAILED, DORMANT, BIT, or STANDBY.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------</td>
<td>-------</td>
</tr>
<tr>
<td>ONLINE</td>
<td>Subsystem is operational and may respond to all requests from CMS. Simulation and diagnostics may be allowed in this state. Radiation is allowed in this state but must be commanded on via Control Emissions. From this state the subsystem may transition to BIT, CALIBRATE, READY, STANDBY, FAILED, or OFFLINE.</td>
</tr>
<tr>
<td>READY</td>
<td>Subsystem is ready for CMS to command full operation. Simulation may be allowed in this state. Ready to transition to ONLINE. Self-tests and calibration has been performed as necessary. Radiation is not allowed in the READY state. From this state the subsystem may transition to STANDBY, ONLINE, FAILED, BIT, CALIBRATE, or OFFLINE.</td>
</tr>
<tr>
<td>STANDBY</td>
<td>Interface between CMS and subsystem is established. Subsystem may not operate fully. Maintenance may be performed in this state. From this state the sub-system may transition to READY, CALIBRATE, BIT, FAILED, DORMANT, or OFFLINE.</td>
</tr>
<tr>
<td>BIT</td>
<td>Subsystem is running Built-In-Test procedure. CMS may communicate with subsystem, but subsystem shall only respond affirmatively to a limited set of commands. From this state the subsystem may transition to READY, FAILED, CALIBRATE, STANDBY (transition may be ordered before completion of BIT if Battle Override is enabled), or OFFLINE.</td>
</tr>
<tr>
<td>CALIBRATE</td>
<td>Subsystem is running calibration procedure. Subsystem shall only respond to a limited set of commands from CMS. From this state the subsystem may transition to READY, FAILED, BIT, STANDBY (transition may be ordered before completion of calibration if Battle Override is enabled), or OFFLINE.</td>
</tr>
<tr>
<td>DORMANT</td>
<td>Interface between CMS and subsystem may or may not exist. Some power is applied to the subsystem and temperature control (e.g. cooling) is active. From this state, the sub-system may transition to FAILED, STANDBY, or OFFLINE.</td>
</tr>
</tbody>
</table>
#### Attribute Notes

**FAILED**

Subsystem is non-operational due to a critical fault such as a primary power supply failure. CMS is able to communicate with subsystem to perform diagnostics. In the FAILED state, the health state of the sub-system and nearly all associated services is NOT AVAILABLE or UNKNOWN (provided via Health State). If the health state of the sub-system or some services is DEGRADED, the sub-system is not required to enter into this state. From this state the sub-system may transition to BIT, STANDBY, READY, CALIBRATE, DORMANT or OFFLINE.

**OFFLINE**

No connection between CMS and Subsystem is open. Main power is usually not applied to subsystem. From OFFLINE, subsystem transitions to FAILED, DORMANT, BIT, or STANDBY.

**ONLINE**

Subsystem is operational and may respond to all requests from CMS. Simulation and diagnostics may be allowed in this state. Radiation is allowed in this state but must be commanded on via Control Emissions. From this state the subsystem may transition to BIT, CALIBRATE, READY, STANDBY, FAILED, or OFFLINE.

**READY**

Subsystem is ready for CMS to command full operation. Simulation may be allowed in this state. Ready to transition to ONLINE, self-tests and calibration has been performed as necessary. Radiation is not allowed in the READY state. From this state the sub-system may transition to ONLINE, FAILED, BIT, CALIBRATE, or OFFLINE.

**STANDBY**

Interface between CMS and subsystem is established. Subsystem may not operate fully. Maintenance may be performed in this state. From this state the sub-system may transition to READY, CALIBRATE, BIT, FAILED, DORMANT, or OFFLINE.

---

**7.4.5.357, 5.5.38 version_type**

Type: IDLStruct

Package: Subsystem_Control

Version of the equipment

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>major_version</td>
<td>Major version number</td>
</tr>
<tr>
<td>minor_version</td>
<td>Minor version number</td>
</tr>
</tbody>
</table>
### Initial Type: Initial State

<table>
<thead>
<tr>
<th>Package</th>
<th>Subsystem_Control</th>
<th>major_version</th>
<th>unsigned short</th>
<th>Major version number</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>minor_version</td>
<td>unsigned short</td>
<td>Minor version number</td>
<td></td>
</tr>
</tbody>
</table>

#### 7.57.6 Sensor_Domain

**Parent Package:** Domain_Model

This package contains the Domain Models for the Clutter Reporting, Plot Reporting, Sensor Control, Sensor Performance, Track Reporting, and Tracking Control services.

#### 7.57.6.1 Clutter_Reporting

**Parent Package:** Sensor_Domain

Contains Structs used within the Clutter Reporting service.
7.6.1 clutter_assessment_request_type

Type: IDLStruct

Package: Clutter_Reporting

CMS generated request for a clutter assessment.

Table 7.22105 - Attributes of IDLStruct clutter_assessment_request_type
### 7.6.1.2 clutter_indication_type

**Type:** Class

**Package:** Clutter_Reporting

Indicates if the clutter within the cell is of a specific type.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAND</td>
<td>clutter caused by land</td>
</tr>
<tr>
<td>SEA</td>
<td>clutter caused by sea surface</td>
</tr>
<tr>
<td>WEATHER</td>
<td>clutter caused by weather phenomena</td>
</tr>
<tr>
<td>NO_STATEMENT</td>
<td>clutter cause unknown or unstated</td>
</tr>
</tbody>
</table>

### 7.6.1.3 clutter_map_cell_type

**Type:** IDLStruct

**Package:** Clutter_Reporting
Indicates the intensity and type of clutter for a defined geometric type.

### Table 7.74clutter_map_cell_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>cell_boundaries</td>
<td>Indicates the boundaries of the cell for which clutter is being reported.</td>
</tr>
<tr>
<td>clutter_type</td>
<td>Indicates whether the clutter is LAND, SEA, WEATHER, or unspecified (NO_STATEMENT).</td>
</tr>
<tr>
<td>clutter_intensity</td>
<td>Intensity of the clutter for the specified cell. Units indicated by the intensity type attribute.</td>
</tr>
</tbody>
</table>

### 7.6.1.4 clutter_report_type

**Type:** IDLStruct  
**Package:** Clutter_Reporting  
Clutter report generated by the subsystem.

### Table 7.75clutter_report_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>intensity_type</td>
<td>Indicates the units of the clutter intensity reported.</td>
</tr>
<tr>
<td>time_of_report</td>
<td>Time of the clutter report.</td>
</tr>
</tbody>
</table>

### 7.6.1.5 concentration_plot_cell_type

**Type:** Class  
**Package:** Clutter_Reporting  
Indicates the plot concentration of a defined geometric type.

### Table 7.76concentration_plot_cell_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>cell_boundaries</td>
<td>Specifies the dimension of the cell for which plot concentration is being reported.</td>
</tr>
<tr>
<td>plot_count</td>
<td>The number of plots generated within the cell.</td>
</tr>
</tbody>
</table>

### 7.6.1.6 intensity_units_type

**Type:** Class  
**Package:** Clutter_Reporting
Units of the clutter intensity

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>POWER_RECEIVED_LINEAR</td>
<td>Direct measurement of power in Watts on a linear scale</td>
</tr>
<tr>
<td>POWER_RECEIVED_LOG_LINEAR</td>
<td>Direct measurement of power on a logarithmic scale (e.g. dBm, dBW)</td>
</tr>
<tr>
<td>RCS_LINEAR</td>
<td>estimated radar cross section in square meters</td>
</tr>
<tr>
<td>RCS_LOG_LINEAR</td>
<td>estimated radar cross section on a logarithmic scale</td>
</tr>
<tr>
<td>SNR_LINEAR</td>
<td>Ratio of the signal and noise amplitudes</td>
</tr>
<tr>
<td>SNR_LOG_LINEAR</td>
<td>Ratio of the signal and noise amplitudes on a logarithmic scale</td>
</tr>
</tbody>
</table>

**7.6.1.7 plot_concentration_report_type**

**7.5.1.7** Type: IDLStruct  
**Package:** Clutter_Reporting  
Plot concentration report as generated by the subsystem.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_of_report time_type</td>
<td>Time of the plot concentration report.</td>
</tr>
</tbody>
</table>

**Table 7.78112 - Attributes of IDLStruct plot_concentration_report_type**

**Table 7.113 - Relations of IDLStruct plot_concentration_report_type**

**7.6.1.8 plot_concentration_request_data_type**

**7.5.1.8** Type: IDLStruct  
**Package:** Clutter_Reporting  
CMS request for plot concentration of a specified region.

**Table 7.79114 - Attributes of IDLStruct plot_concentration_request_data_type**
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>region_of_plot_concentration_request</td>
<td>Region for which the plot concentration was requested.</td>
</tr>
<tr>
<td>general_polar_volume_type</td>
<td></td>
</tr>
</tbody>
</table>

7.5.27.6.2 Media Streaming

**Parent Package:** Sensor_Domain

This package provides a data model for describing the metadata associated with a sensor's media streams.
Figure 7.39.63 Media_Streaming (Class diagram)

### 7.5.2.1 codec_type
**Type:** Class `string`
**Package:** Media_Streaming
The representation of the codec associated with the stream
ElementTag: Length = 20

### 7.6.2.1 media_allocation_type
**Type:** Class `string`
**Package:** Media_Streaming
To represent the allocation of sensor tracks to media streams.

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association media_stream</td>
<td>The media stream relating to this allocation of tracks</td>
</tr>
<tr>
<td>media_stream_metadata_type reference [1]</td>
<td></td>
</tr>
<tr>
<td>Connector</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>Association sensor_track sensor_track_type reference [0, *]</td>
<td>The sensor tracks that are present in the media stream</td>
</tr>
</tbody>
</table>

**7.6.2.3 media_kind_type**

**7.5.2.3 Type:** Class

**Package:** Media_Streaming

The high-level categorisation of types of media
Table 7.116 - Attributes of Class media_kind_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUDIO</td>
<td>Audio media stream</td>
</tr>
<tr>
<td>VIDEO</td>
<td>Video media stream</td>
</tr>
<tr>
<td>OTHER_MEDIA</td>
<td>Another media stream</td>
</tr>
</tbody>
</table>

7.6.2.4 media_name_type

7.5.2.4 **Type:** Class string
7.5.2.4 **Package:** Media_Streaming
The representation for the identifying name of a media stream
ElementTag: Length = 32

7.6.2.5 media_stream_id_type

7.5.2.5 **Type:** Class long
7.5.2.5 **Package:** Media_Streaming
The representation for the unique identifier for the media stream

7.6.2.6 media_stream_metadata_type

7.5.2.6 **Type:** Class
7.5.2.6 **Package:** Media_Streaming
The representation of a media stream such as video or audio

Table 7.80117 - Attributes of Class media_stream_metadata_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>media_kind</td>
<td>media_kind_type</td>
</tr>
<tr>
<td>codec</td>
<td>codec_type</td>
</tr>
<tr>
<td>uri</td>
<td>url_type</td>
</tr>
<tr>
<td>media_name</td>
<td>media_name_type</td>
</tr>
<tr>
<td>«key» media_id</td>
<td>media_stream_id_type</td>
</tr>
</tbody>
</table>

7.5.3.7.6.3 Search

Parent Package: Sensor_Domain
7.6.3.1 cued_search_cue_type

Type: Class
Package: Search

Type used for specifying the constraints on a cued search.

Table 7.81118 - Attributes of Class cued_search_cue_type
Table 7.82 - Attributes of Class cued_search_report_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>found_track_id sensor_track_id [0..1]</td>
<td>The identifier of the track formed as a result of the search request</td>
</tr>
</tbody>
</table>

Table 7.120 - Relations of Class cued_search_report_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: original_cue cued_search_cue_type reference</td>
<td></td>
</tr>
</tbody>
</table>

7.6.3.3 search_pattern_type

7.5.3.3 Type: IDLEnum

Package: Search

The types of search pattern that can be employed for search and surveillance tasks

Table 7.83 - Attributes of IDLEnum search_pattern_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«enum» LAWNMOWER_BY_LENGTH</td>
<td>Coverage by alternating traversal of the area length-wise. Valid for rectangular areas.</td>
</tr>
<tr>
<td>«enum» LAWNMOWER_BY_WIDTH</td>
<td>Coverage by alternating traversal of the area width-wise. Valid for rectangular areas.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>«enum» SPIRAL_IN</td>
<td>Coverage by traversing the perimeter and then progressively smaller traversals of the interior towards the center.</td>
</tr>
<tr>
<td>«enum» SPIRAL_OUT</td>
<td>Coverage by starting at the center and traversing through the interior on a path that is (approximately) tangential to the center and parallel to the perimeter until the perimeter has been traversed.</td>
</tr>
<tr>
<td>«enum» RANDOM_SAMPLE</td>
<td>Search by sensing subsets of the area selected at random.</td>
</tr>
<tr>
<td>«enum» UNSPECIFIED_PATTERN</td>
<td>No search pattern is specified.</td>
</tr>
</tbody>
</table>

**7.6.3.4 search_repeat_type**

**7.5.3.4 Type:** IDLEnum  
**Package:** Search  
Defines the search behavior on repeat / subsequent searches

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«enum» ONCE_ONLY</td>
<td>Complete a single search pattern.</td>
</tr>
<tr>
<td>«enum» REPEAT</td>
<td>Repeat the task indefinitely.</td>
</tr>
<tr>
<td>«enum» REPEAT_ONCE</td>
<td>Repeat the task once.</td>
</tr>
<tr>
<td>«enum» REVERSE</td>
<td>Repeat the task in reverse indefinitely.</td>
</tr>
<tr>
<td>«enum» REVERSE_ONCE</td>
<td>Repeat the task in reverse once.</td>
</tr>
<tr>
<td>«enum» RANDOM_REPEAT</td>
<td>Randomly repeat the elements of the task indefinitely.</td>
</tr>
<tr>
<td>«enum» UNSPECIFIED_REPEAT</td>
<td>No repeat specified.</td>
</tr>
</tbody>
</table>

**Table 7.122 - Attributes of IDLEnum search_repeat_type**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«enum» REPEAT</td>
<td>Repeat the task indefinitely.</td>
</tr>
<tr>
<td>«enum» REPEAT_ONCE</td>
<td>Repeat the task once.</td>
</tr>
<tr>
<td>«enum» REVERSE</td>
<td>Repeat the task in reverse indefinitely.</td>
</tr>
<tr>
<td>«enum» REVERSE_ONCE</td>
<td>Repeat the task in reverse once.</td>
</tr>
<tr>
<td>«enum» RANDOM_REPEAT</td>
<td>Randomly repeat the elements of the task indefinitely.</td>
</tr>
</tbody>
</table>
7.6.3.5 surveillance_area_type

Type: IDLStruct
Package: Search

A 2D area that is included in a surveillance task

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>pattern search_pattern_type</td>
<td>The pattern to apply to the area</td>
</tr>
<tr>
<td>.area area_2d_type</td>
<td>The area to be kept under surveillance.</td>
</tr>
<tr>
<td>repeat search_repeat_type</td>
<td>The search behavior at the end of a search cycle.</td>
</tr>
</tbody>
</table>

7.6.3.6 surveillance_search_type

Type: IDLStruct
Package: Search

The parameters with which to task a sensor to concentrate its surveillance efforts within a spatial and/or frequency band.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>volume general_polar_volume_type</td>
<td>The region of surveillance in the environment to be searched for tracks.</td>
</tr>
<tr>
<td>coordinate_orientation</td>
<td>coordinate_orientation_type The orientation of the polar coordinates used in this class. Note that the origin is always the sensor reference point and that the coordinate system is always polar.</td>
</tr>
<tr>
<td>frequency_band</td>
<td>frequency_band_type [0..1] The frequency band to be searched.</td>
</tr>
</tbody>
</table>

7.6.3.7 surveillance_task_type

Type: IDLStruct
Package: Search

The information for a CMS request to the subsystem (as appropriate to be a directional sensor that can be steered) to undertake a surveillance task.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
</table>
The system tracks to keep under surveillance. The information regarding the system tracks is published using another interface standard, such as the TACSIT Data Exchange specification (TEX), the choice of which may be system specific.

AttributeTag: Length = 100

**Table 7.126 - Relations of IDLStruct surveillance_task_type**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>system_track system_track_id_type [0..*]</td>
<td>The system tracks to keep under surveillance. The information regarding the system tracks is published using another interface standard, such as the TACSIT Data Exchange specification (TEX), the choice of which may be system specific. AttributeTag: Length = 100</td>
</tr>
</tbody>
</table>

### 7.5.47.6.4 Sensor_Assessment

**Parent Package:** Sensor_Domain

This package provides a data model to describe the identification and classification assessment that a sensor can make about its sensor tracks. The approach is to be agnostic to any specific assessment process or classification regime. The model assumes a general process whereby matches (with confidence values) are made between data relating to the sensor track and reference data. Matches can relate to previous matches building up a structured hierarchy of assumptions leading to progressively higher-level identification and classification assessments. The lowest level is to match measurement parameters with reference data; the next level is to match one or more of these with modes; then modes with equipment and finally equipment with platforms.

Assessment Objectives (Equipment, Function, Platform and Activity) are passed from Subsystems to CMS by value in the Sensor Assessment use cases. They contain an Objective Id key value which is used to refer to the assessment values in Track Reporting use cases.
Figure 7.41.65 Sensor_Assessment - plots (Class diagram)
The classes to support assessment of a sensor track at the equipment and platform level.
Figure 7.42.66 Sensor_Assessment - platform (Class diagram)

The classes to support the assessment of a sensor track at the mode level
Figure 7.43.67 Sensor Assessment - modes (Class diagram)

Basic types to support sensor assessment
Open Architecture Radar Interface Standard (OARIS), v2.0
Classes to support the configuration of supported categories for sensor assessment

Figure 7.44.68 Sensor_Assessment - base types (Class diagram)

Classes to support the configuration of supported categories for sensor assessment
Classes to support the assessment of multi-path effects.
7.5.4.1 assessment_object_id_type
Type: Class long
Package: Sensor_Assessment
Unique identifier for the objects to which the sensor assessment is attempting to match the measurement parameters.

7.5.4.2 country_code_type
Type: Class string
Package: Sensor_Assessment
Two character (Alpha 2) country code as defined by ISO 3166-1.
An empty string represents undefined data.
ElementTag: Length = 2

7.5.4.3 descriptor_list_type
Type: Class descriptor_single_value_type
Package: Sensor_Assessment
list of descriptor values
ElementTag: Length = 10

7.5.4.4 descriptor_name_type
Type: Class string
Package: Sensor_Assessment
Represents the name of a descriptor
ElementTag: Length = 24

7.5.4.5 descriptor_single_value_type
Table 7.127 - Attributes of Class
descriptor_single_value_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlCase» value descriptor_single_value_type</td>
<td>The option for a single value</td>
</tr>
<tr>
<td>«idlCase» list descriptor_list_type</td>
<td>The option for a list of values</td>
</tr>
</tbody>
</table>

Table 7.127 - Attributes of IDLUnion descriptor_value_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlCase» value descriptor_single_value_type</td>
<td>The option for a single value</td>
</tr>
<tr>
<td>«idlCase» list descriptor_list_type</td>
<td>The option for a list of values</td>
</tr>
</tbody>
</table>

Table 7.128 - Attributes of IDLStruct equipment_match_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>is_a_threat boolean</td>
<td>Whether the equipment - function combination is considered to be threatening</td>
</tr>
</tbody>
</table>

Table 7.129 - Relations of IDLStruct equipment_mode_match_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: function observable_function_type reference [0..1]</td>
<td>The function the equipment has been matched as performing</td>
</tr>
<tr>
<td>Association: equipment equipment_type reference</td>
<td></td>
</tr>
<tr>
<td>Aggregation: mode match equipment_mode_match_type [0..*]</td>
<td>A match between an equipment an a mode identified in the sensor track’s parametric measurements.</td>
</tr>
<tr>
<td>Aggregation: parameter match equipment_parameter_match_type [0..*]</td>
<td>A match between an equipment an a measurement parameter of the sensor track.</td>
</tr>
</tbody>
</table>
7.5.4.8 Type: IDLStruct match_link_type
Package: Sensor_Assessment
A match between an equipment match and a mode match. For a possible match to an equipment this represents the linkage to a possible parametric mode that has been identified in the assessment of the sensor track.

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: mode parametric_mode_match_type reference</td>
<td>The mode from the sensor track’s parametric measurements being matched.</td>
</tr>
</tbody>
</table>

7.6.4.8 equipment_name_type

7.5.4.9 Type: Class string
Package: Sensor_Assessment
The name of or label for an item of equipment
ElementTag: Length = 32

Table 7 - Attributes of Class equipment_name_type

7.6.4.9 equipment_parameter_match_type

7.5.4.10 Type: IDLStruct match_link_type
Package: Sensor_Assessment
Package: Sensor_Assessment
A match between an equipment match and a measurement parameter match. For a possible match to an equipment this represents the linkage to a possible reference parameter that has been identified in the assessment of the sensor track.
assessment

Table 7.131 - Relations of the sensor track

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association parameter reference parameter match type reference</td>
<td>The parameter from the sensor tracks measurement parameters being matched.</td>
</tr>
</tbody>
</table>

7.5.4.11 IDLStruct equipment parameter match type

7.6.4.10 equipment_type

Type: IDLStruct
Package: Sensor_Assessment
The representation of an item of equipment that is relevant to assessment of parametric sensor measurements.

Table 7.82132 - Attributes of IDLStruct equipment_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>equipment_name_type</td>
</tr>
<tr>
<td>id</td>
<td>assessment_objective_id_type</td>
</tr>
</tbody>
</table>

Table 7.133 - Relations of IDLStruct equipment_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association descriptor reference descriptor type [0..*]</td>
<td>Descriptors for the equipment providing supporting, amplifying or qualifying information</td>
</tr>
<tr>
<td>Association platform platform_class_type reference [0..*]</td>
<td>The platforms known to contain the equipment</td>
</tr>
</tbody>
</table>

7.5.4.127.6.4.11 function_name_type

Type: Class string
Package: Sensor_Assessment
The name of some functional behavior exhibited by an equipment
ElementTag: Length = 20

7.5.4.137.6.4.12 match_id_type

Type: Class long
Package: Sensor_Assessment
The unique identifier for a match instance (within the scope of a sensor).

7.5.4.147.6.4.13 match_link_type

Type: Class
Package: Sensor_Assessment
The representation of a link between an assessment match and an existing lower level match.
Table 7.88134 - Attributes of Class match_link_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>confidence</td>
<td>The confidence in the match between a one match and an existing lower level match for a sensor track. This is the result of a statistical hypothesis test.</td>
</tr>
</tbody>
</table>

7.5.4.157, 6.4.14 match_type

Type: Class
Package: SensorAssessment
An abstract base class for matches between measurements and reference data in the assessment process

Table 7.89135 - Attributes of Class match_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>confidence</td>
<td>The confidence in the match between an equipment and a parametric measurement for a sensor track. This is the result of a statistical hypothesis test.</td>
</tr>
<tr>
<td>id</td>
<td>The unique identifier (within the scope of a sensor) for the match instance. Match links for higher-level assessment objectives refer to lower-level matches using this identifier.</td>
</tr>
</tbody>
</table>

7.6.4.15 measurement_element_match_type

Type: Class
Package: SensorAssessment

Table 7.136 - Relations of Class measurement_element_match_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The discrete parameter measurement value contained in the element of the sequence. Each measurement_element_type instance referred to by an association instance belongs to a different discrete_set_measurement_type instance. The unique identifier (within the scope of a sensor) for the match instance. Match links for higher-level assessment objectives refer to lower-level matches using this identifier.</td>
</tr>
<tr>
<td>reference</td>
<td>The discrete parameter measurement value contained in the element of the sequence. Each measurement_element_type instance referred to by an association instance belongs to a different discrete_set_measurement_type instance. The unique identifier (within the scope of a sensor) for the match instance. Match links for higher-level assessment objectives refer to lower-level matches using this identifier.</td>
</tr>
</tbody>
</table>

7.5.4.16 measurement_element_match_type

Type: Class match_link_type
Package: SensorAssessment
7.5.4.17 mode_name_type
Type: Class
Package: Sensor_Assessment
The name or label for a kind of mode that gives rise to a set of measurements.

7.6.4.16 multipath_set_type
Type: Class
Package: Sensor_Assessment
Represents a set of tracks that correspond to the signal which has been measured through the detection of discrete signals that have arrived at the sensor by means of different paths through the environment.

Table 7.90 - Attributes of Class multipath_set_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«key» multipath_id match_id_type</td>
<td>The unique identifier for the multi-path set</td>
</tr>
<tr>
<td>confidence confidence_type</td>
<td>The probability that the set represents independently routed detections of the same real world object.</td>
</tr>
</tbody>
</table>

Table 7.138 - Relations of Class multipath_set_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: supporting_track sensor_track_type reference [1..*]</td>
<td>One of the tracks in the multipath set</td>
</tr>
<tr>
<td>Association: master sensor_track_type reference [0..1]</td>
<td>The master track for the multipath set</td>
</tr>
</tbody>
</table>

7.6.4.18 observable_function_type
7.5.4.19 Type: IDLStruct
Package: Sensor_Assessment
The representation of a function observable by the sensor that can be exhibited by equipment detected by the sensor and matched to sensor parametric measurements.

Table 7.91 - Attributes of IDLStruct observable_function_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name function_name_type</td>
<td>The name or label of the function</td>
</tr>
<tr>
<td>«key» id assessment_objective_id_type</td>
<td>Unique identifier for the function (within the scope of the sensor).</td>
</tr>
</tbody>
</table>

7.6.4.19 parametric_mode_match_type
7.5.4.20 Type: IDLStruct
Package: Sensor_Assessment

The identification of a mode within a sensor track's parametric data. A mode is a behavior of the real-world object being tracked by the sensor (or a component of that object).

Table 7.140 - Relations of IDLStruct parametric_mode_match_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: reference_mode reference_mode_type reference</td>
<td>The reference mode being matched</td>
</tr>
<tr>
<td>Aggregation: parameter_match reference_parameter_match_type [0..*]</td>
<td>The reference parameters, which have been matched by the measurement parameters in determining the mode assessment</td>
</tr>
</tbody>
</table>

7.5.4.217.6.4.20 platform_activity_name_type
The name of or label for an activity that can be undertaken by a platform. **ElementTag: Length = 32**

### Table 7 Table - Attributes of Class platform_activity_name_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name platform_activity_name_type</td>
<td>The name of the platform's activity</td>
</tr>
<tr>
<td>«key» id assessment_objective_id_type</td>
<td>Unique identifier for the activity (within the scope of the sensor).</td>
</tr>
</tbody>
</table>

#### 7.5.4.23 7.6.4.22 platform_class_type

The class of an individual platform instance - i.e. a common design from which platform instances are manufactured. This contains attributes that apply to all the platform instances of a class

### Table 7 Table - Attributes of IDLStruct platform_class_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name platform_name_type</td>
<td>The name of the platform (or class of platforms)</td>
</tr>
<tr>
<td>pre_identification identity_type [0..1]</td>
<td>The standard identification of the platform</td>
</tr>
<tr>
<td>pre_recognition short [0..1]</td>
<td>The discrete code representing the type of platform.</td>
</tr>
<tr>
<td>country_code country_code_type</td>
<td>The code representing the country of registration of the platform</td>
</tr>
<tr>
<td>«key» id assessment_objective_id_type</td>
<td>Unique identifier for the platform (within the scope of the sensor).</td>
</tr>
</tbody>
</table>

### Table 7.143 - Relations of IDLStruct platform_class_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: equipment equipment_type reference [0..*]</td>
<td>The equipment known to be associated with a platform</td>
</tr>
</tbody>
</table>

#### 7.6.4.23 platform_equipment_match_type

**Type: IDLStruct match_link_type**
Package: Sensor_Assessment
Represents the matching link between a platform and a constituent piece of equipment

Table 7.144 - Relations of IDLStruct platform_equipment_match_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: equipment equipment_match_type reference</td>
<td></td>
</tr>
</tbody>
</table>

7.5.4.25

platform_match_type

Type: IDLStruct match_type
Package: Sensor_Assessment

7.5.4.26

The representation of a match between a sensor track and a platform.

7.5.4.26

Table 7.145 - Relations of IDLStruct platform_mode_match_type
Association: **associated_sensor_track**

- **sensor_track_type** reference [1..*]
  - The set of sensor tracks that are all associated with the same platform instance and hence real world object under the hypothesis of this platform match.

Association: **activity**

- **platform_activity_type** reference [0..1]
  - The activity identified as being undertaken by the platform when matching the sensor track to it.

Association: **platform**

- **platform_class_type** reference
  - The platform being matched.

Aggregation: **mode_match**

- **platform_mode_match_type** [0..1]
  - The observable equipment modes matched by the sensor in determining the platform match.

Aggregation: **match**

- **subplatform_match_type** [0..*]
  - A hierarchical match from a (super) platform to a separable (sub) platform that it is potentially hosting or carrying as one of its constituent parts.

Aggregation: **equipment_match**

- **platform_equipment_match_type** [0..*]
  - A match between a platform and a constituent equipment.

### 7.6.4.25 **platform_mode_match_type**

- **Type:** IDLStruct match_link_type
- **Package:** Sensor_Assessment

Represents the matching link between a platform and a mode of a constituent piece of equipment.

#### Table 7.146 - Relations of IDLStruct platform_mode_match_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: <strong>mode</strong> parametric_mode_match_type reference</td>
<td></td>
</tr>
</tbody>
</table>

### 7.5.4.27 6.4.26 **platform_name_type**

- **Type:** string
- **Package:** Sensor_Assessment

The name or label for a platform or class of platforms. A platform being a discrete independently acting object in the real-world environment.

ElementTag: Length = 32

### 7.5.4.28 6.4.27 **platform_type**

- **Type:** IDLStruct platform_class_type
- **Package:** Sensor_Assessment

The representation of a platform that an assessment of sensor track data can match against.

#### Table 7.52141 - Attributes of IDLStruct platform_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform_name</td>
<td>The name of the platform (or class of platforms)</td>
</tr>
</tbody>
</table>

### 7.6.4.28 **reference_descriptor_type**

- **Type:** string
- **Package:** Sensor_Assessment
The representation of descriptor for a configuration reference data instance. Reference descriptor instances qualify the reference data instance (e.g. mode, equipment) and are a mechanism to specify aliases and other supporting information.

#### Table 7.93148 - Attributes of Class reference_descriptor_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>descriptor_name_type</td>
</tr>
<tr>
<td></td>
<td>The name of the descriptor for the mode</td>
</tr>
<tr>
<td>value</td>
<td>descriptor_value_type</td>
</tr>
<tr>
<td></td>
<td>The value of the descriptor for the mode</td>
</tr>
</tbody>
</table>

#### 7.5.4.307.6.4.29 reference_id_type

| Type:           | Class long                                 |
| Package:        | Sensor_Assessment                          |
| Notes:          | The unique identifier for a reference parameter, sequence or mode. |

#### 7.5.4.317.6.4.30 reference_mode_type

| Type:           | Class reference_type                      |
| Package:        | Sensor_Assessment                          |
| Notes:          | This class represents a label for a reference mode for a sensor tracks measurement parameter. Such tactically significant labels and their underlying data sets may be made available as an encyclopedic library. |

#### Table 7.94149 - Attributes of Class reference_mode_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>mode_name_type</td>
</tr>
<tr>
<td></td>
<td>The name or label of the mode.</td>
</tr>
</tbody>
</table>

#### Table 7.150 - Relations of Class reference_mode_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation:</td>
<td>descriptor reference_descriptor_type</td>
</tr>
<tr>
<td>[0..*]</td>
<td>The descriptors associated with the mode</td>
</tr>
</tbody>
</table>

#### 7.5.4.327.6.4.31 reference_parameter_match_type
**Open Architecture Radar Interface Standard (OARIS), v2.0**

**Type:** Class match_link_type  
**Package:** Sensor_Assessment

The representation of a match to a reference parameter for a sensor track

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: parameter_kind</td>
<td>The kind of the measured parameters that support the mode identification.</td>
</tr>
<tr>
<td>measurement_parameter_kind_type reference</td>
<td></td>
</tr>
<tr>
<td>Association: reference_parameter</td>
<td>The reference parameters matched by this mode identification</td>
</tr>
<tr>
<td>reference parameter_type_reference</td>
<td></td>
</tr>
</tbody>
</table>

7.6.4.32 reference_parameter_type

7.5.4.33 Type: Class reference_type

Package: Sensor_Assessment

This class represents a label for a reference value, set or distribution for a parameter. Such tactically significant labels and their underlying data sets may be made available as an encyclopedic library.

7.5.4.34 7.6.4.33 reference_type

Type: IDLStruct  
Package: Sensor_Assessment

A base class for reference data being compared with measurements in the assessment process.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>id reference_id_type</td>
<td>The unique identifier for the reference data. This may facilitate the retrieval of additional data outside the scope of this specification.</td>
</tr>
<tr>
<td>alert_id long</td>
<td>The identifier for an alerting or warning process associated with the matching of this reference data.</td>
</tr>
</tbody>
</table>

7.6.4.34 sensor_plot_equipment_equipment_assessment_type

7.5.4.35 Type: IDLStruct  
Package: Sensor_Assessment

The sensor subsystem's assessment of the equipment potentially matched by the sensor plot's measurement parameters.

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: best equipment_match_type_reference [0..1]</td>
<td>The equipment assessed by the sensor as being the best match for the sensor plot.</td>
</tr>
<tr>
<td>Association: sensor_plot sensor_plot_type_reference</td>
<td>The sensor plot to which the assessment refers. The plot instance must contain a plot_id attribute.</td>
</tr>
<tr>
<td>Aggregation: equipment_equipment_match_type [0..*]</td>
<td>The equipment assessed as potentially being represented by the sensor plot.</td>
</tr>
</tbody>
</table>
### 7.6.4.35 sensor_plot_mode_assessment_type

**Type:** IDLStruct  
**Package:** Sensor_Assessment  

The sensor subsystem's assessment of the modes potentially matched by the sensor plot's measurement parameters.

#### Table 7.154 - Relations of IDLStruct sensor_plot_mode_assessment_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: sensor_plot sensor_plot_type reference</td>
<td>The sensor plot to which the assessment refers. The plot instance must contain a plot_id attribute.</td>
</tr>
<tr>
<td>Association: best parametric_mode_match_type reference [0..1]</td>
<td>The mode assessed as most likely by the sensor.</td>
</tr>
<tr>
<td>Aggregation: mode parametric_mode_match_type [0..*]</td>
<td>The modes assessed as candidates for the sensor plot.</td>
</tr>
</tbody>
</table>

### 7.6.4.36 sensor_plot_platform_assessment_type

**Type:** IDLStruct  
**Package:** Sensor_Assessment  

The sensor subsystem's assessment of the platforms potentially matched by the sensor plot's measurement parameters.

#### Table 7.155 - Relations of IDLStruct sensor_plot_platform_assessment_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: sensor_plot sensor_plot_type reference</td>
<td>The sensor plot to which the assessment refers. The plot instance must contain a plot_id attribute.</td>
</tr>
<tr>
<td>Association: best platform_match_type reference [0..1]</td>
<td>The platform assessed by the sensor as the best match for the sensor plot.</td>
</tr>
<tr>
<td>Aggregation: platform platform_match_type [0..*]</td>
<td>The platforms assessed as potentially being represented by the sensor plot.</td>
</tr>
</tbody>
</table>

### 7.6.4.37 sensor_track_equipment_assessment_type

**Type:** IDLStruct  
**Package:** Sensor_Assessment  

A representation of an assessment of the equipment that potentially correspond to a sensor track.

#### Table 7.156 - Relations of IDLStruct sensor_track_equipment_assessment_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: sensor_track sensor_track_type reference</td>
<td>The equipment assessed as most likely to correspond to the sensor track.</td>
</tr>
<tr>
<td>Association: selected equipment_match_type reference [0..1]</td>
<td>The match selected as the authoritative assessment by command.</td>
</tr>
<tr>
<td>Association: equipment equipment_match_type reference [0..*]</td>
<td>An item of equipment that has been assessed as a possible match for the sensor track.</td>
</tr>
</tbody>
</table>

### 7.6.4.38 sensor_track_mode_assessment_type

**Type:** Class  
**Package:** Sensor_Assessment  

The representation of the state of the assessment of a sensor track's possible identified modes.
Table 7.157 - Relations of Class sensor_track_mode_assessment_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: sensor_track sensor_track_type reference</td>
<td>The sensor track to which the mode assessment relates</td>
</tr>
<tr>
<td>Association: selected parametric_mode_match_type reference [0..1]</td>
<td>The match that has been authoritatively selected as the mode relating to the sensor track.</td>
</tr>
<tr>
<td>Association: best parametric_mode_match_type reference [0..1]</td>
<td>The sensor's best match</td>
</tr>
<tr>
<td>Aggregation: mode parametric_mode_match_type [0..*]</td>
<td>The parametric modes matched in the assessment of the sensor track's measurement parameters.</td>
</tr>
</tbody>
</table>

7.6.4.39 sensor_track_platform_assessment_type

7.5.4.40 Type: IDLStruct
Package: Sensor_Assessment
A representation of an assessment of the platforms that potentially correspond to a sensor track.

Table 7.158 - Relations of IDLStruct sensor_track_platform_assessment_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: sensor_track sensor_track_type reference</td>
<td>The sensor track corresponding to the platform assessment</td>
</tr>
<tr>
<td>Association: selected platform_match_type reference [0..1]</td>
<td>The match selected as the authoritative assessment by command</td>
</tr>
<tr>
<td>Association: best_match platform_match_type reference [0..1]</td>
<td>The platform assessed as most likely to correspond to the sensor track</td>
</tr>
<tr>
<td>Association: platform platform_match_type reference [0..*]</td>
<td>A platform that has been assessed as a possible match for the sensor track</td>
</tr>
</tbody>
</table>

7.6.4.40 subplatform_match_type

7.5.4.41 Type: IDLStruct
Package: Sensor_Assessment

Table 7.160 - Relations of IDLStruct subplatform_match_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: subplatform platform_match_type reference</td>
<td>A (sub) platform (potentially independently operating real-world object) that is currently contained by the platform. Examples include a helicopter that is currently on a ship's landing deck.</td>
</tr>
</tbody>
</table>
Parent Package: Sensor_Domain
This package provides a data model to describe supplementary parameters that a sensor can provide about a sensor track. The approach is to be agnostic to any specific type of measurement that a sensor may make. Rather, classes are provided that allow the sensor to describe the parameters that it supports and then to describe the measurements that it has made of those parameters. Measurement can be treated as a single instance, a continuous range or a discrete set; the quantity can be scalar, vector, discrete or qualitative; and confidence values can be supplied where appropriate as can accuracy estimates.

Supplementary measurements as relating to a sensor track.
Figure 7.47.71 Supplementary Measurement - tracks (Class diagram)
Base types for supplementary measurements.
Figure 7.49.73 Supplementary Measurement - distributions (Class diagram)
Open Architecture Radar Interface Standard (OARIS), v2.0
Figure 7.50.74 Supplementary_Measurement - continuous distributions (Class diagram)
Figure 7.5 Supplementary_Measurement - base types (Class diagram)
7.5.5.1 continuous_measurement_type

7.6.5.1 continuous_measurement_type
Type: IDLStruct parameter_distribution_type
Package: Supplementary_Measurement
A continuous representation of a parameter measurement value.

7.5.5.2 discrete_measurement_type
7.6.5.2 discrete_measurement_type
Type: Class single_measurement_type
Package: Supplementary_Measurement
This represents a parameter which takes discrete values.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>value long</td>
<td>The discrete value of the measurement</td>
</tr>
</tbody>
</table>

7.6.5.3 discrete_order_type
7.5.5.3 Type: IDLEnum
Package: Supplementary_Measurement
The ordering semantics of a set of measurements of a parameter.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORDER_INCREASING_BY_VALUE</td>
<td>The measurements are ordered by increasing value such that the smallest value is first and the largest value is last.</td>
</tr>
<tr>
<td>ORDER_DECREASING_BY_VALUE</td>
<td>The measurements are ordered by decreasing value such that the smallest value is last and the largest value is first.</td>
</tr>
<tr>
<td>ORDER_OLDEST_FIRST</td>
<td>The measurements are ordered by age such that the value received first is first and the latest value is last.</td>
</tr>
<tr>
<td>ORDER_MOST_RECENT_FIRST</td>
<td>The measurements are ordered by age such that the value received first is last and the latest value is first.</td>
</tr>
<tr>
<td>ORDER_NOT_SPECIFIED</td>
<td>The ordering is not specified as it is not semantically meaningful.</td>
</tr>
</tbody>
</table>
The measurements are ordered by decreasing value such that the smallest value is last and the largest value is first.

The measurements are ordered by age such that the value received first is first and the latest value is last.

The measurements are ordered by age such that the value received first is last and the latest value is first.

The ordering is not specified as it is not semantically meaningful.

7.5.5.4 discrete_set_measurement_type
Type: IDLStruct parameter_distribution_type
Package: Supplementary_Measurement
The values of the measurement parameter follow a discrete distribution

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>repeating</td>
<td>Whether the elements within the discrete distribution repeat (in the same order)</td>
</tr>
<tr>
<td>order</td>
<td>The semantics of the ordering of the elements of the discrete distribution</td>
</tr>
<tr>
<td>known_set</td>
<td>Whether the elements within the discrete distribution correspond to a known set of measurement values</td>
</tr>
</tbody>
</table>

Table 7.106 - Relations of IDLStruct discrete_set_measurement_type
Connector: Aggregation: element measurement_element_type [1..*]
A discrete element within the discrete distribution

7.6.5.5 distribution_mode_type
7.5.5.5 Type: IDLStruct
Package: Supplementary_Measurement
The distribution of one mode independently contributing to a multi-modal distribution.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>proportion</td>
<td>The proportion that this mode contributes to the overall distribution. The sum of all the modes equals 1.</td>
</tr>
</tbody>
</table>

Table 7.106 - Relations of IDLStruct distribution_mode_type
Connector: Aggregation: distribution_pdf_measurement_type
The distribution for this mode,
7.6.5.6 distribution_parameter_measurement_type

Type: IDLStruct

Package: Supplementary_Measurement

The measurement estimation of a (sensor defined) parameter describing a measurement parameter’s distribution

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name distribution_parameter_name_type</td>
<td>The name of a parameter describing a distribution</td>
</tr>
</tbody>
</table>
Table 7.168 - Relations of IDLStruct distribution_parameter_measurement_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: value single_measurement_type</td>
<td>The measured value for a parameter describing a distribution.</td>
</tr>
</tbody>
</table>

7.6.5.7 distribution_parameter_name_type

7.5.5.7 Type: Class string

Package: Supplementary_Measurement

The name of a sensor defined probability density function.

ElementTag: Length = 20

7.6.5.8 measurement_drift_type

7.5.5.8 Type: IDLStruct

Package: Supplementary_Measurement

Describes how a measurement varies with time (on a time-scale longer than that described by modulation measurements).

Table 7.102169 - Attributes of IDLStruct measurement_drift_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>.kind measurement_variation_kind_type</td>
<td>The qualitative measure of the kind of drift detected.</td>
</tr>
<tr>
<td>confidence confidence_type [0..1]</td>
<td>The sensor's confidence in identifying the kind of drift.</td>
</tr>
<tr>
<td>.fit double [0..1]</td>
<td>Sample size independent measure of the closeness by which the measurement sample fit the model of the identified kind of drift.</td>
</tr>
</tbody>
</table>

Table 7.170 - Relations of IDLStruct measurement_drift_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: max_rate scalar_measurement_type [0..1]</td>
<td>The maximum rate of change of the measurement detected (in the units of the measurement's kind per second).</td>
</tr>
<tr>
<td>Aggregation: mean_rate scalar_measurement_type</td>
<td>The mean rate of change of the measurement detected (in the units of the measurement's kind per second).</td>
</tr>
<tr>
<td>Aggregation: repetition_period scalar_measurement_type [0..1]</td>
<td>The time for the drift behavior to repeat itself.</td>
</tr>
<tr>
<td>Aggregation: min_rate scalar_measurement_type [0..1]</td>
<td>The minimum rate of change of the measurement detected (in the units of the measurement's kind per second).</td>
</tr>
</tbody>
</table>

7.6.5.9 measurement_element_type

7.5.5.9 Type: IDLStruct

Package: Supplementary_Measurement

The representation of an element within a discrete distribution

Table 7.103171 - Attributes of IDLStruct measurement_element_type
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>The number of times the parameter measured has corresponded to this</td>
</tr>
<tr>
<td></td>
<td>element in a row. That is since another element was measured.</td>
</tr>
<tr>
<td>total_count</td>
<td>The total number of times this element has been measured with the</td>
</tr>
<tr>
<td></td>
<td>discrete distribution for this parameter value for the sensor track.</td>
</tr>
</tbody>
</table>

Table 7.172 - Relations of IDLStruct measurement_element_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation:</td>
<td>measurement single_measurement_type</td>
</tr>
<tr>
<td></td>
<td>The measurement of the element within the discrete distribution.</td>
</tr>
</tbody>
</table>

7.5.5.10 6.5.10 measurement_interval_type

Type: IDLStruct continuous_measurement_type
Package: Supplementary_Measurement
The representation of parameter measurement values that are distributed within a bounded interval

Table 7.104173 - Attributes of IDLStruct measurement_interval_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>uniform</td>
<td>The measurement values are uniformly distributed within the bounded</td>
</tr>
<tr>
<td></td>
<td>interval</td>
</tr>
</tbody>
</table>

Table 7.174 - Relations of IDLStruct measurement_interval_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation:</td>
<td>drift measurement_drift_type [0..1]</td>
</tr>
<tr>
<td></td>
<td>A qualitative description of how the parameter measurement changes</td>
</tr>
<tr>
<td></td>
<td>over time</td>
</tr>
<tr>
<td>Aggregation:</td>
<td>min_value scalar_measurement_type</td>
</tr>
<tr>
<td></td>
<td>The lower bound of the parameter measurements, which may be</td>
</tr>
<tr>
<td></td>
<td>extrapolated from actual sample measurements received or processed.</td>
</tr>
<tr>
<td>Aggregation:</td>
<td>mean_value scalar_measurement_type</td>
</tr>
<tr>
<td></td>
<td>The mean value of the measurement within the interval.</td>
</tr>
<tr>
<td>Aggregation:</td>
<td>max_value scalar_measurement_type</td>
</tr>
<tr>
<td></td>
<td>The upper bound of the parameter measurements, which may be</td>
</tr>
<tr>
<td></td>
<td>extrapolated from actual sample measurements received or processed.</td>
</tr>
<tr>
<td>Aggregation:</td>
<td>standard_deviation scalar_measurement_type [0..1]</td>
</tr>
<tr>
<td></td>
<td>The standard deviation of the measurement within the interval.</td>
</tr>
</tbody>
</table>

7.5.5.11 measurement_kind_id_type
7.5.5.11 Type: Class long
Package: Supplementary_Measurement
The unique identifier for describing kinds of measurements

**7.6.5.12 measurement_name_type**

**7.5.5.12 Type:** Class **string**  
**Package:** Supplementary_Measurement  
The name or label for a kind of measurement  
ElementTag: Length = 20

**7.6.5.13 measurement_parameter_kind_type**

**7.5.5.13 Type:** Class  
**Package:** Supplementary_Measurement  
Describes a kind of measurement parameters in terms of its meta-data (the information that applies to all measurement_parameter_type instances)

**Table 7.105175 - Attributes of Class measurement_parameter_kind_type**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name measurement_name_type</td>
<td>The name or label for the measurement.</td>
</tr>
<tr>
<td>id measurement_kind_id_type</td>
<td>The unique identifier for this kind of parameter.</td>
</tr>
<tr>
<td>units measurement_unit_type</td>
<td>The units of the measurement.</td>
</tr>
</tbody>
</table>

**7.6.5.14 measurement_parameter_set_name_type**

**7.5.5.14 Type:** IDLTypeDef **string**  
**Package:** Supplementary_Measurement  
Names of sets of parameters for sensor tracks  
ElementTag: Length = 32

**7.6.5.15 measurement_parameter_status_type**

**7.5.5.15 Type:** IDLEnum  
**Package:** Supplementary_Measurement  
The measurement status of the parameter

**Table 7.106 - Attributes of IDLEnum measurement_parameter_status_type**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CURRENT</td>
<td>The parameter is currently subject to measurement for this sensor track</td>
</tr>
<tr>
<td>MISSING</td>
<td>The parameter temporarily cannot be measured for this sensor track</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>---------------------------</td>
<td></td>
</tr>
<tr>
<td>«enum» NOT_PRESENT</td>
<td>The parameter can no longer be measured for this sensor track</td>
</tr>
<tr>
<td>«enum» NO_PARAMETER_STATUS</td>
<td>No statement is available regarding the parameter's measurement status for this sensor track</td>
</tr>
</tbody>
</table>

7.5.5.16 measurement-parameter-type

Type: IDLStruct

Package: Supplementary_Measurement

AThe measurement status of a parameter by a sensor for a sensor track.
### Table 7.107 - Attributes of IDLStructIDLEnum measurement_parameter_status_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>confidence.confidence_type {0..1}enum CURRENT</td>
<td>The probability that the parameter is currently subject to measurement corresponds to measurement labelled for this sensor track.</td>
</tr>
<tr>
<td>continuous boolean {0..1}enum MISSING</td>
<td>Indicates that the phenomenon being measured is in an enduring steady state and hence that complementary/orthogonal measurements of more detailed time-varying characteristics/phenomena are not present for this track. The parameter temporarily cannot be measured for this sensor track.</td>
</tr>
<tr>
<td>count long {0..1}enum NOT_PRESENT</td>
<td>The number of coherent discrete measurements of this quantity. If the sensor detects a qualitative change then the count is reset. The parameter can no longer be measured for this sensor track.</td>
</tr>
<tr>
<td>status measurement_parameter_status_type {enum} NO_PARAMETER_STATUS</td>
<td>The No statement is available regarding the parameter’s measurement status for this sensor track.</td>
</tr>
<tr>
<td>time_of_information time_type</td>
<td>The time at which the parameter was measured.</td>
</tr>
<tr>
<td>intentional boolean {0..1}</td>
<td>Whether or not the phenomenon being measured by this parameter is considered to be design feature of the equipment causing the phenomenon.</td>
</tr>
</tbody>
</table>

### 7.6.5.16 measurement_parameter_type

**Type:** IDLStruct

**Package:** Supplementary_Measurement

A measurement of a parameter by a sensor for a sensor track.

### Table 7.177 - Attributes of IDLStruct measurement_parameter_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>confidence.confidence_type {0..1}</td>
<td>The probability that the measurement corresponds to measurement labelled for the sensor track.</td>
</tr>
<tr>
<td>continuous boolean {0..1}</td>
<td>Indicates that the phenomenon being measured is in an enduring steady state and hence that complementary/orthogonal measurements of more detailed time-varying characteristics/phenomena are not present for this track.</td>
</tr>
<tr>
<td>count long {0..1}</td>
<td>The number of coherent discrete measurements of this quantity. If the sensor detects a qualitative change then the count is reset.</td>
</tr>
</tbody>
</table>
**Table 7.178 - Relations of IDLStruct measurement_parameter_type**

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: kind</td>
<td>measurement_parameter_kind_type</td>
</tr>
<tr>
<td>reference</td>
<td>The description of the parameter and the unique identifier (within the scope of a sensor track) of a measurement parameter instance.</td>
</tr>
<tr>
<td>Aggregation: distribution</td>
<td>parameter_distribution_type</td>
</tr>
<tr>
<td>[0..1]</td>
<td>The representation of the statistical distribution of the measurement parameter.</td>
</tr>
<tr>
<td>Aggregation: sample_range</td>
<td>sample_range_type [0..1]</td>
</tr>
<tr>
<td></td>
<td>The range in which samples contributing to the measurement have occurred.</td>
</tr>
</tbody>
</table>

**7.6.5.17 measurement_unit_type**

**7.5.5.17 Type:** IDLEnum  
**Package:** Supplementary_Measurement  
The units used to quantify the measurement values and accuracies.

**Table 7.108179 - Attributes of IDLEnum measurement_unit_type**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«enum» DIMENSIONLESS</td>
<td>There are no units as the quantity is a dimensionless value</td>
</tr>
<tr>
<td>«enum» DECIBEL</td>
<td>units are in decibels to measure amplitudes</td>
</tr>
<tr>
<td>«enum» DECIBELS_PER_SECOND</td>
<td>units are in decibels per second to measure change in amplitude with time.</td>
</tr>
<tr>
<td>«enum» Hertz</td>
<td>units are in Hertz to measure frequencies</td>
</tr>
<tr>
<td>«enum» HERTZ_PER_SECOND</td>
<td>Units are in Hertz per second to measure change in frequency with time</td>
</tr>
<tr>
<td>«enum» METER</td>
<td>Units are meters</td>
</tr>
<tr>
<td>«enum» METERS_PER_SECOND</td>
<td>Units are in meters per second to measure speeds</td>
</tr>
<tr>
<td>«enum» OTHER_UNIT</td>
<td>Another unit is used to quantify the measurements and accuracies</td>
</tr>
<tr>
<td>«enum» RADIUS</td>
<td>Units are in radians to measure angles</td>
</tr>
<tr>
<td>«enum» RADIANS_PER_SECOND</td>
<td>Units are in radians per seconds to measure the change in angles with time</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>«enum» SECOND</td>
<td>Units are in seconds to measure time or intervals.</td>
</tr>
<tr>
<td>«enum» SECONDS_PER_SECOND</td>
<td>Units are in seconds per second to measure the change in regular intervals over time.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«enum» RADIANS_PER_SECOND</td>
<td>Units are in radians per seconds to measure the change in angles with time</td>
</tr>
<tr>
<td>«enum» SECOND</td>
<td>Units are in seconds to measure time or intervals.</td>
</tr>
<tr>
<td>«enum» SECONDS_PER_SECOND</td>
<td>Units are in seconds per second to measure the change in regular intervals over time.</td>
</tr>
</tbody>
</table>

### 7.6.5.18 measurement_variation_kind_type

This class is used to specify a qualitative description of change in a parameter measurement value over time within a distribution. The characteristic quantitative values of the variation pattern can be represented by other related measurement parameters.

#### Table 7.109180 - Attributes of Class measurement_variation_kind_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«enum» DECREASING</td>
<td>The value is decreasing monotonically; a minimum value is not yet determined</td>
</tr>
<tr>
<td>«enum» DECREASING_WITH_WRAP</td>
<td>The value decreases monotonically until it reaches a minimum value at which point it wraps or resets to a maximum value.</td>
</tr>
<tr>
<td>«enum» DRIFT_NOT_SPECIFIED</td>
<td>The drift behavior is not specified</td>
</tr>
<tr>
<td>«enum» INCREASING</td>
<td>The is increasing monotonically; a maximum value is not yet determined</td>
</tr>
<tr>
<td>«enum» INCREASING_WITH_WRAP</td>
<td>The value increases monotonically until it reaches a maximum asymptotic value.</td>
</tr>
<tr>
<td>«enum» INCREASING_WITH_WRAP</td>
<td>The value increases monotonically until it reaches a maximum asymptotic value.</td>
</tr>
</tbody>
</table>
### Table 7.140181 - Attributes of IDLStruct modulation_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>kind</td>
<td>The qualitative measure of the kind of modulation detected.</td>
</tr>
</tbody>
</table>

Table 7.182 - Relations of IDLStruct modulation_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: modulation_amplitude scalar_measurement_type [0..1]</td>
<td>The amplitude of the modulation of the parameter being measured.</td>
</tr>
<tr>
<td>Aggregation: mean_value scalar_measurement_type</td>
<td>The mean value of the measurement that is subject to modulation.</td>
</tr>
<tr>
<td>Aggregation: modulation_frequency scalar_measurement_type [0..1]</td>
<td>The frequency of the modulation of the parameter being measured.</td>
</tr>
</tbody>
</table>

### 7.6.5.20 multi modal_measurement_type

Type: IDLStruct continuous_measurement_type

Package: Supplementary_Measurement

The representation of parameter measurement values that have a multi-modal distribution normal
### Table 7.183 - Relations of IDLStruct multi_modal_measurement_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: mode_distribution_mode_type [1..*]</td>
<td>A mode within a multi-mode distribution for the parameter measurement</td>
</tr>
</tbody>
</table>

#### 7.6.5.21 normal_measurement_type

**Type:** IDLStruct pdf_measurement_type  
**Package:** Supplementary_Measurement  
The representation of a measurement parameter that is normally distributed.

#### 7.6.5.22 parameter_distribution_type

**Type:** IDLStruct  
**Package:** Supplementary_Measurement  
A representation of the statistical distribution of a parameter.

#### 7.6.5.23 parameter_id_type

**Type:** Class long  
**Package:** Supplementary_Measurement  
A representation of the statistical distribution of a parameter.

#### 7.6.5.23 parameter_id_type

**Type:** Class  
**Package:** Supplementary_Measurement  
The unique identifier for a measurement parameter.

#### 7.5.5.247.6.5.25 pdf_measurement_type

**Type:** IDLStruct continuous_measurement_type  
**Package:** Supplementary_Measurement  
The values of the parameter measurement are distributed according to a probability density function.

### Table 7.184 - Relations of IDLStruct pdf_measurement_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: standard_deviation single_measurement_type [0..1]</td>
<td>The standard deviation of values from the probability density function.</td>
</tr>
<tr>
<td>Aggregation: mean single_measurement_type</td>
<td>The mean (expected) value of the probability density function.</td>
</tr>
</tbody>
</table>

#### 7.5.5.25 pdf_name_type

**Type:** Class string  
**Package:** Supplementary_Measurement  
The name of a sensor defined probability density function.  
**Element Tag:** Length = 20

#### 7.5.5.26 plot_measurement_parameter_set_type

**Type:** IDLStruct  
**Package:** Supplementary_Measurement
The name of a sensor defined probability density function.
ElementTag: Length = 20

7.6.5.26 plot_measurement_parameter_set_type
Type: IDLStruct
Package: Supplementary_Measurement

A set of the measurement parameters relating to a sensor track. Subsystems form measurement parameters into sets for efficient information transfer to the CMS. A subsystem may chose the number and composition of these sets. A subsystem may place all measurements into a single set per track,
create multiple sets or create no sets and report measurement parameters individually instead. For a particular sensor track, measurement parameter names shall be unique across all measurement parameter set instances - i.e. sets shall be non-overlapping.

### Table 7.14185 - Attributes of IDLStruct plot_measurement_parameter_set_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the set of parameters</td>
</tr>
</tbody>
</table>

### Table 7.186 - Relations of IDLStruct plot_measurement_parameter_set_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: parameter measurement_parameter_type</td>
<td>The set of measurement parameters associated with the plot</td>
</tr>
</tbody>
</table>

### 7.6.5.27 poisson_measurement_type

**Type:** IDLStruct *pdf_measurement_type*

**Package:** Supplementary_Measurement

The parameter measurement follows a Poisson distribution

### 7.6.5.28 qualitative_measurement_type

**Type:** Class *single_measurement_type*

**Package:** Supplementary_Measurement

This describes a qualitative measure

### Table 7.142187 - Attributes of Class qualitative_measurement_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>descriptor</td>
<td>The descriptor for the qualitative measurement</td>
</tr>
</tbody>
</table>

### 7.5.5.297.6.5.29 sample_range_type

**Type:** IDLStruct

**Package:** Supplementary_Measurement

The inclusive range of samples sensed that contribute to the measurement value

### Table 7.143188 - Attributes of IDLStruct sample_range_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>min_value</td>
<td>The minimum value of a sample for the measurement</td>
</tr>
<tr>
<td>max_value</td>
<td>The maximum value of a sample for the measurement</td>
</tr>
</tbody>
</table>

### 7.6.5.30 scalar_measurement_type
7.5.5.30 Type: Class single_measurement_type
Package: Supplementary_Measurement
This class represents individual scalar measurements of parameter values.

Table 7.14189 - Attributes of Class scalar_measurement_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>accuracy</td>
<td>double [0..1]</td>
</tr>
<tr>
<td>value</td>
<td>double</td>
</tr>
</tbody>
</table>

7.6.5.31 Type: sensor_defined_pdf_measurement_type
Package: Supplementary_Measurement
The representation of a measurement of generalised probability density function whose definition can be instantiated by a sensor for extensibility.
Table 7.145190 - Attributes of IDLStruct sensor_defined_pdf_measurement_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name_pdf_name_type</td>
<td>The name of the probability density function</td>
</tr>
</tbody>
</table>

Table 7.191 - Relations of IDLStruct sensor_defined_pdf_measurement_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: parameter</td>
<td>The list of additional parameters required to describe the sensor defined probability density function</td>
</tr>
<tr>
<td>distribution_parameter_measurement_type</td>
<td></td>
</tr>
</tbody>
</table>

7.5.5.327.6.5.32 sequence_name_type

Type: Class string
Package: Supplementary_Measurement

To name a sequence

ElementTag: Length = 32

7.5.5.337.6.5.33 single_measurement_type

Type: IDLStruct parameter_distribution_type
Package: Supplementary_Measurement

A single discrete representation of a parameter measurement value.

Table 7.146192 - Attributes of IDLStruct single_measurement_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>confidence_type</td>
<td>The confidence in the parameter measurement value; this is the probability that the value and accuracy represent the true distribution of the physical effect they are labelled as measuring in the real world.</td>
</tr>
</tbody>
</table>

7.6.5.34 track_measurement_parameter_set_type

7.5.5.34 Type: IDLStruct
Package: Supplementary_Measurement

A set of the measurement parameters relating to a sensor track. Subsystems form measurement parameters into sets for efficient information transfer to the CMS. A subsystem may choose the number and composition of these sets. A subsystem may place all measurements into a single set per track, create multiple sets or create no sets and report measurement parameters individually instead.

For a particular sensor track, measurement parameter names shall be unique across all measurement parameter set instances - i.e. sets shall be non-overlapping.

Table 7.147193 - Attributes of IDLStruct track_measurement_parameter_set_type
Table 7.194 - Relations of IDLStruct track_measurement_parameter_set_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the set of parameters</td>
</tr>
</tbody>
</table>

Association: sensor_track sensor_track_type reference

Aggregation: parameter measurement_parameter_type [0..*] The parameter measurement for the element

Table 7.195 - Relations of IDLStruct track_measurement_parameter_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association: sensor_track sensor_track_type reference</td>
<td>The sensor track to which the measurement parameter relates</td>
</tr>
<tr>
<td>Aggregation: parameter measurement_parameter_type</td>
<td>The individual parameter</td>
</tr>
</tbody>
</table>

7.6.5.35 track_measurement_parameter_type

7.5.5.35 Type: IDLStruct

Package: Supplementary_Measurement

To represent parameter measurements for a sensor track reported individually

7.6.5.36 vector_measurement_type

7.5.5.36 Type: Class single_measurement_type

Package: Supplementary_Measurement

This class represents individual vector measurements of parameter values.
### Table 7.196 - Attributes of Class `vector_measurement_type`

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>covariance</strong></td>
<td>double [0..*] The covariance between the elements of the vector value in a 1-dimensional representation of the triangular matrix. The i,j element ((i \geq j)) of a covariance matrix for a vector of size (N) is at position (\text{sum}(0..j-1, N - k) + (i - j)). The covariance is zero length if not specified. AttributeTag: Length = 21</td>
</tr>
<tr>
<td><strong>value</strong></td>
<td>double [1..*] The vector values AttributeTag: Length = 6</td>
</tr>
</tbody>
</table>
open architecture radar interface standard (oaris), v2.0

figure 7.52.76 domain model - sensor plot (class diagram)

idlStruct

sensor_plot_type

+ position: polar_position_type
+ coordinate_specification: coordinate_specification_type
+ range_rate: speed_type [0..1]
+ range_qualification: range_qualification_type [0..1]
+ azimuth_qualification: azimuth_qualification_type
+ elevation_qualification: elevation_qualification_type [0..1]
+ range_rate_accuracy: speed_type [0..1]
+ simulation_status: boolean
+ strength: strength_type [0..1]
+ confidence: confidence_type [0..1]
+ time_of_plot: time_type
+ time_accuracy: duration_type [0..1]
+ additional_information: anonymous_blob_type
  + splash_spotting_area_id: splash_spotting_area_id_type [0..1]
  + jammer_indication: boolean
+ plots
+ plot_summary_type

idlStruct

plot_summary_type

+ range_bounds: range_interval_type [0..1]
+ azimuth_bounds: azimuth_interval_type [0..1]
+ elevation_bounds: elevation_interval_type [0..1]
+ time_interval: absolute_duration_type
+ cell_strength
+ plots
+ distribution

idlUnion

distribution

+ plot_distribution_type

idlStruct

plot_distribution_type

+ range_cell_count: short
+ azimuth_cell_count: short
+ elevation_cell_count: short

idlStruct

sensor_plot_set_type

+ position: polar_position_type
+ coordinate_specification: coordinate_specification_type
+ range_rate: speed_type [0..1]
+ range_qualification: range_qualification_type [0..1]
+ azimuth_qualification: azimuth_qualification_type
+ elevation_qualification: elevation_qualification_type [0..1]
+ range_rate_accuracy: speed_type [0..1]
+ simulation_status: boolean
+ strength: strength_type [0..1]
+ confidence: confidence_type [0..1]
+ time_of_plot: time_type
+ time_accuracy: duration_type [0..1]
+ additional_information: anonymous_blob_type
  + splash_spotting_area_id: splash_spotting_area_id_type [0..1]
  + jammer_indication: boolean
+ plots

idlTypedef

plot_id_type

unsigned short

idlTypedef

strength_type

unsigned long

idlTypedef

plot_summary_type

open architecture radar interface standard (oaris), v2.0

255
Figure 7.77 Domain Model - Supporting Types (Class diagram)
7.5.6.1  error_distribution_kind_type
Type: Class unsigned long IDLEnum
Package: Plot_Reporting

Identifier for a plot, unique within a given sensor. Such plot ids, should not be reused between sensor subsystem restarts.

The class models the kinds of error distribution supported for sensors accuracy.

Table 7.11: Attributes of Class plot_idIDLEnum error_distribution_kind_type
7.5.6.2 sensor_plot_set_type
Type: Class
Package: Plot_Reporting
Set of one or more sensor plots.

Table 7. Attributes of Class sensor_plot_set_type

7.5.6.3 sensor_plot_type
Type: Class
Package: Plot_Reporting
One plot from a sensor.

The additional_information attribute is used for characteristics of the plot that are specific to certain sensors, and therefore not in the general plot type, for example MTI or range rate.

Table 7. Attributes of Class sensor_plot_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>plot_id: plot_id_type [0..1]</td>
<td>A unique identifier for the plot within the scope of the sensor. This attribute is optional as not all sensors need to provide such an identifier for each plot.</td>
</tr>
<tr>
<td>position: position_coordinate_type</td>
<td>The position of the plot. This is the mean, central position. Note the qualification attributes, which give information on accuracy and spread estimates. The error has a Gaussian distribution with zero mean and stated standard deviation.</td>
</tr>
<tr>
<td>coordinate_specification</td>
<td>This attribute defines the characteristics of the coordinate system used.</td>
</tr>
<tr>
<td>range_qualification: range_qualification_type [0..1]</td>
<td>A measure of the spread and accuracy of the plot in range. This is optional as not all sensors measure range.</td>
</tr>
<tr>
<td>azimuth_qualification: azimuth_qualification_type</td>
<td>A measure of the spread and accuracy of the plot in azimuth.</td>
</tr>
<tr>
<td>elevation_qualification: elevation_qualification_type [0..1]</td>
<td>A measure of the spread and accuracy of the plot in elevation. This is optional as not all sensors measure elevation.</td>
</tr>
<tr>
<td>simulation_status: boolean</td>
<td>If true, the plot is simulated. See also simulation support services within this standard. The error has a uniform distribution with zero mean and stated standard deviation.</td>
</tr>
<tr>
<td>strength: strength_type [0..1]</td>
<td>The signal strength of the plot. This attribute is optional as not all sensors measure a quantity which has equivalence to strength. The error has a symmetric triangular distribution with zero mean and stated standard deviation.</td>
</tr>
</tbody>
</table>
### Attribute | Notes
---|---
**time_of_plot** | The time at which the plot was measured.

**additional_information**
- **anonymous_blob_type**: `IDLEnum`
  - **UNDEFINED DISTRIBUTION**
    - Potentially classified information about the plot, which may be used in a system-specific way to distribute information about a plot to other subsystems. Further information about this attribute, including layout semantics, is outside of the scope of this interface standard. The distribution of the error is not defined.

### 7.6.6.2 plot_distribution_type
**Type**: `IDLStruct`
**Package**: `Plot_Reporting`

This class encapsulates the strength of the plot over a grid of higher-resolution range cells. The spatial extent of the distribution in range, azimuth and elevation is defined by the spread attribute of the relevant qualification attribute. The sequence of strength values represents a 3D array over range (inner iteration), azimuth and elevation (outer iteration). The cell at logical index `i,j,k` for range, azimuth and elevation respectively is at index:

\[
i + i \times \text{range_cell_count} + k \times \text{azimuth_cell_count} \times \text{range_cell_count}
\]

The cell indexed zero represents the lowest values of range, azimuth or elevation and cells are equally spaced such that the cell at logical index `i,j,k` for range, azimuth and elevation respectively corresponds to:

- \[\min \text{ range} + i \times \text{range_spread} / \text{range_cell_count} < \text{range} < \min \text{ range} + (i + 1) \times \text{range_spread} / \text{range_cell_count}\]
- \[\min \text{ azimuth} + i \times \text{azimuth_spread} / \text{azimuth_cell_count} < \text{azimuth} < \min \text{ azimuth} + (i + 1) \times \text{azimuth_spread} / \text{azimuth_cell_count}\]
- \[\min \text{ elevation} + i \times \text{elevation_spread} / \text{elevation_cell_count} < \text{elevation} < \min \text{ elevation} + (i + 1) \times \text{elevation_spread} / \text{elevation_cell_count}\]

where \(\text{range_spread} = \text{range_qualification.spread}\) for the plot and range, azimuth and elevation are the mean coordinates reported for the plot.

If no spread is defined for a plot in the range, azimuth or elevation qualification then it is only valid for there to be a count of one defined for that dimension.

### Table 7.198 - Attributes of IDLStruct plot_distribution_type

| Attribute | Notes |
---|---|
**splash_spotting_area_id**
- **splash_spotting_area_id_type**: `[0..1]`
- **range_cell_count**: short
  - Indicates which splash spotting area the plot refers to - if any - hence it is optional. The number of cells in the range dimension. Only one cell is valid if no range spread is defined.

**jammer_indication**: boolean
- **azimuth_cell_count**: short
  - Indication whether or not a plot is from a source of jamming. The number of cells in the azimuth dimension. Only one cell is valid if no azimuth spread is defined.

**elevation_cell_count**: short
- The number of cells in the elevation dimension. Only one cell is valid if no elevation spread is defined.
Table 7.199 - Relations of IDLStruct plot_distribution_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: cell_strength</td>
<td>The strength of plot signal in a higher resolution spatial cell</td>
</tr>
</tbody>
</table>

7.6.6.3 plot_filter_parameters_type

Type: IDLStruct

Package: Plot_Reporting

The criteria that must all be met for a plot to pass the filter. The filter attributes are applied with and-wise logic. For or-wise logic define multiple filter objects.

Table 7.200 - Attributes of IDLStruct plot_filter_parameters_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode filter_mode_type</td>
<td>The mode in which the plots are filtered.</td>
</tr>
<tr>
<td>area area_2d_type [0..1]</td>
<td>An area which is optionally part of the filter.</td>
</tr>
<tr>
<td>height height_interval_type [0..1]</td>
<td>The height values that are optionally part of the filter.</td>
</tr>
<tr>
<td>is_inclusive boolean</td>
<td>If true, tracks that pass the filter are included in transmission and/or reception dependent upon the mode attribute. Otherwise, they are excluded.</td>
</tr>
<tr>
<td>higher_values boolean</td>
<td>When true, for real-valued criteria, plots meet the criteria of the filter if the plot's value is equal to or higher than the corresponding filter criteria value. Otherwise, the criteria is met if equal to or lower.</td>
</tr>
</tbody>
</table>

Table 7.201 - Relations of IDLStruct plot_filter_parameters_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: plot_attributes</td>
<td>The filter criteria for plots that relate to characteristics of the plots themselves.</td>
</tr>
<tr>
<td>Aggregation: track_attributes</td>
<td>The filter criteria for plots that relate to the tracks to which the plots contribute.</td>
</tr>
</tbody>
</table>

7.6.6.4 plot_id_type

7.5.6.4 Type: Class

Package: Plot_Reporting

Identifier for a plot, unique within a given sensor. Such plot ids should not be reused between sensor subsystem restarts.

7.6.6.5 plot_level_filter_attributes_type

Type: IDLStruct

Package: Plot_Reporting

The plot-level criteria which the plot attributes must pass in order to pass the filter.

Table 7.202 - Attributes of IDLStruct plot_level_filter_attributes_type
7.6.6.6 plot_summary_type
Type: IDLStruct
Package: Plot_Reporting
The class provides a summary of plots found by the sensor in a region of the environment. Objects expected to be in the region for which there is no corresponding plot have not been detected by sensor, therefore missed measurements can be identified from this information.

Table 7.203 - Attributes of IDLStruct plot_summary_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>range_bounds</td>
<td>The bounds of the region being summarized in range. If omitted the region in unbounded in range.</td>
</tr>
<tr>
<td>azimuth_bounds</td>
<td>The bounds of the region being summarized in azimuth. If omitted the region in unbounded in azimuth.</td>
</tr>
<tr>
<td>elevation_bounds</td>
<td>The bounds of the region being summarized in elevation. If omitted the region in unbounded in elevation.</td>
</tr>
<tr>
<td>time_interval</td>
<td>The period of time during which the sensor sensed the region</td>
</tr>
</tbody>
</table>

Table 7.204 - Relations of IDLStruct plot_summary_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association:</td>
<td>plots sensor_plot_type reference [0..*] The set of plots found in the region</td>
</tr>
</tbody>
</table>

7.6.6.7 processing_capability_type
Type: IDLTypeDef
Package: Plot_Reporting
Encapsulates a category of sensor processing capability. The set of known categories of sensor processing is defined on an implementation specific basis.
ElementTag: Length = 20

7.6.6.8 sensor_calibration_model_type
Type: IDLStruct
Package: Plot_Reporting
This class models the residual global sensor error estimate after calibration
Table 7.205 - Attributes of IDLStruct sensor_calibration_model_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>azimuth_error</td>
<td>Residual error in azimuth to one standard deviation</td>
</tr>
<tr>
<td>elevation_error</td>
<td>Residual error in elevation to one standard deviation</td>
</tr>
<tr>
<td>orientation_error_distribution</td>
<td>The statistical distribution of the azimuth and elevation errors.</td>
</tr>
<tr>
<td>range_error</td>
<td>Residual error in range to one standard deviation</td>
</tr>
<tr>
<td>range_rate_error</td>
<td>Residual error in range rate to one standard deviation</td>
</tr>
<tr>
<td>signal_error_distribution</td>
<td>The statistical distribution of the range and range rate errors.</td>
</tr>
<tr>
<td>stern_offset_error</td>
<td>Residual error in offset of the sensor bore-sight origin on the stern-bow axis to one standard deviation</td>
</tr>
<tr>
<td>port_offset_error</td>
<td>Residual error in offset of the sensor bore-sight origin on the port-starboard axis to one standard deviation</td>
</tr>
<tr>
<td>mast_offset_error</td>
<td>Residual error in offset of the sensor bore-sight origin on the mast-keel axis to one standard deviation</td>
</tr>
<tr>
<td>offset_error_distribution</td>
<td>The statistical distribution of the sensor origin offset errors.</td>
</tr>
</tbody>
</table>

7.6.6.9 sensor_plot_set_type
Type: Class
Package: Plot_Reporting
Set of one or more sensor plots.

Table 7.206 - Relations of Class sensor_plot_set_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: plots sensor_plot_type [0..*]</td>
<td>The plots in the sensor plot set</td>
</tr>
</tbody>
</table>

7.6.6.10 sensor_plot_type
Type: Class
Package: Plot_Reporting
One plot from a sensor, a plot being a measurement estimate of an object's state in terms of location, motion and optionally size at a particular moment in time.

Table 7.207 - Attributes of Class sensor_plot_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«key» plot_id</td>
<td>A unique identifier for the plot within the scope of the sensor. This attribute is mandatory so that a sensor's plot summary can refer to published plots. AttributeTag: Issue =</td>
</tr>
<tr>
<td><strong>Attribute</strong></td>
<td><strong>Notes</strong></td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>position polar_position_type</strong></td>
<td>The position of the plot in polar coordinates (measurements assumed to be relative to a particular sensor position). This is the mean, central position. Note the qualification attributes, which give information on accuracy and spread estimates. AttributeTag: Issue =</td>
</tr>
<tr>
<td><strong>coordinate_specification</strong></td>
<td>This attribute defines the characteristics of the coordinate system used</td>
</tr>
<tr>
<td><strong>range_rate speed_type [0..1]</strong></td>
<td>The speed of the object detected along the line-of-sight of the sensor; positive values for an object receding from the sensor. Doppler processing can derive this value.</td>
</tr>
<tr>
<td><strong>range_qualification range_qualification_type [0..1]</strong></td>
<td>A measure of the spread and accuracy of the plot in range. This is optional as not all sensors measure range.</td>
</tr>
<tr>
<td><strong>azimuth_qualification azimuth_qualification_type</strong></td>
<td>A measure of the spread and accuracy of the plot in azimuth.</td>
</tr>
<tr>
<td><strong>elevation_qualification elevation_qualification_type [0..1]</strong></td>
<td>A measure of the spread and accuracy of the plot in elevation. This is optional as not all sensors measure elevation.</td>
</tr>
<tr>
<td><strong>range_rate_accuracy speed_type [0..1]</strong></td>
<td>A measure of the accuracy of the plot in range rate equal to one standard deviation of uncertainty. This is optional as not all sensors measure range rate. Note that for rigid objects a continuous spread in the measurement of range rate is not expected. AttributeTag: Issue =</td>
</tr>
<tr>
<td><strong>simulation_status boolean</strong></td>
<td>If true, the plot is simulated. See also simulation support services within this standard.</td>
</tr>
<tr>
<td><strong>strength strength_type [0..1]</strong></td>
<td>The signal strength of the plot. This attribute is optional as not all sensors measure a quantity which has equivalence to strength.</td>
</tr>
<tr>
<td><strong>confidence confidence_type [0..1]</strong></td>
<td>The probability that the plot represents a true object of interest as opposed to clutter, noise or other false objects. AttributeTag: Issue =</td>
</tr>
<tr>
<td><strong>time_of_plot time_type</strong></td>
<td>The time at which the plot was measured.</td>
</tr>
<tr>
<td><strong>time_accuracy duration_type [0..1]</strong></td>
<td>A measure of the accuracy of the time-stamping of the plot's time_of_plot attribute. This is equal to one standard deviation of uncertainty. This is optional as not all sensors estimate time accuracy and for some applications the uncertainty is negligible. AttributeTag: Issue =</td>
</tr>
</tbody>
</table>
### Table 7.208 - Relations of Class sensor_plot_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>additional_information</strong> anonymous_blob_type</td>
<td>Potentially classified information about the plot, which may be used in a system specific way to distribute information about a plot to other subsystems. Further information about this attribute, including layout semantics is outside of the scope of this interface standard.</td>
</tr>
<tr>
<td><strong>splash_spotting_area_id</strong> splash_spotting_area_id_type [0..1]</td>
<td>Indicates which splash spotting area the plot refers to - if any - hence it is optional.</td>
</tr>
<tr>
<td><strong>jammer_indication</strong> boolean</td>
<td>Indication whether or not a plot is from a source of jamming.</td>
</tr>
</tbody>
</table>

### Table 7.209 - Attributes of IDLStruct sensor_precision_model_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>azimuth_precision</strong> azimuth_coordinate_type [0..1]</td>
<td>The precision with which the sensor is capable of measuring azimuth.</td>
</tr>
<tr>
<td><strong>elevation_precision</strong> elevation_coordinate_type [0..1]</td>
<td>The precision with which the sensor is capable of measuring elevation.</td>
</tr>
<tr>
<td><strong>range_precision</strong> range_coordinate_type [0..1]</td>
<td>The precision with which the sensor is capable of measuring range.</td>
</tr>
<tr>
<td><strong>range_rate_precision</strong> range_rate_type [0..1]</td>
<td>The precision with which the sensor is capable of measuring range rate.</td>
</tr>
<tr>
<td><strong>time_precision</strong> duration_type [0..1]</td>
<td>The precision with which the sensor is capable of measuring time.</td>
</tr>
<tr>
<td><strong>strength_precision</strong> strength_type [0..1]</td>
<td>The precision with which the sensor is capable of measuring signal strength.</td>
</tr>
<tr>
<td><strong>confidence_threshold</strong> confidence_type [0..1]</td>
<td>The threshold probability for signal strength to identify a plot.</td>
</tr>
</tbody>
</table>

#### 7.6.6.11 sensor_precision_model_type

**Type:** IDLStruct  
**Package:** Plot_Reporting  

This class models the precision of the sensor - i.e. the smallest changes in measurement quantities that it is capable of distinguishing.
7.6.6.12  sensor_processing_model_type
Type: IDLStruct
Package: Plot_Reporting
This class encapsulates sensor processing parameters to promote the accurate statistical processing of its measurements.

Table 7.210 - Attributes of IDLStruct sensor_processing_model_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>nominal_reporting_interval duration_type [0..1]</td>
<td>The nominal period between successive measurements on the same object for the sensor. AttributeTag: Issue =</td>
</tr>
<tr>
<td>nominal_internal_reporting_latency duration_type [0..1]</td>
<td>The nominal period between the sensor's measurement of an object and its reporting of the object to systems on the same platform. AttributeTag: Issue =</td>
</tr>
<tr>
<td>nominal_external_reporting_latency duration_type [0..1]</td>
<td>The nominal period between the sensor's measurement of an object and its reporting of the object to a system on any other connected platform. AttributeTag: Issue =</td>
</tr>
<tr>
<td>processing_capability processing_capability_type [0..10]</td>
<td>The set of processing capabilities of which the sensor is capable. These capabilities have quality implications for the sensors plot measurement information.</td>
</tr>
</tbody>
</table>

7.6.6.13  sensor_stabilization_model_type
Type: IDLStruct
Package: Plot_Reporting
This class models the sensor error estimate due to sensor stabilization. These are errors that are in addition to any calibration errors.

Table 7.211 - Attributes of IDLStruct sensor_stabilization_model_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>azimuth_error azimuth_coordinate_type [0..1]</td>
<td>Current error in azimuth due to stabilization to one standard deviation</td>
</tr>
<tr>
<td>elevation_error azimuth_coordinate_type [0..1]</td>
<td>Current error in elevation due to stabilization to one standard deviation</td>
</tr>
<tr>
<td>orientation_error_distribution error_distribution_kind_type</td>
<td>The statistical distribution of the azimuth and elevation stabilization errors.</td>
</tr>
<tr>
<td>temporal_error_correlation_interval duration_type [0..1]</td>
<td>The time period, centered on the time of information, such that the coefficient of correlation between stabilization errors at either end is expected to be 0.5. Measurements made within this interval are expected to have stabilization errors that are strongly correlated with each other.</td>
</tr>
</tbody>
</table>
The time for which the stabilization error estimates are valid.

### 7.6.6.14 track_level_filter_attributes_type

**Type:** IDLStruct  
**Package:** Plot_Reporting

The track-level criteria that must be met for the plot to pass the filter. These are criteria applied with respect to any track to which the plot is contributing.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>speed speed_type [0..1]</td>
<td>A speed criterion. A track to which the plot contributes must have an absolute speed greater than this.</td>
</tr>
<tr>
<td>priority track_priority_type [0..1]</td>
<td>A priority criterion. A track to which the plot contributes must have a priority greater than or equal to this.</td>
</tr>
<tr>
<td>association_status confidence_interval_type</td>
<td>A criterion relating to whether the plot is contributing to a track. The cumulative probability of being associated to a track must be within the interval defined to pass the filter. If no filtering is required then an interval including all confidence values is defined.</td>
</tr>
<tr>
<td>external_protocol_name String [0..*]</td>
<td>Filter on the basis of the external protocols on which the track is known. AttributeTag: Issue =</td>
</tr>
<tr>
<td>quality quality_interval_type</td>
<td>A criterion relating to the quality of a track. A track to which the plot contributes must have a track quality within the interval defined to pass the filter. If no filtering is required then an interval including all quality values is defined.</td>
</tr>
</tbody>
</table>

### 7.6.6.15 sensor_orientation_type

**Type:** Class  
**Package:** Plot_Reporting

This class describes the orientation of the sensor at a particular moment in time. This is useful for plot processing functionality such as track extraction as it allows instantaneous coverage of the sensor to be estimated.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>azimuth azimuth_coordinate_type</td>
<td>The (azimuth) direction of the head of the sensor (e.g. antenna, lens or hydro-phone)</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>elevation elevation_coordinate_type [0..1]</code></td>
<td>The (elevation) direction of the head of the sensor (e.g. antenna, lens or hydro-phone). If not supplied either horizontal is assumed or a constant angle is defined through the Manage_Subsystem_Parameters use case.</td>
</tr>
<tr>
<td><code>time_of_validity time_type</code></td>
<td>The time for which is sensor orientation is valid</td>
</tr>
<tr>
<td><code>sensor_coordinate_system</code></td>
<td>This attribute defines the interpretation of azimuth and elevation. Valid enumerates are: NORTH_HORIZONTAL, NORTH_DOWN, STERN_VEEL, STERN_DECK_LEVEL</td>
</tr>
<tr>
<td><code>origin position_coordinate_type [0..1]</code></td>
<td>The position of the origin of the head of the sensor.</td>
</tr>
<tr>
<td><code>azimuth_coverage azimuth_interval_type [0..1]</code></td>
<td>The instantaneous extent of the coverage of the sensor in azimuth with respect the origin of its head.</td>
</tr>
<tr>
<td><code>elevation_coverage elevation_interval_type [0..1]</code></td>
<td>The instantaneous extent of the coverage of the sensor in elevation with respect the origin of its head. If it is only valid to set this when the elevation attribute is also specified.</td>
</tr>
<tr>
<td><code>origin_coordinate_specification</code></td>
<td>This attribute defines the characteristics of the coordinate system used to define the origin. It is only valid to set this when origin attribute is also specified.</td>
</tr>
</tbody>
</table>

### 7.5.7 Sensor_Control

**Parent Package:** Sensor_Domain

This package contains structs and type defs for managing frequency usage, transmission sectors, emission control, and test target scenarios.
```idl
+targets_parameter
«idlStruct»
  «idlStruct»
  «idlStruct»
  «idlStruct»
  «idlStruct»
  «idlStruct»
```

```idl
+targets_parameter
«idlStruct»
  «idlStruct»
  «idlStruct»
  «idlStruct»
  «idlStruct»
  «idlStruct»
```
7.6.7.1 selected_frequency_list_type

**Type:** IDLStruct

**Package:** Sensor_Control

This struct contains zero to many frequencies which may be enabled/disabled by the CMS.

### Table 7.214 - Relations of IDLStruct selected_frequency_list_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
</tr>
</tbody>
</table>

### Table 7.120 - Attributes of Class transmission_frequency_state_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
</tr>
<tr>
<td>frequency_id</td>
<td>frequency_band_type</td>
</tr>
</tbody>
</table>

7.6.7.2 transmission_frequency_state_type

**Type:** Class

**Package:** Sensor_Control

State of frequency transmission.

### Table 7.215 - Attributes of Class transmission_frequency_state_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
</tr>
</tbody>
</table>

7.6.7.3 all_frequencies_state_type

**Type:** Class

**Package:** Sensor_Control

This struct contains zero to many "available" or "not available" frequencies which may be enabled/disabled by the CMS.

### Table 7.121 - Attributes of IDLStruct reported_frequency all_frequencies_state_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
</tr>
<tr>
<td>frequency_id</td>
<td>frequency_band_type</td>
</tr>
</tbody>
</table>
### enable boolean
Indicates whether the CMS is enabling or disabling a transmission frequency.

### Aggregation: reported_frequencies

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>reported_frequency_id</td>
<td>A unique identifier for the transmission frequency.</td>
</tr>
<tr>
<td>frequency_band</td>
<td></td>
</tr>
<tr>
<td>state_type</td>
<td></td>
</tr>
</tbody>
</table>

### available boolean
Indicates whether a transmission frequency is available or not available.

#### 7.6.7.4 transmission_reported_frequency_mode_type

**Type:** Class
**Package:** Sensor_Control

**Notes:**
- AUTOMATIC_FREQUENCY_SELECTION: The sensor always uses the same pre-selected frequency.
- FIXED_FREQUENCY: At each transmission sensor selects the frequency to be used inside a pre-selected subset of frequencies.
- FREQUENCY_DIVERSITY: At each transmission sensor selects the frequency to be used among the least jammed frequencies.
- RANDOM_AGILITY: At each transmission sensor random selects the frequency to be used.
### 7.6.7.6 transmission_sector_set_type

**Type:** IDLStruct  
**Package:** Sensor_Control  

This struct contains zero to many transmission sectors which must be set/reset by the CMS.

### Table 7.219 - Relations of IDLStruct transmission_sector_set_type—Type—Class

**Package:** Sensor_Control  
**Sector for transmission**

<table>
<thead>
<tr>
<th>Attribute/connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>power_level</td>
<td>Indicates the transmission power level of the sector.</td>
</tr>
<tr>
<td>sector_enabled</td>
<td>Indicates whether the CMS is enabling or disabling a transmission sector.</td>
</tr>
<tr>
<td>sector_id</td>
<td>A unique identifier for the transmission sector.</td>
</tr>
<tr>
<td>sector_reference</td>
<td>This indicates the reference system of the transmission sector.</td>
</tr>
<tr>
<td>sector_shape</td>
<td>Note that the azimuth dimension of the sector shape (polar volume) applies to the horizon plane (i.e., elevation=0).</td>
</tr>
<tr>
<td>transmission_mode</td>
<td>Indicates the transmission mode used within the sector.</td>
</tr>
</tbody>
</table>

### Table 7.220 - Attributes of Class transmission_sector_type

**Package:** Sensor_Control  
**Sector for transmission**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>power_level_transmission</td>
<td>Indicates the transmission power level of the sector.</td>
</tr>
<tr>
<td>sector_enabled</td>
<td>Indicates whether the CMS is enabling or disabling a transmission sector.</td>
</tr>
<tr>
<td>sector_id</td>
<td>A unique identifier for the transmission sector.</td>
</tr>
<tr>
<td>sector_reference</td>
<td>This indicates the reference system of the transmission sector.</td>
</tr>
<tr>
<td>sector_shape</td>
<td>Note that the azimuth dimension of the sector shape (polar volume) applies to the horizon plane (i.e., elevation=0).</td>
</tr>
<tr>
<td>transmission_mode</td>
<td>Indicates the transmission mode used within the sector.</td>
</tr>
</tbody>
</table>
7.6.7.8 transmission_sector_power_level_type

Type: Class
Package: Sensor_Control
This enumeration allows specification of a CMS commanded power level for a sector.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>FULL_RADIATE_POWER</td>
<td>radiate with full power</td>
</tr>
<tr>
<td>INHIBIT</td>
<td>inhibit transmission</td>
</tr>
<tr>
<td>REDUCED_RADIATE_POWER</td>
<td>radiate with reduced power</td>
</tr>
</tbody>
</table>
7.6.7.9 sector_reference_type

Type: IDLEnum
Package: Sensor_Control
This enumeration specifies the sectors reference systems.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>NORTH_RELATED</td>
<td>Indicates values referenced with respect to true North</td>
</tr>
<tr>
<td>SHIP_RELATED</td>
<td>Indicates values referenced with respect to ship's heading</td>
</tr>
</tbody>
</table>

7.6.7.10 control_emission_state_type

Type: Class
Package: Sensor_Control
Emission state

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>emission_activated</td>
<td>Indicates whether the CMS is enabling or disabling the sensor emission state.</td>
</tr>
</tbody>
</table>

7.6.7.11 test_target_scenario_type

Type: IDLUnion
Package: Sensor_Control
Scenario for test targets

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlCase» scenario_common_parameter_target</td>
<td>This case is used when a test target scenario is constituted by a number of targets distributed in a defined area/volume and having the same common parameters.</td>
</tr>
<tr>
<td>«idlCase» scenario_independent_target</td>
<td>This case is used when a test target scenario is constituted by a number of independent targets.</td>
</tr>
</tbody>
</table>

7.6.7.12 test_target_scenario_independent_target_type

Type: IDLStruct
Package: Sensor_Control
The scenario is defined by a number of independent targets, with each target having its own characteristic parameters.

**Table 7.128225 - Attributes of IDLStruct test_target_scenario_independent_target_type**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>number_of_test_target</td>
<td>unsigned short This is the number of the test targets composing the scenario.</td>
</tr>
<tr>
<td>test_target_scenario_activated</td>
<td>boolean Indicates whether the CMS is enabling or disabling the generation of a test target scenario.</td>
</tr>
<tr>
<td>test_target_scenario_id</td>
<td>test_target_scenario_id_type A unique identifier for the test target scenario.</td>
</tr>
</tbody>
</table>

**Table 7.226 - Relations of IDLStruct test_target_scenario_independent_target_type**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: targets</td>
<td>test_target_scenario_id [0..*] A unique identifier for the test target scenario.</td>
</tr>
<tr>
<td>test_target_scenario_id_0</td>
<td>test_target_scenario_id_type</td>
</tr>
</tbody>
</table>

**7.6.7.13 test_target_scenario_common_parameter_target_type**

**7.5.7.13 Type:** IDLStruct  
**Package:** Sensor_Control  
The scenario is defined by a number of targets distributed in a defined area/volume and having the same common parameters.

**Table 7.129227 - Attributes of IDLStruct test_target_scenario_common_parameter_target_type**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>initial_time</td>
<td>time_type This indicates the common initial time of the targets.</td>
</tr>
<tr>
<td>number_of_test_target</td>
<td>unsigned short This is the number of the test targets composing the scenario.</td>
</tr>
<tr>
<td>test_target_scenario_activated</td>
<td>boolean Indicates whether the CMS is enabling or disabling the generation of a test target scenario.</td>
</tr>
<tr>
<td>test_target_scenario_id</td>
<td>test_target_scenario_id_type A unique identifier for the test target scenario.</td>
</tr>
<tr>
<td>volume_boundaries</td>
<td>general_polar_volume_type This indicates the area/volume boundaries where the test targets are distributed.</td>
</tr>
</tbody>
</table>

**Table 7.228 - Relations of IDLStruct test_target_scenario_common_parameter_target_type**

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: targets</td>
<td>parameter test_target_plus_scenario_type</td>
</tr>
<tr>
<td>test_target_plus_scenario_type</td>
<td></td>
</tr>
</tbody>
</table>

**7.6.7.14 test_target_type**

**Type:** IDLStruct
### Table 7.130229 - Attributes of IDLStruct test_target_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>initial_time time_type</td>
<td>This attribute defines the relevant initial time.</td>
</tr>
<tr>
<td>position wgs84_position_type</td>
<td>This attribute defines the initial target position.</td>
</tr>
<tr>
<td>test_target_id unsigned short</td>
<td>A identifier for the test targets.</td>
</tr>
<tr>
<td>test_target_parameter anonymous_blob_type</td>
<td>This attribute defines: - the target motion type, with the relevant motion parameters - the target generation parameters, such as injection type (internal / external), attenuation law (constant / variable-with-range), doppler type (0 / PRF/2).</td>
</tr>
</tbody>
</table>

#### 7.6.7.15 test_target_plus_scenario_type

Type: Class

Package: Sensor_Control

Test target with its scenario
Table 7.230 - Attributes of Class test_target_plus_scenario_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>test_target_id</td>
<td>unsigned short A identifier for the test targets.</td>
</tr>
<tr>
<td>test_target_parameter</td>
<td>anonymous_blob_type This attribute defines:</td>
</tr>
<tr>
<td></td>
<td>- the target motion type, with the relevant motion parameters</td>
</tr>
<tr>
<td></td>
<td>- the target generation parameters, such as injection type</td>
</tr>
<tr>
<td></td>
<td>(internal / external), attenuation law (constant / variable-</td>
</tr>
<tr>
<td></td>
<td>with-range), doppler type (0 / PRF/2).</td>
</tr>
</tbody>
</table>

7.6.7.16 test_target_scenario_id_type

7.5.7.16 Type: IDLTypeDef long

Package: Sensor_Control

This typedef is used to identify a specific test target scenario.

7.6.7.17 test_target_scenario_state_type

7.5.7.17 Type: Class

Package: Sensor_Control

scenario state

Table 7.4321 - Attributes of Class test_target_scenario_state_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>test_target_scenario_activated</td>
<td>boolean Indicates whether the CMS is enabling or disabling the execution of the test target scenario.</td>
</tr>
<tr>
<td>test_target_scenario_id</td>
<td>test_target_scenario_id_type A unique identifier for the test target scenario.</td>
</tr>
</tbody>
</table>
7.6.8.1 absolute_performance_type

**Type:** IDLTypeDef

**Package:** Sensor_Performance

Defined as a signal excess in dB above noise floor for a nominal ideal target with 1m² tangential cross-section.

7.6.8.2 interference_report_type

**7.5.8.1 Type:** Class

**Package:** Sensor_Performance

Set of interferer objects in a report.

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: interferers</td>
<td>The interference sources, which are described by the report.</td>
</tr>
</tbody>
</table>

Table 7.232 - Relations of Class interference_report_type
### 7.6.8.3 interferer_kind

**Type:** Class

**Package:** Sensor_Performance

Enumeration of the types of interferers that are known about.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVE_NOISE</td>
<td>Interference from active noise.</td>
</tr>
<tr>
<td>CLUTTER</td>
<td>Interference from clutter.</td>
</tr>
<tr>
<td>SELF_SCREENING_JAMMER</td>
<td>Interference from a jammer, which is self screening.</td>
</tr>
<tr>
<td>STANDOFF_JAMMER</td>
<td>Interference from a stand-off jammer.</td>
</tr>
<tr>
<td>STROBE</td>
<td>Interference from a strobe jammer.</td>
</tr>
<tr>
<td>OTHER_TYPE</td>
<td>The interference source is of a different type to the other declared interference kinds</td>
</tr>
<tr>
<td>NO_STATEMENT</td>
<td>The interference source could not be classified by the sensor subsystem.</td>
</tr>
<tr>
<td>ACTIVE_NOISE</td>
<td>Interference from active noise.</td>
</tr>
<tr>
<td>CLUTTER</td>
<td>Interference from clutter.</td>
</tr>
<tr>
<td>SELF_SCREENING_JAMMER</td>
<td>Interference from a jammer, which is self screening.</td>
</tr>
<tr>
<td>STANDOFF_JAMMER</td>
<td>Interference from a stand-off jammer.</td>
</tr>
<tr>
<td>STROBE</td>
<td>Interference from a strobe jammer.</td>
</tr>
<tr>
<td>OTHER_TYPE</td>
<td>The interference source is of a different type to the other declared interference kinds</td>
</tr>
<tr>
<td>NO_STATEMENT</td>
<td>The interference source could not be classified by the sensor subsystem.</td>
</tr>
</tbody>
</table>

### 7.6.8.4 interferer_type

**Type:** Class

**Package:** Sensor_Performance

A single source of interference.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>STANDOFF_JAMMER</td>
<td>Interference from a stand-off jammer.</td>
</tr>
<tr>
<td>STROBE</td>
<td>Interference from a strobe jammer.</td>
</tr>
<tr>
<td>OTHER_TYPE</td>
<td>The interference source is of a different type to the other declared interference kinds</td>
</tr>
<tr>
<td>NO_STATEMENT</td>
<td>The interference source could not be classified by the sensor subsystem.</td>
</tr>
</tbody>
</table>
### Attribute Table

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>timestamp time_type</td>
<td>Time to which the performance report applies.</td>
</tr>
<tr>
<td>magnitude jamming_magnitude_type [0..1]</td>
<td>The Effective Radiated Power (ERP) of the source of interference. This is an optional attribute, which may not all sensors may be able to calculate.</td>
</tr>
<tr>
<td>affected_bands frequency_band_type [1..*]</td>
<td>A list of frequency bands which are affected by the source of interference.</td>
</tr>
<tr>
<td>position position_coordinate_type [0..1]</td>
<td>The source position of the interference. This is an optional attribute that not all sensors may be able to calculate.</td>
</tr>
<tr>
<td>kind interferer_kind</td>
<td>A classification of the interference source.</td>
</tr>
<tr>
<td>affected_volume general_polar_volume_type [0..1]</td>
<td>The volume in space, which the interference source is affecting. This is an optional attribute, which may not all sensors may be able to calculate.</td>
</tr>
<tr>
<td>position_coordinate_specification coordinate_specification_type</td>
<td>Specifies the coordinate system used to define the interferer.</td>
</tr>
</tbody>
</table>

**7.6.8.5 jamming_magnitude_type**

- **Type:** Class unsigned short
- **Package:** Sensor_Performance

Target strength (Effective Radiated Power - ERP) of a jammer. The precise semantics of this type are sensor subsystem specific, but a typical interpretation is as a signal to noise ratio in dB.

**7.6.8.6 performance_bin_type**

Open Architecture Radar Interface Standard (OARIS), v2.0
7.5.8.5 Type: IDLStruct
Package: Sensor_Performance
Value of performance in a volume of space. This is given as a signal excess in dB above noise floor for a nominal 0dB target strength. For a current performance report, this noise floor shall include clutter and jamming. These are not included in a nominal performance report.
Table 7.134 - Attributes of IDLStruct perfomance_bin_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>start_range range_coordinate_type</td>
<td>The start of the bin in range.</td>
</tr>
<tr>
<td>end_range range_coordinate_type</td>
<td>The end of the bin in range.</td>
</tr>
<tr>
<td>relative_value relative_performance_type [0..1]</td>
<td>The assessed relative level of performance, (comparable with other instances of the sensor or the same sensor in a different context). If no value present, there is no performance data available for this bin.</td>
</tr>
<tr>
<td>absolute_value absolute_performance_type [0..1]</td>
<td>The assessed absolute level of performance (comparable with other sensors). If no value present, there is no performance data available for this bin.</td>
</tr>
</tbody>
</table>

7.6.8.7 performance_assessment_reportparameters_type

7.5.8.6 Type: Class
Package: Sensor_Performance
Contains the results of a performance assessment.

Table 7.135 - Attributes of Class performance_assessment_report_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_of_report time_type</td>
<td>The time of validity of the performance assessment.</td>
</tr>
</tbody>
</table>

7.5.8.7 performance_assessment_request_type

Type: Class
Package: Sensor_Performance
A performance assessment request consists of an overall volume of interest and a specification of a number of 'bins' into which that volume is to be sub-divided. In response the sensor assess performance for each 'bin'. The coordinate origin for the request is the SENSOR_REFERENCE_POINT as defined in coordinate_origin_type.

Table 7.136 - Attributes of Class performance_assessment_requestparameters_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>azimuth_bin_count unsigned short</td>
<td>Number of azimuth bins that the CMS would like in the performance report. The subsystem should try to honour this request but does not have to.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>range_bin_count</td>
<td>Number of range bins that the CMS would like in the report. The subsystem should try to honour this request but does not have to.</td>
</tr>
<tr>
<td>elevation_bin_count</td>
<td>The number of elevation bins that the CMS would like in the report. The subsystem should try to honour this request but does not have to.</td>
</tr>
<tr>
<td>start_azimuth</td>
<td>Defines the start of the arc of azimuth (positive orientation) of the volume in which the sensor's performance is to be assessed.</td>
</tr>
<tr>
<td>end_azimuth</td>
<td>Defines the end of the arc of azimuth (positive orientation) of the volume in which the sensor's performance is to be assessed.</td>
</tr>
<tr>
<td>start_elevation</td>
<td>Defines the start of the arc of elevation (positive orientation) of the volume in which the sensor's performance is to be assessed.</td>
</tr>
<tr>
<td>end_elevation</td>
<td>Defines the end of the arc of elevation (positive orientation) of the volume in which the sensor's performance is to be assessed.</td>
</tr>
<tr>
<td>min_range</td>
<td>Defines the minimum range of the volume in which the sensor's performance is to be assessed.</td>
</tr>
<tr>
<td>max_range</td>
<td>Defines the maximum range of the volume in which the sensor's performance is to be assessed.</td>
</tr>
<tr>
<td>frequency_bands</td>
<td>The set of frequency bands to assess the performance for. Where no bands are specified the performance is assessed for the sensor in general in the specified operational mode.</td>
</tr>
<tr>
<td>applicable_mode</td>
<td>The performance assessment is to be in the context of this operational mode of the sensor subsystem.</td>
</tr>
<tr>
<td>coordinate_orientation</td>
<td>The orientation of the polar coordinates used in this class. Note that the origin is always the sensor reference point and that the coordinate system is always polar.</td>
</tr>
</tbody>
</table>

### 7.6.8.8 performance_assessment_report_type

**Type:** Class

**Package:** Sensor_Performance

Contains the results of a performance assessment.

---

Table 7.237 - Attributes of Class performance_assessment_report_type
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>start_elevation elevation_coordinate_type [0..1]</td>
<td>Defines the start of the arc of elevation (positive orientation) of the volume in which the sensor's performance is to be assessed.</td>
</tr>
<tr>
<td>end_elevation elevation_coordinate_type [0..1]</td>
<td>Defines the end of the arc of elevation (positive orientation) of the volume in which the sensor's performance is to be assessed.</td>
</tr>
<tr>
<td>min_range range_coordinate_type [0..1]</td>
<td>Defines the minimum range of the volume in which the sensor's performance is to be assessed.</td>
</tr>
<tr>
<td>max_range range_coordinate_type [0..1]</td>
<td>Defines the maximum range of the volume in which the sensor's performance is to be assessed.</td>
</tr>
<tr>
<td>frequency_bands frequency_band_type [0..*]</td>
<td>The set of frequency bands to assess the performance for. Where no bands are specified the performance is assessed for the sensor in general in the specified operational mode.</td>
</tr>
<tr>
<td>applicable_mode operational_mode_type</td>
<td>The performance assessment is to be in the context of this operational mode of the sensor subsystem.</td>
</tr>
<tr>
<td>coordinate_orientation coordinate_orientation_type</td>
<td>The orientation of the polar coordinates used in this class. Note that the origin is always the sensor reference point and that the coordinate system is always polar.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: band performance_band_type [1..*]</td>
<td>The performance assessment for the band (or the sensor in general).</td>
</tr>
<tr>
<td>Aggregation: assessment_dimensions performance_assessment_parameters_type [1]</td>
<td>The actual dimensions of the assessment that is performed are reported with the result.</td>
</tr>
</tbody>
</table>

### Table 7.238 - Relations of Class performance_assessment_report_type

#### 7.6.8.9 performance_band_type

**Type:** IDLStruct

**Package:** Sensor_Performance

The performance reported in a particular band (or in general)

### Table 7.137239 - Attributes of IDLStruct performance_band_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>frequency_band frequency_band_type [0..1]</td>
<td>The specific band to which the contained performance assessments refer.</td>
</tr>
<tr>
<td>performance_beam_type:</td>
<td>The specific band to which the contained performance assessments refer.</td>
</tr>
</tbody>
</table>
7.5.8.9 Table 7.240 - Relations of IDLStruct performance_band_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: sector performance_sector_type [1..*]</td>
<td>The list of sectors in the performance assessment</td>
</tr>
</tbody>
</table>

7.6.8.10 performance_beam_type

**Type:** IDLStruct  
**Package:** Sensor_Performance

Set of performance values for a line of points in space. Each value applies to a volume whose boundaries may be inferred from the numbers of bins and the min and max values in the report.

**Table 7.138241 - Attributes of IDLStruct performance_beam_type**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>start_elevation elevation_coordinate_type</td>
<td>The start of the beam in elevation (positive orientation).</td>
</tr>
<tr>
<td>end_elevation  elevation_coordinate_type</td>
<td>The end of the beam in elevation (positive orientation).</td>
</tr>
</tbody>
</table>

**Table 7.242 - Relations of IDLStruct performance_beam_type**

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: bin performance_bin_type [1..*]</td>
<td>The list of 'bins' in a beam of the performance assessment</td>
</tr>
</tbody>
</table>

7.5.8.107.6.8.11 performance_sector_type
Type: Class
Package: Sensor_Performance

A set of performance values for a sector of azimuth [start_azimuth..end_azimuth].

Table 7.43243 - Attributes of Class performance_sector_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>start_azimuth</td>
<td>azimuth_coordinate_type The start of the sector of azimuth (positive orientation).</td>
</tr>
<tr>
<td>end_azimuth</td>
<td>azimuth_coordinate_type The end of the sector of azimuth (positive orientation).</td>
</tr>
</tbody>
</table>

Table 7.244 - Relations of Class performance_sector_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: beam</td>
<td>performance_beam_type [1..*] The list of beams in the sector of the performance report</td>
</tr>
</tbody>
</table>

7.5.8.117, 6.8.12relative_performance_type

Type: Class float
Package: Sensor_Performance

Defined as a signal excess in dB above noise floor for a nominal 0dB target strength, when assessing nominal performance or for the jammer when providing jammer assessment.

7.5.97.6.9 Track_Reporting

Parent Package: Sensor_Domain
This service provides facilities to report different types of sensor tracks.
Figure 7.55.81 Track Reporting - Sensor Track (Class diagram)
Figure 7.56.82 Track Reporting - Type Definitions (Class diagram)
7.5.9.1 sensor_track_assessment_objective_id_type
Type: IDLTypeDef unsigned long
Class: Track_Reporting::Sensor Track Identification
Package: Track_Reporting

Unique identifier for the objects to which the sensor assessment is attempting to match the measurement parameters.
7.5.9.27.6.9.2 _external_track_prioritynumber_type_

Type: Class unsigned short _IDL.Struct_

Package: Track_Reporting

The representation of the track's priority with respect to the allocation of the sensor's resources.
A track _quality number from an external protocol (to OARIS) such as a data-link.

### Table 7.245 - Attributes of IDLStruct external_track_number_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>protocol_name</td>
<td>String The name of the external protocol, system of network</td>
</tr>
<tr>
<td>track_number</td>
<td>String The string representation (human readable) of the external track number value,</td>
</tr>
</tbody>
</table>

#### 7.6.9.3 plot_association_type

**Type:** Class float IDLStruct  
**Package:** Track_Reporting

This class represents an association between a sensor track and a sensor plot, supporting a multi-hypothesis many-to-many mapping between plots and tracks.

### Table 7.246 - Attributes of IDLStruct plot_association_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>association_likelihood</td>
<td>confidence_type The likelihood of this sensor plot given the prior sensor track. This is independent of the likelihood of other track associations. In general, the association likelihood values for a plot do not sum to one. AttributeTag: Issue =</td>
</tr>
<tr>
<td>plot_subsystem</td>
<td>subsystem_id_type The subsystem that published the sensor plot contributing to the sensor track. AttributeTag: Issue =</td>
</tr>
</tbody>
</table>

### Table 7.247 - Relations of IDLStruct plot_association_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association.plot</td>
<td>sensor_plot_type reference [1] The sensor plot that the sensor track is based on</td>
</tr>
</tbody>
</table>

#### 7.6.9.4 protocol_name_type

**Type:** IDLTypeDef  
**Package:** Track_Reporting

The representation of the quality of a track for the purposes of comparison according to system specific criteria.

**ElementTag: Range = 0.0 .. 1.0**

**environment** The name of an external protocol on which objects (e.g. tracks) could also be reported. Values are system implementation specific.

#### 7.6.9.5 sensor_track_id_type

**Type:** Class IDLTypeDef  
**Package:** Track_Reporting  
Sensor Track Identification
7.6.9.6 track_filter_parameters_type

**Type:** IDLStruct

**Package:** Track_Reporting

The criteria that must all be met for a track to pass the filter. The filter attributes are applied with and-wise logic. For or-wise logic define multiple filter objects.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode filter_mode_type</td>
<td>The mode in which the tracks are filtered.</td>
</tr>
<tr>
<td>protocol_name protocol_name_type [0..*]</td>
<td>Filter tracks that are also being reported on these protocols.</td>
</tr>
<tr>
<td>is_inclusive boolean</td>
<td>If true, tracks that pass the filter are included in transmission and/or reception dependent upon the mode attribute. Otherwise, they are excluded.</td>
</tr>
<tr>
<td>priority track_priority_type [0..1]</td>
<td>A priority criterion. A track must have a priority greater than or equal to this.</td>
</tr>
<tr>
<td>area area_2d_type [0..1]</td>
<td>An area which is optionally part of the filter.</td>
</tr>
<tr>
<td>quality quality_interval_type</td>
<td>A track quality criterion. A track must have a track quality within the interval defined to pass the filter. If no filtering is required then an interval including all quality values is defined.</td>
</tr>
<tr>
<td>height height_interval_type [0..1]</td>
<td>The height values that are optionally part of the filter.</td>
</tr>
</tbody>
</table>

7.6.9.7 environment_type

**Type:** Class

**Package:** Track_Reporting

The sensor tracking environment

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>«idlEnum» AIR</td>
<td>In the air</td>
</tr>
<tr>
<td>«idlEnum» LAND</td>
<td>On land</td>
</tr>
<tr>
<td>«idlEnum» SURFACE</td>
<td>On the sea surface</td>
</tr>
<tr>
<td>«idlEnum» SUBSURFACE</td>
<td>Below the sea surface</td>
</tr>
<tr>
<td>«idlEnum» SPACE</td>
<td>Outside the Earth's atmosphere</td>
</tr>
</tbody>
</table>

7.6.9.8 initiation_mode_type
7.5.9.5 Type: Class
Package: Track_Reporting
Type of track initiation

Table 7.144250 - Attributes of Class initiation_mode_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTOMATIC</td>
<td>Automatic track initiation mode</td>
</tr>
<tr>
<td>EXTERNAL REQUEST</td>
<td>Track initiation on external request (e.g. from CMS)</td>
</tr>
</tbody>
</table>

7.5.9.6 recognition_type

Type: IDLTypeDef unsigned short
Package: Track_Reporting
The recognition type indicates the type of the real-world physical object being tracked.
The numeric value is used to map to a system or implementation specific taxonomy of real-world physical objects that are of tactical interest.

7.5.9.7 sensor_track_type

Table 7.142251 - Attributes of IDLStruct sensor_track_type

Type: IDLStruct
Package: Track_Reporting
Encapsulation of a sensor track
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>additional_information</code> anonymous_blob_type</td>
<td>Additional, vendor-specific information</td>
</tr>
<tr>
<td><code>covariance_matrix</code> covariance_matrix_type [0..1]</td>
<td>The number of elements in the covariance matrix is dependent on the sensor. When present, the position_accuracy and velocity_accuracy attributes should not be present.</td>
</tr>
<tr>
<td>environment environment_type [0..1]</td>
<td>Environment of the track (air, surface etc)</td>
</tr>
<tr>
<td><code>initiation_mode</code> initiation_mode_type [0..1]</td>
<td>Initiation mode of track (automatic or externally initiated)</td>
</tr>
<tr>
<td><code>jammer_indication</code> boolean</td>
<td>Indication whether or not a track is jamming.</td>
</tr>
<tr>
<td><code>max_range_limit</code> range_coordinate_type [0..1]</td>
<td>Maximal range for a bearing track</td>
</tr>
<tr>
<td><code>position</code> position_coordinate_type</td>
<td>The location of the track as calculated in the sensor's chosen coordinate system at the stated time.</td>
</tr>
<tr>
<td><code>position_accuracy</code> position_accuracy_coordinate_type [0..1]</td>
<td>The sensor's stated accuracy for its calculated position. When present, the covariance_matrix attribute should not be present.</td>
</tr>
<tr>
<td><code>position_accuracy_coordinate_system</code> coordinate_specification_type [0..1]</td>
<td>The coordinate system chosen by the sensor for reporting accuracy.</td>
</tr>
<tr>
<td><code>position_coordinate_system</code> coordinate_specification_type</td>
<td>The coordinate system chosen by the sensor.</td>
</tr>
<tr>
<td>«key» <code>sensor_track_id</code> sensor_track_id_type</td>
<td>The sensor's unique identifying reference for the track. Sensors may reuse identifiers after they have deleted the corresponding track. The scheme used for identifier reallocation is system dependent.</td>
</tr>
<tr>
<td><code>sensor_track_pre_identification</code> identity_type [0..1]</td>
<td>Identification information for the sensor track (if available)</td>
</tr>
<tr>
<td><code>sensor_track_pre_recognition</code> recognition_type [0..1]</td>
<td>Recognition information for the sensor track (if available)</td>
</tr>
<tr>
<td><code>simulated</code> boolean</td>
<td>Whether the CMS should process the track as having been synthetically generated as opposed to corresponding to an actual detection in the real world.</td>
</tr>
<tr>
<td><code>time_of_information</code> time_type</td>
<td>The time at which the information in this object is valid, in particular its position.</td>
</tr>
<tr>
<td><code>time_of_initiation</code> time_type [0..1]</td>
<td>The time at which the sensor first determined the existence of this track. <code>&lt;AttributeTag: Issue&gt;</code></td>
</tr>
<tr>
<td><code>track_phase</code> track_phase_type</td>
<td>Track phase (e.g. TRACKED, DELETED, LOST)</td>
</tr>
</tbody>
</table>
### Attribute | Notes
--- | ---
`velocity` | The velocity of the track as calculated in the sensor's chosen coordinate system at the stated time.
`velocity_accuracy` | The sensor's stated accuracy for its calculated velocity. When present, the covariance_matrix attribute should not be present.
`velocity_accuracy_coordinate_system` | The coordinate system chosen by the sensor for reporting accuracy.
`velocity_coordinate_system` | The coordinate system chosen by the sensor.
`track_quality` | The sensor specific quality of this track in comparison to its typical tracks.
`time_of_first_detection` | The time at which the sensor first made measurements leading to the detection of the existence of this track (as opposed to the time of initiation when there was sufficient confidence in one or more detection to initiate a track).
`time_of_last_detection` | The time at which the sensor last detected the existence of this track.
`priority` | The relative priority of a track with regard to the sensor's resources.
`amplitude` | The amplitude or strength of the measurement(s) being tracked by the sensor.

### Table 7.252 - Relations of IDLStruct `sensor_track_type`

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>velocity</code></td>
<td><code>velocity_coordinate_type</code></td>
<td>Association: <code>platform platform_class_type reference [0..1]</code>&lt;br&gt;The sensor's assessment of the track as calculated in the sensor's chosen coordinate system at platform or class of platform of the stated real world object.</td>
</tr>
<tr>
<td><code>velocity_accuracy</code></td>
<td><code>velocity_accuracy_coordinate_type</code></td>
<td>Association: <code>activity platform_activity_type reference [0..1]</code>&lt;br&gt;The sensor's stated accuracy for its calculated velocity. When present, the covariance_matrix attribute should not be present.</td>
</tr>
<tr>
<td><code>velocity_accuracy_coordinate_system</code></td>
<td><code>coordinate_specification_type</code></td>
<td>Association: <code>activity platform_activity_type reference [0..1]</code>&lt;br&gt;The sensor's assessment of the activity being undertaken by the real world object represented by the sensor track as observed by the sensor. The coordinate system chosen by the sensor for reporting accuracy.</td>
</tr>
<tr>
<td><code>velocity_coordinate_system</code></td>
<td><code>coordinate_specification_type</code></td>
<td>Association: <code>activity platform_activity_type reference [0..1]</code>&lt;br&gt;The coordinate system chosen by the sensor.</td>
</tr>
<tr>
<td><code>track_quality</code></td>
<td><code>track_quality_type</code></td>
<td>Association: <code>activity platform_activity_type reference [0..1]</code>&lt;br&gt;The sensor specific quality of this track in comparison to its typical tracks.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Connector</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------</td>
<td>-------</td>
</tr>
<tr>
<td>time_of_first_detection</td>
<td>time</td>
<td>Association: observed_function observable_function_type reference [0..1]</td>
</tr>
<tr>
<td>time_of_last_detection</td>
<td>time</td>
<td>Association: equipment equipment_type reference [0..1]</td>
</tr>
<tr>
<td>priority</td>
<td>Aggregation: external_track_priority_number external_track_number_type [0..*]</td>
<td>A track number for this sensor track from another protocol. The relative priority of a track with regard to the sensor's resources.</td>
</tr>
<tr>
<td>amplitude_strength_type</td>
<td>strength_type</td>
<td>Association: based_on plot_association_type reference [0..*]</td>
</tr>
</tbody>
</table>

### 7.6.9.11 sensor_track_set_type

Type: Class
Package: Track_Reporting
A set of sensor tracks (to enable batch reporting)

#### Table 7.253 - Relations of Class sensor_track_set_type

<table>
<thead>
<tr>
<th>Connector</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation: element sensor_track_type [0..*]</td>
<td></td>
</tr>
</tbody>
</table>

### 7.6.9.12 track_phase_type

Type: Class
Package: Track_Reporting
The detection lifecycle phase of the track

#### Table 7.254 - Attributes of Class track_phase_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>#idEnums DEAD_RECKONED</td>
<td>Track provided based on extrapolated position (dead-reckoned)</td>
</tr>
<tr>
<td>#idEnums DELETED LOST</td>
<td>Track has been deleted. The attribute tag DDS_PSM_Name = NOT_USED Attribute tag DDS_PSM_Notes = Delete enumeration not used for DDS, dispose topic instance instead.</td>
</tr>
<tr>
<td>#idEnums LOST TRACKED</td>
<td>Track has been lost. Regular update of new and existing track</td>
</tr>
</tbody>
</table>
7.5.107.6.10 Tracking_Control

Parent Package: Sensor_Domain

This package contains structs and type defs for managing tracking zones and sensor track information.
Figure 7.58.84 Domain Model (Class diagram)
7.5.10.1 track_info_type
Type: Class
Package: Tracking_Control
This struct identifies track information.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>additional_information</td>
<td>This is additional information that is not specified as part of the interface. Candidate information includes:</td>
</tr>
<tr>
<td></td>
<td>- Track type,</td>
</tr>
<tr>
<td></td>
<td>- Track priority,</td>
</tr>
<tr>
<td></td>
<td>- Track Identification Category Assigned (Pending, Friend, Assumed Friend, Neutral, Unknown, Suspect, Hostile).</td>
</tr>
<tr>
<td>system_track_id</td>
<td>Identifier of the system track being described</td>
</tr>
<tr>
<td>track_priority</td>
<td>The priority assigned to the system track</td>
</tr>
<tr>
<td>identification_assigned</td>
<td>The standard identity of the system track</td>
</tr>
</tbody>
</table>

Example of values:
1. Track While Scan (TWS)
2. Low Priority Target (LPT)
3. High Priority Target (HPT)

7.6.10.2 tracking_zone_set_type
7.5.10.3 Type: Class
Package: Tracking_Control
This struct contains zero to many tracking zones which must be set/reset by the CMS.

7.5.10.4 Table 7.256 - Relations of IDLStruct tracking_zone_set_type
Type: Class
Package: Tracking_Control
This struct identifies a tracking zone.

Table 7.143 – Attributes of Class tracking_zone_type
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>enable boolean</td>
<td>Indicates whether the CMS is enabling or disabling a tracking zone.</td>
</tr>
<tr>
<td>shape general polar_volume_type</td>
<td>This is the polar volume of the zone.</td>
</tr>
<tr>
<td>tracking_type tracking_zone_kind_type</td>
<td>This indicates the tracking zone type.</td>
</tr>
</tbody>
</table>

**7.6.10.3 tracking_zone_type**

**Type:** Class  
**Package:** Tracking_Control  
This struct identifies a tracking zone.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>enable boolean</td>
<td>Indicates whether the CMS is enabling or disabling a tracking zone.</td>
</tr>
<tr>
<td>shape general polar_volume_type</td>
<td>This is the polar volume of the zone.</td>
</tr>
<tr>
<td>tracking_type tracking_zone_kind_type</td>
<td>This indicates the tracking zone type.</td>
</tr>
<tr>
<td>tracking_zone_id tracking_zone_id_type</td>
<td>A unique identifier for the tracking zone.</td>
</tr>
</tbody>
</table>

**7.6.10.4 tracking_zone_kind_type**

**7.5.10.5 Type:** Class  
**Package:** Tracking_Control  
Identifies the kind of a tracking zone.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTOMATIC_TRACK_INITIATION</td>
<td>Zones where the sensor is allowed to auto initiate new tracks. Depending on the sensor type and its capabilities, such a type of zones may be delimited in azimuth only, or both in azimuth and elevation, or may have further range bounds, and in some cases also additional constraints (such as target type, velocity bounds, etc.).</td>
</tr>
<tr>
<td>MULTIPATH_DEVOTED_TRACKING</td>
<td>Sectors where the sensor is required to use, for tracking activities, devoted waveforms to reduce the multipath effects. This capability is usually provided by multifunctional radars. Such a type of sectors is usually limited in azimuth only, below a defined elevation.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>TRACK_ON_JAMMER</td>
<td>Sectors where the sensor is allowed to manage Track-On-Jammer. Depending on the sensor type and its capabilities, such a type of sectors may be delimited either in azimuth only or both in azimuth and elevation.</td>
</tr>
</tbody>
</table>

7.5.10.67.6.10.5 tracking_zone_id_type

Type: Class `abcd`
Package: Tracking_Control
This typedef is used to identify a specific tracking zone.

7.6.7 Radar_Domain

Parent Package: Domain_Model
This package contains the Domain Models for the Air Engagement Support, Engagement Support, Missile Guidance, Search, and Surface Engagement Support services.

7.6.47.7.1 Air_Engagement_Support

Parent Package: Radar_Domain
7.7.1.1 _expected_hit_data_type_

**Type:** _IDLStruct_

**Package:** _Air_Engagement_Support_

Expected hit identifies the target and the time a hit is expected. This data is used to initiate the evaluation of a miss indication within the radar.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>expected_hit_time</em></td>
<td>time_type</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>track_id_descriptor</td>
<td>sensor_track_id_type</td>
</tr>
<tr>
<td></td>
<td>The target track id.</td>
</tr>
</tbody>
</table>

### 7.6.1.2 miss_indication_data_type

**Type:** IDLStruct  
**Package:** Air_Engagement_Support  
Is sent once a hit or miss is noted.

#### Table 7.146 - Attributes of IDLStruct miss_indication_expected_hit_data_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>miss_distance</td>
<td>polar_position_type</td>
</tr>
<tr>
<td></td>
<td>The closest distance of the projectile to the target expressed in polar coordinates.</td>
</tr>
<tr>
<td>kinematics_descriptor</td>
<td>projectile_kinematics_type [1]</td>
</tr>
<tr>
<td>time_stamp</td>
<td>time_type</td>
</tr>
<tr>
<td></td>
<td>The closest time of approach of the projectile to the target.</td>
</tr>
</tbody>
</table>

### 7.7.1.2 miss_indication_data_type

**Type:** IDLStruct  
**Package:** Air_Engagement_Support  
Is sent once a hit or miss is noted.

#### Table 7.261 - Attributes of IDLStruct miss_indication_data_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>miss_distance</td>
<td>polar_position_type</td>
</tr>
<tr>
<td></td>
<td>The closest distance of the projectile to the target expressed in polar coordinates.</td>
</tr>
<tr>
<td>time_stamp</td>
<td>time_type</td>
</tr>
<tr>
<td></td>
<td>The closest time of approach of the projectile to the target.</td>
</tr>
</tbody>
</table>

### 7.6.1.3 7.7.1.3 projectile_kinematics_type

**Type:** IDLStruct  
**Package:** Air_Engagement_Support  
Identifies the kinematics of the projectile that is expected to hit the target.

#### Table 7.147 - Attributes of IDLStruct projectile_kinematics_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_stamp</td>
<td>time_type</td>
</tr>
<tr>
<td></td>
<td>The timestamp when the kinematics was valid/measured.</td>
</tr>
<tr>
<td>position_descriptor</td>
<td>position_coordinate_type</td>
</tr>
<tr>
<td></td>
<td>The projectile's position.</td>
</tr>
<tr>
<td>velocity_descriptor</td>
<td>velocity_coordinate_type</td>
</tr>
<tr>
<td></td>
<td>The projectile's velocity.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>position_descriptor</td>
<td>position_coordinate_type</td>
</tr>
<tr>
<td></td>
<td>The projectile's position.</td>
</tr>
<tr>
<td>velocity_descriptor</td>
<td>velocity_coordinate_type</td>
</tr>
<tr>
<td></td>
<td>The projectile's velocity.</td>
</tr>
</tbody>
</table>
7.6.27.7.2 Engagement_Support

Parent Package: Radar_Domain

To offer flexibility, three variants of coordinate system representation are supported, corresponding to the coordinate_kind_type enumerated. An implementation should support one kind for each service as defined by the coordinate_specification_type value and should only send data of that variant and should check that all data received is of that variant. It should not implement conversion of data in an unexpected variant. Receipt of such data constitutes an error in the operation of the interface. Three representations are supported: time derivatives within a Cartesian coordinate system; time derivatives of a polar coordinate system (range rate, bearing rate etc.); course and speed relative to the WGS84 spheroid.

Figure 7.60.86 Domain Model (Class diagram)
7.7.2.1 available_fire_control_channels_type
7.6.2.1 Type: Class unsigned short
Package: Engagement_Support
The number/amount of available fire control channels.

7.7.2.2 fire_control_channel_id_type
7.6.2.2 Type: Class unsigned short
Package: Engagement_Support
The fire control channel ID as assigned by the subsystem.

7.7.2.3 kill_assessment_result_type
7.6.2.3 Type: Class
Package: Engagement_Support
The possible outcomes of a kill assessment.

Table 7.148263 - Attributes of Class kill_assessment_result_type
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROBABLE_KILL</td>
<td>Kill Probability &gt; 50%</td>
</tr>
<tr>
<td>PROBABLE_MISS</td>
<td>Kill Probability &lt; 50%</td>
</tr>
<tr>
<td>NO_RESULT</td>
<td>Assessment indeterminate</td>
</tr>
</tbody>
</table>

### 7.7.2.4 Kinematics_type

**Type:** IDLStruct  
**Package:** Engagement_Support

Target position/kinematics for which a fire control channel is requested to designate.

### Table 7.264 - Attributes of IDLStruct kinematics_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>coordinate_orientation_type</td>
<td>The orientation of the kinematic coordinates</td>
</tr>
<tr>
<td>cartesian_position_type</td>
<td>The positional element of the kinematics in Cartesian coordinates</td>
</tr>
<tr>
<td>coordinate_origin_type</td>
<td>The origin of the Cartesian coordinate frame</td>
</tr>
<tr>
<td>time_type</td>
<td>The absolute time at which the kinematic information is valid</td>
</tr>
<tr>
<td>cartesian_velocity_type</td>
<td>The velocity element of the kinematics in Cartesian coordinates</td>
</tr>
<tr>
<td>coordinate_kind_type</td>
<td></td>
</tr>
</tbody>
</table>

### 7.6.37.7.3 Missile_Guidance

**Parent Package:** Radar_Domain
The track referred to in a missile guidance command may either be a system track (provided by the CMS) or a sensor track (if the object is already tracked by the sensor). Therefore, the track referred in the missile guidance command may be a sensor_track_id or a system_track_id.

A system track may be based on a sensor track (produced by a sensor on the same platform), but may also be based on a link received track (not modelled).

On the same platform, different objects (targets and own missiles) may be tracked by different sensor types (e.g. 3D radar or ESM). Therefore, for the same interface with a sensor, in successive missile guidance commands, the referred system tracks may be a cartesian point_track at one time and polar bearing_track at the next time.
A system track may be based on a sensor track (produced by a sensor on the same platform), but may also be based on a link received track (not modelled).

On the same platform, different objects (targets and own missiles) may be tracked by different sensor types (e.g. 3D radar, or ESM). Therefore, for the same interface with a sensor, in successive missile_guidance commands, the referred system tracks may be a cartesian point_track at one time and polar bearing_track at the next time.

Figure 7.61.87 Missile Guidance - Track (Class diagram)

The track referred to by a missile guidance command may either be a system track (provided by the CMS) or a sensor track (if the object is already tracked by the sensor). Therefore, the track_id(s) in the missile guidance command may be a sensor_track_id or a missile_track_id.

Figure 7.62.88 Illumination (Class diagram)
Figure 7.63.89 Missile Uplink (Class diagram)
### 7.7.3.1 downlink_report_type

**Type:** Class  
**Package:** Missile_Guidance

Information downlinked by the missile to the radar.

#### Table 7.149265 - Attributes of Class downlink_report_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>own_missile_track_id</td>
<td>track_id_type</td>
</tr>
<tr>
<td>time_of_receipt</td>
<td>time_type</td>
</tr>
<tr>
<td>downlink_content</td>
<td>anonymous_blob_type</td>
</tr>
</tbody>
</table>

#### 7.7.3.2 downlink_request_type

**Type:** Class  
**Package:** Missile_Guidance

request to downlink

#### Table 7.150266 - Attributes of Class downlink_request_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>own_missile_track_id</td>
<td>track_id_type</td>
</tr>
<tr>
<td>listening_period</td>
<td>absolute_duration_type</td>
</tr>
<tr>
<td>frequency_channel</td>
<td>frequency_channel_type [0..1]</td>
</tr>
<tr>
<td>additional_parameters</td>
<td>anonymous_blob_type</td>
</tr>
<tr>
<td>frequency_channel_type</td>
<td>unsigned short</td>
</tr>
<tr>
<td>Attribute</td>
<td>Notes</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>own_missile_track_id</td>
<td>The identifier for track that is representing the system's own missile in the engagement.</td>
</tr>
<tr>
<td>listening_period</td>
<td>The start of the absolute period of time during which downlinks shall be received.</td>
</tr>
<tr>
<td>frequency_channel</td>
<td>Optionally the frequency channel to use for the downlink.</td>
</tr>
<tr>
<td>additional_parameters</td>
<td>System specific information to support the downlink.</td>
</tr>
</tbody>
</table>

#### 7.7.3.3 frequency_channel_type

**Type:** Class *unsigned short*

**Package:** Missile_Guidance

A frequency channel identifies a specific radar frequency.

#### 7.7.3.4 illumination_request_type

**Type:** Class

**Package:** Missile_Guidance

The semantics of selects association is implementation specific.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>target_track_id</td>
<td>The identifier for the target track</td>
</tr>
<tr>
<td>own_missile_track_id</td>
<td>The identifier for track that is representing the system's own missile in the engagement.</td>
</tr>
<tr>
<td>illumination_period</td>
<td>The length of time to provide illumination of the target</td>
</tr>
<tr>
<td>frequency_channel</td>
<td>The frequency channel to use for target illumination</td>
</tr>
<tr>
<td>additional_parameters</td>
<td>System specific information to support the illumination</td>
</tr>
</tbody>
</table>

#### 7.7.3.5 track_id_type

**Type:** Class

**Package:** Missile_Guidance

The track referred to by a missile guidance command may either be a system track (provided by the CMS) or a sensor track (if the object is already tracked by the sensor). Therefore, the track_id(s) in the missile guidance command may be a sensor_track_id or a missile_track_id.
### uplink_report_type

**Type:** IDLStruct

**Package:** Missile_Guidance

A report from uplink

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>sensor_track_id</td>
<td>sensor_track_id_type</td>
</tr>
<tr>
<td>system_track_id</td>
<td>system_track_id_type</td>
</tr>
</tbody>
</table>

Table 7.153269 - Attributes of IDLStruct uplink_report_type
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>own_missile_track_id</td>
<td>track_id_type</td>
</tr>
<tr>
<td></td>
<td>The identifier for track that is representing the system's own missile in the engagement.</td>
</tr>
<tr>
<td>uplink_info</td>
<td>anonymous_blob_type [0..1]</td>
</tr>
<tr>
<td></td>
<td><em>optional</em> System specific information to support the uplink.</td>
</tr>
</tbody>
</table>

### 7.7.3.7 uplink_request_type

**Type:** IDLStruct  
**Package:** Missile_Guidance

a request to downlink

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>own_missile_track_id</td>
<td>track_id_type</td>
</tr>
<tr>
<td></td>
<td>The identifier for track that is representing the system's own missile in the engagement.</td>
</tr>
<tr>
<td>frequency_channel</td>
<td>frequency_channel_type [0..1]</td>
</tr>
<tr>
<td></td>
<td><em>optional</em> Optionally, the frequency channel to use for the uplink.</td>
</tr>
<tr>
<td>request_info</td>
<td>anonymous_blob_type</td>
</tr>
<tr>
<td></td>
<td>System specific information regarding the uplink.</td>
</tr>
</tbody>
</table>

### 7.6.47.7.4 Surface_Engagement_Support

**Parent Package:** Radar_Domain
### Figure 7.65.91 Domain Model (Class diagram)

#### 7.7.4.1 splash_spotting_area_id_type
- **Type**: Class *unsigned short*
- **Package**: Surface_Engagement_Support

#### 7.6.4.1
- **Type**: Class *unsigned short*
- **Package**: Surface_Engagement_Support
the area ID assigned by the sensor.

### 7.7.4.2 splash_spotting_area_position_type

**Type:** Class  
**Package:** Surface_Engagement_Support  
The area definition from the User (CMS) when Splash Spotting is defined using the service "activate splash spotting area by position". The minimum and maximum available sizes are defined in "Manage Subsystem Parameters".

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>azimuth_max</td>
<td>azimuth_coordinate_type when max is less than min, areas covers the north azimuth</td>
</tr>
<tr>
<td>azimuth_min</td>
<td>azimuth_coordinate_type when min is less than max, areas covers the north azimuth</td>
</tr>
<tr>
<td>range_max</td>
<td>range_coordinate_type limited to less than or equal to instrumented range</td>
</tr>
<tr>
<td>range_min</td>
<td>range_coordinate_type limited to greater than or equal to minimum visible range</td>
</tr>
</tbody>
</table>

### 7.7.4.3 splash_spotting_area_set_type

**Type:** Class  
**Package:** Surface_Engagement_Support  
A set consisting of splash spotting areas.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>shape_truncated_sector_type</td>
<td>Shape and size of the splash spotting area</td>
</tr>
<tr>
<td>area_id</td>
<td>Area ID of the splash spotting area</td>
</tr>
</tbody>
</table>

### 7.7.4.4 splash_spotting_area_type

**Type:** Class  
**Package:** Surface_Engagement_Support  
Definition of a single splash spotting area.
Table 7.273 - Attributes of Class splash_spotting_area_type

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>shape truncated_sector_type</td>
<td>Shape and size of the splash spotting area.</td>
</tr>
<tr>
<td>area_id splash_spotting_area_id_type</td>
<td>Area ID of the splash spotting area.</td>
</tr>
</tbody>
</table>

7.7.8 Subsystem_Services

Parent Package: Service_Interfaces
Contains services associated with the Subsystem Domain.

7.7.47.8.1 Encyclopaedic_Support

Parent Package: Subsystem_Services

7.7.47.8.1.1 Receive_Encyclopaedic_Data

Parent Package: Encyclopaedic_Support

Receive_Encyclopaedic_Data_CMS
Type: Interface common_use_case_interface
Package: Receive_Encyclopaedic_Data
This interface describes the process whereby the subsystem receives encyclopedic data. Such data is used by the subsystem to perform self-adaptation to the prevailing environmental conditions. This interface is modelled as a control interaction between the CMS and the subsystem rather than a data flow interaction. The CMS controls the loading of subsystem encyclopaedic data by sending the location of the data, rather than sending the data itself. Of course an implementation may move the encyclopaedic data around a file system beforehand, but that is outside the scope of this standard.

The subsystem is aware of its real-time geographic position and orientation. It is expected that the transfer of this data would be initiated at the start of the ‘mission of the day’. Updates would only be envisaged when the current data set became inapplicable to the current mission. Specific encyclopedic data might be requested by the subsystem. Alternatively, a default set of summary data is sent. Such data, which is an example of ‘reference’ data, would generally be non-sensor in origin and static i.e. not changing in real-time. In the simplest case this data might simply define clutter areas and known jammer locations to assist the subsystem in effecting suitable mitigation for these effects. For a subsystem such as a more complex multi-function radar this might include relevant extracts from a commercial shipping database (Lloyd’s etc.), giving shipping lanes or ship movements or civil airline flight plan data (Civil Aviation Authority etc), locations of wind-farms, major highways, significant structures and potential sources of interference, such as other radars, including consorts, cellular phone masts etc. This data would be used by the subsystem to contribute to the tactical picture. Alternatively, it could be used within the automatic tracking function to enable the identification/elimination from the track picture of non-hostile tracks. Such data could also include, for example, the reference data types communicated via Link 16 such as hazard areas and other fixed point type data. Navigational charts might also be a part of such data. The subsystem VOI (volume of interest) or other filter mechanisms might be supplied in a request from the actor.

Pre-condition: Technical State: The subsystem is in technical state STANDBY, READY or ONLINE. Pre-condition: Mastership Required The CMS has mastership
Pre-condition: Mastership Required: The CMS has mastership
Pre-condition: Subsystem Services: Provide Subsystem Services has completed successfully, in particular this service is available.
Post-condition: Success: The subsystem has received updated Encyclopedic Data
Post-condition: No Success: The subsystem has not received updated Encyclopedic Data

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>encyclopaedic_data_loaded()</td>
<td>The subsystem responds to the CMS that the encyclopaedic data previously requested has been loaded.</td>
<td>request_id_type, request_id The unique id for this request - corresponds to the parameter in the load_encyclopaedic_data request</td>
</tr>
</tbody>
</table>

Table 7.15274 - Methods of Interface Receive_Encyclopaedic_Data_CMS

Receive_Encyclopaedic_Data_Sub
Type: Interface
Package: Receive_Encyclopaedic_Data

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>load_encyclopaedic_data()</td>
<td>The CMS requests the subsystem to load encyclopaedic data of a</td>
<td>request_id_type, request_id The unique identifier for this request</td>
</tr>
</tbody>
</table>

Table 7.15275 - Methods of Interface Receive_Encyclopaedic_Data_Sub
**url_type**
The location of the file containing the encyclopaedic data.

**data_descriptor_type**
A description of the type of encyclopaedic data (e.g., name of the data set). It is expected that implementations will specify a list of descriptors known to particular subsystems. Such a list may be accessible at run-time through the Manage Subsystem Parameters interface.

```
load_encyclopaedic_data(request_id_type, url_type, data_descriptor_type)
```

- **Negative Acknowledgement**
- **Positive Acknowledgement**
- **receive_error(request_id_type, error_reason_type)**

```
«idlInterface»
Receive_Encyclopaedic_Data_CMS
```

```
«idlInterface»
Receive_Encyclopaedic_Data_Sub
```
Figure 7.66.92 Alternate Flow - Receive Encyclopaedic Data (Interaction diagram)
### 7.7.2.8.2 Extended_Subsystem_Control

Figure 7.67.93 Basic Flow - Receive Encyclopaedic Data (Interaction diagram)
Parent Package: Subsystem_Services
Contains interfaces for the Extended Subsystem Control service.

7.7.2.17.8.2.1 Manage_Physical_Configuration

Parent Package: Extended_Subsystem_Control
Contains operations and sequence diagrams for the Manage Physical Configuration interface.

Manage_Physical_Configuration_CMS
Type: Interface common_use_case_interface
Package: Manage_Physical_Configuration

The purpose of this interface is to provide a mechanism to exchange a physical configuration data file between a subsystem and the CMS (potentially xml format). The exact format of the file is subsystem specific. The purpose of the file is to support the maintainer with facilities to configure the internal parts of the subsystem; also to be used as integration support.

Additional Information:

There are at least two cases where the CMS would provide a sub-system’s physical configuration. Case 1 is when the sub-system was able to detect a configuration change and the data must be manually entered in sub-system configuration data (e.g. a servo type and serial number). Case 2 is when the sub-system is being developed and changes to the configuration which cause changes in system behavior are being tested.

Pre-condition: Subsystem must be in a STANDBY state in order for the CMS to request changes to Physical Configuration Data. This pre-condition does not apply if the CMS is only requesting current Physical Configuration Data to be provided by the subsystem.

Pre-condition: CMS must have mastership in order for the CMS to request changes to Physical Configuration Data. This pre-condition does not apply if the CMS is only requesting current Physical Configuration Data to be provided by the subsystem.

Post-condition: For a change in Physical Configuration Data Request, configuration data is properly updated.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_physical_config()</td>
<td>Interface used by CMS to receive a url to access physical configuration data from the subsystem.</td>
<td>configuration_url_type configuration_url request_id_type request_id</td>
</tr>
<tr>
<td>receive_physical_configuration_success()</td>
<td>Interface used by CMS to receive an indication from the subsystem that it has successfully changed its physical configuration data.</td>
<td>request_id_type request_id</td>
</tr>
</tbody>
</table>

Manage_Physical_Configuration_Sub
Type: Interface
Package: Manage_Physical_Configuration

Table 7.150276 - Methods of Interface Manage_Physical_Configuration_CMS

Table 7.160277 - Methods of Interface Manage_Physical_Configuration_Sub
<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>change_physical_configuration()</td>
<td>Interface used by the subsystem to receive requests from the CMS to change its physical configuration data to align with data located at the url specified in the request.</td>
<td>request_id_type request_id configuration_url_type configuration_url</td>
</tr>
<tr>
<td>provide_physical_configuration()</td>
<td>Interface used by the subsystem to receive requests from the CMS to provide its current physical configuration data.</td>
<td>request_id_type request_id</td>
</tr>
</tbody>
</table>
Figure 7.68.94 Manage Physical Configuration - Change (Interaction diagram)

Flow of events which depicts the CMS requesting that the subsystem changing its physical configuration data (also depicts alternate rejection and error paths).
provide_physical_configuration(request_id_type)
Flow of events which depicts the CMS requesting that the subsystem report on its current physical configuration data (also depicts alternate rejection and error paths).

Parent Package: Extended_Subsystem_Control
Contains the interface for offline testing.

Type: Interface common_use_case_interface
Package: Perform_Offline_Test

This is used to instruct the subsystem to perform offline test and return the results to the CMS. The nature of the offline tests is subsystem specific.

Pre-condition: Provide Subsystem Services must have executed successfully. Pre-condition: The CMS must have Mastership.
Pre-condition: The CMS must have Mastership.
Pre-condition: The subsystem may be in any Technical State except for ONLINE.
Post-condition: For the response FAILED, the subsystem transitions to Technical State FAILED, but otherwise remains in the previous Technical State.

Table 7.161278 - Methods of Interface Perform_Offline_Test_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_detailed_test_result()</td>
<td>Provides the CMS with subsystem specific information concerning offline test failures</td>
<td>request_id_type request_id offline_test_result_details_type offline_test_detailed_results</td>
</tr>
<tr>
<td>receive_test_results()</td>
<td>Informs the CMS whether the offline tests passed, passed partially, or failed.</td>
<td>request_id_type request_id offline_test_result_type test_results</td>
</tr>
</tbody>
</table>

Perform_Offline_Test_Sub
Type: Interface
Package: Perform_Offline_Test

Table 7.162279 - Methods of Interface Perform_Offline_Test_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>perform_tests()</td>
<td>Instructs the subsystem to perform the offline tests.</td>
<td>request_id_type request_id offline_test_type test_name Allows a particular test to be selected. If null, all tests are performed.</td>
</tr>
<tr>
<td>request_detailed_test_results()</td>
<td>Asks the subsystem to provide detailed information on the failures.</td>
<td>request_id_type request_id</td>
</tr>
</tbody>
</table>

Open Architecture Radar Interface Standard (OARIS), v2.0
Perform offline tests for subsystem

- Testing starts but fails to complete for some reason
- Receive acknowledgement
- Receive error

Test request is rejected for some reason
- Receive acknowledgement
- Receive test results

Detailed results required?

- Request accepted, processing succeeds
- Detailed results returned

In the event of a partial pass or failure, detailed results from the last test may be requested.

The subsystem executes the offline tests

- Request detailed test results
- Receive detailed test results
- Receive acknowledgement

Optionally, the subsystem may request test results

- Receive test results
- Receive acknowledgement
7.7.2.3 **Restart**

**Manage Network Participation**

- **Parent Package:** Extended_Subsystem_Control
- **Package:** Manage_Network_Participation
- **Type:** Interface common_use_case

The purpose of this interface is to provide a mechanism for a CMS to manage the connectivity of the OARIS data exchange through some external network gateway represented by the Subsystem interface in this use case. When connectivity is established, information can be exchanged between the local CMS and local Subsystems with other CMS and Subsystems connected by this network as if they were locally connected from a functional viewpoint.

- **Package:** Restart

The purpose of this interface is to cause a normal transition to STANDBY and then to READY states as defined by Manage Technical State.

---

**Figure 7.70.96 Perform Offline Test (Interaction diagram)**

This shows the required sequential behaviour for Perform_Offline_Test. See diagram embedded notes for further explanation.
Pre-condition: Sub-system is in ONLINE, READY, FAILED, BIT, or CALIBRATION

Pre-condition: CMS has mastership of sub-system

Post-condition: Sub-system is in READY state if successful, otherwise current state is reported by subsystem.

Additional Information:
The management of such network connectivity may be integral for the sharing of plot data between distributed platforms.

Table 7.163280 - Methods of Interface RestartManage_Network_Participation_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_restart_state()</td>
<td>Interface used by CMS to receive an indication from the subsystem that it has successfully performed restart.</td>
<td>request_id_type, request_id, technical_state_type, technical_state</td>
</tr>
</tbody>
</table>

Restart

Table 7.164281 - Methods of Interface RestartManage_Network_Participation_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>perform_restart()</td>
<td>Interface used by the subsystem to receive a request from the CMS to execute a restart.</td>
<td>request_id_type, request_id</td>
</tr>
</tbody>
</table>

Type: Interface
Package: RestartManage_Network_Participation
perform_restart

receive_acknowledgement(request_id, request_ack)

receive_restart_state(request_id, technical_state_type)
7.7.2.4 • Shutdown

<table>
<thead>
<tr>
<th>Method</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_network_status()</td>
<td>String <code>network_name</code> The name identifying the network request id_type <code>request_id</code> The unique identifier for this specific request</td>
<td></td>
</tr>
<tr>
<td>join_network()</td>
<td>String <code>network_name</code> The name identifying the network to be joined</td>
<td></td>
</tr>
</tbody>
</table>
The unique identifier for the specific request
subsystem id The unique identifier of the subsystem to join the network

String network name The name identifying the network to be left
request id The unique identifier for the specific request
subsystem id The unique identifier for the subsystem to leave the network

Figure 7.97 Manage Network Participation (Class diagram)
Figure 7.98 Manage_Network_Participation - alternate flow - unable to join (Interaction diagram)

- **Failure Case**: 
  - **join fails**: join_network(String, request_id_type, subsystem_id_type) 
  - **status request fails**: request_network_status(String, request_id_type) 
  - **leave fails**: leave_network(String, request_id_type, subsystem_id_type) 
  - **network fails**: report_network_status(String, subsystem_id_type, subsystem_id_type)

- **accepted = false**: The platform the CMS is controlling is unable to join the network (e.g. it is not in the current plan for the network, or the network name is not recognized).

- **accepted = false**: The subsystem is unable to report the network status (e.g. the network name is not recognized).
7.8.2.4 Startup

**Parent Package:** Extended_Subsystem_Control

Contains operations and sequence diagrams for the Startup Shutdown interface. The purpose of this interface is to transition the subsystem to the STANDBY state from any other state as defined by Manage Technical State. Note: This shall cause the Subsystem to cease radiating if it is in an ONLINE state with emissions enabled.

**Pre-condition:** Subsystem is in ONLINE, READY, FAILED, BIT, or CALIBRATION

**Pre-condition:** CMS has mastership of subsystem.
Post-condition: Sub-system is in STANDBY state if successful, otherwise the current state is reported by
the subsystem.

Table 7.165 – Methods of Interface Shutdown_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_shutdown_state()</td>
<td>Interface used by CMS to receive an indication from the subsystem that it has successfully performed shutdown.</td>
<td>request_id_type, request_id, technical_state_type, technical_state</td>
</tr>
</tbody>
</table>

**Shutdown_Sub**

| Type: Interface  | Package: Shutdown |

Table 7.166 – Methods of Interface Shutdown_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>perform_shutdown()</td>
<td>Interface used by the subsystem to receive a request from the CMS to execute a shutdown.</td>
<td>request_id_type, request_id</td>
</tr>
</tbody>
</table>

Figure 7.73 Basic Flow - Shutdown (Interaction diagram)

Basic flow for CMS requesting the subsystem to transition to STANDBY.
7.7.2.5

**• Startup**

**Parent Package:** Extended_Subsystem_Control

Contains operations and sequence diagrams for the Startup interface, Startup_CMS

**Type:** Interface common_use_case_interface

**Package:** Startup

The purpose of this interface is to cause a normal transition from the STANDBY state to the READY state using the transitions defined in the Manage Technical State service.
Pre-condition: Subsystem is in STANDBY State. Pre-condition: CMS has mastership of subsystem.
Pre-condition: CMS has mastership of subsystem.
Post-condition: Subsystem is in READY state if successful. If not execute successful, current state shall be reported by subsystem.

**Table 7.46282 - Methods of Interface Startup_CMS**

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_startup_state()</td>
<td>Interface used by CMS to receive an indication from the subsystem that it has successfully performed startup.</td>
<td>request_id_type request_id technical_state_type technical_state</td>
</tr>
</tbody>
</table>

**Startup_Sub**
Type: Interface
Package: Startup

**Table 7.468283 - Methods of Interface Startup_Sub**

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>perform_startup()</td>
<td>Interface used by the subsystem to receive a request from the CMS to execute startup.</td>
<td>request_id_type request_id</td>
</tr>
</tbody>
</table>
Figure 7.75.100 Basic Flow -Startup (Interaction diagram)
Basic flow for CMS requesting the subsystem to transition from STANDBY to READY.
perform_startup(request_id_type)

[Subsystem rejects request to startup]

receive_acknowledgement(request_id_type, request_ack_type)

receive_acknowledgement(request_id_type, request_ack_type)

receive_error(request_id, error_reason)

receive_startup_state(request_id_type, technical_state_type)

receive_startup_state(request_id_type, technical_state_type)

respond_is

successfully

acknowledged but
fails before

completion

Alternative Flows

[Subsystem rejects request to startup]

receive_acknowledgement(request_id_type, request_ack_type)
Figure 7.7.101 Alternative Flow - Startup (Interaction diagram)
Alternate flow for CMS requesting the subsystem to transition from STANDBY to READY (depicts rejection and error paths).

7.7.3 Recording_and_Replay

7.8.2.5 Provide_Networking_Statistics
Parent Package: Extended_Subsystem_ServicesControl
Contains operations and sequence diagrams for the interface controlling recording and replay.

7.7.3.1 Control_Recording
Parent Package: Recording_and_Replay
Contains the Provide Bandwidth_Statistics interface controlling the recording of information.

Control_Recording_CMS
Provide_Networking_Statistics_CMS
Type: Interface common_use_case_interface
Package: Control_Recording
The interface describes how the CMS controls the recording of information. Such information may be used to support...
• Setting to Work/Commissioning
• Equipment monitoring
• Performance monitoring and evaluation
• ‘Black Box’ recording
• Safety of Life at Sea (SOLAS) recording
For the purposes of this interface, 'recording' is defined as the synchronous capture of real-time information at a defined rate. Provision of additional live real-time data for instrumentation purposes, i.e. for display rather than recording, is outside the scope.

Each record within the recording must be identified and time-stamped.

The operation of the recording function must not affect normal operation of the subsystem. Package: Provide Networking Statistics

This is used to inform the CMS of the bandwidth being used by and quality of service achieved by the Subsystem (e.g. an off-platform communications and/or networking device).

Pre-condition: Provide Subsystem Services must have executed successfully.

Pre-condition: The Subsystem must be in Technical State READY or ONLINE.

Pre-condition: The CMS must have Mastership.

Post-condition: After successful termination, the recording is available for replay via Control_Replay, using the identifier specified.

Post-condition: In the case of abnormal termination, there is a possible fault in the recording subsystem.

Control_Recording_Sub

Type: Interface

Package: Control_Recording

Table 7.469284 - Methods of Interface Control_Recording_SubProvide_Networking_Statistics_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>define_recording_set()</td>
<td>Specifies what is to be recorded</td>
<td>request_id_type, request_id, recording_set_type, recording_parameters_list</td>
</tr>
<tr>
<td>start_recording()</td>
<td>Starts the recording as specified.</td>
<td>request_id_type, request_id, recording_id_type_id</td>
</tr>
<tr>
<td>stop_recording()</td>
<td>Stops the recording</td>
<td>request_id_type, request_id</td>
</tr>
</tbody>
</table>
This shows the required sequential behaviour for Control_Recording. See diagram embedded notes for further explanation.

7.7.3.21.1.1.1 Control_Replay

Parent Package: Recording_and_Replay

Contains the interfaces controlling the replay of information, either using the original interfaces or as a data dump for offline processing.

Control_Replay_CMS

Type: Interface common use case interface

Package: Control_Replay

This interface defines how the CMS controls the replay of information previously recorded using Control_Recording.

Replay is supported in two modes: REAL-TIME and RAW. REAL-TIME mode is used to replay in real time, or at a multiple of real time, data that was visible on other OARIS interfaces via the interfaces used during recording. RAW mode is used to replay data that was visible on other OARIS interfaces and/or internal...
subsystem data that was not available on other OARIS interfaces. In this case the data is merely transferred to the CMS as a set of time-tagged values with no attempt made to reconstruct real-time behaviour. One or more recordings must have been made using Control_Recording.
For simplicity, concurrent recording and replay is not supported.

- Pre-condition: Provide Subsystem Services must have executed successfully.
- Pre-condition: The subsystem must be in Technical State READY or ONLINE.
- Pre-condition: The CMS must have Mastership.
- Pre-condition: In the case of abnormal termination, there is a possible fault in the replay subsystem.

<table>
<thead>
<tr>
<th>Table 7.170 - Methods of Interface Control_Replay_CMS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Method</strong></td>
</tr>
<tr>
<td>end_of_recording()</td>
</tr>
<tr>
<td>receive_recording()</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Control_Replay_Sub</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type: Interface</td>
</tr>
<tr>
<td>Package: Control_Replay</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 7.171 - Methods of Interface Control_Replay_Sub</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Method</strong></td>
</tr>
<tr>
<td>resume_replay()</td>
</tr>
<tr>
<td>start_replay()</td>
</tr>
<tr>
<td>stop_replay()</td>
</tr>
</tbody>
</table>

Open Architecture Radar Interface Standard (OARIS), v2.0
| upload_recording | Requests transfer of a raw recording | request_id_type | request_id | recording_id_type | recording_id |
This shows the required sequential behaviour for Control_Replay using real-time mode. See diagram embedded notes for further explanation.
7.7.4 Simulation_Support
Parent Package: Subsystem_Services

7.7.4.1 Define_Simulation_Scenario
Parent Package: Simulation_Support

Define_Simulation_Scenario_CMS
Type: Interface
Package: Define_Simulation_Scenario

This describes how the contents of a simulation scenario are communicated between the CMS and the subsystem.

Figure 7.79
Control Replay (RAW) (Interaction diagram)
This shows the required sequential behaviour for Control_Replay using raw mode. See diagram embedded notes for further explanation.

This shows the required sequential behaviour for Control_Replay using raw mode, See diagram embedded notes for further explanation.
The CMS provides the subsystem with a simulated environment which consists of simulated objects of different kinds.

A subsystem with built-in simulation capability may participate in this simulation not only by being a consumer of the simulated environment but by contributing actively to it.
Radar type subsystems shall typically build simulated plots or tracks from the simulated environment, while contributing simulated electromagnetic emissions to it. These simulated emissions may in turn be used and detected by other (ESM type) simulations.

Weapon type subsystems when in simulation mode shall typically contribute simulated objects to the simulation that represent the launching and movement of own missiles, bullets or torpedoes and their effect on other simulated objects.

Thus CMS and subsystem both contribute to the simulated environment. Together they form a simulation federation.

The actor is the Combat Management System. Relationship to ‘control simulation’

The definition of simulation mode and the flow of commands to start/stop/resume a simulation scenario are defined in ‘control simulation’.

Relationship to provision of tracks

A radar type subsystem shall provide tracks based on information from the simulated environment, as described above. The interfaces that deal with the provision of tracks indicate whether tracks are simulated or not under amplifying information. This indication should be set for all tracks that are reported in the context of this interface.

Relationship to Receive geographic information

Geographic information is received by using ‘Receive geographic information’.

Pre-condition: Subsystem health state. The subsystem and the relevant subsystem services need to be in the health state AVAILABLE or DEGRADED.

Pre-condition: CMS has mastership.

Pre-condition: Subsystem simulation mode.

- The subsystem must be in subsystem simulation mode ON to participate in the simulation federation.

Pre-condition: Simulation scenario started.

Pre-condition: Geographic information.

The subsystem may need geographic information about its simulated surroundings available locally or by means of other interfaces in order to calculate the detectability or reachability of simulated objects due to obstacles in the surroundings.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
</table>
| write_emitter_system_data_CMS | informs the CMS of the most recent bandwidth utilization | anonymous_blob_type emitter_system_data
| bandwidth_statistics() | | percentage_type peak_utilization The greatest utilization of bandwidth since the last update
| | | percentage_type mean_utilization The average utilization of bandwidth since the last update
| | | device_name_type connection The device specific to the connection to which the statistic pertain
| | | absolute_duration_type duration The period of time for which the statistic apply
| write_radar_beam_data_latency_statistics() | informs the CMS of the most recent latency associated with data transfer | anonymous_blob_type radar_beam_data
| latency_type peak_latency The greatest latency |
The average latency experienced across the network scope since the last update.

`time_type` mean `latency` The average latency experienced across the network scope since the last update.

`device_name_type` connection The device specific to the connection to which the statistic pertain.

`absolute_duration_type` `period_of_validity` The period of time for which the statistics apply.

---

**Figure 7.102** Basic Flow - Provide Networking Statistics (Interaction diagram)

### 7.8.2.6 Shutdown

**Parent Package:** Extended Subsystem Control

**Define_Simulation_Scenario_Shutdown**

Contains operations and sequence diagrams for the Shutdown interface,

**Shutdown_CMS**

**Type:** Interface

**Package:** Define_Simulation_Scenario_Shutdown

The purpose of this interface is to transition the sub-system to the STANDBY state from any other state as defined by Manage Technical State. Note: this shall cause the Subsystem to cease radiating if it is in an ONLINE state with emissions enabled.
Pre-condition: Subsystem is in ONLINE, READY, FAILED, BIT, or CALIBRATION

Pre-condition: CMS has mastership of subsystem.
Post-condition: Sub-system is in STANDBY state if successful, otherwise the current state is reported by the subsystem.

Table 7.173285 - Methods of Interface Define_Simulation_Scenario_SubShutdown_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_shutdown_state()</td>
<td>Interface used by CMS to receive an indication from the subsystem that it has successfully performed shutdown.</td>
<td>request_id_type, request_id, technical_state_type, technical_state</td>
</tr>
<tr>
<td>write_emitter_system_data()</td>
<td>Write emitter system data</td>
<td>anonymous_blob_type emitter_system_data</td>
</tr>
<tr>
<td>write_environment_data()</td>
<td>Write environment data</td>
<td>anonymous_blob_type environmental_entity_data</td>
</tr>
<tr>
<td>write_jammer_beam_data()</td>
<td>Write jammer beam data</td>
<td>anonymous_blob_type jammer_beam_data</td>
</tr>
<tr>
<td>write_platform_data()</td>
<td>Write platform data</td>
<td>anonymous_blob_type platform_data</td>
</tr>
</tbody>
</table>
All information is exchanged upon event or change in no specific order.

write_platform_data(anonymous_blob_type)
write_emitter_system_data(anonymous_blob_type)
write_jammer_beam_data(anonymous_blob_type)
write_environment_data(anonymous_blob_type)

Figure 7.80 Basic Flow – Define Simulation Scenario Data (interaction diagram)
7.7.4.2 Control_Simulation

Parent Package: Simulation_Support

Control_Simulation_CMS

Type: Interface common_use_case_interface
Package: Control_Simulation

This service controls the simulation mode of a subsystem. This simulation mode is independent of the operational mode of the subsystem. Simulation mode is either ON or OFF. "ON" has different meanings for different kinds of subsystems. Effector type subsystems shall not engage real targets but shall simulate the engagement instead. Sensor type subsystems may be fed with simulated targets which shall be reported as plots or tracks. In each case while in simulation mode "ON" the subsystem shall strictly avoid any impact on the environment that could be the result if simulation mode was "OFF".

The actor is the Combat Management System. Basic Flow—Control simulation mode

Start event: command of simulation mode

The service is triggered by the actor. The actor commands the simulation mode which may be one of the following:

- **ON**: This indicates that the subsystem shall operate in simulation mode
- **OFF**: This indicates that the subsystem shall stop operating in simulation mode and that any current simulation shall be terminated

On occurrence of the trigger provision of subsystem simulation mode is executed.

Provision of subsystem simulation mode

After receipt of the simulation mode from the actor the subsystem responds with its subsystem simulation mode.

The subsystem simulation mode may be one of the two:

- write_emitter_system_data(anonymous_blob_type)
- write_radar_beam_data(anonymous_blob_type)

All information is exchanged upon event or change in specific order.
ON: This indicates that the subsystem is operating in simulation mode
OFF: This indicates that the subsystem is not operating in simulation mode
Basic Flow – Control Simulation (Start/Resume, Stop/Freeze)
START/RESUME simulation scenario: Only when in simulation mode ON:
Upon provision of the START/RESUME command by the actor the simulation scenario starts or is resumed after a previously issued FREEZE.
STOP/FREEZE simulation scenario: Only when in simulation mode ON:
Upon provision of the STOP/FREEZE command by the actor the simulation scenario stops or stays frozen.
The service ends.

Provision on initialization
The simulation mode shall be provided by the actor after initialization of the CMS.
The flow of information relevant to subsystem simulation are the subject of another service: Define simulation scenario.
If simulation is stopped or frozen simulation time of the subsystem and the actor shall be also stopped. The synchronization of simulation time may be performed using START/RESUME command.
Pre-condition: CMS has mastership.

Package: Shutdown

Table 7.174286 - Methods of Interface Control_Simulation_CMS Shutdown_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>perform_shutdown()</td>
<td>Interface used by the subsystem to receive a request from the CMS to execute a shutdown.</td>
<td>request_id_type request_id</td>
</tr>
</tbody>
</table>
Figure 7.104 Alternative Flow - Shutdown (Interaction diagram)
Alternate flow for CMS requesting the subsystem to transition to STANDBY (depicts rejection and error paths):

**Restart**

- sim_mode_status()  
- Receive the status and mode of simulation.
- perform_shutdown(request_id_type)
- receive_acknowledgement(request_id_type, request_ack_type)
- receive_error(request_id, error_reason)
- receive_shutdown_state(request_id_type, technical_state_type)
- receive_acknowledgement(request_id_type, request_ack_type)

7.8.2.7

**Parent Package:** Extended_Subsystem_Control_Simulation_Sub  
Contains operations and sequence diagrams for the Restart interface.

**Type:** Interface common_use_case_interface
**Package: Restart**

The purpose of this interface is to cause a normal transition to STANDBY and then to READY states as defined by Manage Technical State.

**Pre-condition:** Sub-system is in ONLINE, READY, FAILED, BIT, or CALIBRATION

**Package: Control_Simulation**

**Pre-condition:** CMS has mastership of sub-system;

**Post-condition:** Sub-system is in READY state if successful, otherwise current state is reported by subsystem.

<table>
<thead>
<tr>
<th>Table 7.175287 - Methods of Interface Restart_CMSControl_Simulation_Sub</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Method</strong></td>
</tr>
<tr>
<td>start_resume_session()</td>
</tr>
<tr>
<td>start_stop_sim_mode()</td>
</tr>
<tr>
<td>stop_freeze_session()</td>
</tr>
</tbody>
</table>
Figure 7.82 Basic Flow - Control Simulation Start/Resume (Interaction diagram)

Figure 7.83 Basic Flow - Control Simulation Stop/Freeze (Interaction diagram)
7.7.4.3 Define_Fault_Scripts

Parent Package: Simulation_Support

**Define_Fault_Scripts_CMS**

receive_restart_state() - Interface used by CMS to receive an indication from the subsystem that it has successfully performed restart.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>start_stop_sim_mode(request_id_type, start_stop_sim_mode_request_type)</td>
<td></td>
<td>request_id_type request_id technical_state_type technical_state</td>
</tr>
<tr>
<td>receive_acknowledgement(request_id_type, request_ack)</td>
<td>request_ack.success == true</td>
<td></td>
</tr>
<tr>
<td>receive_error(request_id_type, error_reason_type)</td>
<td></td>
<td>[Rejected by CMS]</td>
</tr>
<tr>
<td>sim_mode_status(request_id_type, sim_mode_status_type)</td>
<td>receive_acknowledgement(request_id_type, request_ack)</td>
<td>[Accepted by CMS]</td>
</tr>
</tbody>
</table>
perform_restart() Interface used by the subsystem to receive a request from the CMS to execute a restart.

request_id_type request_id

Figure 7.105 Basic Flow - Restart (Interaction diagram)

Basic flow for CMS requesting the subsystem to transition to STANDBY followed by a transition to READY.
Figure 7.106 Alternative Flow - Restart (Interaction diagram)
Alternate flow for CMS requesting the subsystem to transition to STANDBY followed by a transition to READY (depicts rejection and error paths):
The script to be added
fault_script_ids_type: The id of the fault scripts to be
removed
request_id: The request id of the subsystem's simulation.
remove_fault_script(): Removes the given fault scripts from
the subsystem's simulation.
add_fault_script(request_id_type, fault_script_type): Applies
to add fault scripts as well
remove_error(request_id_type, error_reason_type): Negative
acknowledgement
receive_error(request_id_type, error_reason_type): Positive
acknowledgement
receive_acknowledgement(request_id_type, request_ack_type): Positive
acknowledgement
receive_rejected(request_id_type, request_reject_type): Rejected
Ack

figure 7.85 alternative flow - define fault scripts (interaction diagram)
### 7.7.4.7.8.3 Control_Fault_Scripts

**Parent Package:** Simulation_Support

**Type:** Interface common use case interface

**Package:** Control_Fault_Scripts

This enables a trainee, at a CMS Console, to cause the generation of predefined fault messages for training purposes (see also Define Fault Scripts). The subsystem shall output Fault Reports to the CMS, which a trainee may respond to via the CMS Console. Fault clearance messages shall also be sent to the CMS in response to the trainee taking the appropriate action.

- **Pre-condition:** Technical State Subsystem is in technical state READY or ONLINE
- **Pre-condition:** Fault Script Subsystem has a fault script which has been defined previously
- **Pre-condition:** Mastership Required. The CMS has Mastership

**Pre-condition:** Subsystem Services Provide Subsystem Services has successfully completed; in particular this service is available

**Post-condition:** Success Subsystem has provided simulated fault and response to clearance action

**Post-condition:** Failure Subsystem has not provided simulated fault and response to clearance action

Control_Fault_Scripts Sub

---

**Figure 7.86 Basic Flow - Define Fault Scripts (Interaction diagram)**
### Type: Interface

**Package:** Control_Fault_Scripts

#### Table 7.178 - Methods of Interface Control_Fault_Scripts_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>enable_fault_script(request_id_type, fault_script_ids_type)</td>
<td>Causes the subsystem to indicate the faults specified by the given fault request_id_type and fault_script_ids_type. The script ids to be enabled.</td>
<td>request_id_type, request_id fault_script_ids_type, scripts.</td>
</tr>
<tr>
<td>clear_faults(request_id_type, fault_script_ids_type)</td>
<td>Clears the faults defined by the given fault scripts. The script ids to be cleared.</td>
<td>request_id_type, request_id fault_script_ids_type, fault_scripts.</td>
</tr>
</tbody>
</table>

#### Diagram

![Diagram](image-url)
Figure 7.87 Alternative Flow - Control Fault Scripts (Interaction diagram)
7.7.5 Subsystem_Control
Parent Package: Subsystem_Services Contains interfaces for the Subsystem Control service.

7.7.5.1 Manage_Technical_State
Parent Package: Subsystem_Control

7.8.3.1 Contains operations and sequence diagrams for the Manage_Technical_State interface.
Manage_Technical_State_CMS
Parent Package: Subsystem_Control
Contains operations and sequence diagrams for the Manage Technical State interface.
Manage_Technical_State_CMS
Type: Interface common_use_case_interface
Package: Manage_Technical_State
Manage Technical State causes the subsystem to provide or change its technical state.

Special Requirements:
Initialization: Upon initialization, reset or power-on, the sub-system shall transition to a pre-defined state and report the current state to the CMS.

Additional Information:
If a critical component of the subsystem becomes NOT AVAILABLE, the technical state shall transition to FAILED.
All states may transition to OFFLINE, but the subsystem shall only do so in emergency situations or catastrophic damage, to indicate an uncontrolled shutdown.

*Startup, Shutdown, and Restart* explain the sequence of actions for nominal progression through the technical states.

Pre-condition: If the CMS requests a Technical State to change, mastership of the subsystem is required.
Pre-condition: CMS is aware of the current subsystem state.
Pre-condition: CMS is aware of the possible technical states supported by the subsystem.
Post-condition: None.

Table 7.129289 - Methods of Interface Manage_Technical_State_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_periodic_technical_state()</td>
<td>Interface used by CMS to receive periodic technical state reports from the subsystem.</td>
<td>technical_state_type_technical_state</td>
</tr>
<tr>
<td>receive_technical_state()</td>
<td>Interface used by CMS to receive technical state reports from the subsystem which were the result of a transition request from the CMS.</td>
<td>request_id_type request_id technical_state_type_technical_state</td>
</tr>
</tbody>
</table>

Manage_Technical_State_Sub
Type: Interface
Package: Manage_Technical_State

Table 7.180200 - Methods of Interface Manage_Technical_State_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>change_technical_state()</td>
<td>Interface used by the subsystem to receive requests from the CMS to change its technical state.</td>
<td>request_id_type request_id technical_state_type_technical_state</td>
</tr>
<tr>
<td>provide_technical_state()</td>
<td>Interface used by the subsystem to receive requests from the CMS to provide its current technical state.</td>
<td>request_id_type request_id</td>
</tr>
</tbody>
</table>
Flow of events which depicts the CMS requesting that the subsystem changing its current technical state.
alt
Alternative Flows
(Invalid State Condition Requested)
receive_acknowledgement(request_id, request_ack)
[Subsystem Rejects State Change Request]
receive_acknowledgement(request_id, request_ack)
[State Change Unsuccessful]
receive_error(request_id_type, error_reason_type)
receive_technical_state(request_id_type, technical_state_type)
command is successfully acknowledged but fails before completion
Figure 7.90.108 Alternative Flow - Manage Technical State - Change (Interaction diagram)
Alternate flow depicting rejection and error cases for a CMS requesting the subsystem to change its Technical State.
Figure 7.51.109 Basic Flow - Manage Technical State - Periodic Reporting (Interaction diagram)

Flow of events which depicts a subsystem that periodically reports its technical state (without the need for a CMS request).
Flow of events which depicts the CMS requesting that the subsystem report on its current technical state.

### 7.7.5.27.8.3.2 Heartbeat_Signal

**Parent Package:** Subsystem_Control

**Type:** Interface

The service describes how the availability of an established communication between CMS and the subsystem as well as the subsystem itself shall be monitored. The heartbeat signal is triggered by Control Interface Connection. The basic flow is asynchronous.

The actor is the Combat Management System.

Pre-condition: Connection established; Provide Subsystem Services has successfully established communication between CMS and the subsystem.

Post-condition: Interface is alive: The heartbeat has been received successful.

Post-condition: Interface is not alive: The heartbeat has not been received.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receiveSubsystemHeartbeatSignal(request_id_type)</td>
<td>Receive the periodic heartbeat signal to verify, that the connection is still alive.</td>
<td>unsigned long count This parameter is used with implementation specific semantics for monitoring interface participant liveliness.</td>
</tr>
</tbody>
</table>

**Heartbeat_Signal_Sub**

**Type:** Interface

**Package:** Heartbeat_Signal
<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_cms_heartbeat_signal()</td>
<td>Receive the periodic heartbeat signal to verify, that the connection is still alive.</td>
<td>unsigned long count This parameter is used with implementation specific semantics for monitoring interface participant liveness.</td>
</tr>
</tbody>
</table>

![Diagram of heartbeat signal methods]

**Diagram:**
- `receive_cms_heartbeat_signal()`: Sends a CMS heartbeat signal periodically.
- `receive_subsystem_heartbeat_signal()`: Sends a subsystem heartbeat signal periodically.

**Legend:**
- `loop_periodic()`: Loop function for periodic heartbeat signal reception.
7.8.3.3 Provide_Subsystem_Identification

Parent Package: Subsystem_Control

Provide_Subsystem_Identification_CMS

Type: Interface common_use_case_interface

Package: Provide_Subsystem_Identification

In order to enable two interface partners to connect to each other and to open mutual communication, one partner shall initiate and the other to answer. The intention is to let the subsystem initiate the communication. Consequently, the subsystem introduces itself to the CMS identifying e.g. the type of subsystem, the product and its version. That allows the CMS to decide whether it may work with that subsystem.

The actor is the Combat Management System.

The possibility that CMS and subsystem are connected without being capable to work with each other is a consequence of a plug-&-play concept. Although the interface is standardized the CMS may need a setup process to prepare it for a subsystem. This process shall introduce the information necessary to configure functions of that particular CMS with respect to the subsystem.
This may also be necessary on side of the subsystem.
The preparation for a subsystem may be done by means of system configuration data which are
implemented on installation of the combat system. It does not address security information.
Pre-condition: CMS and Subsystem can communicate with each other.
Post-condition: CMS and subsystem may work together. CMS and subsystem have verified that
they may work with each other.
They shall do some organization regarding the communication (out of scope).
Post-condition: CMS and subsystem may not work together. The interface between CMS and
subsystem is closed.

Table 7.183-293 - Methods of Interface Provide_Subsystem_Identification_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
</table>
| receive_sub_identification_data() | Receive the identification data from the subsystem. | device_identification_type
|                                 |                                                 | _identification
|                                 |                                                 | request_id_type _the_request_id |

Provide_Subsystem_Identification_Sub
Type: ___________Interface common_use_case_interface
Package: Provide_Subsystem_Identification

Table 7.184-294 - Methods of Interface Provide_Subsystem_Identification_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
</table>
| receive_cms_identification_data() | Receive the identification data from the CMS.   | device_identification_type
|                              |                                                 | _identification
|                              |                                                 | request_id_type _the_request_id |
```
provide_subsystem_identification CMS

receive_subsystem_identification_data(device_identification_type, request_id_type)
expected = false
expected = true
expected = false

receive_acknowledgement(request_id_type, request_ack_type)

receive_cms_identification_data(device_identification_type, request_id_type)

Alternative Flows

expected = false

expected = false
```

Open Architecture Radar Interface Standard (OARIS), v2.0
Figure 7.94.112 Alternative Flow - Introduction of subsystems (Interaction diagram)
7.7.5.47.8.3.4 Provide Health State

Parent Package: Subsystem_Control

Figure 7.95.113 Basic Flow - Introduction of the subsystem (Interaction diagram)
Provide_Health_State_CMS

Type: Interface common_use_case_interface
Package: Provide_Health_State

The service allows the CMS to monitor and evaluate the health state of the subsystem. The health state information describes functional availability of the subsystem and the services it provides.

The service may be triggered by several possible situations:
- Periodic event, for example by internal clock,
- Actor (CMS) request,
- Health state change,
- Initialization (start-up),
- Recovery of the subsystem after a failure.

In addition to the health state being provided, additional information may be provided to the CMS. In case of a service, the information may include a list of detected faults. In case of a subsystem, the information may include the list of services together with their health state, and for every service which has health state other than AVAILABLE, a list of detected faults. This two dimensional structure is called the service availability matrix.

The state NOT AVAILABLE may also describe the situation in which the service is not implemented. In this case the list of faults shall be empty. In the state UNKNOWN, the subsystem may provide the reason for not being able to evaluate health state (e.g. BIT process not running).

The service ends with success when the health state (possibly accompanied by additional information) is provided to the actor.

Relationship to technical state.
The reported health state of the services is dependent on the technical state.
In the technical state ONLINE, the health state of the services is determined based on the detected faults (if any).
In all technical states other than ONLINE (except OFFLINE), the health state of all services, except the service Subsystem_Control, is NOT AVAILABLE.
The health state of the service Subsystem_Control shall then be DEGRADED, since some functions (e.g. Control Battle Override) are not available in those technical states, and some functions are (e.g. Manage Technical State).
In the technical state OFFLINE no communication at all is possible with the CMS so the health state is not reported.

Relationship to battle override.
When Battle Override is set (see service Control Battle Override), certain faults are not taken into account when determining the health state. These overridable faults generally refer to circumstances that may cause damage to own equipments, but do not prohibit executing the requested task.

Relationship to simulation mode.
If the subsystem is in Simulation mode (technical state is ONLINE), only the faults for parts needed for the simulated execution of the service are taken into account when determining the health state of a service. For instance, if the transmitter is defective, the service Track_Reporting is reported AVAILABLE when in Simulation mode, but is reported NOT AVAILABLE when not in Simulation mode.
Faults may also be simulated for training purposes (see service Define Fault Script). Therefore, irrespective of the Simulation mode, all faults (real and simulated) are included in the reported list of detected faults, each with an indication whether the fault is real or simulated.
If a real system part is simulated, faults of the simulated part should have a different identification.
For instance (see previous example) in Simulation mode, a simulated transmitter could be used, for which the trainer has inserted a simulated fault.
Any faults in the real transmitter would be reported (real fault) as well as the injected fault in the simulated
transmitter (simulated fault). However, the health state of the service Track_Reporting would be based only on the status of the simulated transmitter.

Reason for health state
Each reported health state other than AVAILABLE is accompanied by the reason(s) for that health. In this way the CMS may for instance derive that although the technical state of the subsystem is STANDBY (and NOT AVAILABLE for that reason), there are also faults that would prevent the service to become AVAILABLE when the technical state would be switched to ONLINE.

Pre-condition: Subsystem technical state: The subsystem is in technical state ONLINE or READY.
Post-condition: CMS awareness; CMS is aware of the health state of the subsystem and/or its services.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_fault()</td>
<td>Report a fault to CMS</td>
<td>fault_type the_fault</td>
</tr>
<tr>
<td>report_service_health()</td>
<td>Report health of service</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>service_health_type health</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fault_list_type the_fault_list</td>
</tr>
<tr>
<td>report_subsystem_health()</td>
<td>Report health of subsystem</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>subsystem_health_type health</td>
</tr>
</tbody>
</table>

Table 7.185295 - Methods of Interface Provide_Health_State_CMS

Provide_Health_State_Sub
Type: Interface
Package: Provide_Health_State

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_service_health()</td>
<td>Request service health</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>service_name_type service_name</td>
</tr>
<tr>
<td>request_subsystem_health()</td>
<td>Request subsystem health</td>
<td>request_id_type request_id</td>
</tr>
</tbody>
</table>

Table 7.186296 - Methods of Interface Provide_Health_State_Sub
Figure 7.36.114 Basic Flow - Fault Reporting (Interaction diagram)
on subsystem initiative
report_service_health(request_id_type, service_health_type, fault_list)

Service health provision on CMS request

[alternative flow: processing failed]
receive_acknowledgement(request_id_type, request_ack_type)
request_ack.accepted = true
receive_error(request_id_type, error_reason_type)
request_ack.accepted = false
w: request rejected
receive_acknowledgement(request_id_type, request_ack_type)

[alternative flow]
request_service_health(request_id_type, service_name_type)

Service health provision on subsystem initiative due to:
- Initialization (start-up)
- Recovery after failure
- Health state change
- Periodic (timed)

«idlInterface» Provide_Health_State_Sub
«idlInterface» Provide_Health_State_CMS
Service health provision on subsystem initiation due to:
- Initialization (start-up)
- Recovery after failure
- Health state change
- Periodic (timed)

Service health provision on CMS request

Figure 7.97.115 Basic Flow - Service Health Reporting (Interaction diagram)
Subsystem health provision on CMS request:

- request_ack.accepted = true

For all services provided by this subsystem:

- report_service_health(request_id_type, service_health_type, fault_list)

[Alternative flow: processing failed]

- receive_acknowledgement(request_id_type, request_ack_type)
- receive_error(request_id_type, error_reason_type)
- request_ack.accepted = true
- request_ack.accepted = false
  - request_rejected
- receive_acknowledgement(request_id_type, request_ack_type)

Subsystem health provision on subsystem initiative due to:

- Initialization (start-up)
- Recovery after failure
- Health state change
- Periodic (timed)
Subsystem health provision on subsystem initiative due to:
- Initialization (start-up)
- Recovery after failure
- Health state change
- Periodic (timed)

Alternative flow:
- Request rejected
- Processing failed

Request Acknowledgement
- Request Acknowledgement
- Request Acknowledgement

Sequence Diagram: Subsystem Health Reporting

Service health and corresponding fault lists shall accompany subsystem health report only when subsystem health is reported on request. For subsystem health provision on subsystem initiative, the service health and corresponding fault lists shall be reported on subsystem initiative separately (see sequence diagram Service Health Reporting).

Figure 7.98.116 Basic Flow - Subsystem Health Reporting (Interaction diagram)
7.7.5.57.3.5 Manage_Operational_Mode
Operational modes summarise a set of subsystem parameters optimising the subsystem with respect to an operational purpose.

The names of modes of a specific type of subsystem (e.g., a radar) differ from supplier to supplier. Consequently, they shall be handled as configuration parameters. They shall be offered to the operator to enable him for a selection and shall be transferred to the subsystem to achieve the intended reaction.

The definition of names of operational modes is not within the scope of this standard.

It is the CMS’s responsibility to initiate the determination of initial state by making a request for information to the subsystem.

In the case where the CMS does not have mastership of the subsystem, a change of the operational mode shall be indicated by informing the CMS about the new operational mode (see service "Provide health state").

Configuration data like the set of available operational modes may be received at runtime but may also be inserted by means of an automatic or manual setup process. Although automatic runtime transfer of such information may be achieved through "Manage Subsystem Parameters" it is not a mandatory requirement of this standard for that mechanism to be used.

Pre-condition: Technical state READY or ONLINE;
Pre-condition: "Manage Subsystem Parameters" executed successfully;
Pre-condition: CMS must have Mastership;
Post-condition: Service ends with success: - the subsystem is in the commanded operational state, the CMS is informed that this is the case
Post-condition: Service ends with fail: - the subsystem is still in the original operational state, the CMS has the correct information regarding that state.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_operational_mode()</td>
<td>The current operational mode is reported via this interface method.</td>
<td>request_id, type, operational_mode, current_mode</td>
</tr>
<tr>
<td>configure_operational_function()</td>
<td>function_type function A function operated by the subsystem</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Notes</td>
<td>Parameters</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
<td>------------</td>
</tr>
<tr>
<td>request_get_operational_mode()</td>
<td>The subsystem is requested to report the current operational mode.</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td>request_get_operational_mode()</td>
<td>The subsystem is requested to report the current operational mode.</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td>request_set_operational_mode()</td>
<td>The subsystem is requested to change the current operational mode to the given new operational mode.</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td>request_set_operational_mode()</td>
<td>The subsystem is requested to change the operational mode to the given new operational mode.</td>
<td>request_id_type request_id</td>
</tr>
</tbody>
</table>

**Manage_Operational_Mode_CMS**

Type: ActivityPartition  
Package: Manage_Operational_Mode

**Manage_Operational_Mode_Sub**

Type: ActivityPartition  
Package: Manage_Operational_Mode

![Diagram of interface management](image_url)
This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "get current operational mode" of the service "Manage Operational Mode".
alt operational mode change
request_set_operational_mode(request_id_type, operational_mode_type)

receive_acknowledgement(request_id_type, request_ack_type)
receive_error(request_id_type, error_reason_type)

request_set_operational_mode(request_id_type, operational_mode_type)

report_operational_mode(request_id_type, operational_mode_type)

For spontaneous operational mode change, request_id == 0.

'request_set_operational_mode' is the current operational mode that differs from the requested mode.
This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "set operational mode" of the service "Manage Operational Mode".

7.7.5.67.8.3.6 _Control_Battle_Override_

Parent Package: Subsystem_Control

Parent Package: Subsystem_Control

This package contains interfaces for the Control Battle Override service.

Control_Battle_Override_CMS

Type: Interface common_use_case_interface

Package: Control_Battle_Override

The subsystem is requested to set/reset the Battle Override. When Battle Override is set the subsystem disregards warnings on circumstances which may cause damage to own equipment, typically the overtemperature protections.

It is the CMS's responsibility to initiate the determination of initial state by making a request for information to the subsystem.

Provision of the Battle Override state

Subsystem shall keep CMS informed about the current Battle Override state and its changes (if any).
Lack of mastership
In the case where CMS does not have mastership of the subsystem, CMS shall be informed about the current Battle Override state and its changes (if any).
current Battle Override state and its changes (if any).

Relationship to the subsystem health state
As long as the Battle Override is set, the subsystem internal overtemperature indications shall not result in
any heath state set to “NOT AVAILABLE” (see Provide health state).
Pre-condition: Mastership Required: CMS has mastership of the subsystem
Pre-condition: Subsystem Services: Provide subsystem services has been completed successfully.
Post-condition—Success: The subsystem Battle Override is set/reset as requested and CMS is
informed that this is the case.
Post-condition—No Success: The subsystem Battle Override is still equal to the original one and
CMS has the correct information regarding that state.

Table 7.180399 - Methods of Interface Control_Battle_Override_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
</table>
| battle_override_setting()| This method is used by the subsystem to return the current Battle Override state. | request_id_type request_id
battle_override_state_type
battle_override_state |

Control_Battle_Override_Sub
Type: Interface
Package: Control_Battle_Override

Table 7.180300 - Methods of Interface Control_Battle_Override_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
</table>
| set_battle_override()   | This method is used by the CMS to send a Battle Override set/reset request to the subsystem, | request_id_type request_id
battle_override_state_type
battle_override_state |

Set_Battle_Override_CMS
Request_Battle_Override_CMS
Control_Battle_Override_CMS
Figure 7.101.119 Basic Flow - Control Battle Override - Set/Reset (Interaction diagram)
Figure 7.10.1 Old Alternative Flow - Control Battle Override - Set/Reset - loss of mastership (Interaction diagram)
7.8.3.7 Manage_Subsystem_Parameters

Parent Package: Subsystem_Control

Manage_Subsystem_Parameters_CMS

Type: Interface common_use_case_interface

Package: Manage_Subsystem_Parameters

The service allows the actor to obtain and modify the values of parameters of the subsystem. It also provides the facilities to retrieve the descriptions of parameters available in a certain subsystem.

The actor of the service is the Combat Management System.

The service starts when the CMS requests one of the following:

- Parameter value retrieval
- Parameter value modification
- Retrieval of parameter descriptor, with a list of parameter names (and values in case of modification).

A parameter value may be structured (e.g. a vector or a table).

The service ends when the subsystem has provided the requested information or modified the parameter value.

It is the CMS's responsibility to initiate the determination of initial state by making a request for information to the subsystem.

Parameter names used by a subsystem are to be unique within the scope of that subsystem. Requests for parameter descriptions and to get and set current values are consequently well-defined. Parameter names may be structured using a namespace scheme to promote uniqueness.

Unknown parameter
On receipt of a request for parameter value retrieval, parameter value modification or parameter descriptor retrieval for an unknown parameter name, the subsystem responds with an indication “unknown parameter”. Other (correctly identified) parameters in the same request are processed as requested.

Illegal parameter value
On receipt of a request for parameter value modification with a parameter value that is outside the allowable range of the specified parameter, the subsystem responds with an indication “illegal parameter value” and does not change the parameter value.
This includes inconsistencies of parameter type (e.g. real where integer is expected) and structure (e.g. vector of 2 elements, where a vector of 3 is expected).
Other parameters with legal values in the same request are modified as requested.
In case of an illegal value for an element of a structured parameter, the entire parameter remains unchanged.

Modification of parameter value
A parameter value may only be modified in the technical state(s) as specified in the descriptor of that parameter.

Security
Access to the service may be restricted to certain parts of the CMS because of security restrictions. Pre-condition: Subsystem technical state. The subsystem is in a technical state other than OFFLINE. Pre-condition: Mastership. The CMS has mastership of the subsystem in case of parameter value modification.

Pre-condition: Subsystem technical state. The subsystem is in a technical state other than OFFLINE.
Pre-condition: Mastership. The CMS has mastership of the subsystem in case of parameter value modification.

Table 7.194301 - Methods of Interface Manage_Subsystem_Parameters_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_parameter_values()</td>
<td></td>
<td>request_id_type, request_id, name_value_sequence_type, the_name_value_set, name_error_sequence_type, the_name_error_set</td>
</tr>
<tr>
<td>report_parameter_descriptors()</td>
<td></td>
<td>request_id_type, request_id, descriptor_sequence_type, the_descriptor_sequence, name_error_sequence_type, the_name_error_set</td>
</tr>
</tbody>
</table>

Manage_Subsystem_Parameters_Sub
Type: Interface
Package: Manage_Subsystem_Parameters

Table 7.192302 - Methods of Interface Manage_Subsystem_Parameters_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>retrieve_parameter_values()</td>
<td></td>
<td>request_id_type, request_id</td>
</tr>
</tbody>
</table>
modify_parameter_values()  
request_id  
name_value_sequence_type  
the_name_value_set

retrieve_parameter_descriptors()  
request_id  
parameter_name_sequence_type  
the_name_set

retrieve_parameter_values()  
request_id  
parameter_name_sequence_type  
the_name_set

If name_sequence is empty, all shall be retrieved
[alternative flow: request rejected]

receive_acknowledgement()  
request_ack_type

receive_error()  
error_reason_type

receive_acknowledgement()  
request_ack.accepted = true

report_parameter_values()  
alt

[basic flow]
Figure 7.103.121 Basic Flow - Parameter Retrieval (Interaction diagram)
Manage_Subsystem_Parameters_CM

modify_parameter_values(request_id_type, name_value_sequence_type)
Mastership is required for modification of parameters.
Not satisfying this precondition shall lead to rejection of the request.

receive_acknowledgement(request_id_type, request_ack_type)

For each of the parameters in the name_value_sequence, the subsystem shall check whether:
- the parameter has a known parameter name,
- the new parameter value is valid,
- the parameter may be modified in the subsystems actual technical state,
- the parameter may be modified in the subsystems actual operational mode.
Each parameter not satisfying all conditions shall not be modified (for structured parameters all elements need to satisfy these conditions), and a corresponding name_error_pair shall be returned in the name_error_sequence.

Parameters satisfying the conditions shall be modified directly during the processing of the request, taking into account that for structured parameters all elements shall be modified at the same moment, and a corresponding name_value_pair shall be returned in the name_value_sequence.

request_ack.accepted = true

report_parameter_values(request_id_type, name_value_sequence_type, name_error_sequence_type)

[alternative flow: processing failed]

receive_acknowledgement(request_id_type, request_ack_type)
request_ack.accepted = false

[alternative flow: request rejected]

receive_error(request_id_type, error_reason_type)
request_ack.accepted = true
Mastership is required for modification of parameters. Not satisfying this precondition shall lead to rejection of the request.

For each of the parameters in the name_value_sequence the subsystem shall check whether:
- the parameter has a known parameter name,
- the new parameter value is valid,
- the parameter may be modified in the subsystem's actual technical state,
- the parameter may be modified in the subsystem's actual operational mode.

Each parameter not satisfying all conditions shall not be modified (for structured parameters all elements need to satisfy these conditions), and a corresponding name_error_pair shall be returned in the name_error_sequence.

Parameters satisfying the conditions shall be modified directly during the processing of the request, taking into account that for structured parameters all elements shall be modified at the same moment, and a corresponding name_value_pair shall be returned in the name_value_sequence.
receive_acknowledgement(request_id_type, request_ack_type)

report_parameter_descriptors(request_id_type, descriptor_sequence, name_error_sequence_type)

If the name_sequence is empty, all shall be retrieved

receive_error(request_id_type, error_reason_type)

request_ack.accepted = true
7.8.3.8 Provide_Subsystem_Services

7.7.5.8 Parent Package: Subsystem_Control

PackageTag: No_PSM = DDS
Provide_Subsystem_Services_CMS
Type: Interface common_use_case_interface
Package: Provide_Subsystem_Services

Subsystems offer a number of services to a CMS. Some of the services are mandatory for the type of subsystem, others are optional. New services may be known to the CMS or may not be known. Consequently, the CMS needs to know which services are provided by a subsystem and the subsystem needs to know which services the CMS is able to interact with.

The services considered here are the final versions of those that are specified and defined by the rest of this standard. Some of them are not necessarily implemented by each product of the type of subsystem but also not necessarily supported by each CMS.

The service-related information provided by the subsystem to the CMS deals with both, the interfaces offered by the subsystem and the interfaces expected on CMS side which are necessary to use the service.

Lack of mastership
Mastership of the subsystem must not have an impact upon this interface.

Plug-&-Play aspect
Both sides, subsystem and CMS, shall follow a technical evolution process which is not necessarily
coordinated. Therefore, the latest subsystem version may provide a service which is not yet supported by the CMS or the CMS may be prepared to use a service which is not provided by the subsystem. This may also cause inconsistencies regarding the interfaces to be made available on both sides. As the subsystem may not have an own operator display, it is intended to use the health state of the subsystem if an indication at CMS is to be achieved saying that the interface to the CMS is not implemented properly.

Configuration data of services
The information to be provided to the CMS as information about the implemented services may include related configuration data and may include the information which parts of the service interfaces are supported.

System integration test
After installation of a subsystem on-board, connecting the hardware interfaces with the related CMS hardware interfaces and performing a setup process if applicable it is expected that an interface verification procedure shall be performed. This procedure shall apply all negotiated interfaces so that an improper implementation shall turn-up at that occasion, already. Insofar, the alternative flows should be considered as an integration aid, only.

Spontaneous reporting
Interfaces for which registration/de-registration is considered as an optional facility are written, accordingly. Registration/de-registration of recipients is done using standard registration mechanism (register interest).

Pre-condition: Subsystem identification. Provide subsystem identification has been passed successfully. Post-condition: The CMS is aware of the services and related interfaces supported by the subsystem.

Pre-condition: Subsystem identification. Provide subsystem identification has been passed successfully.

Post-condition: The CMS is aware of the services and related interfaces supported by the subsystem.

Post-condition: The subsystem is aware of the service-related interfaces the CMS may interact with.

Post-condition: The Services do not match. Each of the alternative flows indicates a fatal error which means that the interface is not implemented properly. The CMS does not take any further action but alerts the operator, accordingly.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_implemented_services()</td>
<td>Receive services which are implemented by a subsystem</td>
<td>request_id_type, the_request_id, service_indication_list_type, service_indication_list</td>
</tr>
</tbody>
</table>

**ProvideSubsystem_Services_Sub**

Type: Interface *common-use-case-interface*

Package: *ProvideSubsystem_Services*

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_supported_services()</td>
<td>Receive services which are supported by the CMS</td>
<td>request_id_type, the_request_id, service_list_type, supported_service_list</td>
</tr>
</tbody>
</table>
Figure 7.106.124 Alternative Flow - Service negotiation (Interaction diagram)
Figure 7.107.125 Basic Flow - Service negotiation (Interaction diagram)
Manage_Mastership

Parent Package: Subsystem_Control
This package contains interfaces for the Manage Mastership service.
Manage_Mastership_CMS

Type: Interface common_use_case_interface

Package: Manage_Mastership

Besides the CMS, the subsystem may be controlled via other control points, e.g. the subsystem local control unit. This interface describes how the CMS, as any other actor, shall handle the exclusive control of the subsystem (mastership). In fact, every subsystem may be controlled by only one actor at the same time. Only the actor who has the mastership of a subsystem may have exclusive control of the subsystem. Exclusive control means that the subsystem may accept only commands sent by the actor who has its mastership.

The subsystem Mastership may be acquired in two ways:

1. PERIODIC MASTERSHIP REQUEST: The actor who wants to acquire the mastership of a subsystem send to it a periodic Mastership request; the subsystem may accept or deny. Once acquired, the subsystem Mastership is released giving up the periodic Mastership requests sending. This happens both in case of intentional decision and critical event as CMS unavailability or connection loss. As long as CMS wants to maintain the Mastership of the subsystem, it shall continue the periodic Mastership requests sending. The CMS is informed about the Mastership control state by receiving a periodic message sent by the subsystem.

2. ASYNCHRONOUS MASTERSHIP REQUEST: The actor who wants to acquire the mastership of a subsystem send to it an asynchronous request. The subsystem may accept or deny. Once acquired, the mastership is until the mastership owner decides to intentionally release it or until a critical event, which is mastership owner unavailability or connection failure, occurs. In case of intentional mastership release, the CMS shall send an asynchronous mastership release request. In case of critical event, the mastership of the subsystem is automatically released. This happens when the subsystem does no longer receive the CMS heartbeat. The CMS is informed about the Mastership control state by receiving an asynchronous message sent on change by the subsystem.

Mastership management rules
The subsystem Mastership assignmen is controlled by the subsystem itself according to the following rules:

• no more than one Master at any time, so the subsystem may not be commanded by more than one control point
• the actor which wants to acquire the subsystem Mastership shall ask the subsystem for it, so no request no assignment
• the subsystem assigns the Mastership to any actor asking for it without any priority policy, no actor is "more important" than any other.
• on each request, the mastership may be assigned only if it's free, that is not already assigned (unless a Mastership override request is received)

The Mastership management protocol is managed as follows:

• in case of periodic request for Mastership assignment, as long as the actual Master wants to maintain the Mastership, it shall continue the periodic Mastership requests sending
• if the actual Master wants to release the Mastership in case of periodic request for Mastership management, it shall give up the periodic Mastership requests sending, otherwise, in case of asynchronous request, it shall send an asynchronous request for mastership release
subsystem keeps informed about the actual Mastership state and its changes (if any).
At any time the subsystem Mastership may be either "free", that is assigned to none and then available to anybody asks for it, or assigned to somebody, where this somebody may be CMS or not. At the subsystem power-on the Mastership is "free", then:

- as long as the Mastership state is "free", the first received Mastership request shall be satisfied (whether the requestor is CMS or not)
- as long as the Mastership is assigned (to CMS or to somebody other than CMS), the current Master shall maintain the Mastership possession until the Mastership owner is no longer available or decides to release it
- as long as the Mastership is assigned (to CMS or to somebody other than CMS), Mastership requests received from other than the current Master shall be no satisfied, unless a Mastership Override is received, which shall force a Mastership switch to another Master

Note that the Mastership possession is required to control the subsystem (e.g. execute write commands to it), but it is not required to communicate with subsystem and receive information from it.

Mastership Override
The Mastership management protocol could include a Mastership Override to force a Mastership switch from one Master to another one.

Pre-condition: Subsystem Services: Provide subsystem services is successfully passed
Post-condition: Success: The subsystem Mastership state is assigned to CMS or not assigned to CMS, according to the CMS requests, and CMS is informed about.
Post-condition: No Success: The subsystem Mastership state is not according to the CMS requests and CMS has the correct information regarding that state (except in the case of connection loss).

Table 7.185305 - Methods of Interface Manage_Mastership_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_mastership()</td>
<td>This method is used by the subsystem to return the mastership state.</td>
<td>mastership_state_type_control_state</td>
</tr>
</tbody>
</table>

Manage_Mastership_Sub
Type: Interface
Package: Manage_Mastership

Table 7.196306 - Methods of Interface Manage_Mastership_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>acquire_mastership()</td>
<td>This method is used by the CMS to acquire the mastership.</td>
<td>unsigned long count</td>
</tr>
<tr>
<td>release_mastership()</td>
<td>This method is used by the CMS to release the mastership.</td>
<td>unsigned long count</td>
</tr>
</tbody>
</table>
semantics to manage subsystem mastership.
The diagram shows the interaction between two interfaces: `Manage_Mastership_CMS` and `Manage_Mastership_Sub`.

- **acquire_mastership()**
  - This function attempts to acquire mastership.

- **request_ack.success = true**
  - Indicates successful request acknowledgment.
  - `request_ack_type` is also specified.

- **report_mastership_state()**
  - Reports the current mastership state.

- **receive_acknowledgement(request_id, request_ack_type)**
  - Receives an acknowledgment for a request.
  - The acknowledgment type is specified.

- **receive_error(request_id, error_reason)**
  - Handles errors related to requests.
  - The error reason is specified.

- **report_error(request_id, error_reason)**
  - Reports an error.
  - The error reason is specified.

- **receive_acknowledgement(request_id_type, request_ack_type)**
  - Receives an acknowledgment for a request.
  - Both request ID and acknowledgment type are specified.

- **receive_mastership_error(error_reason)**
  - Handles mastership errors.
  - The error reason is specified.

- **request_ack_success = true**
  - Indicates successful request acknowledgment.

The diagram illustrates the flow of messages and error handling between the two interfaces, including the handling of mastership requests and errors.

Additional notes:
- The subsystem no longer receives Heartbeat from CMS (CMS unavailability or connection loss).
- CMS unavailability or connection loss results in the subsystem returning the current mastership state as not assigned to CMS at timeout expiration.
- If a request is not acknowledged successfully or acknowledged but fails before completion, it is ejected.
Figure 7.108.126 Basic Flow - Mastership Acquisition - asynchronous request (Interaction diagram)
The subsystem no longer receives Heartbeat from CMS (CMS unavailability or connection loss). The subsystem enters the current Mastership state as not assigned to CMS, at timeout expiration.

loop

request = receive_acknowledgement(request_id_type, request_ack_type)

report_mastership_setting (mastership State_Type)

receive_acknowledgement(request_id_type, request_ack_type)

alt

[basic flow]

acquire_mastership()
Figure 7.109.127 Basic Flow - Mastership Acquisition - periodic request (Interaction diagram)
Figure 7.110.128 Basic Flow - Mastership Release - asynchronous request (Interaction diagram)
Figure 7.111.129 Basic Flow - Mastership Release - periodic request (Interaction diagram)
7.7.5.10 Register_Interest

Parent Package: Subsystem_Control

PackageTag: No_PSM = DDS

Register_Interest_CMS

Type: Interface common_use_case_interface

Package: Register_Interest

This service allows the CMS to register (and deregister) interest in other services. It is explicitly meant to address the possibility of CMS "subscribing" to information supplied by the subsystem, with the understanding that the information shall be provided by the subsystem, without the need for further request. Such mode of operation may be applicable for those services, which have been reported as such in Provide subsystem services. This includes typically track and plot reporting services, but may involve other services as well.

The service starts when the actor registers interest in information provided by a service. The registration shall include information on:

- The service for which the actor wants to register / deregister his interest
- The information within the service for which the actor wants to register / deregister his interest
- The intended (direct or indirect) recipient(s) of the information provided by the subsystem.
- Any parameters of the provision needed such as Quality of Service parameters.

The service ends when the subsystem confirms registration / deregistration of interest.

Pre-condition: Sensor health state: The sensor and the service need to be in the health state AVAILABLE or DEGRADED.
### Table 7.307 - Methods of Interface Register_Interest_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>confirm_registration()</td>
<td>Confirm registration of interest</td>
<td>request_id_type request_id</td>
</tr>
</tbody>
</table>

Register_Interest_Sub
Type: Interface
Package: Register_Interest

### Table 7.308 - Methods of Interface Register_Interest_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>register_interest()</td>
<td>Register interest in the service</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>interest_list_type the_interest_list</td>
</tr>
</tbody>
</table>

[Diagram of interface interactions]
7.8.4 Sensor Recording and Replay

7.8 Parent Package: Subsystem Services
Contains the interfaces controlling recording and replay.

7.8.4.1 Control Recording
Parent Package: Recording and Replay
Contains the interface controlling the recording of information.
Control Recording CMS
Type: Interface
Package: Control Recording
The interface describes how the CMS controls the recording of information. Such information may be used to support:
• Setting-to Work/Commissioning
• Equipment monitoring
• Performance monitoring and evaluation
• ‘Black Box’ recording
• Safety of Life at Sea (SOLAS) recording
• De-briefing
For the purposes of this interface, ‘recording’ is defined as the synchronous capture of real-time information at a defined rate. Provision of additional ‘live’ real-time data for instrumentation purposes, i.e. for display rather than recording, is outside the scope.

Each record within the recording must be identified and time-stamped.

The operation of the recording function must not affect normal operation of the subsystem.

For simplicity, concurrent recording and replay is not supported.

Pre-condition: Provide Subsystem Services must have executed successfully.

Pre-condition: The subsystem must be in Technical State READY or ONLINE.

Post-condition: The CMS must have Mastership.

Post-condition: After successful termination, the recording is available for replay via Control_Replay, using the identifier specified.

Post-condition: In the case of abnormal termination, there is a possible fault in the recording subsystem.

Control Recording Sub
Type: Interface
Package: Control Recording

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>define_recording_set()</td>
<td>Specifies what is to be recorded</td>
<td>request_id_type request_id recording_set type recording_parameters_list</td>
</tr>
<tr>
<td>start_recording()</td>
<td>Starts the recording as specified. Note that only one recording may be running at a time.</td>
<td>request_id_type request_id recording_id_type id</td>
</tr>
<tr>
<td>stop_recording()</td>
<td>Stops the recording</td>
<td>request_id_type request_id</td>
</tr>
</tbody>
</table>
This shows the required sequential behaviour for Control Recording. See diagram embedded notes for further explanation.

7.8.4.2 Control Replay

Parent Package: Recording and Replay

Contains the interfaces controlling the replay of information; either using the original interfaces or as a data dump for offline processing.

Control Replay CMS

Type: Interface

Package: Control Replay

This interface defines how the CMS controls the replay of information previously recorded using Control Recording.

Replay is supported in two modes: REAL-TIME and RAW. REAL-TIME mode is used to replay in real time, or at a multiple of real-time, data that was visible on other OARIS interfaces via the interfaces used during recording. RAW mode is used to replay data that was visible on other OARIS interfaces and/or internal subsystem data that was not available on other OARIS interfaces. In this case the data is merely transferred to the CMS as a set of time-tagged values with no attempt made to reconstruct real-time behaviour. One or more recordings must have been made using Control Recording. For simplicity, concurrent recording and replay is not supported.
**Pre-condition:** Provide Subsystem Services must have executed successfully.
**Pre-condition:** The subsystem must be in Technical State READY or ONLINE.
**Pre-condition:** The CMS must have Mastership.
**Pre-condition:** In the case of abnormal termination, there is a possible fault in the replay subsystem.

### Table 7.310 - Methods of Interface Control_Replay_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>end_of_recording()</td>
<td>The subsystem has reached the end of the recording before a stop command was received.</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td>receive_recording()</td>
<td>Used to transfer a raw recording to the CMS.</td>
<td>request_id_type request_id recording_type requested_recording The raw recording data.</td>
</tr>
</tbody>
</table>

**Control_Replay_Sub**  
Type: Interface  
Package: Control_Replay

### Table 7.311 - Methods of Interface Control_Replay_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>resume_replay()</td>
<td>Resumes replay following a stop command</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>actual_time_type actual_time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The current time (time of day) at which playback should start. This allows synchronisation of playback from different subsystems.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>replay_speed_type replay_speed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Controls the replay speed. 1.0 represents real time.</td>
</tr>
<tr>
<td>start_replay()</td>
<td>Starts replay as specified</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>replay_set_type replay_parameters_list</td>
</tr>
<tr>
<td></td>
<td></td>
<td>recording_type actual_time_type recorded_time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The time in the recording at which playback should start. This allows synchronisation of playback from different subsystems.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>replay_speed_type replay_speed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Controls the replay speed. 1.0 represents real time.</td>
</tr>
<tr>
<td>stop_replay()</td>
<td>Stops replay</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td>upload_recording()</td>
<td>Requests transfer of a raw recording</td>
<td>request_id_type request_id recording_id_type</td>
</tr>
</tbody>
</table>
Figure 7.132 Control Replay (Interaction diagram)
This shows the required sequential behaviour for Control Replay using real-time mode. See diagram embedded notes for further explanation.
This shows the required sequential behaviour for Control_Replay using raw mode. See diagram embedded notes for further explanation.

### 7.8.5 Simulation Support

**Parent Package:** Subsystem_Services

#### 7.8.5.1 Define Simulation Scenario

**Parent Package:** Simulation_Support

**Define Simulation Scenario_CMS**

**Type:** Interface

**Package:** Define Simulation Scenario

This describes how the contents of a simulation scenario are communicated between the CMS and the subsystem.

The CMS provides the subsystem with a simulated environment which consists of simulated objects of different kinds.

A subsystem with built-in simulation capability may participate in this simulation not only by being a consumer of the simulated environment but by contributing actively to it.
Radar type subsystems shall typically build simulated plots or tracks from the simulated environment, while contributing simulated electromagnetic emissions to it. These simulated emissions may in turn be used and detected by other (ESM type) simulations.

Weapon type subsystems when in simulation mode shall typically contribute simulated objects to the simulation that represent the launching and movement of own missiles, bullets or torpedoes and their effect on other simulated objects.

Thus CMS and subsystem both contribute to the simulated environment. Together they form a simulation federation.

The actor is the Combat Management System.

Relationship to ‘control simulation’
The definition of simulation mode and flow of commands to start/stop/freeze/resume a simulation scenario are defined in ‘control simulation’.

Relationship to provision of tracks
A radar type subsystem shall provide tracks based on information from the simulated environment, as described above. The interfaces that deal with the provision of tracks indicate whether tracks are simulated or not under amplifying information. This indication should be set for all tracks that are reported in the context of this interface.

Relationship to Receive geographic information
Geographic information is received by using ‘Receive geographic information’.

Pre-condition: Subsystem health state: The subsystem and the relevant subsystem services need to be in the health state AVAILABLE or DEGRADED.

Pre-condition: CMS has mastership.

Pre-condition: Subsystem simulation mode: The subsystem must be in subsystem simulation mode ON to participate in the simulation federation.

Pre-condition: Simulation scenario started: The actor must have started or resumed a simulation scenario.

Pre-condition: Geographic information: The subsystem may need geographic information about its simulated surroundings available locally or by means of other interfaces in order to calculate the detectability or reachability of simulated objects due to obstacles in the surroundings.

Table 7.312 - Methods of Interface Define_Simulation_Scenario_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>write_emitter_system_data_CMS()</td>
<td>Write emitter system data</td>
<td>anonymous_blob_type_emitter_system_data</td>
</tr>
<tr>
<td>write_radar_beam_data()</td>
<td>Write radar beam data</td>
<td>anonymous_blob_type_radar_beam_data</td>
</tr>
</tbody>
</table>

Define_Simulation_Scenario_Sub

Type: Interface
Package: Define_Simulation_Scenario

Table 7.313 - Methods of Interface Define_Simulation_Scenario_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>write_emitter_system_data_Sub()</td>
<td>Write emitter system data</td>
<td>anonymous_blob_type_emitter_system_data</td>
</tr>
<tr>
<td>write_environment_data()</td>
<td>Write environment data</td>
<td>anonymous_blob_type_environmental_entity_data</td>
</tr>
</tbody>
</table>
```plaintext
write_jammer_beam_data(anonymous_blob_type)
write_platform_data(anonymous_blob_type)
write_environment_data(anonymous_blob_type)
write_emitter_system_data(anonymous_blob_type)
```

**Figure 7.134 Basic Flow - Define Simulation Scenario Data (Interaction diagram)**
7.8.5.2 Control_Simulation

**Parent Package:** Simulation_Support

**Control_Simulation_CMS**

**Type:** Interface

**Package:** Control_Simulation

This service controls the simulation mode of a subsystem. This simulation mode is independent of the operational mode of the subsystem. Simulation mode is either ON or OFF. "ON" has different meanings for different kinds of subsystems. Effector type subsystems shall not engage real targets but shall simulate the engagement instead. Sensor type subsystems may be fed with simulated targets which shall be reported as plots or tracks. In each case while in simulation mode "ON" the subsystem shall strictly avoid any impact on the environment that could be the result if simulation mode was "OFF".

The actor is the Combat Management System.

**Basic Flow – Control simulation mode**

**Start event – command of simulation-mode**

The service is triggered by the actor. The actor commands the simulation mode which may be one of the following:

- **ON**: This indicates that the subsystem shall operate in simulation mode
- **OFF**: This indicates that the subsystem shall stop operating in simulation mode and that any current simulation shall be terminated

On occurrence of the trigger provision of subsystem-simulation-mode is executed.

**Provision of subsystem-simulation-mode**

After receipt of the simulation mode from the actor the subsystem responds with its subsystem simulation mode. The subsystem simulation mode may be one of the two:
• **ON:** This indicates that the subsystem is operating in simulation mode
• **OFF:** This indicates that the subsystem is not operating in simulation mode

**Basic Flow – Control Simulation (Start/Resume, Stop/Freeze)**

**START/RESUME simulation scenario**

Only when in simulation mode ON:

Upon provision of the START/RESUME command by the actor the simulation scenario starts or is resumed after a previously issued FREEZE.

**STOP/FREEZE simulation scenario**

Only when in simulation mode ON:

Upon provision of the STOP/FREEZE command by the actor the simulation scenario stops or stays frozen. The service ends.

**Provision on initialization**

The simulation mode shall be provided by the actor after initialization of the CMS.

The flow of information relevant to subsystem simulation are the subject of another service: Define simulation scenario.

If simulation is stopped or frozen simulation time of the subsystem and the actor shall be also stopped.

The synchronization of simulation time may be performed using START/RESUME command.

**Pre-condition:** CMS has mastership.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_sim_mode_status()</td>
<td>Receive the status and mode of simulation.</td>
<td>request_id_type request_id sim_mode_status_type the_status</td>
</tr>
</tbody>
</table>

**Control_Simulation_Sub**

**Type:** Interface

**Package:** Control_Simulation

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>start_resume_session()</td>
<td>This request shall be initiated on demand of the CMS. If the subsystem is in simulation mode it shall start/resume its simulation session and acknowledges the request.</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td>start_stop_sim_mode()</td>
<td>This request shall be initiated on demand of the CMS to activate/deactivate the simulation mode of the subsystem. The subsystem needs to acknowledge the request.</td>
<td>request_id_type request_id start_stop_sim_mode request_type the_request</td>
</tr>
<tr>
<td>stop_freeze_session()</td>
<td>This request shall be initiated on demand of the CMS. If the subsystem is in simulation mode and the session state is running the</td>
<td>request_id_type request_id stop_freeze_session request_type the_request</td>
</tr>
</tbody>
</table>
Subsystem needs to stop/freeze its session and acknowledges the request.

Figure 7.136 Basic Flow - Control Simulation Start/Resume (Interaction diagram)

Figure 7.137 Basic Flow - Control Simulation Stop/Freeze (Interaction diagram)
7.8.5.3 Define_Fault_Scripts

Parent Package: Simulation_Support

Define_Fault_Scripts_CMS

Type: Interface
Package: Define_Fault_Scripts

This enables a maintainer trainer to script a set of subsystem faults, the effects of which would be simulated for training purposes. The faults may be scripted in relation to a specific simulation scenario. Each fault script shall include a unique identifier.

Pre-condition: Subsystem Services: Provide subsystem services has been completed successfully, in particular this service is available.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>fault_script_summary()</td>
<td>This provides a list of all fault scripts for a subsystem to the CMS for confirmation.</td>
<td>request_id_type, request_id, fault_scripts_type, faults</td>
</tr>
</tbody>
</table>

Define_Fault_Scripts_Sub

Type: Interface
Package: Define_Fault_Scripts

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>start_stop_sim_mode(request_id_type, start_stop_sim_mode_request_type)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>receive_acknowledgement(request_id_type, request_ack)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>receive_error(request_id_type, error_reason_type)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sim_mode_status(request_id_type, sim_mode_status_type)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
add_fault_scripts(request_id_type request_id, fault_scripts_type scripts)
Adds the given fault scripts to the subsystem's simulation.

remove_fault_scripts(request_id_type request_id, fault_script_ids_type fault_scripts)
Removes the given fault scripts from the subsystem's simulation.

Figure 7.139 Alternative Flow - Define Fault Scripts (Interaction diagram)
7.8.5.4 Control_Fault_Scripts

**Parent Package:** Simulation_Support

**Type:** Interface

**Package:** Control_Fault_Scripts

This enables a trainee, at a CMS Console to cause the generation of predefined fault messages for training purposes (see also Define Fault Scripts). The subsystem shall output Fault Reports to the CMS which a trainee may respond to via the CMS Console. Fault clearance messages shall also be sent to the CMS in response to the trainee taking the appropriate action.

**Pre-condition:**
- Technical State: Subsystem is in technical state READY or ONLINE
- Fault Script: Subsystem has a fault script which has been defined previously
- Mastership Required: The CMS has Mastership
- Subsystem Services: Provide Subsystem Services has successfully completed; in particular this service is available
- Simulation Mode: Simulation Mode is ON

**Post-condition:**
- Success: Subsystem has provided simulated fault and response to clearance action
- Failure: Subsystem has not provided simulated fault and response to clearance action

**Control_Fault_Scripts_Sub**

**Type:** Interface

**Package:** Control_Fault_Scripts

---

**Figure 7.140 Basic Flow - Define Fault Scripts (Interaction diagram)**
### Table 7.318 - Methods of Interface Control_Fault_Scripts_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>enable_fault_script()</td>
<td>Causes the subsystem to indicate the faults specified by the given fault scripts when appropriately stimulated. The faults remain in place until they are cleared either by a call to clear_fault or by an action on another interface that would clear the equivalent non-simulated fault.</td>
<td>request_id_type request_id fault_script_ids_type scripts The script ids to be enabled</td>
</tr>
<tr>
<td>clear_faults()</td>
<td>Clears the faults defined by the given fault scripts.</td>
<td>request_id_type request_id fault_script_ids_type fault_scripts The script ids to be cleared</td>
</tr>
</tbody>
</table>

### Figure 7.141 Alternative Flow - Control Fault Scripts (Interaction diagram)
7.9 Sensor_Services

**Parent Package:** Service_Interfaces
Contains services associated with the Sensor Domain.

7.9.1 Clutter_Reporting

**Parent Package:** Sensor_Services

```plaintext
- enable_fault_script(request_id_type,
  fault_script_ids_type)
- receive_acknowledgement(request_id_type,
  request_ack_type)
- clear_faults(request_id_type,
  fault_script_ids_type)
- receive_acknowledgement(request_id_type,
  request_ack_type)
```

Figure 7.142 Basic Flow - Control Fault Scripts (Interaction diagram)
This package contains interfaces for the Clutter Reporting service.

### 7.9.1.1 Provide_Area_with_Plot_Concentration

**Parent Package:** Clutter_Reporting

Contains operations and sequence diagrams for the Provide Area with Plot Concentration interface.

**Type:** Interface common_use_case_interface

**Package:** Provide_Area_with_Plot_Concentration

The Radar provides the combat management system with the number of plots in a specific sector. The sector information consists of range, azimuth, and elevation. The number of plots observed in the region may provide an indication of high clutter.

Additional Information:

The information may be developed when requested or based on scan histories. The choice of methods depends upon radar design. The timestamp should indicate the oldest data used to create the report to allow the CMS or an operator to determine the validity of the report (i.e. day old data mixed with recent is still only as good as day old data).

Sector Information must consist of a measurement time stamp, range extents, azimuth extents, and elevation extents in platform coordinates.

For radars which report plot concentration without a CMS request, the CMS shall begin to receive reports upon registration of the Provide Plot Concentration interface.

**Pre-condition:** Radar in ONLINE State

**Post-condition:** None

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_periodic_plot_concentration()</td>
<td>Interface used by CMS to receive periodic plot concentration reports from the subsystem.</td>
<td>plot_concentration_report_type plot_concentration_report</td>
</tr>
<tr>
<td>receive_plot_concentration()</td>
<td>Interface used by the CMS to receive a requested plot concentration report from the subsystem.</td>
<td>request_id_type request_id plot_concentration_report_type plot_concentration</td>
</tr>
</tbody>
</table>

### Package: Provide_Area_with_Plot_Concentration

**Type:** Interface

**Package:** Provide_Area_with_Plot_Concentration

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7.108319 - Methods of Interface Provide_Plot_Concentration_CMS

Table 7.109320 - Methods of Interface Provide_Plot_Concentration_Sub
| provide_plot_concentration() | Interface used by the subsystem to receive a plot concentration request from the CMS. | request_id_type | request_id | plot_concentration_request_data_type | plot_request | type | plot_request |
provide_plot_concentration(request_id_type, plot_concentration_request_data_type)

receive_acknowledgement(request_id, request_ack)

receive_plot_concentration(request_id_type, plot_concentration_request_type)

unable to comply with request

receive_acknowledgement(request_id_type, request_ack_type)

error encountered following an accepted request

receive_acknowledgement(request_id_type, request_ack_type, error_reason_type)
Figure 7.113.143 Provide Plot Concentration - Report Requested by CMS (Interaction diagram)

Flow of events which depicts a subsystem that reports plot concentration following an explicit request from the CMS (also depicts alternate rejection and error paths).
Flow of events which depicts a subsystem that periodically reports plot concentration reports (without the need for a CMS request).

**Provide_Clutter_Assessment**

**7.9.1.2 Parent Package:** Clutter_Reporting

Contains operations and sequence diagrams for the Provide Clutter Assessment interface.

**Provide_Clutter_Assessment_CMS**

**Type:** Interface common_use_case_interface

**Package:** Provide_Clutter_Assessment
The radar reports visible clutter to the combat management system. The report shall include a map (collection of cells) with information on range, azimuth, elevation and intensity in platform relative coordinates. Clutter may be classified by type, Land, Sea, Weather (optional), etc.. Intensity may be indicated by linear signal-to-noise ratio (SNR), log-linear SNR, linear power received, log-linear power received (e.g. dBm, dBW), linear Radar Cross Section (square meters), or log-linear RCS (dbsm).

For radars which report clutter assessment without a CMS request, the CMS shall begin to receive reports upon registration of the Provide Clutter Assessment interface.

Pre-condition: Radar is in ONLINE State;
Pre-condition: The Radar is capable of distinguishing clutter from targets;
Post-condition: None;

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_clutter_assessment()</td>
<td>Interface used by the CMS to receive a requested clutter assessment report from the subsystem.</td>
<td>request_id_type request_id clutter_report_type_clutter_report</td>
</tr>
<tr>
<td>receive_periodic_clutter_assessment()</td>
<td>Interface used by CMS to receive periodic clutter assessment reports from the subsystem.</td>
<td>clutter_report_type_clutter_report</td>
</tr>
</tbody>
</table>

Provide_Clutter_Assessment_Sub
Type: Interface
Table 7.201322 - Methods of Interface Provide_Clutter_Assessment_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>provide_clutter_assessment()</td>
<td>Interface used by the subsystem to receive a clutter assessment request from the CMS.</td>
<td>request_id_type request_id clutter_assessment_request_type clutter_request</td>
</tr>
</tbody>
</table>

**Diagram:**
- `provide_clutter_assessment(request_id_type, clutter_assessment_request_type)`
- `receive_acknowledgement(request_id, request_ack)`
- `receive_error(request_id_type, request_ack_type)`
- `receive_clutter_assessment(request_id_type, clutter_report_type)`
- `receive_acknowledgement(request_id_type, request_ack_type)`
- `receive_error(request_id_type, request_ack_type)`
- `provide_clutter_assessment(request_id_type, clutter_assessment_request_type)`

Package: Provide_Clutter_Assessment
Figure 7.145.145 Provide Clutter Assessment (Interaction diagram)
Flow of events which depicts a subsystem that reports a clutter assessment following an explicit request from the CMS (also depicts alternate rejection and error paths).
Flow of events which depicts a subsystem that periodically reports a clutter assessment (without the need for a CMS request).

**Media_Streaming**

This package contains interfaces for the Media Streaming service.
### 7.9.2.1 Allocate_Tracks_To_Stream

**Parent Package:** Media_Streaming

**Allocate_Tracks_To_Stream_Sub**
- **Type:** Interface
- **Package:** Allocate_Tracks_To_Stream

#### Table 7.202 – Methods of Interface Allocate_Tracks_To_Stream_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add_Track_To_Stream()</td>
<td>A request to add the sensor’s track to the specified stream.</td>
<td>media_stream_id_type Stream_Id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sensor_track_id_type Track_Id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>request_id_type Request_Id</td>
</tr>
<tr>
<td>Remove_Track_From_Stream()</td>
<td>A request to remove the sensor’s track from the specified stream.</td>
<td>media_stream_id_type Stream_Id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sensor_track_id_type Track_Id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>request_id_type Request_Id</td>
</tr>
<tr>
<td>Add_All_Tracks_To_Stream()</td>
<td>A request to add all the sensor’s tracks to the specified stream.</td>
<td>media_stream_id_type Stream_Id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>request_id_type Request_Id</td>
</tr>
<tr>
<td>Remove_All_Tracks_From_Stream()</td>
<td>A request to remove all the sensor’s tracks from the specified stream.</td>
<td>media_stream_id_type Stream_Id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>request_id_type Request_Id</td>
</tr>
</tbody>
</table>
Allocate_Tracks_to_Stream_CMS

Type: Interface common_use_case_interface
Package: Allocate_Tracks_To_Stream

This service allows the CMS to receive the allocation of sensor tracks to media streams.

Table 7.203323 - Methods of Interface Allocate_Tracks_to_Stream_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_media_stream_allocation()</td>
<td>For a sensor to report on the allocation of tracks to one of its media streams</td>
<td>media_allocation_type allocation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The allocation of sensor tracks to a media stream</td>
</tr>
</tbody>
</table>

Allocate_Tracks_To_Stream_Sub

Type: Interface
Package: Allocate_Tracks_To_Stream

Table 7.324 - Methods of Interface Allocate_Tracks_To_Stream_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add_Track_To_Stream()</td>
<td>A request to add the sensor's track to the specified stream.</td>
<td>media_stream_id type Stream_Id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sensor_track_id type Track_Id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>request_id type Request_Id</td>
</tr>
<tr>
<td>Remove_Track_From_Stream()</td>
<td>A request to remove the sensor's track from the specified stream.</td>
<td>media_stream_id type Stream_Id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sensor_track_id type Track_Id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>request_id type Request_Id</td>
</tr>
<tr>
<td>Add_All_Tracks_To_Stream()</td>
<td>A request to add all the sensor's tracks to the specified stream.</td>
<td>media_stream_id type Stream_Id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>request_id type Request_Id</td>
</tr>
</tbody>
</table>
Remove All Tracks From Stream
A request to remove all the sensor's
tracks from the specified stream.

media_stream_id_type Stream_Id
request_id_type Request_Id

Figure 7.447.147 Allocate_Tracks_To_Stream (Class diagram)
Open Architecture Radar Interface Standard (OARIS), v2.0

Add_Track_To_Stream(media_stream_id_type, sensor_track_id_type, request_id_type)

Remove_Track_From_Stream(media_stream_id_type, sensor_track_id_type, request_id_type)

report_media_stream_allocation(media_allocation_type)

acknowledgement

receive_acknowledgement(request_id_type, request_ack_type)

negative acknowledgement

Remove_Track_From_Stream(media_stream_id_type, sensor_track_id_type, request_id_type)

Negative acknowledgement

report_media_stream_allocation(media_allocation_type)

Allocate_Tracks_To_Stream_Sub «idlInterface»

Allocate_Tracks_to_Stream_CMS «idlInterface»

debug «idlInterface»

unable to remove track from media stream

[track inval]

em

unable to add track to media stream

[track inval for media stream or unknown to sensor]
Allocate_Tracks_to_Stream_CMS

Allocate_Tracks_To_Stream_Sub

Subsystem unable to add track to media stream
[track invalid for media stream or unknown to sensor]
receive_acknowledgement(request_id_type, request_ack_type)
negative acknowledgement

remove_media_stream_allocation(media_allocation_type)

Allocate_Tracks_To_Stream(media_stream_id_type, sensor_track_id_type, request_id_type)

Subsystem unable to remove track from media stream
[track mandatory for media stream]
receive_acknowledgement(request_id_type, request_ack_type)
negative acknowledgement

Add_Track_To_Stream(media_stream_id_type, sensor_track_id_type, request_id_type)

Remove_Track_From_Stream(media_stream_id_type, sensor_track_id_type, request_id_type)

report_media_stream_allocation(media_allocation_type)
receive_acknowledgement(request_id_type, request_ack_type)

Figure 7.148 Allocate_Tracks_To_Stream - Alternate Flow (Interaction diagram)
Initial allocation of tracks
Add an additional track
Remove one of the existing tracks from the media stream
Remove all the tracks
All the tracks back

Add_Track_To_Stream(media_stream_id_type, sensor_track_id_type, request_id_type)
report_media_stream_allocation(media_allocation_type)

Remove_Track_From_Stream(media_stream_id_type, sensor_track_id_type, request_id_type)
report_media_stream_allocation(media_allocation_type)

Remove_All_Tracks_From_Stream(media_stream_id_type, request_id_type)
report_media_stream_allocation(media_allocation_type)

Add_Track_To_Stream_CMS(media_stream_id_type, sensor_track_id_type, request_id_type)
report_media_stream_allocation(media_allocation_type)

Allocate_Tracks_To_Stream_Sub«idlInterface»
Allocate_Tracks_to_Stream_CMS
7.9.2.2 Configure_Media_Streams

**Parent Package:** Media_Streaming

Configure_Media_Streams_CMS

**Type:** Interface

**Package:** Configure_Media_Streams

This service allows the CMS to be informed about the configuration of media streams provided by the sensor.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allocate_Tracks_to_Stream_CMS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Allocate_Tracks_To_Stream_Sub</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add an additional track</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Remove one of the existing tracks from the media stream</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Remove all the tracks</td>
<td></td>
<td></td>
</tr>
<tr>
<td>All the tracks back</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 7.119.149 Allocate_Tracks_To_Stream - Basic Flow (Interaction diagram)**
| configure_media_stream() | media_stream_metadata_type
| metadata The metadata for the stream |

Configure_Media_Streams_Sub
Type: Interface
Package: Configure_Media_Streams

```
+   configure_media_stream(media_stream_metadata_type): void
```

Figure 7.120 Configure_Media_Streams (Class diagram)

Figure 7.121 Configure_Media_Streams (Interaction diagram)
7.8.3.7.9.3 Search

Parent Package: Sensor_Services
This package contains interfaces for the Search service.

7.9.3.1 Perform_Cued_Search

7.8.3.1 Parent Package: Search

Perform_Cued_Search_CMS

Type: Interface common_use_case_interface

Package: Perform_Cued_Search

The CMS Search Interface.
The subsystem is requested to undertake a cued search in the requested cue volume or to the requested track. The cue may be 1D (azimuth only), 2D (has an additional elevation constraint), 3D (has a further range constraint) or 4D (has a further target velocity constraint). The response of the subsystem is either to reject the cued search request if it is invalid within the current mode/configuration or to provide a cue request reply containing data relating to any resulting tracks.

Depending upon the individual radar it may be possible to predefine a cued search waveform...
The cued search request may contain azimuth, elevation and range data along with time of the positional data.

Pre-condition: Technical State: The Subsystem is in Technical State ONLINE. Pre-condition: Mastership: The CMS has Mastership
Pre-condition: Subsystem Services: The Provide Subsystem Services Service has been executed successfully.

Post-condition: Success: The CMS has received a 'Cued Search Report'
Post-condition: Failure: The CMS has not received a 'Cued Search Report'

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_cued_search_result()</td>
<td>Send a report to the CMS containing the results of a previously cued search.</td>
<td>cued_search_report_type result_report The result of the search. request_id_type request_id The unique id relating to this cued search request as supplied by the CMS.</td>
</tr>
</tbody>
</table>

Perform_Cued_Search_Sub
Type: Interface
Package: Perform_Cued_Search
The Subsystem Search Interface.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>perform_cued_search()</td>
<td>Request to subsystem to perform a cued search in accordance with the given set of constraints.</td>
<td>cued_search_cue_type constraint The details of the constraints on where the radar is to look for tracks. request_id_type request_id The unique id for this request. The radar includes this in all replies relating to this request.</td>
</tr>
<tr>
<td>perform_cue_to_track()</td>
<td>Request to subsystem to perform a cue to the position of a track produced by a different subsystem.</td>
<td>sensor_track_id_type sensor_track_id The identifier of the track to cue to. string subsystem_name The name of the subsystem that produced the track to cue to. request_id_type request_id The unique id for this request. The radar includes this in all replies relating to this request.</td>
</tr>
<tr>
<td>perform_surveillance()</td>
<td></td>
<td>surveillance_task_type surveillance_task The surveillance task to be performed request_id_type request_id The unique id for this request. The sensor</td>
</tr>
</tbody>
</table>
stop_surveillance()

request_id_type request_id The unique id for this request. The sensor includes this in all replies relating to this request.

receive_acknowledgement(request_id_type, request_ack_type)

perform_surveillance(surveillance_task_type, request_id_type)

receive_acknowledgement(request_id_type, request_ack_type)

stop_surveillance(request_id_type)
Figure 7.122.152 Basic Flow - Perform Surveillance (Interaction diagram)
perform_cued_search(cued_search_cue_type, request_id_type)

Negative Acknowledgement
[Subsystem has encountered an error while performing a cued search]
receive_acknowledgement(request_id, request_ack)

Failure to form a track from a cued search is not an error condition. This results in a report without a track identifier being returned.
Failure to form a track from a cued search is not an error condition. This results in a report without a track identifier being returned.

Figure 7.123.153 Alternative Flow - Sensor does not Perform Cued Search (Interaction diagram)
The cued search report may not contain a track identifier resulting from the search.

The cued search report may not contain a track identifier resulting from the search.

Failure to form a track from a cue to track is not an error condition. This results in a report without a track identifier being returned.

Figure 7.124.154 Basic Flow - Perform Cued Search (Interaction diagram)
Failure to form a track from a cue to track is not an error condition. This results in a report without a track identifier being returned.

Figure 7.155 Alternative Flow - Sensor does not Perform Cued To Track (Interaction diagram)
7.8.4.1.4 _Sensor_Assessment_

Parent Package: Sensor_Services

This package contains interfaces for the Sensor Assessment service.

---

7.8.4.4.1 _Assess_Sensor_Plot_

Parent Package: Sensor_Assessment

The cued search report may not contain a track identifier resulting from the search.

```
loop
[More than one track found]

perform_cue_to_track(sensor_track_id_type, string, request_id_type)

report_cued_search_result(cued_search_report_type, request_id_type)

receive_acknowledgement(request_id_type, request_ack_type)
```

---

Figure 7.126,156 Basic Flow - Perform Cued To Track (Interaction diagram)

Figure 7.127,157 Sensor_Assessment (Class diagram)
Provide_Sensor_Plot_Assessment_CMS
Type: Interface
Package: Assess_Sensor_Plot
The interface for a sensor to provide assessments (identification and classification) of sensor plots to the CMS. It is expected that the assessment relates to matching the plot's measurement parameters to reference data. The sensor provides a set of mode, equipment and/or platform matches relating to a particular plot (referenced by the plot id). Therefore the sensor plot must have a plot_id attribute defined.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>write_equipment_assessment()</td>
<td>To report on the overall equipment assessment for a sensor plot.</td>
<td>sensor_plot_equipment_assessment_type_equipment_assessment The assessment of the equipment to which the sensor track's data may correspond.</td>
</tr>
<tr>
<td>write_platform_assessment()</td>
<td>To report on the overall platform assessment for a sensor plot.</td>
<td>sensor_plot_platform_assessment_type_platform_assessment The assessment of the platform to which the sensor track's data may correspond.</td>
</tr>
<tr>
<td>write_mode_assessment()</td>
<td>To report on the overall mode assessment for a sensor plot.</td>
<td>sensor_plot_mode_assessment_type_mode_of_the_detected_equipment assessment for a sensor track</td>
</tr>
</tbody>
</table>

Provide_Sensor_Plot_Assessment_Sub
Type: Interface
Package: Assess_Sensor_Plot
The interface by which a CMS can control the sensor's assessment of the plot data. The sensor matches parametric measurements to reference data and then reports each of these sets as an assessment for each plot for the categories of equipment modes, equipment marks (build standards / versions) and
platform instances (or platform classes). The sensor also reports what it has assessed to be the best match.

Figure 7.128 Assess_Sensor_Plot (Class diagram)
7.8.4.2 Assess_Sensor_Track

Parent Package: Sensor_Assessment

Provide_Sensor_Track_Assessment_CMS

Type: Interface common_use_case_interface

Package: Assess_Sensor_Track

The interface for a sensor to provide assessments (identification and classification) of sensor tracks to the CMS. The sensor matches parametric measurements to reference data and then reports each of these sets as an assessment for each track for the categories of equipment modes, equipment marks (build standards / versions) and platform instances (or platform classes). The sensor also reports what it has
assessed to be the best match and the match currently selected. The currently selected match influences the attributes reported for the sensor track (including its recognition and identification). The CMS uses the select and deselect methods to set or override the match that is selected for an assessment of a sensor track. Sensors report a track to the CMS using the Track Reporting use case before providing an assessment. Assessments are only reported for tracks whilst the sensor track is in the TRACKED track state.

Table 7.208329 - Methods of Interface Provide_Sensor_Track_Assessment_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>write_equipment_assessment()</td>
<td>To report on the overall equipment assessment for a sensor track.</td>
<td>sensor_track_equipment_assessment_type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>equipment_assessment The assessment of the equipment to which the sensor track's data may correspond.</td>
</tr>
<tr>
<td>write_platform_assessment()</td>
<td>To report on the overall platform assessment for a sensor track.</td>
<td>sensor_track_platform_assessment_type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>platform_assessment The assessment of the platform to which the sensor track's data may correspond.</td>
</tr>
<tr>
<td>write_multipath_set()</td>
<td>To report on the assessment of a set of sensor track representing the same real world object through multiple paths.</td>
<td>multipath_set_type_set</td>
</tr>
<tr>
<td>write_mode_assessment()</td>
<td>To report on the overall mode assessment for a sensor track.</td>
<td>sensor_track_mode_assessment_type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mode_assessment The overall assessment of mode (of the detected equipment) for a sensor track</td>
</tr>
</tbody>
</table>

Provide_Sensor_Track_Assessment_Sub

Type: Interface

Package: Assess_Sensor_Track

The interface by which a CMS can control the sensor’s assessment of the track data.

Table 7.209330 - Methods of Interface Provide_Sensor_Track_Assessment_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>select_equipment_assessment()</td>
<td>The CMS selects a particular equipment match as being the authoritative assessment for the sensor track with regard to the equipment it is a detection of. The Subsystem, thereafter reports the sensor track in accordance with this assessment.</td>
<td>request_id_type request_id The unique identifier of the request to select the match long_match_id The identifier of the match to be selected sensor_track_id_type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sensor_track_id The sensor track to which the assessment applies</td>
</tr>
<tr>
<td>select_platform_assessment()</td>
<td>The CMS selects a particular platform match as being the authoritative assessment for the</td>
<td>request_id_type request_id The unique identifier of the request to select the match</td>
</tr>
</tbody>
</table>

Open Architecture Radar Interface Standard (OARIS), v2.0
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Request IDs</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>select_mode_assessment()</td>
<td>The CMS selects a particular mode match as being the authoritative assessment for the sensor track with regard to the equipment it is a detection of. The Subsystem, thereafter reports the sensor track in accordance with this assessment.</td>
<td>request_id</td>
<td>request_id_type request_id The unique identifier of the request to select the match long_match_id The identifier of the match to be selected sensor_track_id_type sensor_track_id The sensor track to which the assessment applies sensor_track_id_type sensor_track_id The sensor track to which the assessment applies</td>
</tr>
<tr>
<td>deselect_equipment_assessment()</td>
<td>The CMS deselects equipment match as being the authoritative assessment for the sensor track with regard to the equipment it is a detection of. The Subsystem, stops reporting the sensor track in accordance with the previously selected assessment.</td>
<td>request_id</td>
<td>request_id_type request_id The unique identifier of the request to deselect matches long_match_id The identifier of the match to be deselected sensor_track_id_type sensor_track_id The sensor track to which the assessment applies</td>
</tr>
<tr>
<td>deselect_platform_assessment()</td>
<td>The CMS deselects platform match as being the authoritative assessment for the sensor track with regard to the platform it is a detection of. The Subsystem, stops reporting the sensor track in accordance with the previously selected assessment.</td>
<td>request_id</td>
<td>request_id_type request_id The unique identifier of the request to deselect matches long_match_id The identifier of the match to be deselected sensor_track_id_type sensor_track_id The sensor track to which the assessment applies</td>
</tr>
<tr>
<td>deselect_mode_assessment()</td>
<td>The CMS deselects mode match as being the authoritative assessment for the sensor track with regard to the equipment it is a detection of. The Subsystem, stops reporting the sensor track in accordance with the previously selected assessment.</td>
<td>request_id</td>
<td>request_id_type request_id The unique identifier of the request to deselect matches long_match_id The identifier of the match to be deselected sensor_track_id_type sensor_track_id The sensor track to which the assessment applies</td>
</tr>
<tr>
<td>select_mode_assessment()</td>
<td>The CMS selects a particular mode match as being the authoritative assessment for the sensor track with regard to the equipment it is a detection of. The Subsystem, thereafter reports the sensor track in accordance with this assessment.</td>
<td>request_id</td>
<td>request_id_type request_id The unique identifier of the request to select the match long_match_id The identifier of the match to be selected sensor_track_id_type sensor_track_id The sensor track to which the assessment applies sensor_track_id_type sensor_track_id The sensor track to which the assessment applies</td>
</tr>
</tbody>
</table>
deselect_equipment_assessment()
The CMS deselects equipment match as being the authoritative assessment for the sensor track with regard to the equipment it is a detection of. The Subsystem stops reporting the sensor track in accordance with the previously selected assessment.

request_id_type request_id The unique identifier of the request to deselect matches
sensor_track_id_type sensor_track_id The sensor track to which the assessment applies

deselect_platform_assessment()
The CMS deselects platform match as being the authoritative assessment for the sensor track with regard to the platform it is a detection of. The Subsystem stops reporting the sensor track in accordance with the previously selected assessment.

request_id_type request_id The unique identifier of the request to deselect matches
sensor_track_id_type sensor_track_id The sensor track to which the assessment applies

deselect_mode_assessment()
The CMS deselects mode match as being the authoritative assessment for the sensor track with regard to the equipment it is a detection of. The Subsystem stops reporting the sensor track in accordance with the previously selected assessment.

request_id_type request_id The unique identifier of the request to deselect matches
sensor_track_id_type sensor_track_id The sensor track to which the assessment applies
<table>
<thead>
<tr>
<th>Provide_Sensor_Track_Assessment_OWS</th>
<th>Provide_Sensor_Track_Assessment_Sub</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ write_equipment_assessment(sensor_track_equipment_assessment_type): void</td>
<td></td>
</tr>
<tr>
<td>+ write_platform_assessment(sensor_track_platform_assessment_type): void</td>
<td></td>
</tr>
<tr>
<td>+ write_multipath_set(multipath_set_type): void</td>
<td></td>
</tr>
<tr>
<td>+ write_mode_assessment(sensor_track_mode_assessment_type): void</td>
<td></td>
</tr>
<tr>
<td>+ select_equipment_assessment(request_id_type, long, sensor_track_id_type): void</td>
<td></td>
</tr>
<tr>
<td>+ select_platform_assessment(request_id_type, long, sensor_track_id_type): void</td>
<td></td>
</tr>
<tr>
<td>+ select_mode_assessment(request_id_type, long, sensor_track_id_type): void</td>
<td></td>
</tr>
<tr>
<td>+ deselect_equipment_assessment(request_id_type, sensor_track_id_type): void</td>
<td></td>
</tr>
<tr>
<td>+ deselect_platform_assessment(request_id_type, sensor_track_id_type): void</td>
<td></td>
</tr>
<tr>
<td>+ deselect_mode_assessment(request_id_type, sensor_track_id_type): void</td>
<td></td>
</tr>
</tbody>
</table>

Figure 7.130.160 Assess_Sensor_Track (Class diagram)
Unable to deselect [No assessment for sensor track]

deselect_equipment_assessment(request_id_type)
receive_acknowledgement(request_id_type, request_ack_type)
Negative acknowledgement

Unable to select [Mode match doesn't exist]

write_mode_assessment(sensor_track_mode_assessment_type)
select_mode_assessment(request_id_type, long)
receive_acknowledgement(request_id_type, request_ack_type)
Negative acknowledgement

[Equipment match doesn't exist]

write_equipment_assessment(sensor_track_equipment_assessment_type)
select_equipment_assessment(request_id_type, long)
receive_acknowledgement(request_id_type, request_ack_type)
Negative acknowledgement

[Platform match doesn't exist]

write_platform_assessment(sensor_track_platform_assessment_type)
select_platform_assessment(request_id_type, long)
receive_acknowledgement(request_id_type, request_ack_type)
Negative acknowledgement
Figure 7.131.161 Assess_Sensor_Track - alternate flows (Interaction diagram)
open architecture radar interface standard (oaris), v2.0

provide sensor track assessment

write mode assessment (sensor_track_mode_assessment_type)
select mode assessment (request_id_type, long)
receive acknowledgement (request_id_type, request_ack_type)
write mode assessment (sensor_track_mode_assessment_type)

write equipment assessment (sensor_track_equipment_assessment_type)
select equipment assessment (request_id_type, long)
receive acknowledgement (request_id_type, request_ack_type)
write equipment assessment (sensor_track_equipment_assessment_type)

write platform assessment (sensor_track_platform_assessment_type)
select platform assessment (request_id_type, long)
receive acknowledgement (request_id_type, request_ack_type)
write platform assessment (sensor_track_platform_assessment_type)
Provide_Sensor_Track_Assessment(CMS)

- select_mode_assessment(request_id_type, request_ack_type)
- write_mode_assessment(sensor_track_mode_assessment_type)
- select_equipment_assessment(request_id_type, request_ack_type)
- write_equipment_assessment(sensor_track_equipment_assessment_type)
- select_platform_assessment(request_id_type, request_ack_type)
- write_platform_assessment(sensor_track_platform_assessment_type)
- receive_acknowledgement(request_id_type, request_ack_type)
- receive_acknowledgement(request_id_type, request_ack_type)

Provide_Sensor_Track_Assessment_Sub

- select_mode_assessment(request_id_type, request_ack_type)
- write_mode_assessment(sensor_track_mode_assessment_type)
- select_equipment_assessment(request_id_type, request_ack_type)
- write_equipment_assessment(sensor_track_equipment_assessment_type)
- select_platform_assessment(request_id_type, request_ack_type)
- write_platform_assessment(sensor_track_platform_assessment_type)
- receive_acknowledgement(request_id_type, request_ack_type)
- receive_acknowledgement(request_id_type, request_ack_type)
CMS user selects an equipment match for the sensor track

CMS user reconsiders the sensor track and deselects the equipment match

Figure 7.132.162 Assess_Sensor_Track - assessment and selection (Interaction diagram)

Figure 7.133.163 Assess_Sensor_Track - equipment deselection (Interaction diagram)
CMS user selects a mode match assessment for a track

CMS user reconsiders the sensor track and deselects the mode match

Figure 7.134.164 Assess_Sensor_Track - mode deselection (Interaction diagram)
Figure 7.135.165 Assess_Sensor_Track - multipath (Interaction diagram)
7.8.5.7.9.5 Supplementary_Measurement

Parent Package: Sensor_Services
This package contains interfaces for the Supplementary Measurement service.

7.9.5.1 Configure_Measurement_Parameters
Parent Package: Supplementary_Measurement

Configure_Measurement_Parameters_CMS
Type: Interface
Package: Configure_Measurement_Parameters
The configuration of measurement parameters allows integrated systems to specify the set of measurement types for which the installed equipment has a measurement capability and semantics associated with these measurement types.
Measurement types have the potential to be classified and also the set of measurement types can be expected to grow as technology advances. Therefore the meaning of the associated identifiers are systems specific and determined from configuration data.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>configure_parametric_measurement</td>
<td>measurement_parameter_kind_type</td>
<td>measurement_parameter_kind_type parameter: A kind of parameter supported by the sensor</td>
</tr>
</tbody>
</table>

Configure_Measurement_Parameters_Sub
Type: Interface
Package: Configure_Measurement_Parameters

Figure 7.137.167 Configure_Measurement_Parameters (Class diagram)
7.8.5.27.9.5.2 Provide_Sensor_Plot_Parameters

Parent Package: Supplementary_Measurement

Provide_Sensor_Plot_Parameters_CMS
Type: Interface
Package: Provide_Sensor_Plot_Parameters

Interface for a sensor to provide its supplementary parametric data with respect to plots to the CMS. A sensor can pass a set of measurements with plot data when reporting to the CMS.

### Table 7.211332 - Methods of Interface Provide_Sensor_Plot_Parameters_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>write_parameter_measurement_set()</td>
<td>For the sensor to report on the measurement of a set of parameters with a plot defined by configuration data.</td>
<td>plot_measurement_parameter_set_by_plot_measurement_parameter_set_type The sensor's measurement of a set of parameters</td>
</tr>
</tbody>
</table>

**Provide_Sensor_Plot_Parameters_Sub**

Type: Interface
Package: Provide_Sensor_Plot_Parameters
7.8.5.3 Provide_Sensor_Track_Parameters

Parent Package: Supplementary_Measurement

Provide_Sensor_Track_Parameters_CMS

Type: Interface

Package: Provide_Sensor_Track_Parameters

Interface for a sensor to provide its supplementary parametric data with respect to tracks to the CMS. A sensor can pass measurements to the CMS individually or as a set (relating to the same track). Sensors report a track to the CMS using the Track Reporting use case before reporting any of its supplementary measurements. Supplementary measurements are only reported for tracks whilst the sensor track is in the TRACKED track state.

Table 7.212333 - Methods of Interface Provide_Sensor_Track_Parameters_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>write_parameter_measurement()</td>
<td>For the sensor to report on the measurement of an individual parameter defined by configuration data.</td>
<td>track_measurement_parameter_type Parameter The sensor's measurement of a parameter</td>
</tr>
<tr>
<td>write_parameter_measurement_set()</td>
<td>For the sensor to report on the measurement of a set of parameters for a track defined by configuration data.</td>
<td>track_measurement_parameter_set_type Parameter Set The sensor's measurement of a set of parameters</td>
</tr>
<tr>
<td>parameter defined by configuration data.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>measurement of a parameter</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
For the sensor to report on the measurement of a set of parameters for a track defined by configuration data.

provide_Sensor_Track_Parameters_Sub

**Type:** Interface

**Package:** Provide_Sensor_Track_Parameters

Figure 7.140.170 Provide_Sensor_Track_Parameters (Class diagram)
Figure 7.141.171 Provide_Sensor_Track_Parameters - parameter sets (Interaction diagram)
7.8.67.9.6 Plot_Reporting

Parent Package: Sensor_Services
This package contains interfaces for the Plot Reporting service.

7.8.67.9.6.1 ProvideFilter_Plots

Parent Package: Plot_Reporting

Provide This package contains interfaces for the Filter Plots service.
Filter_Plots_CMS
Type: Interface
Package: Filter_Plots
The interface to the CMS for receiving information relating to the filters used to control which plots are made available to other network segments.

The plot sharing architecture recognizes that connectivity between different platforms hosting sensors may not support the bandwidth required to share all plot and track updates. It is possible for a sensor also to provide the networking functionality in which case it is providing an additional role in the interface.

### Table 7.334 - Methods of Interface Filter_Plots_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_plot_filter()</td>
<td>Reports the parameters of one of the filters that are active for plots in the communication and networking subsystem. Plots are transmitted or received, according to their mode, if they pass the conditions of at least one of the active filters.</td>
<td>filter_id_type, filter_id, The identifier for the filter filter_value, The criteria for the filter</td>
</tr>
<tr>
<td>plot_filter_removed()</td>
<td>Reports that a particular plot filter has been removed.</td>
<td>filter_id_type, filter_id, The identifier of the filter removed</td>
</tr>
</tbody>
</table>

### Filter_Plots_Sub

**Type:** Interface  
**Package:** Filter_Plots

The interface to the subsystem for receiving updates to the filters used to control which plots are made available to other network segments.

The plot sharing architecture recognizes that connectivity between different platforms hosting sensors may not support the bandwidth required to share all plot and track updates.

In this use case the subsystem is the network component providing connectivity to other platforms, as distinct from the local sensors providing the plots. It is possible for a sensor to also provide the networking functionality, in which case it is providing an additional role in the interface.

### Table 7.335 - Methods of Interface Filter_Plots_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>add_plot_filter()</td>
<td>Adds an active filter for plots to the communication and networking subsystem.</td>
<td>request_id_type, request_id, unique reference for the request filter plot_filter_parameters, filter, the values to be used to filter plots by</td>
</tr>
<tr>
<td>remove_plot_filter()</td>
<td>Removes a filter for plots from the communication and networking subsystem.</td>
<td>request_id_type, request_id, The unique reference for the request filter id_type, filter_id, The identifier for the filter to be removed</td>
</tr>
</tbody>
</table>
alt failure cases
[Add fail]

- add_plot_filter(request_id_type, plot_filter_parameters_type)
- receive_acknowledgement(request_id_type, request_ack_type)
- plot_filter_removed(filter_id_type)
- remove_plot_filter(request_id_type, filter_id_type)
- report_plot_filter(filter_id_type, plot_filter_parameters_type)
- receive_acknowledgement(request_id_type, request_ack_type)

Figure 7.173 Filter Plots - alternative flows (Interaction diagram)

- add_plot_filter(request_id_type, plot_filter_parameters_type)
- receive_acknowledgement(request_id_type, request_ack_type)
- report_plot_filter(filter_id_type, plot_filter_parameters_type)
- receive_acknowledgement(request_id_type, request_ack_type)
- plot_filter_removed(filter_id_type)
7.9.6.2 Provide_Sensor_Characteristics

Parent Package: Plot_Reporting

This package contains interfaces for the Provide Sensor Characteristic service. Provide_Sensor_Characteristics_CMS

Type: Interface

Package: Provide_Sensor_Characteristics

The interface to the CMS for providing information about the characteristics of a sensor. This enables sensor-agnostic processing of sensor data particularly plot data. Sensor characteristics are sent by the subsystem when it receives a request from the CMS.

Table 7.336 - Methods of Interface Provide_Sensor_Characteristics_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_sensor_calibration_model()</td>
<td>Method for a sensor subsystem to inform CMS of its calibration model.</td>
<td>request_id_type request_id The unique identifier of the request for the sensor's calibration model sensor_calibration_model_type model The sensor's calibration model</td>
</tr>
<tr>
<td>report_sensor_precision_model()</td>
<td>Method for a sensor subsystem to inform CMS of its precision model.</td>
<td>request_id_type request_id The unique identifier of the request for the sensor's precision model sensor_precision_model_type model The sensor's model of its precision</td>
</tr>
<tr>
<td>report_sensor_stabilization_model()</td>
<td>Method for a sensor subsystem to inform CMS of its stabilization model.</td>
<td>request_id_type request_id The unique identifier of the request for the sensor's stabilization model sensor_stabilization_model_type model The sensor's model of stabilization characteristics</td>
</tr>
<tr>
<td>report_sensor_processing_model()</td>
<td>Method for a sensor subsystem to inform CMS of its processing model.</td>
<td>request_id_type request_id The unique identifier of the request for the sensor's processing model sensor_processing_model_type model The sensor's model of its own processing algorithms</td>
</tr>
</tbody>
</table>

Provide_Sensor_Characteristics_Sub

Type: Interface

Package: Provide_Sensor_Characteristics

The interface to the Subsystem for requesting sensor characteristics.

Table 7.337 - Methods of Interface Provide_Sensor_Characteristics_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_characteristics()</td>
<td>This requests the subsystem to send its characteristics to the CMS.</td>
<td>request_id_type request_id The unique identifier for the request</td>
</tr>
</tbody>
</table>
7.9.6.3 Provide_Plots

Figure 7.175 Basic Flow - Provide_Sensor_Characteristics on request (Interaction diagram)

Figure 7.176 Provide_Sensor_Characteristics - Alternate Flow - models unavailable (Interaction diagram)
Parent Package: Plot_Reporting

Provide_Plots_CMS

Type: Interface

Package: Provide_Plots

Interface to the CMS for receiving plot updates. This interface provides sensor plots to the CMS (filterable to air, surface, land and space environments). The transfer of data is expected to take place asynchronously, although for certain classes of sensor it may appear periodic.

Pre-condition: Subsystem Services: Provide Subsystem Services has successfully executed

Pre-condition: Register Interest: The CMS has successfully registered interest in this service

Post-condition: Success: CMS has received plot datastream

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>write_sensor_plot()</td>
<td>This method receives a individual plot update from the sensor. It is expected to be called periodically from the sensor.</td>
<td>sensor_plot_type_plots</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The set of plots</td>
</tr>
<tr>
<td>write_sensor_plot_set()</td>
<td>This method receives a set of one or more plot updates from the sensor. It is expected to be called periodically from the sensor.</td>
<td>sensor_plot_set_type_plots</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The set of plots</td>
</tr>
</tbody>
</table>
This sequence diagram shows the style of transferring plots individually:

```
loop [periodic]
  loop [for each return]
    write_sensor_plot(sensor_plot_type)
```

Figure 7.143.177 Basic Flow - Provide Plots (Individual) (Interaction diagram)
7.9.6.4 Provide_Sensor_Orientation

7.8.6.2 Parent Package: Plot_Reporting

Provide_Sensor_Orientation_CMS

Type: Interface

This sequence diagram shows the batched style of updating plots, with whole sets being transformed atomically.

Figure 7.144.178 Basic Flow - Provide Plots (Sets) (Interaction diagram)
Package: Provide_Sensor_Orientation
The interface to the CMS for receiving sensor orientation updates. The sensor provides its orientation in the case that it has movement that is independent of that for the overall platform. It is provided periodically with a frequency defined using the manage subsystem parameters use case.

Pre-condition: Subsystem Services: Provide Subsystem Services has successfully executed
Pre-condition: Register Interest: The CMS has successfully registered interest in this service
Post-condition: Success: CMS has received sensor orientation data stream

Table 7. Methods of Interface Provide_Sensor_Orientation_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>write_sensor_orientation()</td>
<td>Informs the CMS of the orientation of the sensor</td>
<td>sensor_orientation_type orientation The orientation of the sensor</td>
</tr>
</tbody>
</table>

Sensor's with independent movement (e.g., surveillance and navigation radars that rotate) provide regular updates on its orientation. The frequency of updates is defined using the manage subsystem parameters use case.
Sensor’s with independent movement (e.g. surveillance and navigation radars that rotate) provide regular updates on its orientation. The frequency of updates is defined using the manage subsystem parameters use case.

Figure 7.145.179 Basic Flow - Provide Sensor Orientation (Interaction diagram)
::Provide_Sensor_Orientation_CMS
+   write_sensor_orientation(sensor_orientation_type): void

::Provide_Plots_CMS
+   write_sensor_plot(sensor_plot_type): void
+   write_sensor_plot_set(sensor_plot_set_type): void

\text{idlInterface}
Service-Level-Interfaces & Actors-Templates: plot_reporting one

-   \text{idlInterface}
Provide_Plots_CMS
-   \text{idlInterface}
Provide_Sensor_Orientation_CMS
   \text{write_sensor_orientation(sensor_orientation_type): void}
   \text{write_sensor_plot(sensor_plot_type): void}
   \text{write_sensor_plot_set(sensor_plot_set_type): void}
7.9.7 Sensor Control

Parent Package: Sensor_Services

This package contains interfaces for the Sensor Control service.

7.9.7.1 Manage_Frequency_Usage

Parent Package: Sensor_Control

This package contains interfaces for the Manage Frequency Usage service.

Manage_Frequency_Usage_CMS

Type: Interface common_use_case_interface

Package: Manage_Frequency_Usage

This controls the sensor behaviour with respect to the transmission frequency management. Basing on a discrete set of transmission frequencies offered by the sensor, CMS may disable/enable the use of a subset;
of them. As well CMS may select the sensor transmission mode, i.e. how the sensor shall select the transmission frequencies, among the set of transmission modes supported by the sensor.
The transmission mode defines how the sensor selects the transmission frequencies, which may be:

- **Fixed Frequency**: sensor always uses the same pre-selected frequency
- **Frequency Diversity**: at each transmission sensor selects the frequency to be used inside a pre-selected subset of frequencies
- **Automatic Frequency Selection**: at each transmission sensor selects the frequency to be used among the least jammed frequencies
- **Random Agility**: at each transmission sensor random selects the frequency to be used.

The availability of each of the above listed transmission modes depends on the sensor type and its capabilities (not all the sensor types support all them). Besides a transmission mode supported by the sensor may be “selectable” or “not selectable” according to the specific sensor rules and the state of transmission frequencies.

Both the set of transmission frequencies offered by the sensor and the supported transmission modes (names and characteristics) differ from sensor to sensor, so they shall be handled as configuration parameters. The sensor reports all supported frequencies whether or not currently available or enabled. Sensors cannot enable/disable the setting of the frequency usage at its own initiative, but at any time a transmission frequency could become not available because of a fault (e.g., fault of the relevant oscillator), and this could affect the effective availability of one or more sensor supported transmission modes.

**Provision of the frequency usage state**
Sensor shall keep CMS informed about the current availability of the frequency usage and its changes (if any).

**Provision of the transmission mode**
Sensor shall keep CMS informed about the currently selected transmission mode, with the relevant parameters, and its changes (if any).

It is the CMS's responsibility to initiate the determination of initial state by making a request for information to the subsystem.

**Lack of mastership**
In the case where CMS does not have mastership of the sensor, CMS shall be informed about both the actual setting of the frequency usage and the actual transmission mode, with its changes (if any).

**State of transmission frequencies**
With respect to its operational use each sensor transmission frequency may be "enabled" or "disabled", according to the relevant setting. On the other hand, with respect to its health status, each transmission frequency may be "available" or "not available" according to the presence of faults. Note that a transmission frequency may be effectively selectable for the sensor transmission if it is both "enabled" and not in fault.

**Relationship to Manage Transmission Sectors**
As well as the overall transmission mode, here specified, CMS may define sectors where a devoted transmission mode is to be applied (see Manage Transmission Sectors).

**Pre-condition**: Mastership Required; CMS has mastership of the sensor.

**Pre-condition**: Subsystem Services; Provide subsystem services is successfully passed.

**Pre-condition**: Transmission Frequencies; CMS knows the transmission frequencies offered by the sensor and their actual availability.

**Pre-condition**: Selectable Transmission modes and frequencies; CMS is aware of the currently selectable transmission modes and transmission frequencies.

**Post-condition**: Success; Both the setting of the frequency usage and the sensor transmission mode are according to the request and CMS is informed that this is the case.
Post-condition: No Success: Both the setting of the frequency usage and the sensor transmission mode are unchanged with respect to the original one and CMS is informed that this is the case.
**Table 7.215340 - Methods of Interface Manage_Frequency_Usage_CMS**

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_frequencies_state()</td>
<td>Method used by the sensor to return the current availability of the frequency usage and its changes (if any).</td>
<td>all_frequencies_state_type frequencies_state</td>
</tr>
</tbody>
</table>
| report_transmission_mode_state() | Method used by the sensor to return the selected transmission mode, with the relevant parameters, and its changes (if any). | request_id_type request_id  
transmission_frequency_mode_type  
transmissionModeSetting         |
| transmission_frequency_state_response() | Method used by the sensor to return the actual setting of the frequency usage modified according to the request. | request_id_type request_id  
selected_frequency_list_type  
setting_message                |

**Manage_Frequency_Usage_Sub**

**Type:** Interface

**Package:** Manage_Frequency_Usage

This is the Subsystem interface for managing frequency usage.

**Table 7.216341 - Methods of Interface Manage_Frequency_Usage_Sub**

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
</table>
| set_frequencies()       | Method used by the CMS to enable or disable frequency bands or discrete frequencies. | request_id_type request_id  
selected_frequency_list_type  
request                      |
| set_transmission_mode() | Method used by the CMS to select the available sensor transmission mode. | request_id_type request_id  
transmission_frequency_mode_type  
transmissionmode         |

**Diagram**

The sensor reports all supported frequencies, whether or not currently available or enabled. The sensor can return the current availability of the frequency usage and its changes (if any) periodically or upon change. The sensor also responds to requests to set or change the transmission mode and frequency state.
Figure 7.147.181 Basic Flow - Frequency Availability Change Notification (Interaction diagram)
Figure 7.148.182 Basic Flow - Enable/Disable Frequency Usage (Interaction diagram)
set_frequencies(request_id_type, frequencies_set_request)
and isssfullywledged before deletion:
transmission_frequency_state_response(request_id_type, selected_frequency_list_type)

receive_error(request_id, commsuccessfails, error_reason)

receive_acknowledgement(request_id_type, request_ack_type)
Figure 7.149.183 Alternative Flow - Enable/Disable Frequency Usage - loss of mastership (Interaction diagram)
Figure 7.150.184 Basic Flow - Transmission Mode Selection (Interaction diagram)
set_transmission_mode(request_id_type, transmission_frequency_mode_type)

alt [Subsystem rejects request]
receive_acknowledgement(request_id_type, request_ack_type)

alt [Subsystem fails]
receive_acknowledgement(request_id_type, request_ack_type)
receive_error(request_id, error_reason)

report_transmission_mode_state(request_id_type, transmission_frequency_mode_type)

mand successfully acknowledged but before completion
report_transmission_mode_state(request_id_type, transmission_frequency_mode_type)
7.9.7.2 Manage_Transmission_Sectors

Parent Package: Sensor_Control

This package contains interfaces for the Manage Transmission Sectors service.

Type: Interface common_use_case_interface

Package: Manage_Transmission_Sectors

Figure 7.151.185 Alternative Flow - Transmission Mode Selection - loss of mastership (Interaction diagram)
This determines the sectors where the sensor is allowed to radiate together with the relevant transmission modes and parameters. Sectors may be delimited in azimuth only, or both in azimuth and elevation; for each sector the sensor may be requested either to no transmit at all or to apply a proper transmission mode.

Typical transmission sectors types are:

- **Transmit Inhibit Sectors**
  sectors where the sensor is not allowed to radiate. Depending on the sensor type and its capabilities, such a type of sectors may be delimited in azimuth only, or both in azimuth and elevation.

- **Reduced Radiate Power Sectors**
  sectors where the sensor shall radiate at reduced power. Depending on the sensor type and its capabilities, such a type of sectors may be delimited in azimuth only or both in azimuth and elevation.

- **Transmission Mode Sectors**
  sectors where the sensor is required to apply a devoted transmission mode (see Manage Frequency Usage). Depending on the sensor type and its capabilities, such a type of sectors may be delimited in azimuth only or both in azimuth and elevation, but they may not overlap each other.

- **Blind Arc Sectors**
  sectors where the sensor is not allowed to radiate. Such a type of sectors may be delimited in azimuth only, or both in azimuth and elevation, depending on the sensor type and its capabilities. (Note: the same as “Transmit Inhibit Sectors”, with the difference that sectors are defined in Ship’s Reference System.)

Provision of the sensor transmission sectors setting
Sensor shall keep CMS informed about the actual setting of the transmission sectors and its changes (if any).

It is the CMS's responsibility to initiate the determination of initial state by making a request for information to the subsystem.

Lack of mastership
In the case where CMS does not have mastership of the sensor, CMS shall be informed about the actual setting of the transmission sectors and its changes (if any).

Pre-condition: Mastership Required; CMS has mastership of the sensor
Pre-condition: Subsystem Services: Provide subsystem services is successfully passed

Post-condition—Success: The setting of the transmission sectors has been modified according to the request and CMS is informed that this is the case.
Post-condition—No Success: The setting of the transmission sectors is unchanged with respect to the original one and CMS is informed that this is the case.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>transmission_sector_setting()</td>
<td>Method used by the sensor to return the actual setting of the transmission sectors and its changes (if any).</td>
<td>request_id_type request_id transmission_sector_set_type setting_message</td>
</tr>
</tbody>
</table>

**Table 7.211342 - Methods of Interface Manage_Transmission_Sectors_CMS**

Manage_Transmission_Sectors_Sub
Type: __Interface
Package: Manage_Transmission_Sectors
This is the Subsystem interface for managing transmission sectors.
Table 7.343 - Methods of Interface Manage_Transmission_Sectors_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>set_transmission_sector()</td>
<td>Method used by the CMS to send a set/reset transmission sector request to the sensor.</td>
<td>request_id_type request_id transmission_sector_set_type_sector</td>
</tr>
</tbody>
</table>

Figure 7.344 | Basic Flow - Manage Transmission Sectors - Enable/Disable (Interaction diagram)
set_transmission_sector(request_id_type, transmission_sector_set_type)

The transmission_sector_set parameter must be not null.

receive_error(request_id_type, error_reason_type)

command is successfully acknowledged but fails before completion.

receive_acknowledgement(request_id_type, request_ack_type)

rejects request.

transmit_transmission_settings(request_id_type, request_set_type)
7.8.7.3 Control_Emissions

Parent Package: Sensor_Control

This package contains interfaces for the Control Emissions service.

Control_Emissions_CMS

Type: Interface common_use_case_interface

Package: Control_Emissions

The sensor is requested to inhibit/enable own emissions. In the case where the sensor is a radar, this shall result in the Radiation on/off command. Note that this interface just covers the software managed control of the emission state. For safety reasons many sensors are supplied with an additional hardware control of own emission state, such as a pushbutton directly connected to the transmitter.

Provision of the Emission state
Sensor shall keep CMS informed about the current state of emissions and its changes (if any).
It is the CMS's responsibility to initiate the determination of initial state by making a request for information to the subsystem.

Lack of mastership
In the case where CMS does not have mastership of the sensor, CMS shall be informed about the current emissions state and its changes (if any).

Relationship to the Transmission Sectors management
As long as emissions are on, the sensor shall transmit in the sectors where transmission is allowed and according to the relevant transmission modes and parameters, as determined through Manage Transmission Sectors.

Pre-condition: Mastership Required; CMS has mastership of the sensor
Pre-condition: Subsystem Services; Provide subsystem services is successfully passed

Pre-condition: Emissions State; CMS is aware that actually the sensor may switch its emissions state, e.g. both the technical state and the health state allow the sensor to switch to Radiation on, no engagement in execution to switch to Radiation off, and so on.

Post-condition: Success: The sensor emissions state is on/off as requested and CMS is informed that this is the case.
Post-condition: No Success: The sensor emissions state is still equal to the original one and CMS has the correct information regarding that state.

Table 7.218344 - Methods of Interface Control_Emissions_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
</table>
| control_emission_setting() | Method used by the sensor to return the current state of emissions and its changes (if any). | request_id_type request_id
|                       |                                                                       | control_emission_state_type control_emission_state | emission_state |

Control_Emissions_Sub
Type: Interface
Package: Control_Emissions
This is the Subsystem interface for controlling emissions.

Table 7.219345 - Methods of Interface Control_Emissions_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>set_control_emission()</td>
<td>Method used by the CMS to send an Emissions on/off request to the sensor.</td>
<td>request_id_type request_id control_emission_state_type control_emission_state</td>
</tr>
</tbody>
</table>
Figure 7.154.188 Basic Flow - Control Emissions - On/Off (Interaction diagram)
set_control_emission(request_id_type, control_emission_state)

alt

Subsystem rejects request

receive_acknowledgement(request_id_type, request_ack_type)

Subsystem fails

receive_error(request_id_type, error_reason_type)

message is successfully acknowledged but lost before completion

control_emission_setting(request_id_type, control_emission_state_type)
7.9.7.4 Define_Test_Target_Scenario

**Parent Package:** Sensor_Control

This package contains interfaces for the Define Test Target Scenario service.

**Define_Test_Target_Scenario_CMS**

**Type:** Interface common_use_case_interface

**Package:** Define_Test_Target_Scenario

This specifies the interactions for defining and modifying a test scenario. A Test Target scenario consists of a number of Test Targets to be generated according to their characteristics (positions, motion law, generation parameters) with the purpose of producing stimuli devoted to the execution of an internal functional test of the sensor.

A number of Test Target scenarios may be maintained in a sensor internal Test Targets scenarios database, where each scenario is identified by a unique identification number. Write accesses to this database shall be rejected if the sensor Mastership is not actually assigned to CMS, but the possession of the sensor Mastership is not required for executing read accesses.

The generation of the so defined Test Target scenarios may be activated as specified in Control Test Target Facility. For the generation mechanism see the interface Control Test Target Facility.

One or more Test Target scenarios may be maintained in a sensor internal Test Targets scenarios database, where each scenario is identified by a unique identification number. The number of available Test Target scenarios is accessed by Manage subsystem parameters.
Depending on the sensor type and its capabilities, a Test Target scenario may be constituted by:
a) a number of independent targets, with each target having own characteristic parameters; so the scenario is defined by:

- number of targets
- for each target
  - the initial target position with the relevant initial time
  - target parameters

b) a number of targets distributed in a defined area/volume and having the same common parameters, so the scenario is defined by:

- number of targets
- area/volume boundaries
- common initial time
- common targets parameters

Target parameters define:

- the target motion type, with the relevant motion parameters
- the target generation parameters, such as injection type (internal / external), attenuation law (constant / variable-with-range), doppler type (0 / PRF/2).

Pre-condition: Mastership Required: CMS has mastership of the sensor
Pre-condition: Subsystem Services: Provide subsystem services is successfully passed
Pre-condition: Test Target Facility: Test Target facility is supported by the sensor and CMS is aware of which types of Test Target the sensor may manage

Post-condition:
Success: Write access:
The specified Test Target scenario is modified according to the request and CMS is informed that this is the case.

Read access:
The requested Test Target scenario is reported to CMS.

Post-condition: No Success: Write access:
The specified Test Target scenario is unchanged and CMS is informed about the denial reason.

Read access:
The requested Test Target scenario is not reported to CMS and CMS is informed about the denial reason.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>test_target_scenario_setting()</td>
<td>Method used by the sensor to return the identification number of the modified or created test target scenario.</td>
<td>request_id_type request_id test_target_scenario_id_type test_target_scenario_id</td>
</tr>
<tr>
<td>test_target_scenario_setting_all_features()</td>
<td>Method used by the sensor to return the required test target scenario with its parameters.</td>
<td>request_id_type request_id test_target_scenario_type test_target_features</td>
</tr>
</tbody>
</table>

Table 7.220346 - Methods of Interface Define_Test_Target_Scenario_CMS

Open Architecture Radar Interface Standard (OARIS), v2.0 529
Package: Define_Test_Target_Scenario
This is the Subsystem interface for defining test target scenarios.
Table 7.24347 - Methods of Interface Define_Test_Target_Scenario_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>read_test_target_scenario()</td>
<td>Method used by the CMS to send to the sensor a read request of a specified Test Target scenario.</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>test_target_scenario_id_type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>test_target_scenario_id</td>
</tr>
<tr>
<td>write_test_target_scenario()</td>
<td>Method used by the CMS to send to the sensor a write request of a specified Test Target scenario.</td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>test_target_scenario_id_type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>test_target_scenario</td>
</tr>
</tbody>
</table>

*idlInterface* Define_Test_Target_Scenario_CMS

*idlInterface* Define_Test_Target_Scenario_Sub

write_test_target_scenario(request_id_type, test_target_scenario_type)

receive_acknowledgement(request_id_type, request_ack_type)

test_target_scenario_setting(request_id_type, test_target_scenario_id_type)
Figure 7.156.190 Basic Flow - Write a Target Test Target Scenario (Interaction diagram)
write_test_target_scenario(request_id_type, test_target_scenario_type)

command is successfully acknowledged but fails before completing

receive_error(request_id_type, error_reason_type)

rejects request

receive_acknowledgement(request_id_type, request_ack_type)

receive_error(request_id_type, error_reason_type)

receive_acknowledgement(request_id_type, request_ack_type)

receive_error(request_id_type, error_reason_type)

receive_acknowledgement(request_id_type, request_ack_type)
Figure 7.157.191 Alternative Flow - Write a Target Test Target Scenario - loss of mastership (Interaction diagram)
7.8.7.5.192 Test_Target_Facility

Parent Package: Sensor_Control
This package contains interfaces for the Test Target Facility service.

**Test_Target_Facility_CMS**

**Type:** Interface *common_use_case_interface*

**Package:** Test_Target_Facility

The sensor is requested to activate/deactivate the execution of its internal functional test and stimulation realized by means of test targets generation. A number of Test Target scenarios may be defined and modified as specified in *Define Test Target Scenario*, each scenario is identified by a proper identification. At any time no more than one Test Target scenario may be active.

### Test Target generation mechanism (applicable to some sensors)

The Test Target generation consists of the injection of proper signals at different points of the receiver chain in order to produce the relevant detections in input to the RMC (Radar Management Computer); these Test Target detections are processed by the RMC as the real ones, so they shall generate one or more plots ("Test Target" plots) and tracks ("Test Target" tracks).

Such a generation mechanism is controlled by the RMC driving a devoted hardware, its purpose is to execute an on-line BITE of the complete receiver chain.

Test Target generation is executed while the radar is working in operational mode, so Test Target detections and real detections live together, forming "Test Target" plots and tracks at the same time as real plots and tracks. This implies that CMS shall receive "Test Target" plots and tracks together with real plots and tracks.

### Lack of mastership

In the case where CMS does not have mastership of the sensor, CMS shall be informed about the actual state of the Test Target generation and its changes (if any).

### Provision of the Test Target generation state

Sensor shall keep CMS informed about the actual state of the Test Target generation and its changes (if any).

### Relationship to the subsystem health state

As long as a Test Target scenario is in generation sensor checks the relevant returns at different points of the receiver chain, up to form plots in the same positions where Test Targets have been generated. The relevant results contribute to the sensor health state.

**Pre-condition:** Mastership Required; CMS has mastership of the sensor

**Pre-condition:** Subsystem Services: *Provide subsystem services* is successfully passed

**Pre-condition:** Test Target facility: Test Target facility is supported by the sensor and CMS is aware of the current availability of the Test Target generation.

**Post-condition:** Success: The state of the Test Target generation is modified according to the request and CMS is informed that this is the case.

**Post-condition:** No Success: The state of the Test Target generation is unchanged with respect the original one and CMS is informed about the denial reason.

### Table 7.222348 - Methods of Interface Test_Target_Facility_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
</table>
| notify_test_target()  | Method used by the sensor to return the actual state of the Test Target generation consistent with the request. | request_id_type request_id
|                       |                                                                      | test_target_scenario_state_type test_target_scenario_state |

**Test_Target_Facility_Sub**

**Type:** Interface

**Package:** Test_Target_Facility

This is the Subsystem interface for testing target facilities.
### Table 7.349 - Methods of Interface Test Target Facility Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>set_test_target_facility_state()</td>
<td>Method used by the CMS to send an activation request of a specified Test Target scenario.</td>
<td>request_id_type, request_id, test_target_scenario_state_type, scenario_state</td>
</tr>
</tbody>
</table>

![Basic Flow - Activate/Deactivate Test Target Facility (Interaction diagram)]
set_test_target_facility_state(request_id_type, test_target_scenario_state)

receive_termination(request_id, request_ack_type)

receive_termination(request_id, request_ack_type)

receive_error(request_id, error_reason)

notify_test_target(request_id_type, test_target_scenario_state_type)

command is successfully acknowledged but fails before completion

set_test_target_facility_state(request_id_type, test_target_scenario_state)

receive_termination(request_id, request_ack_type)

receive_error(request_id, error_reason)

notify_test_target(request_id_type, test_target_scenario_state_type)
Figure 7.160 Alternative Flow - Activate/Deactivate Test Target Facility - loss of mastership (Interaction diagram)
7.8.7.9.8 Sensor_Performance

Parent Package: Sensor_Services
This package contains interfaces for the Sensor Performance service.

7.9.8.8 Sensor_Performance

7.9.8.1 Provide_Interference_Reports

7.8.8.1 Parent Package: Sensor_Performance

Provide_Interference_Reports_CMS

Type: Interface common_use_case_interface
Package: Provide_Interference_Reports

This describes the process whereby the subsystem provides a set of reports on sources of interference, including jammers. The data shall, therefore, in general, be non-real-time but should, where appropriate, be time-tagged and shall be updated when any observed data changes. The sensor need not be radiating but shall at least be receiving. The subsystem VOI (volume of interest) or other filter mechanisms might be supplied in a request to the subsystem.

For a nominal effect assessment, the request might contain data on number, strength/Effective Radiated Power (ERP), type and deployment of jammers and other interferers affecting radar operations. For example, for each interferer:

- Sensor time-tag
- Interference type - active noise, self-screening jammer, standoff jammer etc
- Strength/Effective Radiated Power
- Locations - strobes etc.
- Affected sectors
- Frequency bands affected

Pre-condition: Technical State: The subsystem is in technical state ONLINE.
Pre-condition: Subsystem Services: The Provide Subsystem Services Service has been completed successfully
Pre-condition: Register Interest: The Register Interest Service has been executed successfully to register interest in Interference Reports.
Post-condition: Success: The CMS has received Interference Reports
Post-condition: Failure: The CMS receives no Interference Reports

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>interference_report_response()</td>
<td>Provides an updated set of interference reports to the CMS.</td>
<td>request_id_type request_id interference_report_type interference_report The report on interference</td>
</tr>
<tr>
<td>interference_report_periodic()</td>
<td>Provides an updated set of interference reports to the CMS.</td>
<td>interference_report_type interference_report The report on interference</td>
</tr>
</tbody>
</table>

Provide_Interference_Reports_Sub

Type: Interface

540 Open Architecture Radar Interface Standard (OARIS), v2.0
<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>volume_for_interference_reports()</td>
<td>This allows definition of the volume in space which is of interest with regard to the provision of interference reports.</td>
<td>request_id_type request_id The unique identifier for this request. This is referenced in acknowledgement and any error reporting regarding this definition of the volume of interest. polar_volume_type volume The volume in space coordinate_orientation_type coordinate_orientation specifies the orientation of the polar volume</td>
</tr>
</tbody>
</table>
«idlInterface»
Provide_Interference_Reports_CMS

«idlInterface»
Provide_INTERFERENCE_REPORTS_SUB

opt
Volume
of
Interest
Supplied

[CMS supplies volume of interest]

volume_for_interference_reports(request_id_type, polar_volume_type, coordinate_orientation_type)

alt Unsuccessful Request

[Subsystem unable to filter interference reports to the requested volume of interest]

receive_acknowledgement(request_id_type, request_ack_type)

negative acknowledgment

positive acknowledgment

receive_error(request_id_type, error_reason_type)

[Error occurs whilst preparing interference reports as requested]

receive_acknowledgement(request_id_type, request_ack_type)

[Subsys
Figure 7.161.195 Alternative Flow - Provide Interference Reports (Interaction diagram)
Provide Interference Reports (CMS)

loop periodic

volume_for_interference_reports(request_id_type, volume_type, volume_orientation_type)

location_for_interference_reports(request_id_type, volume_type, volume_orientation_type)

interference_report_periodic(request_id_type)

interference_reportresponse(request_id_type, interference_report_periodic)

interference_report_periodic(request_id_type)

interference_reportresponse(request_id_type, interference_report_periodic)

loop periodic

provide_interference_reports

volume_for_interference_reports(request_id_type, volume_type, volume_orientation_type)

location_for_interference_reports(request_id_type, volume_type, volume_orientation_type)

interference_report_periodic(request_id_type)

interference_reportresponse(request_id_type, interference_report_periodic)

interference_report_periodic(request_id_type)

interference_reportresponse(request_id_type, interference_report_periodic)
7.9.8.2 Provide_Nominal_Performance

Parent Package: Sensor_Performance

Provide_Nominal_Performance_CMS
Type: Interface common_use_case_interface
Package: Provide_Nominal_Performance

This is incremental to Register Interest, which deals with the subscription to subsystem functions. It provides an indication of the expected performance of the available subsystem services such as those presented in Provide Subsystem Services, based upon the current environmental conditions (See Receive Meteorological Data - METOC). The subsystem need not be radiating to provide this assessment. This interface is more targeted towards a subsystem such as the complex MFR than the 2D surveillance radar. The most basic example of performance would be reporting of the nominal coverage, in elevation, azimuth and range, given an assumed operating regime with no jamming and with default clutter conditions. Other examples might be that the actor requests the probability of detection for a specified target type or perhaps the probability of correct automatic classification of such a target within a specified sector of coverage under current environmental conditions.

Pre-condition: Technical State: The Subsystem is in the Technical State ONLINE.
Pre-condition: Subsystem Services: The Provide Subsystem Services Service has been executed successfully.
Post-condition: Success: The CMS is aware of the Nominal Performance of the Subsystem
Post-condition: Failure: The CMS is not aware of the Nominal Performance of the Subsystem
Table 7.352 - Methods of Interface Provide_Nominal_Performance_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>nominal_performance_response()</td>
<td>The subsystem responds to the previous nominal performance request with its determination of the requested aspect of nominal performance.</td>
<td>request_id_type, request_id The unique id from the request. performance_assessment_report_type The report on nominal performance.</td>
</tr>
</tbody>
</table>

Provide_Nominal_Performance_Sub

**Type:** Interface

**Package:** Provide_Nominal_Performance

Subsystem interface for provision of nominal performance assessment.

Table 7.225353 - Methods of Interface Provide_Nominal_Performance_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>nominal_performance_request()</td>
<td>The CMS requests nominal performance of the subsystem in the current environmental conditions. The aspect of performance requested is a parameter of the request.</td>
<td>request_id_type, request_id The unique id which identifies this request. It is used to mark replies from the sensor relating to this request. performance_assessment_request_type The details of the performance request.</td>
</tr>
</tbody>
</table>
Figure 7.163.197 Alternative Flow - Provide Nominal Performance (Interaction diagram)
7.9.8.3 Provide_Performance_Assessment

Parent Package: Sensor_Performance

Type: Interface common_use_case_interface

Package: Provide_Performance_Assessment

This is incremental to Register Interest, which deals with the subscription to subsystem functions and Provide Nominal Performance which provides the subsystem nominal performance. This interface reports the real-time performance of the available subsystem functions against the goals of the mission. The reported performance is that currently being attained by the subsystem subject to the current operating regime and environmental conditions, including any clutter and jamming and taking account of any mitigation/cancellation of such effects by the subsystem.

This interface is aimed at a subsystem such as an MFR radar. Information is provided to the Command function allowing decisions to be made on the achieved performance, which is often considerably different to the anticipated performance level as reported through the Provide Nominal Performance Service.

The most basic example of performance would be reporting of the radar coverage, in elevation, azimuth and range, for the current operating regime and environmental conditions. This would take account of any clutter and jamming present. Other examples might be that the actor requests the probability of detection for a specified target type or perhaps the probability of correct automatic classification of such a target within a specified range under current environmental conditions N.B. if the radar is operating in an appropriate mode.
then real-time clutter and/or jamming data might be available to the radar subsystem. Otherwise the actor would have to supply any known data to the subsystem for performance assessment (see Receive Encyclopaedic Data and Receive Geographic Information). If no environmental data is specified then the design performance would be reported.

Pre-condition: Technical State: The Subsystem is in the technical state ONLINE.
Pre-condition: Subsystem Services: The Provide Subsystem Services Service has completed successfully.

Post-condition: Success: The CMS is aware of the assessed performance of the subsystem
Post-condition: Failure: The CMS is not aware of the assessed performance of the subsystem

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>performance_assessment_response()</td>
<td>The subsystem responds to the previous performance assessment request with its assessment of the requested aspect of actual performance.</td>
<td>request_id_type request_id The unique identifier for this assessment. This identifier is supplied by the CMS when the assessment is requested. performance_assessment_report_type performance_assessment The details of the assessment</td>
</tr>
</tbody>
</table>

Table 7.226354 - Methods of Interface Provide_Performance_Assessment_CMS

Provide_Performance_Assessment_Sub
Type: Interface
Package: Provide_Performance_Assessment
Subsystem interface for provision of current performance assessment.
Note that the coordinates are always polar for this service and that the origin is always the sensor reference point as per the coordinates and positions package.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>performance_assessment_request()</td>
<td>The CMS requests assessment of actual performance of the subsystem. The aspect of performance requested is a parameter of the request.</td>
<td>request_id_type request_id The unique identifier for this assessment. This identifier is contained in all related replies from the sensor. performance_assessment_request_type parameters_type request Details of the assessment</td>
</tr>
</tbody>
</table>

Table 7.227355 - Methods of Interface Provide_Performance_Assessment_Sub
Figure 7.165.199 Alternate Flow - Provide_Performance_Assessment (Interaction diagram)
7.9.8.4 Provide_Jammer_Assessment

**Parent Package:** Sensor_Performance

**Type:** Interface common_use_case_interface

**Package:** Provide_Jammer_Assessment

This interface describes the process whereby the subsystem provides a periodic assessment of the effects of actual jamming on the detection and tracking performance of the subsystem. The actual subsystem performance vs the nominal (see Provide Nominal Performance) shall be reported so that this data is current and real-time. This should include the effects on (spatial) coverage caused by any jamming. The impact on frequencies used e.g. operating band limitations is dealt with in Provide Interference Reports Mastership is not required.

The radar need not be radiating in the ONLINE state but shall at least be receiving. The subsystem VOI (volume of interest) or other filter mechanisms might be supplied in a request to the subsystem.

The kind of information which could be provided in the returned assessment, depending on any jamming mitigation strategy (frequency agility, moving target indication, low side-lobe levels, main beam or side-lobe cancellation, side-lobe blanking etc.) might then include:

- Noise floor pre-/post-jammer cancellation, as applicable
- Degradation in detectability (compared with the nominal)
Pre-condition: Technical State: The subsystem is in the technical state ONLINE
Pre-condition: Subsystem Services: The Provide Subsystem Services Service has been successfully executed
Pre-condition: Register Interest: The Register Interest Service has completed successfully.
Post-condition: Success: CMS has received Jamming Effect Assessments
Post-condition: No Success: The CMS has not received Jamming Effect Assessments.

Table 7.228356 - Methods of Interface Provide_Jammer_Assessment_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>jammer_assessment_response()</td>
<td></td>
<td>request_id_type request_id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>performance_assessment_report_typ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>report</td>
</tr>
</tbody>
</table>

Provide_Jammer_Assessment_Sub
Type: Interface
Package: Provide_Jammer_Assessment
## Table 7.229357 - Methods of Interface Provide_Jammer_Assessment_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>jammer_assessment_request()</td>
<td></td>
<td>request_id_type request_id performance_assessment_request_type parameters_type jammer_assessment_request</td>
</tr>
</tbody>
</table>
Figure 7.167.201 Alternate Flow - Provide Jammer Assessment (Interaction diagram)
7.9.9 Track Reporting

**Parent Package:** Sensor_Services

This package contains interfaces for the Track Reporting service.

7.9.9.1 Filter Tracks

**Parent Package:** Track_Reporting

**Filter_Tracks_CMS**

**Type:** Interface

**Package:** Filter_Tracks

The interface to the CMS for receiving information relating to the filters used to control which tracks are made available to other network segments. The plot (and track) sharing architecture recognizes that connectivity between different platforms hosting sensors may not support the bandwidth required to share all plot and track updates. It is possible for a sensor also to provide the networking functionality in which case it is providing an additional role in the interface.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_track_filter()</td>
<td>Reports the parameters of one of the filters that are active for tracks in the communication and networking subsystem. Tracks are transmitted or received, according to their performance</td>
<td>filter_id: The identifier for the filter</td>
</tr>
</tbody>
</table>
Filter Tracks Sub

Type: Interface
Package: Filter Tracks

The interface to the subsystem for receiving updates to the filters used to control which tracks are made available to other network segments.

The plot (and track) sharing architecture recognizes that connectivity between different platforms hosting sensors may not support the bandwidth required to share all plot and track updates. In this use case the subsystem is the network component providing connectivity to other platforms, as distinct from the local sensors providing the plots. It is possible for a sensor to also provide the networking functionality, in which case it is providing an additional role in the interface.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>add_track_filter()</td>
<td>Adds a track filter to the communication and networking subsystem.</td>
<td>request_id_type request_id unique reference for the request</td>
</tr>
</tbody>
</table>
|                         |                                                                      | track filter parameters_type filter the values to be used to filter tracks by | accounts
| remove_track_filter()   | Removes a track filter from the communication and networking subsystem. | request_id_type request_id The unique reference for the request            |
|                         |                                                                      | filter_id_type filter_id The identifier for the filter to be removed        |
Figure 7.168 Basic Flow – Provide Jammer Assessment

Example Flow Filter_Tracks (Interaction diagram)

*idlInterface*

Filter_Tracks_Sub

Filter_Tracks_CMS

add_track_filter(request_id_type, track_filter_parameters_type)

receive_acknowledgement(request_id_type, request_ack_type)

report_track_filter(filter_id_type, track_filter_parameters_type)

remove_track_filter(request_id_type, filter_id_type)

receive_acknowledgement(request_id_type, request_ack_type)

track_filter_removed(filter_id_type)

alt failure cases

accepted = false

E.g., because there is no such sensor track, the external protocol is not supported or the track number is not in a recognized format.

Figure 7.204 Basic Flow Filter_Tracks (Interaction diagram)
7.9.9.2 Label_Tracks

**Parent: Track_Reporting**

**Parent Package:** Track_Reporting

**Package:** Label_Tracks

The interface to the CMS for track number labeling. The subsystem uses the mappings received in transmissions to other platforms. This enables distributed local pictures to be formed that are coherent with other protocols being used for sharing data.

### Table 7.360 - Methods of Interface Label_Tracks_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>external_track_label_response()</td>
<td></td>
<td>request_id_type request_id The unique identifier for the request sensor track id type sensor_track_id The identifier for the sensor track external_track_number_type external_track_number The external track numbers labelling the sensor track</td>
</tr>
</tbody>
</table>

### Table 7.361 - Methods of Interface Label_Tracks_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>label_track_for_external_protocol()</td>
<td>This operation is used to instruct the subsystem to label a sensor track with an external track number when transmitting it off-platform.</td>
<td>request_id_type request_id The unique identifier for the request sensor track id type sensor_track_id The identifier for the sensor track external_track_number_type external_track_number The external track number on an external protocol to label the track with</td>
</tr>
<tr>
<td>unlabel_track_for_external_protocol()</td>
<td>This operation is used to instruct the subsystem to no longer label a sensor track with a track number for a particular external protocol when transmitting it off-platform.</td>
<td>request_id_type request_id The unique identifier for the request sensor track id type sensor_track_id The identifier for the sensor track String external_protocol The name of the external protocol</td>
</tr>
</tbody>
</table>
E.g. because there is no such sensor track, the external protocol is not supported or the track number is not in a recognized format.

Figure 7.205 Alternate Flow Label Tracks (Interaction diagram)

Figure 7.206 Basic Flow Label Tracks (Interaction diagram)
7.9.9.3 Provider_Sensor_ServicesTracks
This package contains interfaces for the Track Reporting service.

7.8.9.1 Provider_Sensor_Tracks Parent Package: Track_Reporting

Provide_Sensor_Tracks_CMS
Type: Interface common_use_case_interface
Package: Provider_Sensor_Tracks
This service allows the CMS to obtain an overview of (real and/or simulated) air / land / space / surface objects observed or simulated. Information may cover all aspects of a track such as kinematic and amplifying information.

The service does not cover:
** additional track information provision dedicated for engagement support,
** special search functions such as cued search, volume search and horizon search (however, if such a search function is initiated by means of another service, the tracks shall be provided by this service).

Although the service focuses on radar as an example of a sensor, the service also applies to other sensors, like IR/EO sensors and ECM/ESM sensors.

The actor is the Combat Management System.

The service starts when:
** if the service does provide registration capabilities: the service "Register interest" has completed successfully or
• if the service does not provide registration capabilities: the service "Provide subsystem services" has completed successfully for this service.

The sensor provides, periodically or on event, a set of sensor tracks observed by the sensor. These may be sensor point or bearing tracks. The set of sensor tracks includes:

- Track updates of existing and new sensor tracks. These are provided when there are sufficient measurements (e.g. plots) in the last observation cycle, which may be associated with the sensor track;
- Dead-reckoned tracks. These are sensor track updates for which in the last observation cycle there
are no measurements that may be associated with the sensor track. For dead-reckoned tracks, the sensor track information (e.g. kinematics) is extrapolated. The dead-reckoned tracks may become "normal" tracks again if, in the next scan, there are measurement(s) that may be associated with the track. Alternatively, dead-reckoned tracks (after n unsuccessful scans) may become lost tracks.

**Lost tracks.** These are sensor track updates that are reported once, if in the last n scans, there are no measurements that may be associated with the sensor track. The value of n is typically a sensor parameter that is managed by the service "Manage subsystem parameters". Some sensors are not capable of reporting lost and/or dead-reckoned tracks. The sensor may also provide single sensor tracks periodically or on event.

The service ends with success when:

- if the service does provide registration capabilities: the service "Register interest" has completed successfully for a deregistration request, or
- if the service does not provide registration capabilities: the sensor is shutdown using service "Shut down".

**Pre-condition:**
- Sensor health state: The sensor and the service need to be in the health state AVAILABLE or DEGRADED
- Sensor parameters: The relevant sensor parameters (e.g. allowed frequencies, transmission sectors) need to be set.

---

1 The manner in which this is done is described in other services of the OARIS (“Manage frequency usage”, “Manage transmission sectors”, “Control emissions” and “Manage subsystem parameters”).

## Table 7.230362 - Methods of Interface Provide Sensor Tracks CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>write_sensor_track()</td>
<td>The method represents a write of a single sensor track (air, land, space or surface) to the CMS. The write may be periodic or not. MethodTag = ea_guid = {C4602AC-1306-5578-9E2C-7A0C-42EAA3A2}</td>
<td>sensor_track_type, the_sensor_track</td>
</tr>
<tr>
<td>write_sensor_track_set()</td>
<td>The method represents a single write of a set of sensor tracks to the CMS. The write may be periodic or not. Include all tracks observed during a sensor scan. Be an update of just one track (a set of 1) if this is how the sensor works.</td>
<td>sensor_track_set_type, the_track_set</td>
</tr>
<tr>
<td>write_sensor_track()</td>
<td>The method represents a write of a single sensor track (air, land, space or surface) to the CMS. The write may be periodic or not.</td>
<td>sensor_track_type, the_sensor_track</td>
</tr>
<tr>
<td>write_sensor_track_set()</td>
<td>The method represents a single write of a set of sensor tracks to the CMS. The write may be periodic or not.</td>
<td>sensor_track_set_type, the_track_set</td>
</tr>
</tbody>
</table>
periodic or not
- include all tracks observed during a
sensor scan
- be an update of just one track (a set
of 1) if this is how the sensor works

delete_sensor_track() The method represents a deletion of
a single sensor track (air, land, space
or surface) to the CMS. After a
deletion, no further writes for that
sensor track instance are made.
Subsystems may subsequently use
the same sensor_track_id to denote
another sensor track instance
possibly after a 'cooling off period'.
Such behavior is implementation
specific.

sensor_track_id_type
sensor_track_id

Method
Notes
Parameters
prioritize_track() CMS requests the subsystem to
report the referenced sensor track
with the stated priority.
MethodTag: ea_guid =
{47B331AA-50E2-4afD-B7C3-
86E27DADBFB55}
request_id_type
request_id
sensor_track_id_type
sensor_track_id
track_priority_type
priority

remove_track_priority() The CMS requests the subsystem to
report the track with the default
priority for that subsystem.
MethodTag: ea_guid =
{0FD4DA0F-644B-4128-A064-
5888E3B04FF8}
request_id_type
request_id
sensor_track_id_type
sensor_track_id

Open Architecture Radar Interface Standard (OARIS), v2.0
prioritize_track()

CMS requests the subsystem to report the referenced sensor track with the stated priority.

MethodTag: Issue = request_id_type, request_id, sensor_track_id_type, sensor_track_id, track_priority_type

remove_track_priority()

The CMS requests the subsystem to report the track with the default priority for that subsystem.

request_id_type, request_id, sensor_track_id_type, sensor_track_id

alt Failure Conditions

- Sensor track does not exist
- Priority level not supported
- Insufficient resources

receive_acknowledgement(request_id_type, request_ack_type)

receive_error(request_id_type, error_reason_type)
Figure 7. Alternative Flow - Track Prioritization (Interaction diagram)
prioritized_reporting

- Write sensor track(sensor_track_type)
- Prioritize track(request_id_type, sensor_track_id_type, track_priority_type)
- Receive acknowledgment(request_id_type, request_ack_type)

Normal Reporting

- Write sensor track(sensor_track_type)
- Remove track priority(request_id_type, sensor_track_id_type)
- Receive acknowledgment(request_id_type, request_ack_type)

Normal Rep outing

- Write sensor track(sensor_track_type)
Figure 7.209 Basic Flow - Change Priority (Interaction diagram)
This sequence diagram shows the style of reporting tracks individually. Depending on the requested services, all tracks are reported or for instance only tracks with a certain environment or jamming indication. The messages may be sent periodically or on event (when a new track update is available).

Figure 7.210 Basic Flow - Sensor Track Reporting (Individual) (Interaction diagram)
This sequence diagram shows the style of reporting tracks in batches; sets containing one or more tracks are reported atomically. Depending on the requested services, all tracks are reported or for instance only tracks with a certain environment or jamming indication. The messages may be sent periodically or on event (when a new track update is available).

Figure 7.422.211 Basic Flow - Sensor Track Reporting (Sets) (Interaction diagram)

7.8.107.9.10 Tracking_Control
Parent Package: Sensor_Services
This package contains interfaces for the Tracking Control service.

7.9.10.1 Delete Sensor Track

7.8.10.14.1.1 Delete Sensor Track
Open Architecture Radar Interface Standard (OARIS), v2.0

Parent Package: Tracking_Control
This package contains interfaces for the Delete Sensor Track service.

Delete_Sensor_Track_CMS
Type: Interface common_use_case_interface
Package: Delete_Sensor_Track
The sensor is requested to remove a specified track from its internal Track Data Base; obviously the deleted track may come back (with another track identification number) within a few seconds if it was a living track.

Pre-condition: Mastership Required: CMS has mastership of the sensor
Pre-condition: Subsystem Services: Provide subsystem services is successfully passed
Pre-condition: Tracking capability: Tracking capability is supported by the sensor, and CMS is aware that actually the sensor may delete that track.

Post-condition—Success: CMS is informed of the successful deletion of the required track, and the next track reporting shall no contain the deleted track. Obviously, the deleted track may come back within a few seconds if it was a living target, but with another identification number.

Post-condition—No Success: CMS is informed of the request rejection and of the denial reason. No impact on the sensor track management evolution.

Delete_Sensor_Track_Sub
Type: _Interface
Package: Delete_Sensor_Track
This is the Subsystem interface for deleting sensor tracks.

Table: Methods of Interface Delete_Sensor_Track_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>delete_track()</td>
<td>Method used by the CMS to send a track deletion request, specifying the identification number of the track to be deleted.</td>
<td>sensor_track_id_type trackId, request_id_type request_id</td>
</tr>
</tbody>
</table>
Figure 7.212 Basic Flow - Delete Sensor Track (Interaction diagram)

The deleted track is not included in the next track reporting returned by the sensor.

definition

delete_track(sensor_track_id_type, request_id)

receives

acknowledgement(request_id_type, request_ack_type)
Figure 7.174.213 Alternative Flow - Delete Sensor Track (Interaction diagram)

7.9.10.2 Receive_Track_Information
7.8.10.2 Parent Package: Tracking_Control
This package contains interfaces for the Receive Track Information service.

Receive_Track_Information_CMS

Type: Interface common_use_case_interface

Package: Receive_Track_Information

CMS may provide information belonging to a sensor track in order to enable for a coordinated presentation of the sensor track both on CMS consoles and a dedicated radar console. The track information which may be supplied are:

1. External track identification number
2. Additional Information – this is not specified as part of the interface, candidate information includes:
   - Track type
   - Track priority
   - Track Identification Category Assigned (Pending, Friend, Assumed Friend, Neutral, Unknown, Suspect, Hostile)

Track identities management
Each sensor track shall have an “Internal Track Identification Number” and may one or more additional “External Track Identification Numbers”. The former shall be assigned by the sensor when the track is formed and, as long as the track is alive, it cannot be changed for any reason. The latter shall be set to “none” when the track is formed and then overwritten, during the track life, to report the track identity/ies externally assigned to the track.

All track identification numbers shall be reported together with the track data, but the track identification shall be made through the “Internal Track Identification Number”.

Pre-condition: Mastership Required: CMS has mastership of the sensor
Pre-condition: Subsystem Services: Provide subsystem services is successfully passed
Pre-condition: Tracking capability: Tracking capability is supported by the sensor, and CMS is aware that
Actually the sensor may manage that track

Pre-condition: Technical State: Sensor is working in Operational

Post-condition: Success: CMS is informed of the successful execution of the request, and the next track reporting shall contain the identified track with the provided information.

Post-condition: No Success: CMS is informed of the request rejection and of the denial reason. No impact on the sensor track management evolution.

Receive_Track_Information_Sub

Type: Interface

Package: Receive_Track_Information

This is the Subsystem interface for receiving track information.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
</table>
| insert_info_track()   | Method used by the CMS to send a receive track information request, specifying the track identification number and related track information. | request_id_type
|                       |                                                                       | request_id
|                       |                                                                       | sensor_track_id_type
|                       |                                                                       | trackId
|                       |                                                                       | track_info_type
|                       |                                                                       | trackInfo

Table 7.233365 - Methods of Interface Receive_Track_Information_Sub
The sensor shall provide the track updates as per "Provide Sensor Tracks".

Figure 7.125.214 Basic Flow - Receive Track Information (Interaction diagram)
The sensor shall not provide the track updates as per “Provide Sensor Tracks”.

The command is successfully acknowledged but fails before completion.

Subsystem rejects request

Subsystem fails

insert_info_track(request_id_type, sensor_track_id_type, track_info)

receive_acknowledgement(request_id_type, request_ack_type)

receive_error(request_id, error_reason)
7.8.10.3.1 Initiate_Track

Parent Package: Tracking_Control

This package contains interfaces for the Initiate Track service.

Package: Initiate_Track

Initiate_Track_CMS

Type: Interface common_use_case_interface

The sensor is requested to start tracking on a new target based on given information, such as positional data and additionally also kinematic data. Sensor replies indicating the request acceptance or rejection. If accepted, the initiation of a new track shall be attempted as required, and the relevant result shall be reported later through an "externally designated track initiation report" containing the identification number of the resulting track (if any).

Additional Information

Data reported in the "externally designated track initiation request"

The provided information depends on the sensor type and its capabilities, typically they are:

- Identification number of the designation (mandatory)
- Position and time (mandatory)
- Accuracy of the provided positional data (optional)
- Velocity and relevant accuracy (optional)
- Track characteristics (optional)

Data reported in the "externally designated track initiation report"

The purpose is this report is to inform CMS about the final result of the track initiation request, i.e. it
reports to CMS if the track has been successfully initiated or not, and (in case of success) the identification number of the new formed track. The provided information depends on the sensor type and its capabilities, typically they are:

- Identification number of the designation (mandatory)
- Initiation result (mandatory)
- Identification number of the initiated track, if any (mandatory)
- Other info (optional).

Pre-condition: Mastership Required
CMS has mastership of the sensor
Pre-condition: Subsystem Services
Provide subsystem services is successfully passed
Post-condition: Success
The setting of the tracking zones has been modified according to the request and CMS is informed that this is the case.
Post-condition: No Success
The setting of the tracking zones is unchanged with respect to the original one and CMS is informed that this is the case.

Table 7.234366 - Methods of Interface Initiate_Track_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
</table>
| report_track()  | Method used by the sensor to issue an "externally designated track initiation report" containing data of the successfully initiated track. | request_id_type request_id
sensor_track_id_type id_report |

Initiate_Track_Sub
Type: Interface
Package: Initiate_Track
This is the Subsystem interface for initiating tracks.

Table 7.235367 - Methods of Interface Initiate_Track_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
</table>
| initiate_track()| Method used by the CMS to send an "externally designated track initiation request", specifying a timed position and kinematic. | request_id_type request_id
system_track_type track_info |
The sensor shall provide the track updates as per “Provide Sensor Tracks.”
The sensor shall provide the track updates as per "Provide Sensor Tracks".

**Figure 7.216 Basic Flow Initiate Track (Interaction diagram)**

- `initiate_track(request_id_type, system_track)`
- `report_track(request_id_type, sensor_track_id_type)`
- `receive_acknowledgement(request_id_type, request_ack_type)`
- `receive_error(request_id, error_reason)`
7.9.10.4 **Manage_Tracking_Zones**

Parent Package: Tracking_Control

This package contains interfaces for the Manage Tracking Zones service.

Manage_Tracking_Zones_CMS

Type: Interface common_use_case_interface

Package: Manage_Tracking_Zones

This controls the sensor tracking behaviour in selected zones, which may be 1D (delimited in azimuth only), 2D (have additional elevation bounds) or 3D (have further range bounds). Depending on the zone type the sensor may be requested to modify its normal tracking behaviour, such as enable/disable the capability to auto initiate new tracks, or the capability of managing Track-On-Jammer. A list of typical tracking zones is:

- **Automatic Track Initiation Zones**
  zones where the sensor is allowed to auto initiate new tracks. Depending on the sensor type and its capabilities, such a type of zones may be delimited in azimuth only, or both in azimuth and elevation, or may have further range bounds, and in some cases also additional constraints (such as target type, velocity bounds, etc.).

- **Track-On-Jammer Sectors**
  sectors where the sensor is allowed to manage Track-On-Jammer. Depending on the sensor type and its capabilities, such a type of sectors may be delimited either in azimuth only or both in azimuth and elevation.

- **Multipath Devoted Tracking Sectors**
  sectors where the sensor is required to use, for tracking activities, devoted waveforms to reduce the multipath effects. This capability is usually provided by multifunctional radars. Such a type of sectors is usually limited in azimuth only, below a defined elevation.
The supported tracking zone types (names and characteristics) differ from sensor to sensor, so they shall be handled as configuration parameters. They shall be offered to the operator to enable him for a selection and then transferred to the sensor to achieve the intended response.

Special Requirements
Provision of the sensor tracking zones setting
Sensor shall keep CMS informed about the actual setting of the tracking zones and its changes (if any).

It is the CMS’s responsibility to initiate the determination of initial state by making a request for information to the subsystem.

Additional Information
Lack of mastership
In the case where CMS does not have mastership of the sensor, CMS shall be informed about the actual setting of the tracking zones and its changes (if any).

Pre-condition: Mastership Required; CMS has mastership of the sensor
Pre-condition: Subsystem Services: Provide subsystem services is successfully passed
Pre-condition: Tracking zones setting: CMS is aware of which types of tracking zones the sensor may manage and of their current setting.

Post-condition: Success: The setting of the tracking zones has been modified according to the request and CMS is informed that this is the case.
Post-condition: No Success: The setting of the tracking zones is unchanged with respect to the original one and CMS is informed that this is the case.

<table>
<thead>
<tr>
<th>Table 7.236368 - Methods of Interface Manage_Tracking_Zones_CMS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Method</strong></td>
</tr>
<tr>
<td>tracking_zone_setting()</td>
</tr>
</tbody>
</table>

Manage_Tracking_Zones_Sub
Type: Interface
Package: Manage_Tracking_Zones
This is the Subsystem interface for managing tracking zones.

<table>
<thead>
<tr>
<th>Table 7.237369 - Methods of Interface Manage_Tracking_Zones_Sub</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Method</strong></td>
</tr>
<tr>
<td>set_tracking_zone()</td>
</tr>
</tbody>
</table>
If tracking_zone_set dimension is null, the operation set_tracking_zone get all the current tracking zones.
set_tracking_zone(request_id_type, tracking_zone_set)

alt
[Subsystem rejects request]

receive_acknowledgement(request_id_type, request_ack_type)

[Subsystem fails]

receive_error(request_id, error_reason)

and
is
succesfully
wledged but
before
letion

tracking_zone_setting(request_id_type, tracking_zone_set)

In the operation set_tracking_zone, the tracking_zone_set parameter must be not null.
set_tracking_zone(request_id_type, tracking_zone_set)

Figure 7.219 Alternative Flow - Manage Tracking Zone - Enable/Disable - loss of Mastership (Interaction diagram)

7.97.10 Radar_Services
Parent Package: Service Interfaces
**7.9.10.1 Air_Engagement_Support**

**Parent Package:** Radar_Services

**Parent Package:** Service_Interfaces

Contains services associated with the Radar Domain.

**7.10.1.1 Provide_Projectile_Positional_Information**

**Parent Package:** Air_Engagement_Support

**Type:** Interface common_use_case_interface

**Package:** Provide_Projectile_Positional_Information

Fire control radars suitable for Close-In-Weapon-Systems need the capability to observe the projectiles in flight, to measure at which distance they pass the target so that related shot corrections for the gun may be calculated, automatically. The measured distance in azimuth and elevation is called miss indication in the following.

This capability may be available in a non-close-in-weapon-system environment, too. It may also be available for phased-array radars.

Mastership of the subsystem must not have any impact upon the miss indication capability.

See also service 'Process Target Designation'.

Pre-condition: "Process Target Designation" was successfully carried out and a target is being tracked.

Pre-condition: CMS must have mastership.

Pre-condition: CMS must have mastership.

---

**Table 7.238370 - Methods of Interface Provide_Projectile_Positional_Information_CMS**

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_miss_indication()</td>
<td>Via this message, the subsystem reports to the CMS the miss indication.</td>
<td>miss_indication_data_type MissIndicationData request_id_type RequestID</td>
</tr>
</tbody>
</table>

---

**Provide_Projectile_Positional_Information_Sub**

**Type:** Interface

**Package:** Provide_Projectile_Positional_Information

---

**Table 7.238371 - Methods of Interface Provide_Projectile_Positional_Information_Sub**

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_miss_indication()</td>
<td>Request the subsystem to report a miss indication.</td>
<td>request_id_type RequestID expected_hit_data_type ExpectedHitData</td>
</tr>
</tbody>
</table>
Figure 7.220 Provide projectile positional information - Request reporting of miss indications (Interaction diagram)

«idlInterface»
Provide_Projectile_Positional_Information_CMS

«idlInterface»
Provide_Projectile_Positional_Information_Sub

request_miss_indication(request_id_type, expected_hit_data_type)

receive_acknowledgement(request_id_type, request_ack_type)
request_ack.success = true

request_miss_indication(request_id_type, expected_hit_data_type)

receive_acknowledgement(request_id_type, request_ack_type)
request_ack.success = true
request_miss_indication(miss_indication_data_type, request_id_type)
request_ack.success = true

receive_error(request_id, error_reason)
request_ack.success = false
receive_acknowledgement(request_id_type, request_ack_type)
request_ack.success = true

receive_acknowledgement(request_id_type, request_ack_type)
report_miss_indication(miss_indication_data_type, request_id_type)
request_ack.success = true

report_miss_indication(miss_indication_data_type, request_id_type)
receive_acknowledgement(request_id_type, request_ack_type)
request_ack.success = true

report_miss_indication(miss_indication_data_type, request_id_type)
receive_error(request_id, error_reason)
request_ack.success = false
receive_acknowledgement(request_id_type, request_ack_type)
request_ack.success = true
This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "request reporting of miss indications" of the service 'Provide projectile position information'.

7.9.2.10.2 Engagement_Support
Parent Package: Radar_Services

7.9.2.17.10.2.1 Process_Target_Designation Parent Package: Engagement_Support
Parent Package: Engagement_Support

Process_Target_Designation_CMS
Type: Interface common_use_case_interface
Package: Process_Target_Designation

Fire control radars are designed to perform one target engagement at a time with respect to an air, surface or land target and provide the necessary information for a fire control solution regarding that target.

The CMS selects a track and requests the fire control radar to acquire and track the target behind that track. If the acquisition is successful the radar starts tracking the target and reporting fire control information.

Some fire control radars provide information about one or more other targets appearing in its field of view and may even provide associated sensor tracks. This is, however, not within the scope of this service interface but covered by "Provide sensor tracks".

The fire control information may be plots and/or tracks, depending on the product.
On receiving the de-designation request the fire control radar stops following the target and stops providing fire control information.

Phased array radars may include fire control capabilities as well. If they do, they provide a number of ‘virtual fire control radars’. To the extent that these virtual fire control radars are comparable in function and performance, there may be no need for the CMS to select a specific fire control channel to be used for a particular engagement.

In the case where the CMS loses or releases mastership of the subsystem, the subsystems ceases all fire control activities.

A target designation to a weapon with its own fire control capabilities may be done in an analogous way. In that sense, the service (interface) may also be employed by weapon systems.

Pre-condition: CMS must have Mastership.
Pre-condition: Technical state READY or ONLINE.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive_fire_control_channel_released()</td>
<td>Via this message, the subsystem confirms the release of a target acquisition.</td>
<td>request_id_type RequestID fire_control_channel_id_type FireControlChannelID</td>
</tr>
<tr>
<td>receive_target_acquired()</td>
<td>Via this message, the subsystem confirms the target acquisition.</td>
<td>request_id_type RequestID sensor_track_id_type TrackID fire_control_channel_id_type FireControlChannelID</td>
</tr>
<tr>
<td>receive_target_dedesignation()</td>
<td>Via this message, the subsystem reports the de-designation of a target</td>
<td>request_id_type RequestID sensor_track_id_type TrackID</td>
</tr>
</tbody>
</table>

Process_Target_Designation_Sub
Type: Interface
Package: Process_Target_Designation

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>dedesignate_target()</td>
<td>The subsystem is requested to de-designate a fire control channel.</td>
<td>request_id_type RequestID fire_control_channel_id_type FireControlChannelID</td>
</tr>
<tr>
<td>designate_target_by_position()</td>
<td>The subsystem is requested to designate a fire control channel based on a position/kinematics.</td>
<td>request_id_type RequestID kinematics_type PositionVelocity</td>
</tr>
<tr>
<td>designate_target_by_track()</td>
<td>The subsystem is requested to designate a fire control channel based on a track.</td>
<td>request_id_type RequestID sensor_track_id_type TrackID</td>
</tr>
</tbody>
</table>
Sensor Track Reporting
Type: InteractionOccurrence
Package: Process_Target_Designation
The sensor track reporting itself is not covered in this service interface. See the corresponding service interface 'Sensor Track Reporting'.
Sensor Track Reporting
Type: InteractionOccurrence

Package: Process_Target_Designation

The sensor track reporting itself is not covered in this service interface. See the corresponding service interface 'Sensor Track Reporting'.
This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "designate (target) by track" of the service "Process Target Designation".

Figure 7.221 Process Target Designation - Designation by track (Interaction diagram)
This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "designate (target) by position" of the service "Process Target Designation".
This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.

This sequence diagram applies to a fire control channel that has been previously designated by another path.
Figure 7.484.223 Process Target Designation - De-designation (Interaction diagram)

This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "de-designate (target)" of the service "Process Target Designation". It applies to a fire control channel that has been designated by position or by track.

7.9.2.27.10.2 Support_Kill_Assessment

Parent Package: Engagement_Support

Support_Kill_Assessment_CMS
Type: Interface common_use_case_interface
Package: Support_Kill_Assessment

With this service the subsystem provides kill assessment information to the CMS. The information relates to an above water engagement primarily against an air target.

The kill assessment report of the subsystem may be one of the three:
- **PROBABLE-KILL.** This indicates that the subsystem assumes the target to be killed.
- **PROBABLE-MISS.** This indicates that the subsystem assumes the target to be missed by the used weapon system.
- **NO-RESULT.** This indicates that the subsystem was not able to determine a valid result for this request.

See also service (interface) "Process Target Designation".

Pre-condition: Service "Process Target Designation" successfully carried out. **Pre-condition:** CMS must have Mastership.
Pre-condition: CMS must have Mastership.

### Table 7.242374 - Methods of Interface Support_Kill_Assessment_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_kill_assessment_result()</td>
<td>Via this message, the subsystem reports the kill assessment to the CMS.</td>
<td>request_id_type_RequestID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>kill_assessment_result_type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>KillAssessmentReport</td>
</tr>
</tbody>
</table>

**Support_Kill_Assessment_Sub**
- **Type:** Interface
- **Package:** Support_Kill_Assessment

### Table 7.243375 - Methods of Interface Support_Kill_Assessment_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_kill_assessment()</td>
<td>The subsystem is requested to evaluate and report a kill assessment.</td>
<td>request_id_type_RequestID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>expected_hit_data_type_kill_assessment_report</td>
</tr>
<tr>
<td></td>
<td></td>
<td>KillAssessmentData</td>
</tr>
</tbody>
</table>

**alt request kill assessment support**

**[loop kill assessment update]**

**[request kill assessment support]**

**[receive acknowledgement]**

**[processing errors]**

**[processing errors]**

**[report kill assessment result]**

[Diagram: Support_Kill_Assessment_CMS - Support_Kill_Assessment_Sub]
Figure 7.185.224 Basic Flow - Support Kill Assessment - Request Kill Assessment Support (Interaction diagram)

This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "request kill assessment support " of the service "Support Kill Assessment".

7.9.2.37.10.2.3 Support_Surface_Target_Engagement
Support_SURFACE_TARGET_ENGAGEMENT_CMS

Type: Interface common_use_case_interface

Package: Support_SURFACE_TARGET_ENGAGEMENT

This service is intended for fire control radars, as well as surveillance radar systems that have facilities to perform surface target engagements by means of dedicated fire control channels. These fire control channels may need a differently parameterized or more elaborate track algorithm, and they may be combined with related splash spotting video.

The CMS requests the surface track to be engaged. The maximum number of tracks that may be engaged simultaneously is determined by the radar.

The functionality may also be available for land targets, provided they may be tracked by the radar.

In the case where the CMS loses or releases mastership of the subsystem, a change of the availability of fire control channels shall be indicated to the CMS. Fire control radars shall cease all fire control activities.

The set of operational modes that make fire control channels available, as well as the number of available channels shall be provided by means of service "Manage Subsystem Parameters".

Pre-condition: Technical state ONLINE.

Pre-condition: CMS must have Mastership.

Post-condition: Service ends with success - check availability: The CMS is informed about the availability of fire control channels.

Post-condition: Service ends with success - target designation: The radar provides a fire control track for the selected sensor track.

Post-condition: Service ends with success - reporting: The CMS receives regular updates of the fire control track.

Post-condition: Service ends with success - de-designation: The fire control channel is de-assigned and has become available.

Post-condition: Service ends with fail - target designation: The fire control channel is not assigned; no fire control track.

Post-condition: Service ends with fail - surface track is lost: The fire control channel is not assigned; the fire control track is terminated. The CMS is informed about the availability of fire control channel.

Post-condition: Service ends with fail - de-designation: The fire control channel is not assigned.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_availability_state_of_fire_control_channels()</td>
<td>Via this interface method, the number of available fire control channels are returned from the subsystem to the CMS. If no channel is available, the value '0' is returned.</td>
<td>request_id_type RequestID available_fire_control_channels_binary AvailableFireControlChannels</td>
</tr>
<tr>
<td>report_available_fire_control_channels()</td>
<td>Via this interface method, the number of available fire control channels are returned from the subsystem to the CMS.</td>
<td>request_id_type RequestID fire_control_channel_id_type FireControlChannelID</td>
</tr>
</tbody>
</table>
| report_selected_fire_control_channel() | Via this interface method, the selected fire control channel is returned from the subsystem to the CMS. | request_id_type RequestID
fire_control_channel_id_type FireControlChannelID
sensor_track_id_type SensorTrackId |
Table 7.445377 - Methods of Interface Support_Surface_Target_Engagement_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>dedesignate_fire_control_channel()</td>
<td>Request to the subsystem to dedesignate a fire control channel.</td>
<td>request_id_type RequestID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fire_control_channel_id_type FireControlChannelID</td>
</tr>
<tr>
<td>designate_fire_control_channel()</td>
<td>Request to the subsystem to designate a fire control channel.</td>
<td>request_id_type request_id track_id</td>
</tr>
<tr>
<td>request_availability_of_fire_control_channels()</td>
<td>Request to the subsystem to report the available fire control channels.</td>
<td>request_id_type RequestID</td>
</tr>
</tbody>
</table>

Sensor track reporting

Type: InteractionOccurrence
Package: Support_Surface_Target_Engagement

Support_Surface_Target_Engagement_CMS
Type: ActivityPartition
Package: Support_Surface_Target_Engagement

Support_Surface_Target_Engagement_Sub
Type: ActivityPartition
Package: Support_Surface_Target_Engagement

Sensor track reporting

Type: InteractionOccurrence
Package: Support_Surface_Target_Engagement
This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "check availability" of the service "Support surface target engagement".

Figure 7.186.225 Support surface target engagement - Check availability (Interaction diagram)
**A. Designate Fire Control Channel**

```plaintext
signMessage(request_id_type, sensor_track_id_type)
```

**B. Receive Acknowledgement**

```plaintext
receiveAck(request_id_type, request_ack_type)
```

**C. Report Selected Fire Control Channel**

```plaintext
reportFireControlChannel(request_id_type, fire_control_channel_id_type, sensor_track_id_type)
```

**D. Loop Report Fire Control Channel**

```plaintext
while fire_control_channel

reportFireControlChannel(request_id_type, fire_control_channel_id_type, sensor_track_id_type)
```

The reporting of fire control tracks is part of sensor track reporting.

When the reporting ends, the number of available fire control channels is reported.

**E. Request Acknowledgement**

- Success
  ```plaintext
  request_ack.success = true
  receiveAcknowledgement(request_id_type, request_ack_type)
  ```

- Error
  ```plaintext
  request_ack.success = false
  receiveAcknowledgement(request_id_type, request_ack_type)
  ```

**F. Error Processing**

- Alternate Flow: Processing error
  ```plaintext
  request_ack.success = false
  receiveAcknowledgement(request_id_type, request_ack_type)
  ```

- Alternate Flow: Invalid track
  ```plaintext
  request_ack.success = false
  receiveAcknowledgement(request_id_type, request_ack_type)
  ```

This message corresponds with the COMPLETE message.

Internally, the asynchronous reporting of the fire control channel has been triggered.

This message corresponds with the COMPLETE message.

The reporting of fire control tracks is part of sensor track reporting.

The reporting of fire control tracks is part of sensor track reporting.

The reporting of fire control tracks is part of sensor track reporting.

The reporting of fire control tracks is part of sensor track reporting.

This message corresponds with the COMPLETE message.

Internally, the asynchronous reporting of the fire control channel has been triggered.
This sequence diagram shows how the CMS and the subsystem operate with each other during the operation “designate fire control channel” of the service “Support surface target engagement”.

Figure 7.187.226 Support surface target engagement - Designate fire control channel (Interaction diagram)

This message corresponds with the COMPLETE message. Internally, the asynchronous reporting of the fire control channel has been triggered.

The reporting of fire control tracks is part of sensor track reporting. When the reporting ends, the number of available fire control channels is reported.
This sequence diagram shows how the CMS and the subsystem operate with each other during the operation “De-designate fire control channel” of the service “Support surface target engagement”.

**Figure 7.158.227 Support surface target engagement - Dedesignate fire control channel (Interaction diagram)**
7.9.3.10.3 Missile_Guidance
Parent Package: Radar_Services

7.9.3.10.3.1 Perform_Illumination
Parent Package: Missile_Guidance

Perform_Illumination_CMS
Type: Interface common_use_case_interface
Package: Perform_Illumination

This service covers the control of target illumination to support a semi-active homing missile engagement.

The actor is the Combat Management System.

The service is triggered by the illumination request of the actor. Typically, illumination takes place during a specific period within the engagement sequence.

The actor sends an illumination request to the radar.

On the requested start time, the radar starts illuminating the target with specified parameters.

During the illumination, the actor may provide updates of illumination parameters, e.g. to change the stop time.

The service ends at stop time of the illumination.

If the radar may not fulfill the illumination request, this is reported to the actor and the service stops.

If during the illumination a radar fault takes place that prevents execution of illumination (e.g. illumination
frequency not more available), the health state of the Missile Guidance service (of which this service is part) becomes DEGRADED (if the Missile Guidance service is still capable of performing uplinks and/or downlinks) or NOT AVAILABLE, and the service stops.

If the target track becomes lost during the illumination, the service stops.

Pre-condition: Sensor health state: The sensor and the Missile Guidance service are in the health state AVAILABLE or DEGRADED.

Pre-condition: Sensor parameters: The relevant sensor parameters (e.g. allowed frequencies, transmission sectors) are set.

1 The manner in which this is done is described in other services of the OARIS (“Manage frequency usage”, “Manage transmission sectors”, “Control emissions” and “Manage subsystem parameters”).

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>complete()</td>
<td>report_illumination_completed()</td>
<td>request_id_type_request_id</td>
</tr>
</tbody>
</table>

Perform_Illumination_Sub
Type: Interface
Package: Perform_Illumination

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_illumination()</td>
<td></td>
<td>request_id_type_request_id</td>
</tr>
<tr>
<td></td>
<td></td>
<td>illumination_request_type_request</td>
</tr>
<tr>
<td>provide_track()</td>
<td></td>
<td>system_track_type_track</td>
</tr>
</tbody>
</table>
Target to be illuminated

If subsystem is not tracking the target

For all missiles in engagement (if required)

request_illumination(request_id_type, illumination_request_type)

Same method is used when requesting illumination for the first time, as well as modifying the request later. In the latter case, a new request (with new request_id) shall be issued for the same target.

Alternative flow: request rejected

Alternative flow: processing failed

Although not shown in this sequence diagram, processing may also fail after one or more successful illuminations but before the end of the illumination period.

perform_track(system_track)
It is assumed at the moment of the illumination request, the kinematics of the sensor tracks for target and
own missile(s) is reported to the subsystem as required and available for the illumination request.

This may be achieved in two ways:

1. The CMS provides the kinematics periodically to the subsystem,
2. the subsystem itself is tracking the target and own missile(s).

If the pre-condition is not satisfied, the receive_ acknowledgment shall indicate that the request is not accepted.

When after some time the target and/or own missile tracks are no longer available, the subsystem shall send the receive_ error
message with an appropriate error_reason.
Perform_Illumination_CMS «idlInterface»

Perform_Illumination_Sub «idlInterface»

Same method is used when requesting illumination for the first time, as well as modifying the request later. In the latter case, a new request (with new request_id) shall be issued for the same target.

It is assumed that, at the moment of the illumination request, the kinematics of the sensor tracks for target and own_missile(s) as referred to by the illumination_request are available to the subsystem.

This may be achieved in two ways:
1. The CMS provides the kinematics periodically to the subsystem, or
2. The subsystem itself is tracking the target and own_missile(s).

If this pre-condition is not satisfied, the receive_acknowledgement shall indicate that the request is not accepted.

When after some time the target and/or missile tracks are no longer available, the subsystem shall send receive_error message with an appropriate error_reason.

For all missiles in engagement (if required)

Although not shown in this sequence diagram, processing may also fail after one or more successful illuminations but before the end of the illumination period.
Perform_Missile_Downlink

Parent Package: Missile_Guidance

Type: Interface common_use_case_interface

Package: Perform_Missile_Downlink

The service describes the reception and provision of missile downlink information to the CMS. Downlink consists of transmission of energy by the missile. The radar subsystem may track a missile based on these downlink transmissions (beacon track). Provision of the beacon track of the missile to the CMS is covered by service Provide sensor tracks. This service handles the situation where the downlink also has content. Generally, a sequence of downlinks is transmitted by the missile, on periodic basis or triggered by an uplink. However, the CMS (or a dedicated missile subsystem) is responsible for evaluating the downlinks in this sequence. The radar subsystem only receives downlinks and provides them to the CMS, and does not keep track of the sequence. In the special case where the downlink contains own missile kinematics, this data may also be used internally by the radar subsystem.

The actor is the Combat Management System.

Although the downlink may be evaluated by a missile subsystem (which is not part of the CMS), the downlink is assumed to be passed to that missile subsystem via the CMS.

The service is triggered by the downlink request of the actor. The actor sends a downlink request to the radar. During the request listening period, the radar listens to transmissions that are in accordance with the provided downlink parameters. The radar reports to the actor the occurrence of the downlink, including the (decoded) content of the downlink. The information provided by the missile may vary depending on the applied missile fire control principle, and lies outside the scope of the OARIS standard. The information within the downlink may be used internally by the radar. The service ends at the end of the listening period.

If the downlink transmission is interrupted, this is reported to the actor, and the service stops.

If the downlink contains kinematic information about the missile, the radar subsystem may use this information internally to improve the own missile track (provided service Provide sensor tracks or service Process target designation). It is also possible that the missile is tracked based on the fact that it transmits energy and not based on the contents of the downlink. This so-called beacon tracking is covered by service Provide sensor tracks.
condition: Sensor health state The sensor and the Missile Guidance service are in the health state
Pre-condition: Sensor health state: The sensor and the Missile Guidance service are in the health state AVAILABLE or DEGRADED.

Pre-condition: Sensor parameters: The relevant sensor parameters (e.g. allowed frequencies, transmission sectors) are set. (The manner in which this is done is described in other services of the specification; see “Manage frequency usage”, “Manage transmission sectors”, “Control emissions” and “Manage subsystem parameters”).

Pre-condition: Engagement phase: An engagement must be taking place.

Pre-condition: Missile downlink parameters: The parameters of the missile downlink transmission must be known to the radar. Note that this does not concern the content of the transmission, but rather the transmission characteristics (e.g. frequency).

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_downlink()</td>
<td></td>
<td>request_id_type, request_id,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>downlink_report_type,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the_downlink_info</td>
</tr>
<tr>
<td>complete()</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>provide_track()</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7.248390 - Methods of Interface Perform_Missile_Downlink_CMS

Perform_Missile_Downlink_Sub
Type: Interface
Package: Perform_Missile_Downlink

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_downlink()</td>
<td></td>
<td>request_id_type, request_id,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>downlink_request_type,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>request</td>
</tr>
<tr>
<td>provide_track()</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7.248391 - Methods of Interface Perform_Missile_Downlink_Sub
Open Architecture Radar Interface Standard (OARIS), v2.0

Although not shown in this sequence diagram, processing may also be interrupted at any time during the downlink process by an error condition. In this case, the interface should not be processed until the error condition is resolved.
The request downlink operation has not been identified in the service description.

The reasons for introducing it here are:
1. There are no provisions on p. 1995 to carry out the downlink downlink parameters generation.
2. The CMS is only interested in downlink information from own missiles in flight belonging to an active engagement.
3. Frequency and data related parameters (e.g. frequency) are engagement dependent.
The request_missile_downlink operation has not been identified in the service description. The reasons for introducing it here are:
1. There are no provisions (e.g., services) to satisfy the missile downlink parameters precondition.
2. The CMS is only interested in downlink information from own missiles in flight belonging to an active engagement.
3. Generally, the missile downlink parameters (e.g., frequency) are engagement dependent.

Downlink report may be periodic or aperiodic.
7.9.3.3 Perform_Missile_Uplink

Parent Package: Missile_Guidance

Perform_Missile_Uplink_CMS

Type: Interface common_use_case_interface

Package: Perform_Missile_Uplink

The service describes the execution of uplink of relevant information from the radar to the missile in flight during an engagement.

Generally, a sequence of uplinks (of various types) must be transmitted to a missile during an engagement. However, the CMS (or a dedicated missile subsystem) is responsible for planning and requesting the correct sequence of uplinks. The radar subsystem only transmits an uplink on request of the CMS. Therefore, this service starts with the request of a single uplink and ends when the radar subsystem has transmitted the uplink.

The actor is the Combat Management System. Although the uplink may be initiated by a missile subsystem (which is not part of the CMS), the uplink is assumed to be passed through the CMS to the radar subsystem.

The service is triggered by the uplink request of the actor.

The actor sends an uplink request to the radar.

At the requested time, the radar sends the uplink to the missile in accordance with the provided uplink parameters.

The information provided to the missile may vary depending on the applied missile fire control principle, and lies outside the scope of the OARIS standard.

The service ends when the radar has confirmed the transmission of the uplink.

If the radar may not fulfill the uplink request, this is reported to the actor and the service stops.

If during the uplink a radar fault takes place that prevents execution of the uplink (e.g. uplink frequency not more available), the health state of the Missile Guidance service (of which this service is part) becomes DEGRADED (if the Missile Guidance service is still capable of performing illumination and/or downlinks) or NOT AVAILABLE, and the service stops.

If the missile track becomes lost during the uplink, the service stops.

Network Centric engagements

In Network-Centric or Network-Enabled systems, guidance of the missile may be transferred during the flight of the missile to another surface platform. As the related technologies are still being developed, it shall be too early to include specific NEC requirements here. However, care should be taken in the design of OARIS that such capabilities could be included at a later date. This means that there should be no built-in restrictions in the standard, which would prevent addition of such facilities in the future.

Relationship to missile downlink

For some missile types an uplink transmission may trigger the transmission of a downlink by the missile (e.g. an acknowledge of receipt). This relation depends heavily on the missile type and lies outside the scope of the OARIS standard.

Pre-condition: Sensor health state: The sensor and the Missile Guidance service are in the health state AVAILABLE or DEGRADED.

Pre-condition: Sensor parameters: The relevant sensor parameters (e.g. allowed frequencies, transmission sectors) are set1.

1. ---
The manner in which this is done is described in other services of the OARIS (*Manage frequency*)
usage”, “Manage transmission sectors”, “Control emissions” and “Manage subsystem parameters”).

Pre-condition: Engagement phase: An engagement must be taking place.
Pre-condition: Known position of missile: The position of the missile must be known, i.e. own missile track must exist. The missile track may be provided by the CMS or by the radar subsystem itself.

Table 7.250382 - Methods of Interface Perform_Missile_Uplink_CMS

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>report_uplink_completed()</td>
<td></td>
<td>request_id_type request_id uplink_report_type report</td>
</tr>
</tbody>
</table>

Perform_Missile_Uplink_Sub  
Type: Interface  
Package: Perform_Missile_Uplink

Table 7.254383 - Methods of Interface Perform_Missile_Uplink_Sub

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_uplink()</td>
<td></td>
<td>request_id_type request_id uplink_request_type request</td>
</tr>
<tr>
<td>provide_track()</td>
<td></td>
<td>system_track_type track</td>
</tr>
</tbody>
</table>
Open Architecture Radar Interface Standard (OARIS), v2.0

receive acknowledgment (request_id_type, request_ack_type)

request rejected

[alternative flow: request_uplink(request_id_type, uplink_request_type)]

 Missile to which the uplink shall be transmitted

provide track (system_track)

[alternative flow: processing failed]

receive acknowledgment (request_id_type, request_ack_type)

receive error (request_id_type, error_reason_type)

request_ack.accepted = true

request_ack.accepted = false
7.9.4.10.4.1 Perform_Splash_Spotting

**Parent Package:** Surface_Engagement_Support

**Type:** Interface common_use_case

**Package:** Perform_Splash_Spotting

Surveillance radar systems may support engagements against surface targets by means of a splash spotting video or measured splash positions. In the vicinity of the target a signal processing is applied which is optimized to observe splashes of the shells hitting the sea surface.
The splash spotting information may be used to achieve shot corrections for a running engagement. The engagement may use a fire control channel of the radar but also of another device like fire control radar. The CMS requests the radar to localize a splash spotting area at a defined position derived from the target kinematics.
The use of splash spotting areas may be limited to fire control channels of the radar. Then, only the localization of a splash spotting area may be done in accordance with this service. Normally, it shall be localized at the predicted hitting point.

These splash spotting areas shall not differ in terms of function and performance so that the selection of the area to be applied to an engagement may be done by the radar, automatically. The CMS just indicates where to localize it.

If mastership is lost during execution in any of the flows the services are terminated. Pre-condition: Technical state ONLINE.

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>confirm_reposition_splash_splotting_area()</td>
<td>Via this method, the request for the repositioning of a splash spotting area is confirmed by the subsystem.</td>
<td>request_id_type RequestID splash_splotting_area_id_type SplashSpottingAreaID</td>
</tr>
<tr>
<td>confirm_splash_splotting_area_deactivation()</td>
<td>Via this method, the request for the deactivation of a splash spotting area is confirmed by the subsystem.</td>
<td>request_id_type RequestID splash_splotting_area_id_type SplashSpottingAreaID</td>
</tr>
<tr>
<td>receive_splash_splotting_area_position()</td>
<td>Via this method, the request for a new splash spotting area based on a position is confirmed by the subsystem.</td>
<td>request_id_type RequestID splash_splotting_area_id_type SplashSpottingAreaID</td>
</tr>
<tr>
<td>receive_splash_splotting_area_track('x', 'y')</td>
<td>Via this method, the request for a new splash spotting area based on a track is confirmed by the subsystem.</td>
<td>request_id_type RequestID splash_splotting_area_id_type SplashSpottingAreaID</td>
</tr>
<tr>
<td>report_splash_splotting_area_activation_state()</td>
<td>Via this interface, the splash spotting areas are reported to the CMS.</td>
<td>request_id_type RequestID splash_splotting_area_set_type SplashSpottingAreaSet</td>
</tr>
</tbody>
</table>

Perform_Splash_Spotting_Sub
Type: Interface
Package: Perform_Splash_Spotting

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 7.252384</td>
<td>Methods of Interface Perform_Splash_Spotting_CMS</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Notes</td>
<td>Parameters</td>
</tr>
<tr>
<td>confirm_reposition_splash_splotting_area()</td>
<td>Via this method, the request for the repositioning of a splash spotting area is confirmed by the subsystem.</td>
<td>request_id_type RequestID splash_splotting_area_id_type SplashSpottingAreaID</td>
</tr>
<tr>
<td>confirm_splash_splotting_area_deactivation()</td>
<td>Via this method, the request for the deactivation of a splash spotting area is confirmed by the subsystem.</td>
<td>request_id_type RequestID splash_splotting_area_id_type SplashSpottingAreaID</td>
</tr>
<tr>
<td>receive_splash_splotting_area_position()</td>
<td>Via this method, the request for a new splash spotting area based on a position is confirmed by the subsystem.</td>
<td>request_id_type RequestID splash_splotting_area_id_type SplashSpottingAreaID</td>
</tr>
<tr>
<td>receive_splash_splotting_area_track('x', 'y')</td>
<td>Via this method, the request for a new splash spotting area based on a track is confirmed by the subsystem.</td>
<td>request_id_type RequestID splash_splotting_area_id_type SplashSpottingAreaID</td>
</tr>
<tr>
<td>report_splash_splotting_area_activation_state()</td>
<td>Via this interface, the splash spotting areas are reported to the CMS.</td>
<td>request_id_type RequestID splash_splotting_area_set_type SplashSpottingAreaSet</td>
</tr>
<tr>
<td>Method</td>
<td>Notes</td>
<td>Parameters</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>activate_splash_spotting_area_by_position()</td>
<td>Requests the subsystem to activate a new splash spotting area based on a position.</td>
<td>request_id_type RequestID splash_sighting_area_position_type SplashSpottingAreaPosition</td>
</tr>
<tr>
<td>activate_splash_spotting_area_by_track()</td>
<td>Requests the subsystem to activate a new splash spotting area based on a sensor track.</td>
<td>request_id_type RequestID sensor_track_id_type TrackID SplashSpottingAreaID</td>
</tr>
<tr>
<td>deactivate_splash_spotting_area()</td>
<td>Requests the subsystem to deactivate a splash spotting area.</td>
<td>request_id_type RequestID splash_sighting_area_id_type SplashSpottingAreaID</td>
</tr>
<tr>
<td>report_splash_sighting_information()</td>
<td>Requests the subsystem to report splash spotting information/splash positions for an existing splash spotting area.</td>
<td>request_id_type RequestID splash_sighting_area_id_type SplashSpottingAreaID</td>
</tr>
<tr>
<td>activate_reposition_splash_sighting_area_by_track()</td>
<td>Requests the subsystem to activate a new existing splash spotting area based on a sensor track.</td>
<td>request_id_type RequestID sensor_track_id_type TrackID SplashSpottingAreaID splash_sighting_area_id_type SplashSpottingAreaID splash_sighting_area_position_type SplashSpottingAreaPosition</td>
</tr>
<tr>
<td>request_splash_sighting_areas()</td>
<td>Request the subsystem to report the splash spotting areas to the CMS.</td>
<td>request_id_type RequestID SplashSpottingAreaID</td>
</tr>
<tr>
<td>request_splash_sighting_positions()</td>
<td>Request the subsystem to report the splash spotting areas to the CMS.</td>
<td>request_id_type RequestID SplashSpottingAreaID</td>
</tr>
</tbody>
</table>

Perform_Splash_Spotting_CMS
Type: ActivityPartition
Package: Perform_Splash_Spotting

Perform_Splash_Spotting_Sub
Type: ActivityPartition
Package: Perform_Splash_Spotting

Open Architecture Radar Interface Standard (OARIS), v2.0
Report measured splash positions

Type: InteractionOccurrence

Package: Perform_Splash_Spotting

Figure 7.192.231 Perform Splash Spotting - Check Activation (Interaction diagram)

This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "check activation" of the service "Perform splash spotting".
Figure 7.193.232 Perform Splash Spotting - Activate Splash Spotting Area by Position (Interaction diagram)
This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "activate splash spotting area by position" of the service "Perform Splash Spotting".

```
reposition_splash_spotting_area(request_id_type, splash_spotting_area_id_type, splash_spotting_area_position_type)
```

- If the operation is successful:
  - `request_ack.success = true`
  - receive_acknowledgement(request_id_type, request_ack_type)
  - confirm_reposition_splash_spotting_area(request_id_type, splash_spotting_area_id_type)
  - receive_acknowledgement(request_id_type, request_ack_type)

- If an error occurs:
  - `request_ack.success = false`
  - receive_error(request_id, error_reason)
  - [alternate flow: invalid splash spotting area parameters]
  - receive_acknowledgement(request_id_type, request_ack_type)

[Diagram showing sequence of operations and interactions between CMS and subsystem]
Figure 7.194.233 Perform Splash Spotting - Re-position Splash Spotting Area (Interaction diagram)

This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "reposition splash spotting area" of the service "Perform splash spotting".
This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "activate splash spotting area by fire control track" of the service "Perform splash spotting".
This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "report on splash spotting information" of the service "Perform splash spotting".
This sequence diagram shows how the CMS and the subsystem operate with each other during the operation "deactivate splash spotting area" of the service "Perform splash spotting".
8. Platform-Specific Models

8.1 DDS Data Model PSM

The DDS Data Model PSM defines a set of IDL files for the Data Model packages defined by the PIM. Comments are added to the IDL files to reflect the mapping rules below.

The detailed rules for the MDA code generation from the Data Model PIM to the DDS PSM IDL are as follows:

• PIM attributes and compositions are mapped to IDL attributes;
• Optional PIM attributes are mapped to a union type with a single member present when the exists case attribute is true;
• Collections in the PIM are mapped to IDL sequences; a Length tag determines the sequence bounds;
• Specialization / Generalization PIM relationships are mapped to IDL unions. Additional data classes are introduced for non-abstract generalization classes that have attributes.

8.2 DDS Services PSM

The DDS Services PSM defines IDL files for each package defined in the Services PIM. For each method on each interface class an IDL struct for a DDS topic named for the method is generated; each parameter is mapped to an attribute of the IDL struct. Note that the PIM only defines parameters with an ‘in’ mode, there are no ‘return’ parameters defined and all methods have at least one parameter. Comments are generated to match the PIM notes and to include the version number of this standard in each file.

Additionally, the struct contains a subsystem_id key attribute of type subsystem_id_type. This indicates which subsystem published the data or is intended to read it as a subscriber.

Operations that require a response contain a request_id in the PIM that logically links request and response instances. In the DDS PSM, each request_id operation parameter is mapped to a keyed attribute of the DDS topic so that distinct request and response pairs can be retrieved from the DDS data space.

To robustly and efficiently ensure that the data exchanged between a particular subsystem and a CMS is recognized correctly, topic samples pertaining to a particular subsystem are published on the partition corresponding to the name used in the Subsystem Identification use case. Also, the CMS uses the receive_subsystem_identification data topic to allocate a subsystem_id to a subsystem; the subsystem sets the subsystem_id to zero for the receive_subsystem_identification data topic for which the CMS subscribes on the wildcard partition “*”. Subsequently, for data intended for all subsystems, the CMS publishes samples on partition “*” with a subsystem id of zero.

However, the Register Interest use case is mapped to the DDS DCPS Reader Listener interface and the Provide Subsystem Services use case is mapped to the DDS DCPS Data Reader and Data Writer interfaces, so there are no IDL files for these use cases.

8.3 GraphQL Data Model and Services PSM

The GraphQL PSM defines a set of schema definition language files, one for each Service interface defined by the PIM; each of these files represents a self-contained service and contains definition for the types represented. Comments are added to these files to reflect the mapping rules below.
The detailed rules for the MDA code generation from the Data Model PIM to theGraphQL are as follows:

- Enumerations are mapped to GraphQL enum.
PIM Classes with an 'idlStruct' stereotype are mapped to both a GraphQL object and input type;

Scalar idlTypedef stereotyped classes are inlined to primitive GraphQL types in the types that use them;

PIM attributes and compositions are mapped to GraphQL object and input attributes;

Non-optional PIM attributes are mandatory GraphQL attributes;

Collections in the PIM are mapped to GraphQL lists (which are unbounded);

Specialization / Generalization PIM relationships are mapped to GraphQL union and interface object types and an input type with optional attributes (and the same semantics). Additional data classes are introduced for non-abstract generalization classes that have attributes.

The GraphQL services derived from CMS interfaces allow the CMS to query and subscribe to operations invoked by a Subsystem, whilst Subsystem can invoke the interface by making mutations. Services derived from Subsystem interfaces allow the Subsystem to query and subscribe and the CMS to mutate. Each GraphQL service contains:

- A schema object declaring query, mutation and subscription attributes;
- A query (also used for subscription) with an argument list allowing filtering by subsystem and whether simulated;
- Mutation object types each returning lists of operations;
- A union type with choices for each operation on the interface;
- A options input type with optional attributes for each operation on the interface for mutations;
- An object type for each operation including a argument list containing each key in the operation types (as well as request_id if present) and an additional list of subsystem names returning a list of operations;
- An input type for each operation including an additional list of subsystem name;
- Sensor Assessment, Supplementary Measurement and Track Reporting operations also support additional arguments to filter by environment and area;