# SAPIENT Middleware
# Interface Control Document

11th May 2020

# Administration page

| Customer Information | |
|---|---|
| Project title | SAPIENT |
| Customer Organisation | Dstl |
| Customer contact | Paul Thomas (dstlsensors@dstl.gov.uk, SAPIENT Interface Management Panel, Chairman) |
| | Stuart Colley (dstlsensors@dstl.gov.uk, SAPIENT Interface Management Panel, Secretary) |



| Principal authors | |
|---|---|
| Dr G.F. Marshall | QinetiQ Ltd, Malvern Technology Centre, St Andrews Road, Malvern, WR14 3PS |
| D.A.A. Faulkner | QinetiQ Ltd, Malvern Technology Centre, St Andrews Road, Malvern, WR14 3PS |



| Release Authority | |
|---|---|
| Name | Paul Thomas |
| Post | Chairman, SAPIENT Interface Management Panel (SIMP) |
| Date of issue | See Record of Changes |

**Record of changes**

| Issue | Date | Detail of Changes |
|---|---|---|
| 1.0 | 10th Jan 2014 | First issue |
| 1.1 | 14th Jan 2014 | For distribution to SAPIENT project |
| 2.0 | 31st March 2014 | Second issue, providing further clarification |
| 3.0 | 7th November 2014 | Third Issue for Review |
| 3.1 | 18th December 2014 | Revised following Review |
| 4.0 | 22nd October 2015 | Minor revisions following September 2015 Demonstration |
| 4.1 | 27th November 2015 | Updated for wider release |
| 5.0-DRAFT | 29th August 2019 | Draft Rewrite to simplify and clarify ICD and move Taxonomy to Appendix |
| 5.0 | 11th May 2020 | Version 5 approved by SAPIENT Interface Management Panel (SIMP) |

# Abstract

This document is the Interface Control Document (ICD) for the SAPIENT project. It defines the interfaces between the High Level Decision Making Module (HLDMM), the Autonomous Sensor Modules (ASM) and the SAPIENT Middleware. It also defines how a Graphical User Interface (GUI) such as that provided by the Lead Systems Integrator for the initial SAPIENT system may ingest the output of the system. The purpose of this document is to enable individual module developers to build component modules compatible with the overall system.

For this issue of the ICD, the document has been re-organised and rewritten based on 5 years of experience of running SAPIENT and feedback from suppliers with the intention that it is clearer and easier to maintain by the SAPIENT Interface Management Panel (SIMP).

# List of contents

1 Introduction 10
1.1 The SAPIENT Project 10
1.2 Purpose of the document 10
1.3 Scope of the document 10
1.4 Document Structure 11
1.5 Intellectual Property Considerations 11
2 System Overview 12
2.1 Network Interface 12
2.1.1 SAPIENT XML Messages 13
2.1.2 Null Termination 13
2.1.3 Additional Data 13
2.2 System Clock 13
3 Architecture Design 14
3.1 Structural Design 14
3.2 Behavioural Design 15
4 Interface Description Overview 17
4.1 Message Descriptions 17
4.2 High level process walkthrough 17
4.2.1 Initialisation 17
4.2.2 Normal Operation 17
4.2.3 ASM Alerts during Normal Operations 18
4.2.4 Control (Sensor Task) messages 18
4.2.5 ASM Shutdown 18
4.2.6 Lost connection 18
4.3 Initialisation 18
4.3.1 Registration Message Description 18
4.3.2 Registration Message Protocol 19
4.3.3 Location Information 19
4.3.4 Sensor ID / Source ID 19
4.3.5 Sensor ID Allocation 19
4.3.6 Initial Status (Heartbeat) message 20
4.4 Ongoing messages from ASMs during normal operation 20
4.4.1 Status (Heartbeat) Messages 20
4.4.2 Detection Messages 20
4.4.2.1 Object ID 21
4.5 ASM Alert Messages during normal operation 21
4.5.1 Message Protocol 21
4.6 Control Messages 22
4.6.1 Sensor Task Message Protocol 22
4.6.2 Sensor Modes 23

| | | |
|---|---|---|
| 4.6.3 | Task Message ‘Control’ Field | 23 |
| 4.7 | Acknowledgement Messages | 23 |
| 4.8 | Alert Response Messages | 23 |
| 4.8.1 | Reject Responses | 23 |
| 5 | Guidance on using the xml schema | 24 |
| 5.1 | General Guidance | 24 |
| 5.2 | Location Information | 24 |
| 5.3 | Location XML fields | 25 |
| 5.3.1 | location | 25 |
| 5.3.2 | locationList | 25 |
| 5.3.3 | rangeBearing | 25 |
| 5.3.4 | rangeBearingCone | 26 |
| 5.4 | Time Information | 26 |
| 5.5 | Further Guidance on XML fields | 27 |
| 5.5.1 | Text Fields | 27 |
| 5.5.2 | Class and Behaviour Taxonomy | 27 |
| 5.6 | Managing Real-Time System Performance | 27 |
| 6 | Message Detailed Description - Initialisation | 28 |
| 6.1 | Registration Message | 28 |
| 6.2 | Registration – heartbeatDefinition | 28 |
| 6.2.1 | Registration - heartbeatInterval | 29 |
| 6.2.2 | Registration - locationType | 29 |
| 6.2.3 | Registration - heartbeatReport | 29 |
| 6.2.4 | Registration – heartbeatReport – Typical Values | 30 |
| 6.3 | Registration - modeDefinition | 31 |
| 6.3.1 | Registration - modeDefinition - settleTime | 31 |
| 6.3.2 | Registration – modeDefinition - maximumLatency | 31 |
| 6.3.3 | Registration – modeDefinition – scanType | 31 |
| 6.3.4 | Registration – modeDefinition – trackingType | 32 |
| 6.3.5 | Registration - modeDefinition – duration | 32 |
| 6.3.6 | Registration - modeDefinition – modeParameter | 32 |
| 6.4 | Registration – detectionDefinition | 32 |
| 6.4.1 | Registration – detectionDefinition – geometricError | 33 |
| 6.4.2 | Registration – detectionDefinition – detectionReport | 33 |
| 6.4.3 | Registration – detectionDefinition – detectionReport Typical Values | 33 |
| 6.4.4 | Registration – detectionDefinition – detectionPerformance | 34 |
| 6.4.5 | Registration – detectionDefinition – detectionClassDefinition | 35 |
| 6.4.6 | Registration – detectionDefinition - ClassPerformance | 35 |
| 6.4.7 | Registration – detectionDefinition – ClassDefinition | 35 |
| 6.4.8 | Registration – detectionDefinition - behaviourDefinition | 36 |
| 6.5 | Registration – taskDefinition | 37 |
| 6.5.1 | Registration – taskDefinition - regionDefinition | 37 |
| 6.5.2 | Registration - regionDefinition - regionType | 37 |

6.5.3 Registration – regionDefinition - classFilterDefinition 38
6.5.4 Registration – regionDefinition - filterParameter 38
6.5.5 Registration – regionDefinition - subClassFilterDefinition 38
6.5.6 Registration – regionDefinition - behaviourFilterDefinition 38
6.5.7 Registration – taskDefinition – command 38
6.5.8 Task Commands 39
6.6 Registration Acknowledgement Message 40
7 Message Detailed Description – Normal Operation 41
7.1 Status Report Message 41
7.1.1 Status Report – Mode 42
7.1.2 Status Report - Power 42
7.1.3 Status Report – Sensor Location 42
7.1.4 Status Report – Field Of View 42
7.1.5 Status Report – Coverage 42
7.1.6 Status Report – Obscuration 42
7.1.7 Status Report – Status Region 43
7.1.8 Status Report – Status 43
7.2 Detection Report Message 44
7.2.1 Detection Report - Class 45
7.2.2 Detection Report - Behaviour 45
7.2.3 Detection Report - Associated File 45
7.2.4 Detection Report - Track Info 45
7.2.5 Detection Report – Predicted Location 45
7.2.6 Detection Report - Object Info 46
7.3.1 Alert - Associated File 47
7.3.2 Alert - Associated Detection 47
7.4 SensorTask Message 47
7.4.1 Sensor Task - Task ID 48
7.4.2 Sensor Task – Control 48
7.5 SensorTask - Region 48
7.5.1 SensorTask – Region - Class Filter 49
7.5.2 SensorTask – Region - Behaviour Filter 49
7.6 SensorTask – Command 49
7.6.1 Request Commands 49
7.6.2 SensorTask – Command – detectionThreshold 50
7.6.3 SensorTask – Command – detectionReportRate 50
7.6.4 SensorTask – Command – mode 50
7.6.5 SensorTask – Command – Look At 50
7.6.6 SensorTask – Command – Look For 50
7.7 SensorTask Acknowledgement Message 51
7.8 Error Messages 51
8 Example Messages 52
8.1 Example Registration Message 52

8.2 Example Heartbeat Messages 55
8.2.1 Initial heartbeat message 55
8.2.2 Additional Obscuration Detected, Illumination Level changed 56
8.2.3 Status Unchanged 56
8.3 Example Detection Messages 56
8.3.1 Example Detection 56
8.3.2 Track Lost from View 57
8.3.3 Track Lost in View 57
8.4 Example Control and Acknowledgement Messages 57
8.4.1 Default Tasking 57
8.4.2 Acknowledgement Message for Default Task 58
8.4.3 Task 1 to over-ride Default tasking 58
8.4.4 Acknowledgement Message for Task 1 58
8.4.5 Stop Task 1 and return to default task 59
8.4.6 Acknowledgment of Stop Message 59
8.4.7 Set Detection Threshold High Command 59
8.4.8 Acknowledgement of Successful Threshold High Command 59
8.4.9 Acknowledgement of Rejected Threshold High Command 59
8.4.10 Request Reset Command 59
8.4.11 LookAt Command 60
8.4.12 Detection Report response to LookAt Command 60
8.4.13 Null Response to LookAt Command 60
9 HLDMM – ASM Database Interaction 61
9.1 ASM – HLDMM Message Protocol Summary 61
9.1.1 Initialisation 61
9.1.2 Ongoing Messages from ASMs during normal operation 61
9.1.3 Control Messages 61
9.2 Initialisation 61
9.2.1 Sensor_Registration Table 62
9.3 Heartbeat Message 62
9.3.1 Status_Report Table 62
9.3.2 Status_Report_Region Table 63
9.3.3 Status_Report_Range_Bearing Table 64
9.3.4 Status_Report_Messages Table 65
9.4 Detection Message 65
9.4.1 Detection_Report_Location Table 65
9.4.2 Detection_Report_Range_Bearing Table 66
9.4.3 Detection_Report_Track_Info Table 67
9.4.4 Detection_Report_Class Table 68
9.4.5 Detection_Report_Subclass Table 69
9.4.6 Detection_Report_Behaviour Table 69
9.4.7 Detection_Report_Assoc_File Table 70
9.4.8 Ground Truth Table 70

9.5 Control Message 71
9.5.1 Sensor_Task Table 71
9.5.2 Sensor_TaskAck Table 72
9.6 Alert Message 72
9.6.1 Alert Table 73
9.6.2 Alert_Location Table 73
9.6.3 Alert_Range_Bearing Table 74
9.6.4 Alert_Assoc_File Table 75
9.6.5 Alert_Assoc_Detection Table 75
10 Interface between HLDMM and GUI 77
10.1 HLDMM – GUI Database 77
10.1.1 Initialisation 77
10.1.2 Heartbeat Message 77
10.1.3 Detection Messages 78
10.1.4 Control Messages 78
10.1.5 Alert Messages 78
10.2 Tracks and Detection Points 78
10.3 Alerts 78
10.4 Sensor Field of View and Coverage 78
10.5 High-Level Tasking and Regions of Interest 78
10.6 Operator Response to Alerts 79
A XML Message Structure 80
A.1 Registration Message 80
A.2 Status Report Message 82
A.3 Detection Message 83
A.4 Task Message 84
A.5 Alert Message 85
A.6 Registration Acknowledgement Message 85
A.7 Task Acknowledgement Message 85
A.8 Alert Response Message 85
A.9 Error Message 85
B Cross Reference Table for XML fields 86
C SAPIENT Demonstration System Implementation -Specific Information 92
C.1 SAPIENT Demonstration System -Specific Architecture 92
C.1.1 Physical Connection 92
C.1.2 Firewalls and Network Security 92
C.1.3 Data Interface 92
C.1.4 File Share 93
C.1.5 Database 93
C.1.6 Sensor ID Allocation 93
C.1.7 Sensor Location 93
C.1.8 System Clock 93
C.1.9 Power Supplies 93

C.2 System Architecture 93
C.2.1 LSI Server 94
C.2.2 ASM Engineering PCs 94
C.2.3 HLDMM Server 94
C.2.4 GUI PC 95
C.3 Implementation Specific requirements 95
C.3.1 Sensor Locations 95
C.3.2 Sensor Coverage and Field Of View 95
C.4 ASM Database Configuration 95
C.4.1 Demonstration Configuration 95
C.4.2 Database Server Installation 95
C.4.3 Database Creation 95
C.4.4 HLDMM Database access 95
C.5 File Share 96
C.6 Cubica HLDMM Specific Constraints 96
C.6.1 Reporting Classes, Sub-classes and Behaviour 96
C.6.2 Status Report Messages 96
C.6.3 Bearings Only Detection Work around 96
D SAPIENT Taxonomy 97
D.1 Object Classification Taxonomy 97
D.1.1 Human Sub Class Types 97
D.1.2 Vehicle Sub Class Types 97
D.1.3 Air Vehicle Sub Class Types 97
D.1.4 RF Communications Sub Class Types 98
D.2 Object Behaviour Taxonomy 98
E Line of Bearing Sensors 99

# 1 Introduction

## 1.1 The SAPIENT Project

Sensing for Asset Protection with Integrated Electronic Networked Technology (SAPIENT) is a concept that combines modular autonomous sensing with fusion and sensor management. Since 2013 the SAPIENT project has developed the concept, standards and demonstrator systems that embody this concept. The key principles of SAPIENT are:

- Reduction in operator workload during monitoring activities;
- Improved autonomy for decision making (automatic identification of threat activities to reduce false alarm rates, allowing for autonomous context driven decision making);
- Improved autonomy for sensor management;
- Lower bandwidth requirements for network traffic;
- Sensor modularity – sensors of different types use the same interface for communication.

The SAPIENT project has investigated the following use cases:

- Protection of land-based high-value assets with defined borders;
- Situational Awareness over large areas;
- Counter Unmanned Aerial Systems (C-UAS);
- Situation Awareness in contested urban environments.

The SAPIENT concept considers a system of systems approach, consisting of multiple Autonomous Sensor Modules (ASM) that are all connected to a single High Level Decision Making Module (HLDMM).

## 1.2 Purpose of the document

This document describes the interface specification for the SAPIENT Middleware.

This Interface Control Document (ICD) has been up-issued following the successful demonstration of the SAPIENT Middleware.

This version of the SAPIENT ICD superceeds previous versions and is published on the official UK government (.gov.uk) website where a link to the test harness and middleware software can also be found.

## 1.3 Scope of the document

This document covers the interfaces between the Autonomous Sensor Modules (ASM), the High Level Decision Making Modules (HLDMM) and the SAPIENT Middleware. It also defines how a Graphical User Interface (GUI) such as that provided by the Lead Systems Integrator for the initial SAPIENT demonstration may ingest the output of the system.

The SAPIENT concept is intended to support a range of different sensing modalities, a significant range of levels of object detection, localisation and classification, and a range of interpretations of what “autonomy” means. In developing this ICD, a balance has therefore been struck between the specific needs of the SAPIENT Middleware demonstration system and a desire to provide a flexible and extensible interface that could cope with unknown future sensor and processing modules in potential future iterations of the SAPIENT Middleware system beyond the current instantiation.

## 1.4 Document Structure

This document is organised as follows:

Sections 1 to 3 outline the System Design.

Sections 4 and 5 give an overview of the SAPIENT Middleware messages and protocol.

Section 6 gives a detailed description of the messages during initialisation.

Section 7 gives a detailed description of the messages during normal operation

Section 8 gives some example messages.

Section 9 defines the interface between the HLDMM and the Middleware database. Section 10 defines the interface between the HLDMM and the GUI.

Annex A contains the blank XML messages to show the structure of the messages.

Annex B contains a list of all the XML data fields and their usage.

Annex C contains the implementation-specific information from the SAPIENT Phase 2 Demonstration September 2015.

Annex D contains the extendable Taxonomy of classes that will be supported by the SAPIENT Middleware. This has been placed as an Appendix so that it can be revised without amending the rest of the ICD.

Annex E provides advice on implementing a Line of Bearing (LoB) ASM.

## 1.5 Intellectual Property Considerations

This Interface Control Document is Crown-owned Copyright. It has been generated through collaboration with QinetiQ as Lead Systems Integrator (LSI), various ASM and HLDMM suppliers from across the SAPIENT phases (2013-2019), and the members of the SAPIENT Interface Management Panel (SIMP). Each of them has agreed that they will not assert any essential patent rights (those which are in practice essential for the use of any part of the standard, as opposed to merely representing one of multiple practical implementations thereof), nor any rights in information they have contributed towards the standard.

This does not prevent any supplier from developing Intellectual Property in implementations of technology that connects to this interface.

It is the nature of an interface specification that it needs to describe fully all of the possible interactions across the interface: the HLDMM supplier in particular needs to understand the types of information it could receive from each of the candidate ASMs. Care has therefore been taken to generalise the message specifications wherever possible, and to anonymise the details of what the sensors can provide, so that there is no traceability from this document to any individual sensors' capabilities.

This Document is released and licensed under the terms of the Open Government Licence (see front page). Any contributions or requested amendments must be provided to Dstl under the Creative Commons Zero (CC-0) licence. Any information not provided to Dstl with the CC-0 licence will be protected appropriately and will not be used or reflected in this document.

# 2 System Overview

SAPIENT is intended to work at the “information level” rather than continually streaming raw data. The SAPIENT Middleware system architecture is based around a central database. A number of data agents manage the data-flow into the database. Each ASM communicates with a data agent to populate the database with declarations and detections, including as much information as the sensor modules are capable of providing. NB This is not intended to include streaming raw data. All ingest into the database is carried out by the data agents to ensure the data is valid. The data agent may monitor and if necessary limit data bandwidth. The ASMs will communicate with the data agent using messages as described in sections 6 and 7.

The HLDMM performs higher-level fusion on the ASM messages and reasoning based on the data therein. The output of this reasoning is fused tracks and alerts for display on a GUI and tasking of ASMs to provide further information such as imagery of objects of interest.

Similar to the ASMs, the HLDMM communicates with an HLDMM data agent to pass its messages to the system. This populates the database with data describing fused tracks and alerts for passing to a GUI and tasks for ASMs. The task messages are then forwarded to the relevant ASM via the data agent connected to the ASM. A separate set of database tables is used for HLDMM messages.

To provide scalability it is envisaged that each SAPIENT database/HLDMM combination will only work with a local set of sensors. Where multiple sets of sensors are deployed a hierarchy of SAPIENT systems could be deployed.

The specifics of the interface to the GUI are not relevant to the ASM providers. A high-level view of the interface to the GUI is provided in section 10.

## 2.1 Network Interface

SAPIENT as a standard is intended to specify the content of the SAPIENT messages rather than constrain the means of communicating the messages between modules.

Currently, the different modules communicate using TCP socket connections in an architecture described in section 3. As a result, any standard Internet technology such as Ethernet or Wi-Fi that supports TCP may be used for the SAPIENT Middleware. The only constraint is that the network must allow ASM messages to be received by the HLDMM with sufficiently low latency to allow it to perform fusion and tasking in real-time. This latency constraint will vary by implementation but in current implementations is typically of the order of 1 second to allow timely autonomous tasking of other modules.

The SAPIENT Middleware does not place a constraint on the Network topology because this is likely to be deployment specific. For reference a network architecture used in one of the SAPIENT Middleware trials is included in Appendix C.1.

### 2.1.1 SAPIENT XML Messages

This ICD defines a set of SAPIENT XML messages for each module to populate the system, with its detections and declarations. Only valid SAPIENT XML messages should be sent over the SAPIENT connections. It is intended that the autonomous behaviour of the system should be encapsulated within these XML messages. Whilst the ICD describes the messages, XSD files will be used as the master definition of the messages.

The advantage of XML is that it is readily extendable, and also is easy to adapt for the varying levels of information provided by each ASM. No individual sensor will be expected to populate all the fields; however, every ASM will be expected to populate the minimal sub-set of these fields.

This ICD defines the following key messages:

- Registration
- Status Report
- Detection Report
- Alert
- Sensor Task

### 2.1.2 Null Termination

To improve the reliability of the system in the presence of communication errors, each XML message will be terminated by a null (0) character. This allows the system to reject incomplete or corrupted messages.

### 2.1.3 Additional Data

Additional data such as image snapshots may be needed to allow a system operator to respond appropriately to an alert generated by the SAPIENT Middleware system. This will be referenced by URLs in the SAPIENT Middleware XML messages. To provide persistent storage of this additional data, a file share or webserver may be required as discussed in Appendix C.5. This is likely to be implementation specific.

## 2.2 System Clock

All network devices shall synchronise to a central Network Time Protocol (NTP) Server on the SAPIENT Middleware network. All modules will be expected to synchronise with this at regular intervals during system operation.

# 3 Architecture Design

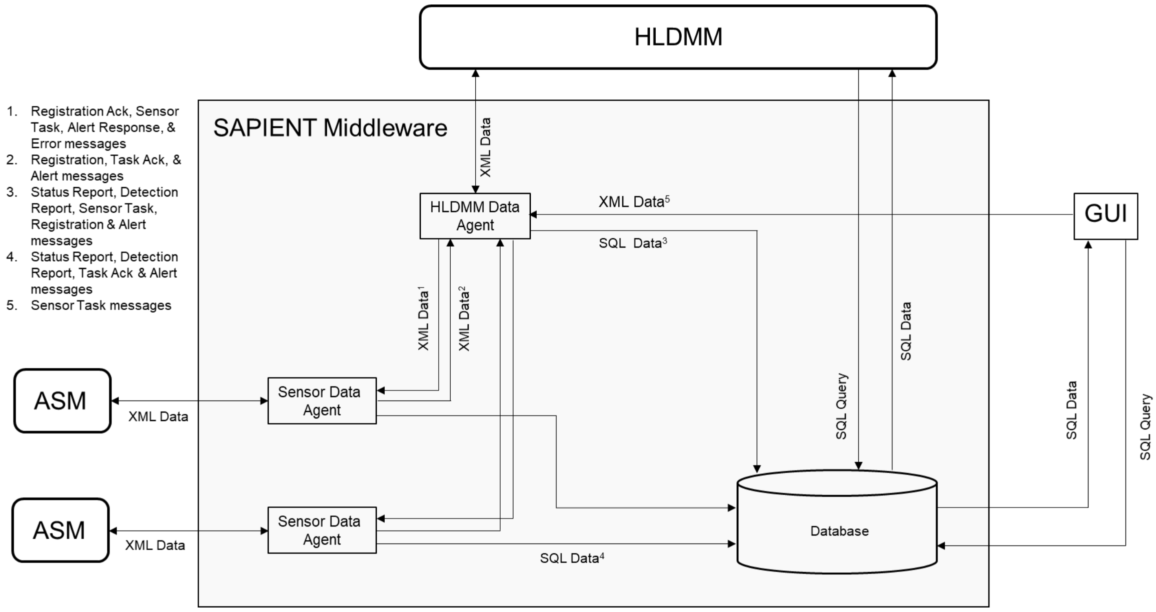


*Figure 1: SAPIENT Middleware Architecture illustrating data flow from ASMs via Middleware to HLDMM and GUI. The shaded area shows the extent of the SAPIENT Middleware. The interfaces that cross the boundary are defined in this ICD.*

## 3.1 Structural Design

Figure 1 shows the SAPIENT Middleware Architecture. For each SAPIENT Middleware system, a single HLDMM interacts with the Information from one or more ASMs. The SAPIENT Middleware, consisting of a number of Data Agents and a database provides a consistent method of routing and storing SAPIENT Middleware messages. A separate instance of the Data Agent is deployed for each instance of an ASM or HLDMM. This allows the system to be expanded or reconfigured without affecting existing modules.

The ASMs are decoupled from the database to allow the SAPIENT Middleware to control data flow and to allow future expansion of the database.

The HLDMM is allowed direct access to the database for read but not for write. This ensures that the responsibility for the integrity of the database lies with the SAPIENT Middleware.

The Graphical User Interface can read HLDMM and ASM data from the database but sends any messages via the Middleware.

If an ASM consists of a network of multiple sensors and/or processing nodes, the ASM supplier will be responsible for providing an appropriate bridge onto the SAPIENT Middleware system. Where a collection of neighbouring identical sensors may be considered as one sub-system, they can be considered as a single ASM. Where the sensors have different modalities or widely differing locations, it may be better to consider them as separate ASMs.

## 3.2 Behavioural Design

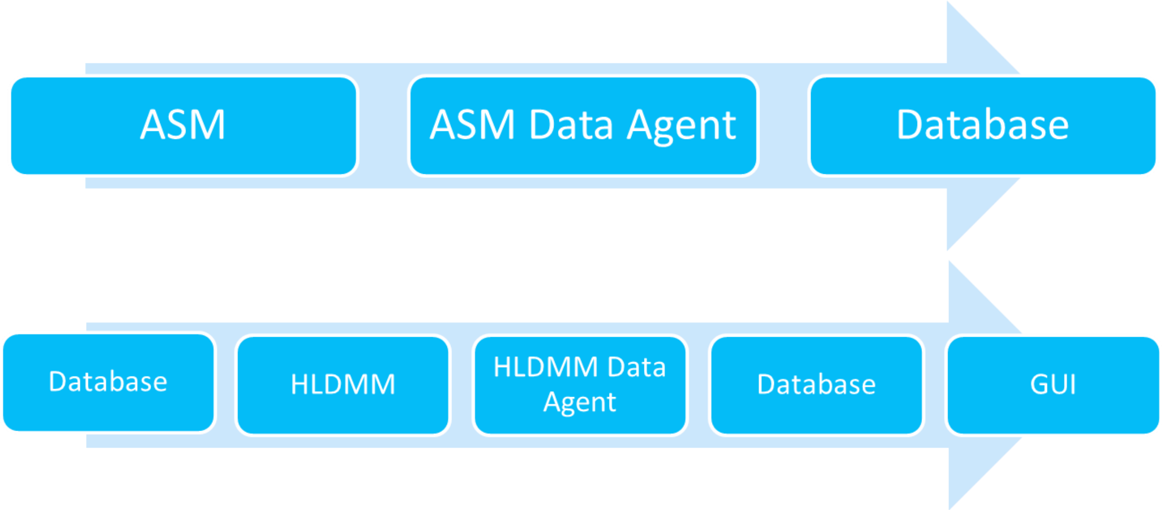


*Figure 2: SAPIENT Middleware Detection Data Flow*

Figure 1 and Figure 2 illustrate the data flow between the ASMs, the SAPIENT Middleware and the HLDMM in normal operation.

- Detection reports, Status Reports and Alerts are generated by the ASM and passed to the ASM Data Agent to be stored in the Database.
- The HLDMM directly queries the database and performs processing to determine whether to raise an alert to an operator and/or task an ASM.
- When the HLDMM generates an alert that it wishes to raise up to the operator, it passes an Alert message to the HLDMM Data Agent, which populates the database, which in turn is queried by the GUI.

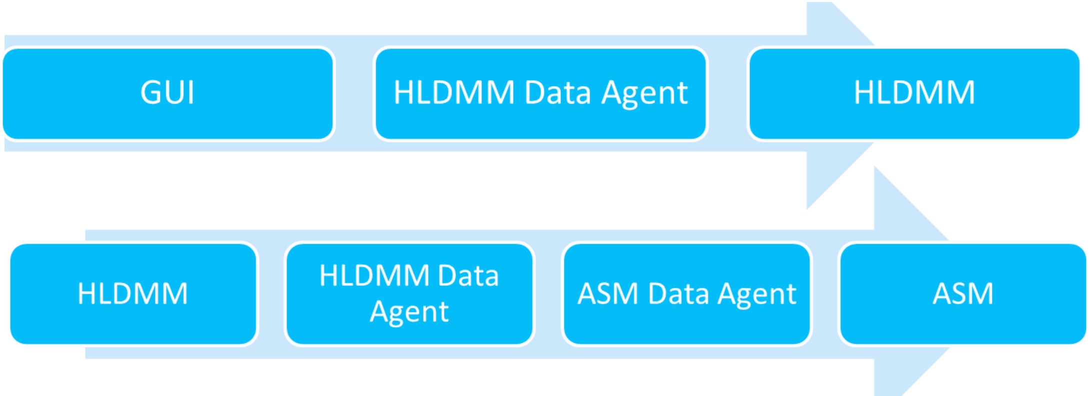


*Figure 3: SAPIENT Middleware Control (Tasking) Data flow.*

Figure 1 and Figure 3 illustrate the control flow 1) from the GUI to the HLDMM and 2) from the HLDMM to the ASMs.

Provision has been made for an operator to provide high level instructions from the GUI, via the SAPIENT Middleware, to inform the HLDMM of the “Commander’s Intent”, e.g. at mission start-up. This can also support manual interventions such as control of an effector where a man-in-the-loop is required.

Once set-up, in normal operation, most tasks and instructions will be generated autonomously by the HLDMM, and passed via the HLDMM Data Agent and the ASM Data Agent to the appropriate ASM.

# 4 Interface Description Overview

## 4.1 Message Descriptions

All interaction between the ASMs and the HLDMM will be via the SAPIENT Middleware, and consists of these activities:

1. Initialisation;
2. Ongoing messages during normal operation;
3. (Optional) ASM Alerts during normal operation
4. Control messages from the HLDMM to individual ASMs;

These will now be described in more detail.

## 4.2 High level process walkthrough

### 4.2.1 Initialisation

- ASM connects as a client to an ASM SAPIENT Data Agent (SDA) in the SAPIENT Middleware;
- ASM sends Registration message to HLDMM Data Agent (HDA) via the SDA;
- HDA sends Registration Acknowledgement message to the ASM via the SDA;
- ASM sends initial Status message to SDA;
- SDA stores messages in database;
- ASM begins event detection as per its default tasking (for those sensors which have defaults) ;

NB: If the HLDMM defines a task for an ASM, that task over-rides any ASM-generated default task previously defined.

### 4.2.2 Normal Operation

- ASM performs activity detection;
- ASM sends Detection messages (incorporating location and classification information on the detected entity) as per current tasking to SDA;
- ASM sends regular Status messages to SDA;
- ASM optionally sends Alert messages to SDA (see below);
- SDA stores messages in database;
- HLDMM queries the database, and reasons over/fuses the detections;
- HLDMM sends detection messages containing fused track points to HDA;
- HLDMM sends alert messages to HDA;
- HLDMM send task messages to ASMs via HDA and SDAs;
- HDA stores messages in database;

- GUI queries the database to show the output from the HLDMM and/or ASMs.

#### 4.2.3 ASM Alerts during Normal Operations

- If ASM wishes to inform the HLDMM that it is about to do something, the alert message can be used to inform or alert the HLDMM of this. An example of this might be a sensor deciding to do a detailed scan of an object or area.
- If the alert message merits a response from the HLDMM, the Alert Response message can used.
- If the HLDMM chooses to deny permission to carry out the action, the HLDMM will respond with an Alert Response message with 'status' field of value 'Reject'. The 'reason' field may be used to provide a strategy for retrying the action in future. (See section 4.8.1)

#### 4.2.4 Control (Sensor Task) messages

- HLDMM sends Sensor Task Message to ASM via the HDA and SDA connected to the ASM ;
- ASM sends acknowledgement of message to HLDMM via SDA and HDA – confirming acceptance or otherwise of tasking or command;
- ASM changes detection behaviour as per command;
- ASM returns to Normal Operation.

#### 4.2.5 ASM Shutdown

- ASM sends Status message with "goodbye" tag to show impending loss of communication from this ASM;
- ASM closes network connection to data agent.

#### 4.2.6 Lost connection

It is the HLDMM and ASM module's responsibility to monitor connection to the Middleware and attempt to reconnect if connection is lost. If a module is unable to connect to the Middleware, then it should attempt to retry every 10 seconds until it is successful.

If reconnection occurs over a short period of time (a few minutes), re-sending a Registration message may not be necessary. If a longer time has passed, the ASM should re-send a Registration message.

### 4.3 Initialisation

#### 4.3.1 Registration Message Description

As part of the system initialisation, each ASM must produce a Registration message stating what its capabilities are. This will include sensor type, supported detection modes, a list of what it is able to detect, track and/or classify, a declaration of error characteristics associated with the detections/tracks/classifications, a description of the contents of its Status (heartbeat) message (see below) and a list of how it is able to interact with the HLDMM – e.g. tasks it is able to accept. The message will also include the performance implications of different detection modes. Performance can be specified in terms of geometric (location) error, detection performance and classification performance.

The Registration messages will contain declarations of sensor capability. The detail of the Registration message is given in section 6.

### 4.3.2 Registration Message Protocol

The Registration message shall be sent by the ASM to the SAPIENT Data Agent (SDA) before any other messages. No other messages shall be sent until a Registration Acknowledgement message or Error message is received in response. If successful, the Registration message only needs to be passed once, when the ASM connects to the system.

The SAPIENT Data Agent (SDA) forwards the message to the HLDMM Data Agent (HDA). The HDA generates a Registration Acknowledgement message either confirming the ASM's existing sensor ID or if required allocating a new sensor ID. The HDA sends the acknowledgement to the SDA which forwards it on to the ASM.

If the ASM does not receive a response within 30 seconds of sending the Registration message, the ASM should close the existing socket connection, open a new socket connection to the SDA and send the message again.

The Registration message is stored in the database for use by the HLDMM and GUI.

### 4.3.3 Location Information

The Registration message is used to declare the real-world coordinates system that the ASM will use to provide location information to the SAPIENT Middleware system in all subsequent messages. Section 5.2 provides guidance on how location information should be provided.

### 4.3.4 Sensor ID / Source ID

Each module instance must use a single, unique module identifier number in all messages. The purpose of this identifier is to allow the middleware and HLDMM to identify the origin of messages. The Source ID is the ID of the source of the message, while the Sensor ID is the ID of the sensor the message refers to. In the case that a sensor is sending a message, these IDs will be the same.

Registration messages, control messages and their acknowledgements are specific to an ASM and so 'sensorID' is used in these messages. Heartbeat, detection and alert messages can be generated by ASMs or HLDMMs and so 'sourceID' is used in these messages. The same identifier number shall be used across all messages. High-Level Control messages from the GUI to the HLDMM shall use 'sensorID' value zero (0).

### 4.3.5 Sensor ID Allocation

One of the key steps in initialisation is allocating a Sensor ID to an ASM. The ICD allows for the Sensor ID identifier to be either pre-assigned or allocated during the initialisation phase by the Middleware. If an ASM has already been allocated a Sensor ID, the sensorID field should be populated with that value. If the field is omitted or left empty, the Middleware will allocate an ID to this ASM.

In all cases, the ASM must use the SensorID value returned in the Registration Acknowledgement message for all subsequent messages.

#### 4.3.6 Initial Status (Heartbeat) message

As part of system initialisation, an initial heartbeat message will be sent after the Registration message. This will indicate the initial state and optionally the physical location and field of view of the ASM. For ASMs providing detection messages in RangeBearing (Spherical) coordinates, a sensor location must be provided. For ASMs that provide detection messages in Cartesian coordinates, providing a sensor location is optional but can be useful for calibrating sensor location if the ASM does not have its own GPS or GPS accuracy is degraded.

### 4.4 Ongoing messages from ASMs during normal operation

During normal operation, ASMs provide Detection and Status messages. These follow the XML schema in the attached XSD files. The XML fields are described further in section 7. The messages are stored in the database by the SAPIENT Data Agent (SDA) for the HLDMM and GUI to read.

#### 4.4.1 Status (Heartbeat) Messages

These are regular messages that must be produced by all ASMs. This message will contain a time-stamp and the sensor's unique ID, and will provide the HLDMM with reassurance that the sensor is still operating correctly, even in the absence of any valid detections. In addition, this message can provide additional information on the ASM. This information could include for example, some of the following fields:

> Sensor location, field of view; coverage; sensor mode; obscuration polygon(s); rain detected; degraded sensor performance; remaining battery power; camera exposure time.

In addition to regular Status messages, one-off Status messages may be sent immediately on important changes. Examples of this might include the following:

> ASM tamper, ASM fatal error, ASM completely obscured

It is recommended that a regular Status message should be sent at least once every 10 seconds. They shall be sent at least once every 60 seconds and not more than once a second. If no Status or Detection messages are received by the Middleware within a 60 second period, the Data Agent will close the socket connection and wait for the ASM or HLDMM to re-establish communication.

At least the latest status information will be stored in the database. Storing some historical status information may be useful for diagnostic purposes, but to limit redundant data, this may be limited to where information has changed.

The Status message is described further in section 7.1

#### 4.4.2 Detection Messages

These messages are output by each ASM when they wish to declare a detection of an object or other result (e.g. classification) to the wider system. The message declaration contains the sensor's unique ID, a timestamp, an object unique ID number, the real-world location of the detected object (at least 2D location, some sensors will be able to produce 3D locations and/or location error ellipses), and as much additional information about the detected object as the ASM can provide (in line with the declaration made in the Registration message). This could include for example the detection confidence level, track information, and/or classification information (with confidence levels). A URL to additional data such as an image snap-shot or point-cloud may also be provided.

Sensors that can only provide a bearing to a detected object should provide a ‘dummy’ range value of half the maximum range of the sensor and a range error also of half the range of the sensor.

The format of the messages from all of the ASMs shall be consistent with the given xml schema, however the actual content of the messages will potentially be different for each ASM, depending on its capabilities. No individual sensor module will be expected to fill in all the possible fields – each ASM will provide as much information as it can, as described in its registration message, and the HLDMM will be expected to reason across it and fuse it appropriately.

The Detection message is described further in section 7.2

#### 4.4.2.1 Object ID

Object ID is an integer number generated by an ASM that uniquely identifies an object. If an ASM is not capable of associating multiple detections of the same object then the object ID generated should be unique to each detection message.

When an object remains static in the field of view for a long time such as a vehicle parking up, it is assumed that it may merge into the background in time. If detected as such, it should be reported as “lostInView”. If supported, it may then be reported as an obscuration.

## 4.5 ASM Alert Messages during normal operation

ASM generated Alert messages provide a means for an ASM to report something that needs a response from the HLDMM. An example might be that an ASM wishes to autonomously change mode, but would like permission from the HLDMM. Modes requiring permission should include modeParameter type="PermissionToChange" value="Required" in the registration message.

They can also be used to provide extra information directly from the ASM to the GUI. In this case no response would be expected.

Finally, if the system includes sensors that can only provide alerts, such as door alarms, these can be reported using Alert Messages. Where a response is required an Alert Response message is used.

HLDMM generated Alert messages provide the means for the HLDMM to present alerts to the operator via the GUI and may allow the operator to provide a response to the system.

The Alert message is described further in section 7.3

### 4.5.1 Message Protocol

As per other messages from ASMs, Alert messages are stored in the database by the SAPIENT Data Agent (SDA) for the HLDMM to read.

Alert messages may also be additionally forwarded to the HLDMM via the HDA to allow the HLDMM to provide an immediate response, such as when an ASM wishes to autonomously change mode, but would like permission from the HLDMM.

## 4.6 Control Messages

Control (Sensor Task) messages are sent by the HLDMM via the Middleware to task individual ASMs. The overall philosophy used is that Sensor Task messages contain two sections

1. A Region Definition related to the task
2. A command or action related to the task

Command based tasks tend to be short lived or instantaneous, whereas Region based tasks tend to persist until another task is received. The particular message information will be very much dependent on the capabilities of the individual ASM, as described in the Registration message. However, the general form will be request ID, sensor ID, and a tasking request, for example

- In Region X look for threats of Type Y;
- Ignore all detections in Region Z;
- Look at Location (X,Y)
- Request an image snapshot
- Another sensor has detected something at location (X, Y), can you confirm the detection, track the object and classify the activity. This is effectively a cross-cue from one sensor to another;
- Your false alarm rate is unacceptably high (or your P (d) is unacceptably low) – adjust your thresholds accordingly.

Command based tasks such as ‘Request Snapshot’ do not require the region section to be populated. Similarly if a Region of Interest is defined for an ASM in the Region section of the message, the command section does not need to be populated.

The Sensor Task message is described further in 7.4, the region section in 7.5 and the command section in 7.6

### 4.6.1 Sensor Task Message Protocol

The HLDMM sends Sensor Task Messages to ASMs via the HDA and SDA connected to the relevant ASM. The HDA routes the message to the correct SDA.

The ASM responds to the Sensor Task message with one or more SensorTaskACK acknowledgement messages during the lifetime of the task. The acknowledgement message states whether the ASM accepts or rejects the task and may include any additional data such as a URL to an image snapshot. If an ASM cannot comply with a control message, it sends a SensorTaskACK acknowledgement message with a description as to why in the ‘reason’ field of the message.

The SDA and HDA forward the Acknowledgement message back to the HLDMM. The SDA stores the response in the database.

When a task completes or is stopped by a control message, the ASM will revert to any previously defined task (such as the default one). If an ASM does not have a default task it will stop detecting until it receives a new task.

#### 4.6.2 Sensor Modes

An ASM will operate in one or more sensor modes. A sensor mode is defined as a distinct way of operating such that the method of operation, the detection and error characteristics as reported in the initialisation message remain consistent. Typically a non-steerable sensor will operate in a single sensor mode. A steerable sensor could have multiple modes that define how it will respond to tasking from the HLDMM. E.g. a PTZ camera could have one mode where it continually scans an area and another where it lingers at a number of fixed locations. The Registration message provides the means to communicate the characteristics of the mode. The Sensor Task message provides the means to tell the ASM to change its mode.

#### 4.6.3 Task Message ‘Control’ Field

The ‘control’ field in the Sensor Task message defines what action to perform on that task, (see section 7.4.2).

### 4.7 Acknowledgement Messages

Acknowledgement Messages are used to complete the handshake between the components of the system. These are only used in response to specific messages. There are three types of ACK message:

- Registration Acknowledgement from the HLDMM to the ASM that registration has occurred (see section 6.6);
- Task Acknowledgement from the ASM that it has received a sensor tasking message from the HLDMM (see section 7.7);
- LookAt Null Acknowledgement. The ASM provides a ‘Nothing found’ response to a LookAt message – i.e. the ASM had a look at the required coordinates, but was unable to detect an object there (see section 7.6.5, and sections 8.4.11 to 8.4.13 showing an example of how to use this message).

### 4.8 Alert Response Messages

Alert Response Messages are used where a response is required to an Alert Message. An example of this might be a sensor deciding to do a detailed scan of an object or area and it wants to ask permission from the HLDMM first. The HLDMM would send an Alert Response message with ‘Acknowledge’ or ‘Reject’ status field. Modes requiring permission should include modeParameter type="PermissionToChange" value="Required" in the registration message.

#### 4.8.1 Reject Responses

Where an Alert Response Message has ‘Reject’ in the status field, the reason field can be used to allow the HLDMM to inform the ASM when it can try to send this request again. An empty reason field implies that this request should not be sent again. To allow the ASM to send this request again use ‘Retry:’ in the reason field, followed by the time interval and time units.

e.g. ‘Retry:10, seconds’

# 5 Guidance on using the xml schema

## 5.1 General Guidance

The SAPIENT Middleware system potentially contains a wide range of sensor types and capabilities. The XML Schema has been written in such a way that it should be fit for purpose not only for the modules developed for the SAPIENT Demonstration System outlined in C.1 but also for developing interfaces to other current and future sensors which could be subsequently developed and integrated. It is recognised that no one sensor will be expected to fill in all of the fields. Additional guidance is given in sections 6, 7 and Appendix A as to which fields are compulsory and which are optional.

## 5.2 Location Information

To support a variety of sensor types, the SAPIENT Middleware system will accept location information in a number of different real-world coordinates systems. Currently GPS and UTM coordinate systems are supported for global locations (also referred to Cartesian locations). The system will also support detections and status information in Spherical coordinates (range/bearing/elevation) relative to a known sensor location. Providing object locations in any of these formats is acceptable, depending on what the native format is of the individual ASM.

To keep this flexibility bounded, this will be limited to Cartesian (x, y, z coordinates, together with a corresponding reference frame, either UTM or GPS) or Spherical (Range-bearing-elevation). This could be easily extended with additional coordinates systems in future if required. Note that bearing-only coordinates are regarded as a sub-set of Spherical coordinates.

There are a number of different types of locations provided in the XML messages; (i.e. detections, sensor location, sensor field of view, areas of obscuration, regions). Table 1 provides guidance on which coordinates system to use for the different types of location. Where supported by an ASM, the coordinates system used shall be declared in the Registration message. This coordinate system shall be used consistently for all messages that contain location data of that type. For Cartesian coordinates, the ASM shall use the same reference frame (i.e. GPS or UTM) for all location types.

*Table 1: Guidance on Coordinates Systems for different types of location information*

| **Location Type** | **Coordinates System** |
|---|---|
| Detections | Any |
| Sensor Location | Cartesian |
| Sensor Field of View and Coverage | Any (Range-Bearing may be preferable for steerable sensors) |
| Areas of obscuration | Any |
| Regions | Any – Cartesian preferred because regions may be independent of sensors |

Any object sizes, speeds etc. reported shall also be given in real-world units. Internal frames of reference (e.g. camera-centric, measured in terms of pixels) should be calibrated and translated into real-world values (with associated errors if appropriate) before being reported.

The HLDMM shall task the individual ASMs using the chosen coordinate system of the ASM as specified by the TaskDefinition section of the ASM Registration message.

All azimuth bearings shall be reported relative to north, Grid North should be used by default. If the HLDMM support Magnetic or True North and one of these is the preferred form for the ASM, this can be specified in the ASM's Registration message.

## 5.3 Location XML fields

Location information is provided by one of the following XML fields:

### 5.3.1 location

```
<location>
<X/> <Y/> <Z/> <eX/> <eY/> <eZ/>
</location>
```

The location element defines a single point in space in Cartesian form as follows:

- X is Eastings in metres up to 2 decimal places.
- Y is Northings in metres up to 2 decimal places.
- Z is Altitude of object in metres up to 2 decimal places - if omitted then object on ground-plane will be assumed. Altitude may be calculated by summing the relative value to the sensor and the known sensor altitude so that terrain data is not required.
- eX is x error of location in metres up to two decimal places
- eY is y error of location in metres up to two decimal places
- eZ is z error of location in metres up to two decimal places

### 5.3.2 locationList

This is a list of location elements that define the boundary polygon of a region or sensor field of view

### 5.3.3 rangeBearing

```
<rangeBearing>
<R/> <Az/> <Ele/> <Z/>
<eR/> <eAz/> <eEle/> <eZ/>
</rangeBearing>
```

The rangeBearing element defines a single point in space in Spherical form as follows:

- R, (Range) is Distance from sensor to object in metres up to 2 decimal places.
- Az, (Azimuth) is Angle of detection relative to North - based on HLDMM wide concept of north. This is typically in degrees.
- Ele, (Elevation) is Angle above/below horizontal in degrees up to 2 decimal places - if omitted then object will be assumed to be on the ground-plane.
- Z is Altitude of object in metres up to 2 decimal places - if omitted then object on ground-plane will be assumed. Altitude may be calculated by summing the relative value to the sensor and the known sensor altitude so that terrain data is not required. It is not expected that this and Ele will both be populated.
- eR is range error of location in metres up to two decimal places.
- eAz is azimuth error of location in degrees up to two decimal places.
- eEle is elevation error in degrees up to two decimal places.
- eZ is altitude error of location in metres up to two decimal places

Further advice on the practicalities of reporting observations from line of bearing sensors can be found in Appendix E.

5.3.4 rangeBearingCone

```
<rangeBearingCone>
<R/> <Az/> <Ele/>
<hExtent/><vExtent/>
<eR/> <eAz/> <eEle/>
<ehExtent/> <evExtent/>
</rangeBearingCone>
```

The rangeBearingCone element defines a Cone or field of view in Spherical form as follows:

- R, (Radius) is Distance from sensor to edge of region in metres up to 2 decimal places.
- Az, (Azimuth) is Horizontal Angle of centre or boresight of region relative to North - based on system wide concept of north. This is typically in degrees.
- Ele, (Elevation) is Angle of centre or boresight of region above/below horizontal in degrees up to 2 decimal
- hExtent is Horizontal extent angle of region in degrees up to 2 decimal places.
- vExtent is Vertical extent angle of region in degrees up to 2 decimal places
- eR is range error of location in metres.
- eAz is azimuth error of location in degrees.
- eEle is elevation error in degrees.
- ehExtent is error in horizontal extent angle.
- evExtent is error in vertical extent angle

## 5.4 Time Information

Note that all timestamps are in UTC Zulu using ISO8601 format;
i.e.: yyyy-MM-ddTHH:mm:ss.fffZ

Detection and Status messages should have sub-second accuracy, to ease the task of the HLDMM in cross-correlating them. Other messages only need specify to the nearest seconds i.e.: yyyy-MM-ddTHH:mm:ssZ

The SAPIENT Middleware system will provide a network time reference, Time information shall be synchronised to this time reference. Modules should typically synchronise to the time reference every 10-15minutes. However this may be adjusted depending on what is appropriate on the particular network.

## 5.5 Further Guidance on XML fields

To keep the XML messages concise, it is valid to omit fields that are not mandated. This is equivalent to specifying 'no' in the report lists in the initialisation message.

### 5.5.1 Text Fields

Where XML fields include text, it is assumed that the first letter will be capitalised.

### 5.5.2 Class and Behaviour Taxonomy

It is up to each ASM supplier to define the list of class types that they can classify. The Taxonomy that the SAPIENT Middleware supports will evolve and so has been placed in Appendix D to allow it to evolve without impacting the rest of the document.

## 5.6 Managing Real-Time System Performance

Whilst the SAPIENT Middleware system architecture has been designed to provide sufficient data throughput for the HLDMM and end-user real-time requirements, it is prudent to have the ability to manage data bandwidth if issues do arise.

If the HLDMM decides it is receiving too much data from one or more ASMs to process in real-time, then the DetectionThreshold and DetectionReportRate commands can be used to limit the data being sent across the system.

If the Middleware detects an issue that may not be apparent to the HLDMM, it can send an Alert Message to the HLDMM to appraise it of the issue so that it can then decide on the most appropriate course of action.
It is recommended that the TCP_NO_DELAY flag is set for all socket connections to minimise the latency through the system.

# 6 Message Detailed Description - Initialisation

This section describes the content of the SAPIENT Middleware messages used during initialisation.

The status column describes whether the field is Mandatory (M), Mandatory List (M List), Optional (O), an Optional, List of values (O List) or (M*) where one of 2 fields must be specified. M* is mostly used where a coordinate must be specified but can be either Spherical coordinates (RangeBearing) or Cartesian Coordinates (Location)

## 6.1 Registration Message

On first connection to the SAPIENT Middleware system, an ASM must send a Registration message. This will include a number of fields declaring the capabilities of the HLDMM. For any given field where the option Yes/No is provided in the message schema, a "No" response is equivalent to omitting the field altogether.

NB. Care should be taken to ensure that fields declared in the Registration message are consistent with those actually provided in the other message types.

| Type | Field Name | Status | Description |
|---|---|---|---|
| dateTime | timestamp | M | UTC time of message |
| int | sensorID | O | The Registration Message is used to allocate a Sensor ID to an ASM. This is described in section 4.3.5. If blank or omitted the Middleware will allocate an ID. |
| string | sensorType | M | Sensor Type should be one of a number of strings agreed with the system owner and the HLDMM provider to allow the HLDMM to uniquely identify each type of ASM. |
| heartbeatDefinition | heartbeatDefinition | M | See 6.2 |
| modeDefinition[] | modeDefinition | M List | See 6.3 |

## 6.2 Registration – heartbeatDefinition

The heartbeat definition section of the Registration message defines which fields will be present in Status Report messages generated by the ASM and what format and units the fields will use.

| Type | Field Name | Status | Description |
|---|---|---|---|
| heartbeatInterval | heartbeatInterval | M | Time interval between regular 'heartbeat' Status Report messages being issued by the ASM. See 6.2.1 |
| locationType | sensorLocationDefinition | O | Define the Coordinates system to use for sensor location. E.g. UTM |

| | | | |
|---|---|---|---|
| | | | or GPS. NB. RangeBearing is not a valid value for sensorlocation |
| locationType | fieldOfViewDefinition | O | If providing sensor field of view, define the Coordinates system and units to use. |
| locationType | coverageDefinition | O | If providing sensor coverage, define the Coordinates system and units to use. |
| locationType | obscurationDefinition | O | If providing sensor obscuration regions, define the Coordinates system and units to use. |
| heartbeatReport[] | heartbeatReport | O List | |

### 6.2.1 Registration - heartbeatInterval

This states the Time interval between regular 'heartbeat' Status Report messages being issued by the ASM. If a Status Report is not received within this interval, the HLDMM can assume this ASM has been disabled, turned off or lost communication.

| Type | Field Name | Status | Description |
|---|---|---|---|
| string | Units | M | Typically 'Seconds' |
| int | Value | M | Typically between 1and 60 |

### 6.2.2 Registration - locationType

This defines the format and units for each field that can provide location information. Whilst it is valid for an ASM to provide some fields in Spherical (RangeBearing) coordinates and others in Cartesian (UTM, GPS) coordinates, it is recommended that UTM and GPS coordinates systems are not both used within the same ASM.

| Type | Field Name | Status | Description |
|---|---|---|---|
| string | Units | O | e.g. decimal degrees-metres |
| string | Datum | O | If using UTM or GPS this is typically WGS84 |
| string | Zone | O | If using UTM this is the, UTM Zone e.g. 30U, otherwise omit. |
| string | North | O | Typically 'Grid' |
| string | Value | O | UTM, GPS, RangeBearing |

### 6.2.3 Registration - heartbeatReport

This lists the fields that will be included in the Status Report message. Only fields relevant to the particular ASM should be included; all others should be omitted. So, for example, a radar sensor would not report on its exposure or white balance, and a mains-powered sensor would not report battery level. A list of typical values is provided in 6.2.4.

| Type | Field Name | Status | Description |
|---|---|---|---|
| string | Category | M | The category of field to report. This maps to the XML elements of the Status Report message i.e. sensor, power, mode, status |
| string | Type | M | The type or name of the information being provided |
| string | Units | O | A description of the units or valid values that will be reported |
| bool | onChange | O | If True, only report when the value changes. If False, always report it |

### 6.2.4 Registration – heartbeatReport – Typical Values

Below is a list of typical entries in the heartbeat report list section of the Registration message. The type and units fields should be consistent with those provided in the Status Report. (See 7.1.8)

| Category | Type | Units |
|---|---|---|
| sensor | sensorLocation | (BLANK) |
| sensor | fieldOfView | locationList or rangeBearingCone |
| sensor | coverage | locationList or rangeBearingCone |
| sensor | obscurations | locationList or rangeBearingCone |
| power | status | OK, Fault |
| power | level | percentage |
| mode | (BLANK) | Default, Others as defined in registration message |
| status | InternalFault | (BLANK) |
| status | ExternalFault | (BLANK) |
| status | Illumination | Bright, Dark, Normal |
| status | Weather | text |
| status | Clutter | Low, Medium, High |
| status | Exposure | F Stop |
| status | MotionSensitivity | probability |
| status | PTZStatus | Moving, Stopped |
| status | PD | probability |
| status | FAR | probability |

## 6.3 Registration - modeDefinition

| Type | Field Name | Status | Description |
|---|---|---|---|
| string | modeName | M | Name of mode e.g. Default, Follow |
| string | modeDescription | O | Free text description |
| settleTime | settleTime | M | See 6.3.1 |
| maximumLatency | maximumLatency | O | See 6.3.2 |
| string | scanType | O | Fixed, Scanning, Steerable, See 6.3.3 If omitted will assume Fixed. |
| string | trackingType | O | None, Tracklet, Track, TrackWithReID. See 6.3.4. If omitted will assume none |
| Duration | Duration | O | See 6.3.5 |
| modeParameter[] | modeParameter | O List | See 6.3.6 |
| detectionDefinition | detectionDefinition | M | See 6.4 |
| taskDefinition | taskDefinition | M | See 6.5 |
| string | modeType | M | Temporary, Permanent |

### 6.3.1 Registration - modeDefinition - settleTime

This gives the HLDMM an indication of the time the ASM takes to settle to normal performance after switching to this mode. An example might be where sensor processing takes a number of seconds to form a background model before producing reliable information.

| Type | Field Name | Status | Description |
|---|---|---|---|
| String | Units | M | Typically ‘seconds’ |
| Int | Value | M | e.g.5 |

### 6.3.2 Registration – modeDefinition - maximumLatency

This defines the maximum detection latency, i.e. the maximum discrepancy between actual detection and a DetectionReport timestamp.

| Type | Field Name | Status | Description |
|---|---|---|---|
| String | Units | M | Typically ‘seconds’ |
| Int | Value | M | e.g.1 |

### 6.3.3 Registration – modeDefinition – scanType

This defines the sensor field of view behaviour. If omitted will assume fixed.

- Fixed - means a field of view that does not change and all the areas are detected at all times e.g. a fixed video sensor.
- Scanning - means that the sensor will only be detecting over part of its field of view at any time e.g. a rotating radar.

- Steerable - means that the sensor can be steered to point at a location e.g. a Pan-Tilt-Zoom Camera.

#### 6.3.4 Registration – modeDefinition – trackingType

This defines the ASM tracking capabilities. If omitted will assume none.

- None - means there is no data association between detections of the same object. A unique objectID is generated by the ASM for each detection of an object.
- Tracklet - means the ASM attempts to maintain the objectID between detections of the same object but no attempt is made to join broken tracks.
- Track - means that the ASM applies a tracking algorithm to allow consistent objectID even behind occlusions.
- TrackWithReID - means that the ASM can provide tracks that can be re-associated based on features somehow.

#### 6.3.5 Registration - modeDefinition – duration

For a task that takes a fixed period e.g. 3D scan, this can be provided to give the HLDMM an indication of the time that the task will take. For ‘permanent’ mode types this can be omitted.

| Type | Field Name | Status | Description |
|---|---|---|---|
| String | Units | M | Typically ‘seconds’ |
| Int | Value | M | e.g.30 |

#### 6.3.6 Registration - modeDefinition – modeParameter

ModeParameter is a list of descriptive parameters for the mode as agreed with the HLDMM supplier at design/integration time. These can be useful for HLDMM to determine the level of Autonomy within the ASM (e.g. dynamic behaviour). This is particularly relevant to scanning or steerable sensors that can alter their behaviour such as following a target or viewing multiple zones

| Type | Field Name | Status | Description |
|---|---|---|---|
| String | Type | M | e.g. SelfAdaptation |
| Int | Value | M | e.g. ROI, Range |

### 6.4 Registration – detectionDefinition

This lists the fields that will be included in the Detection message whilst in this mode. Only fields relevant to the particular ASM should be included; all others should be omitted.

| Type | Field Name | Status | Description |
|---|---|---|---|
| locationType | locationType | M | Location units to be used by detection messages |
| geometricError | geometricError | O | See 6.4.1 |
| detectionReport[] | detectionReport | M | See 6.4.2 |
| detectionPerformance[] | detectionPerformance | O List | See 6.4.3 |
| detectionClassDefinition | detectionClassDefinition | O | See 6.4.5 |
| behaviourDefinition[] | behaviourDefinition | O List | See 6.4.8 |

### 6.4.1 Registration – detectionDefinition – geometricError

This is a list of location error characterisations to allow the HLDMM to understand the detection performance of the ASM.

| Type | Field Name | Status | Description |
|---|---|---|---|
| string | type | M | e.g. Standard Deviation |
| string | units | M | e.g. metres |
| string | variationType | M | e.g. Linear with Range, Squared with Range |
| performanceValue[] | performanceValue | O List | A list of types and values |

### 6.4.2 Registration – detectionDefinition – detectionReport

This lists the fields that will be included in the Detection message. Only fields relevant to the particular ASM should be included; all others should be omitted. A list of typical values is provided in 6.4.3

| Type | Field Name | Status | Description |
|---|---|---|---|
| string | category | M | The category of field to report. This maps to the XML elements of the Detection message i.e. detection, track, object |
| string | type | M | The type or name of the information being provided |
| string | units | M | A description of the units or valid values that will be reported |
| bool | onChange | O | If True, only report when the value changes. If omitted or False, always report it |

### 6.4.3 Registration – detectionDefinition – detectionReport Typical Values

Below is a list of typical entries in the detection report list section of the Registration message and the associated field in the Detection message. The type and units fields should be consistent with those provided in the Detection Report. (See 7.2)

| Category | Type | Units | DetectionReport field |
|---|---|---|---|
| detection | confidence | probability | detectionConfidence |
| track | confidence | probability | trackInfo, ‘confidence’ |
| track | speed | m-s | trackInfo, ‘speed’ |
| track | az | degrees | trackInfo, ‘az’ (Heading) |
| track | dR | metres | trackInfo, ‘dR (Object Change in Range in metres’) |
| track | dAz | degrees | trackInfo, ‘dAz’ (Object Change in Heading) |
| track | predictedLocation | geo | predictedLocation |
| track | predictionTimestamp | UTC | predictionTimestamp |
| object | dopplerSpeed | m-s | objectInfo, ‘dopplerSpeed’ |
| object | dopplerAz | degrees | objectInfo, ‘dopplerAz’ |
| object | majorLength | metres | objectInfo, ‘majorLength’ |
| object | majorAxisAz | degrees | objectInfo, ‘majorAxisAz’ |
| object | minorLength | metres | objectInfo, ‘minorLength’ |
| object | height | metres | objectInfo, ‘height’ |
| object | colour | none | colour |
| object | state | none | state |

6.4.4 Registration – detectionDefinition – detectionPerformance

A list of Performance measures useful to the HLDMM - only provide if available. Some examples are shown below:

```
<detectionPerformance type = "PD" units ="Per Frame" unitValue="1"
variationType="None">
  <performanceValue type = "scalar" value="0.9" />
</detectionPerformance>

<detectionPerformance type = "FAR" units ="Per Period" unitValue="1"
variationType="Linear with Range">
  <performanceValue type = "eRmin" value="0.1" />
  <performanceValue type = "eRmax" value="0.5" />
</detectionPerformance>

<detectionPerformance type = "FAR" units ="Per Period" unitValue="1"
variationType="Inverse Square with Range">
  <performanceValue type = "eRmin" value="0.1" />
      </detectionPerformance>
```

### 6.4.5 Registration – detectionDefinition – detectionClassDefinition

| Type | Field Name | Status | Description |
|---|---|---|---|
| string | confidenceDefinition | O | Single Class, Multiple Class. Omit if not providing confidence<br>Single Class - only provide confidence for the most likely class.<br>Multiple Class - provide confidence for all classes |
| confusionMatrix[] | confusionMatrix | O List | This is an optional list of string values defining the confusion matrix for a particular ASM. The exact usage to date has been agreed between the HLDMM and ASM providers and is awaiting endorsement by the SIMP. |
| classPerformance[] | classPerformance | O List | See 6.4.6 |
| classDefinition[] | classDefinition | O List | See |

It is preferred that each detection message should include a list of probabilities (confidence values) for each class, sub-class and behaviour that the ASM has said it can provide in the initialisation message. In this case the 'confidenceDefinition' tag in the initialisation message should be included with the value 'Multiple Class'.

If an ASM can only provide the most likely class for each object, then the 'confidenceDefinition' tag in the initialisation message should be included with the value 'Single Class'.

### 6.4.6 Registration – detectionDefinition - ClassPerformance

This is an optional list of classification performance measures. Some examples are shown below:

```
<classPerformance type = "FAR" units ="Per Period" unitValue="1"
variationType="Linear with Range">
  <performanceValue type = "eRmin" value="0.1" />
  <performanceValue type = "eRmax" value="0.5" />
     </classPerformance>
```

### 6.4.7 Registration – detectionDefinition – ClassDefinition

This defines the structure as to how object classes and sub-classes are provided for this ASM. Sub-classes allow a hierarchy to more narrowly define an object. All classes must be mutually exclusive and conform to the Taxonomy in D.1. Some examples are shown below:

```
<!-- list all the object classification types the ASM can detect -->
<classDefinition type = "Human">
  <confidence units="probability" />
  <!-- omit if not providing confidence -->

  <!-- list of sub-classes that can be detected for each class - can be omitted if
there are none -->
  <subClassDefinition level = "1" type="PersonID" values = "names">
```

```
    <confidence units="probability" />
    <!-- omit if not providing confidence -->
  </subClassDefinition>
</classDefinition>

<classDefinition type = "Vehicle">
  <!-- list of sub-classes that can be detected for each class - can be omitted if
there are none -->
  <subClassDefinition level = "1" type="Vehicle Class" values = "4 Wheeled, 2
Wheeled">
    <confidence units="probability" />
    <!-- omit if not providing confidence -->

    <!-- list of sub-classes that can be detected for each sub-class - can be
omitted if there are none -->
    <subClassDefinition level = "2" type="Size" values = "Heavy, Medium, Light">
      <confidence units="probability" />
      <!-- omit if not providing confidence -->

      <!-- list of sub-classes that can be detected for each sub-class - can be
omitted if there are none -->
      <subClassDefinition level = "2" type="Vehicle Type" values = "Truck, Bus,
Van, People Carrier, 4X4, Car, Motorbike, Bicycle">
        <confidence units="probability" />
        <!-- omit if not providing confidence -->
      </subClassDefinition>

    </subClassDefinition>
  </subClassDefinition>
</classDefinition>

<classDefinition type = "Animal">
  <confidence units="probability" />
  <!-- omit if not providing confidence -->
</classDefinition>

<classDefinition type = "Unknown">
  <confidence units="probability" />
  <!-- omit if not providing confidence -->
</classDefinition>
```

### 6.4.8 Registration – detectionDefinition - behaviourDefinition

This defines the structure as to how object behaviour or activity is provided for this ASM. All behaviours must conform to the Taxonomy in D.1.4. Further details of these definitions can be found in section 10.6D. Some examples are shown below:

```
<!-- list of possible behaviours that can be detected and reported -->
<behaviourDefinition type = "Walking">
  <confidence units="probability" />
  <!-- omit if not providing confidence -->
</behaviourDefinition>

<behaviourDefinition type = "Running">
  <confidence units="probability" />
  <!-- omit if not providing confidence -->
</behaviourDefinition>

<behaviourDefinition type = "Crawling">
```

```
  <confidence units="probability" />
  <!-- omit if not providing confidence -->
</behaviourDefinition>

<behaviourDefinition type = "Climbing">
  <confidence units="probability" />
  <!-- omit if not providing confidence -->
</behaviourDefinition>

<behaviourDefinition type = "Throwing">
  <confidence units="probability" />
  <!-- omit if not providing confidence -->
</behaviourDefinition>
```

## 6.5 Registration – taskDefinition

| Type | Field Name | Status | Description |
|---|---|---|---|
| int | concurrentTasks | O | The number of different tasks that can run at a time in this mode |
| regionDefinition | regionDefinition | M | See 6.5.1 |
| command[] | command | O List | See 6.5.7 |

### 6.5.1 Registration – taskDefinition - regionDefinition

| Type | Field Name | Status | Description |
|---|---|---|---|
| regionType | regionType | M List | See 6.5.2 |
| settleTime | settleTime | O | Time to settle to normal performance in this mode |
| locationType | locationType | M List | UTM, GPS, RangeBearing location type to use for region definition |
| classFilterDefinition[] | classFilterDefinition | O List | See 6.5.3 |
| behaviourFilterDefinition{} | behaviourFilterDefinition | O List | See 6.5.6 |

### 6.5.2 Registration - regionDefinition - regionType

| Type | Field Name | Status | Description |
|---|---|---|---|
| string | value | M | Area of Interest, Ignore, Boundary |

### 6.5.3 Registration – regionDefinition - classFilterDefinition

| Type | Field Name | Status | Description |
|---|---|---|---|
| filterParameter[] | filterParameter | O List | See 6.5.4 |
| subClassFilterDefinition[] | subClassFilterDefinition | O List | List of sub-class filters See 6.5.5 |
| string | type | M | Class type to filter on e.g. “Human”, “Vehicle” |

### 6.5.4 Registration – regionDefinition - filterParameter

This is a list of class parameters the ASM can filter on. Typically confidence is the main characteristic to filter on and the operator is typically ‘Greater Than’.

```
<filterParameter name="confidence" operators="All, Greater Than, Less Than, Equals, None" />
```

### 6.5.5 Registration – regionDefinition - subClassFilterDefinition

| Type | Field Name | Status | Description |
|---|---|---|---|
| filterParameter[] | filterParameter | O List | See 6.5.4 |
| string | type | M | Sub-Class type to filter |
| string | level | M | Sub class hierarchy level that this type is at. Typically 1-3. |

### 6.5.6 Registration – regionDefinition - behaviourFilterDefinition

| Type | Field Name | Status | Description |
|---|---|---|---|
| filterParameter[] | filterParameter | O List | See 6.5.4 |
| string | type | O List | Behaviour type to filter on e.g. “Running” |

### 6.5.7 Registration – taskDefinition – command

The command list in the Registration message gives the list of commands that an ASM supports, including the default commands that all ASMs should provide. Some examples are provided below:

| Type | Field Name | Status | Description |
|---|---|---|---|
| string | name | M | E.g. Request, DetectionThreshold, Mode. See list at 7.6 |
| string | units | M | Valid Values for this command e.g. Heartbeat |
| int | completionTime | M | E.g. 1 |
| string | completionTimeUnits | M | Typically 'seconds' |

```
<command name="Request" units="Registration, Reset, Heartbeat, Sensor Time, Stop,
Start, Take Snapshot" completionTime="10" completionTimeUnits="seconds" />
<command name="DetectionThreshold" units="Auto, Low, Medium, High"
completionTime="1" completionTimeUnits="seconds" />
<command name="DetectionReportRate" units="Auto, Low, Medium, High"
completionTime="1" completionTimeUnits="seconds" />
<command name="ClassificationThreshold" units="Auto, Low, Medium, High"
completionTime="1" completionTimeUnits="seconds" />
<command name="Mode" units="Default, Others as defined in registration message"
completionTime="2" completionTimeUnits="seconds" />
<command name="LookAt" units="locationList, rangeBearingCone" completionTime="30"
completionTimeUnits="seconds" />
     <command name="LookFor" units="image" completionTime="60"
     completionTimeUnits="seconds" />
```

### 6.5.8 Task Commands

Below is a list of commands that an ASM may typically support. Further commands may be defined in the Registration message.

| Type | Units | Description |
|---|---|---|
| Request | Registration | Request the ASM sends a Registration message. |
| Request | Reset | Request the ASM 'reboots' itself. |
| Request | Heartbeat | Request the ASM sends a Status Report message. |
| Request | Stop | Request the ASM stops sending Detection and Alert messages until told to start again. |
| Request | Start | Request the ASM starts sending Detection and Alert messages after a previous 'Stop' request. |
| Request | Take Snapshot | Task a camera to take a still image. The URL to the image should be provided in the SensorTaskACK message |
| DetectionThreshold | Auto, Lower, Higher | Allows the HLDMM to adjust the ASMs detection threshold or return it to its own 'Auto' threshold. |
| DetectionReportRate | Auto, Lower, Higher | Allows the HLDMM to tell the ASM to |

| | | |
|---|---|---|
| | | report more or less detections or return it to its own ‘Auto’ rate. |
| Mode | Default, Others as defined in registration message e.g. ‘Follow’ | Allows the HLDMM to tell the ASM to change to a different operating mode as defined list of mode definitions in the Registration message. |
| LookAt | locationList, rangeBearingCone | Allows the HLDMM to task a steerable ASM to look at a particular location. |
| LookFor | | A placeholder for future use |

## 6.6 Registration Acknowledgement Message

Upon receipt of a Registration message, the Middleware generates a Registration Acknowledgement message either confirming the ASM’s existing sensor ID or if required allocating a new sensor ID.

If the ASM does not receive a response within 30 seconds of sending the Registration message, the ASM should close the existing socket connection, open a new socket connection to the SDA and send the message again. The Registration Acknowledgement message has the following format:

```
<SensorRegistrationACK>
<sensorID></sensorID>
</SensorRegistrationACK>
```

The ASM must use the sensorID value provided as the sensorID/sourceID in all subsequent messages.

# 7 Message Detailed Description – Normal Operation

This section describes the content of the SAPIENT Middleware messages used during normal operation.

The status column describes whether the field is Mandatory (M), Mandatory List (M List), Optional (O), an Optional, List of values (O List) or (M*) where one of 2 fields must be specified. M* is mostly used where a coordinate must be specified but can be either Spherical coordinates (RangeBearing) or Cartesian Coordinates (Location)

## 7.1 Status Report Message

This message is sent regularly to give the HLDMM information about the ASM operating mode, and to provide confidence that it is still working correctly (or conversely that performance can be expected to degrade). Only fields relevant to the particular ASM should be included; all others should be omitted. So, for example, a radar sensor would not report on its exposure or white balance, and a mains-powered sensor would not report battery level.

| Type | Field Name | Status | Description |
|---|---|---|---|
| dateTime | timestamp | M | UTC time of message |
| int | sourceID | M | Unique identifier of ASM or HLDMM |
| long | reportID | M | Numerical identifier of this report, incremented by the source |
| string | system | M | Overall status of sender: OK, Error, Tamper, GoodBye.<br>ASM sends goodbye to close the communication cleanly in case it needs to shut down and reboot. |
| string | info | M | Typically 'New'. If no new information provided then 'Unchanged' is also valid but may be ignored by the HLDMM. |
| int | activeTaskID | O | current task being performed by the ASM |
| string | mode | O | Transition, Stopped, Failed, Default, Others as defined in registration message. If a fatal error has occurred then report "Failed" here. |
| StatusReportPower | power | O | See 7.1.2 |
| location | sensorLocation | O | See 7.1.3 |
| StatusReportFieldOfView | fieldOfView | O | See 7.1.4 |
| StatusReportCoverage | coverage | O | See 7.1.5 |
| StatusReportObscuration[] | obscuration | O List | See 7.1.6 |
| StatusReportStatusRegion[] | statusRegion | O List | See 7.1.7 |
| StatusReportStatus[] | status | O List | See 7.1.8 |

### 7.1.1 Status Report – Mode

Mode is an optional field reported where a sensor provides multiple modes or states. E.g. Transition, Stopped, Failed Default, 3D Scanning, Scan and Search.

If omitted, then 'Default' mode will be assumed.

If a fatal error has occurred, then 'Failed' should be reported.

Normally, the ASM will work in 'Default' mode but the HLDMM can request it changes into any other mode specified in the ASM Registration message. When a sensor is "on the move" and not detecting it should report as 'Transition'. If an ASM has received a 'Stop' Request command, it should report 'Stopped'.

### 7.1.2 Status Report - Power

This is used to report the status of the ASM power supply if available.

| Type | Field Name | Status | Description |
|---|---|---|---|
| string | source | O | Mains or Battery |
| string | status | O | OK, Fault |
| int | level | O | Battery level, typically as a percentage e.g. 50 |

### 7.1.3 Status Report – Sensor Location

Sensor location is an optional field that reports the current geographical location of the sensor. With a fixed sensor, this may be reported once during initialisation although typically it is reported in every status report to aid debugging. Location is provided as a <location> element with optional fields for altitude and error (eX, eY, eZ)

### 7.1.4 Status Report – Field Of View

Sensor Field of View is an optional field that reports the current field of view of the sensor. With a fixed sensor, this may be reported once during initialisation. For steerable sensors, this will be reported regularly.

This should either be reported as a <rangebearingCone> element or a <locationList> element. For a location list, points should be in order around the boundary of the field of view polygon.

### 7.1.5 Status Report – Coverage

Sensor Coverage is an optional field that reports the extent of the area a sensor can cover but not necessarily all at once. Typically, this will only be reported for scanning or steerable sensors. For example, a Pan Tilt Zoom camera may have 360degree coverage but only a field of view of 45degrees.

This should either be reported as a <rangebearingCone> element or a <locationList> element. For a location list, points should be in order around the boundary of the coverage polygon.

### 7.1.6 Status Report – Obscuration

Obscuration is an optional field that can define one or more regions that the sensor can report as being unable to detect within. This refers to the region beyond a static obstacle, rather than the region in the field of view beyond a moving target. If the HLDMM wants to know what is obscured behind a moving target, it can work it out itself from the reported object size and location in the detection report.

This should either be reported as a <rangebearingCone> element or a <locationList> element. For a location list, points should be in order around the boundary of the obscuration polygon.

### 7.1.7 Status Report – Status Region

This is used for other intelligent functions such as defining a region of interest for an ASM to only report on activity within. This is a placeholder for future use.

| Type | Field Name | Status | Description |
|---|---|---|---|
| String | type | M | |
| int | regionID | M | |
| string | regionName | M | |
| string | regionStatus | O | |
| string | description | O | |
| locationList | locationList | M* | |
| rangeBearingCone | rangeBearingCone | M* | |

### 7.1.8 Status Report – Status

The ‘Status’ element of the status message provides a way for ASMs to report conditions that may affect the performance of this or other ASMs. The ‘Level’ field indicates the severity of the condition.

- Error is something severely affecting capability
- Warning is performance degradation
- Information is detected environmental condition that might affect other sensors
- Sensor is information about the current sensor state

NB If a fatal error has occurred then report "Failed" under sensor mode as per 7.1.1.

For ‘sensor’ status values, the type and value fields should be consistent with those defined in the Registration message. (See 6.2.4).

For example, the following indicates that rain has been detected that does not affect this sensor but may affect other sensors.

<status level = "**Information**" type = "Weather" value = "Rain" />

The following indicates that the performance of the ASM has been degraded by external conditions e.g. rain snow, but it is still able to perform its current tasking

<status level = "**Warning**" type = "**ExternalFault**" />

The following indicates that the performance of the ASM has been degraded by external conditions e.g. illumination, and so it is unable to perform its current tasking:

<status level = "**Error**" type = "**ExternalFault**" value = "**Illumination**" />

The 'type' field may be one of the following but is not limited to this list:

InternalFault, ExternalFault, Illumination, Weather, Clutter, Exposure, MotionSensitivity, PTZStatus, PD, FAR. (See also 6.2.4.)

| Type | Field Name | Status | Description |
|---|---|---|---|
| string | level | O | Error, Warning, Information, Sensor |
| string | type | M | See examples above |
| string | value | O | See examples above and in 6.2.4 |

## 7.2 Detection Report Message

This message is sent by the ASM to report each detection (or, optionally, group of detections or track). Track data is an optional element used to provide the HLDMM with more information about the movement of an object, where available.
Where sensor performance allows, classification information should also be included for each detected object.

| Type | Field Name | Status | Description |
|---|---|---|---|
| dateTime | timestamp | M | UTC time of detection |
| Int | sourceID | M | Unique identifier of ASM or HLDMM |
| Long | reportID | M | Unique identifier of this message. Typically increments from start-up |
| Long | objectID | M | Unique identifier of object detected. If ASM / HLDMM is not capable of tracking or identifying an object then it can match the reportID |
| Int | taskID | M | Identifier of task that the ASM was carrying out when this detection occurred. By default zero |
| String | State | O | Whether special case such as object lost |
| location | location | M* | Either location or rangeBearing must be provided to give the location of the detection |
| rangeBearing | rangeBearing | M* | |
| Float | detectionConfidence | O | Probability that the ASM has detected a real object Value between 0 and 1. 0 is low confidence, 1 is certain |
| DetectionReport TrackInfo[] | trackInfo | O List | Additional metadata about the track |
| DetectionReport PredictedLocation | predictedLocation | O | Prediction of where the object will be at a later time |
| DetectionReport ObjectInfo[] | objectInfo | O List | Additional metadata about the detected object |
| string | colour | O | Text colour description |

| | | | |
|---|---|---|---|
| DetectionReport Class[] | class | O List | List of possible types of object this could be. E.g. Human, Vehicle. Also what the probability of each class is |
| DetectionReport Behaviour[] | behaviour | O List | List of possible activities this object could be doing. E.g. Walking, running. Also what the probability of each behaviour is |
| associatedFile[] | associatedFile | O List | List of urls to associated media such as image snapshots, 3d point clouds etc. |

### 7.2.1 Detection Report - Class

| Type | Field Name | Status | Description |
|---|---|---|---|
| float | confidence | O | Probability that the object is of this object type (class) |
| subClass[] | subClass1 | O List | List of possible sub classes of object that this object could be |
| string | type | M | Object type e.g. Human, Vehicle |

### 7.2.2 Detection Report - Behaviour

| Type | Field Name | Status | Description |
|---|---|---|---|
| float | confidence | O | Probability that the object is undertaking each activity is |
| String | type | M | Object activity e.g. walking, running. |

### 7.2.3 Detection Report - Associated File

| Type | Field Name | Status | Description |
|---|---|---|---|
| String | type | M | Type of file e.g. image |
| String | url | M | URL to the media. Typically the sensor identifier will be included as a sub folder in the URL |

### 7.2.4 Detection Report - Track Info

| Type | Field Name | Status | Description |
|---|---|---|---|
| String | type | M | Name value pairs of metadata associated with track |
| Float | value | M | |
| Float | e | O | Error value associated with value |

### 7.2.5 Detection Report – Predicted Location

| Type | Field Name | Status | Description |
|---|---|---|---|
| location | location | M* | Either location or rangeBearing must be provided to give the location of the predicted detection. |
| rangeBearing | rangeBearing | M* | |

| | | | |
|---|---|---|---|
| dateTime | predictionTimestamp | O | Timestamp of predicted detection in UTC if different from detection report timestamp |

7.2.6 Detection Report - Object Info

| Type | Field Name | Status | Description |
|---|---|---|---|
| String | type | M | Name value pairs of metadata associated with track |
| Float | value | M | |
| Float | e | O | Error value associated with value |

## 7.3 Alert Message

The primary use of the Alert message is for the HLDMM to provide Alerts to the Operator via the GUI. It can also be used by an ASM to inform the HLDMM that it is about to do something or that something unusual has happened. An example of this might be a sensor deciding to do a detailed scan of an object or area.

For information only alert messages, where no response is expected, place 'Information' in the alertType field. If the alert message merits a response from the HLDMM, the Alert Response message can be used.

This description field within the Alert message can also be used to pass auxiliary information on the status of the ASM directly to the SAPIENT Middleware GUI. Whilst not currently implemented, this could be used to pass information on the ASM internal behaviour (e.g. rejection of false alarms, detection of benign events) for debugging purposes and display on a simple "Engineering display window" via the SAPIENT Middleware GUI.

| Type | Field Name | Status | Description |
|---|---|---|---|
| dateTime | timestamp | M | UTC time of message |
| int | sourceID | M | Unique identifier of ASM or HLDMM |
| int | alertID | M | |
| string | alertType | O | Type of alert. 'Information' implies no response required. |
| string | status | O | Current status of alert:<br>Active, Acknowledge, Reject, Ignore, Clear |
| string | description | O | Text Description of Alert |
| location | location | O* | Providing a Cartesian location is optional and should only be done if relevant to do so |
| rangeBearing | rangeBearing | O* | Providing a RangeBearing location is optional and should only be done if relevant to do so |
| int | regionID | O | Optional field to refer to an existing Region of Interest as designated by the HLDMM |
| string | priority | O | 'Low', 'Medium' 'High' |
| float | ranking | O | 0.0 – 1.0 |
| float | confidence | O | 0.0 – 1.0 |

| | | | |
|---|---|---|---|
| string | debugText | O | HLDMM debug information |
| associatedFile[] | associatedFile | O List | List of associated data files. See 7.3.1 |
| associatedDetection[] | associatedDetection | O List | List of associated detection points. See 7.3.2. |

### 7.3.1 Alert - Associated File

| Type | Field Name | Status | Description |
|---|---|---|---|
| String | type | M | Type of file e.g. image |
| String | url | M | URL to the media. Typically the sensor identifier will be included as a sub folder in the URL |

### 7.3.2 Alert - Associated Detection

| Type | Field Name | Status | Description |
|---|---|---|---|
| location | location | O* | Optional field giving one or more Cartesian locations associated with the alert |
| string | description | O | Optional field to allow additional description of the location |
| dateTime | timestamp | O | Timestamp of associated detection |
| int | sourceID | M | Unique identifier of ASM or HLDMM that was the source of the detection |
| long | objectID | M | Unique identifier of the detection |

## 7.4 SensorTask Message

These are messages sent by the HLDMM to an individual ASM, commanding it to undertake a specific task. The overall philosophy used is that commands and filtering options are task-based with an optional associated region defined in the task command. An ASM acknowledges a SensorTask

If an ASM cannot comply with a control message, it sends a SensorTaskACK acknowledgement message with an appropriate rejection message.

| Type | Field Name | Status | Description |
|---|---|---|---|
| dateTime | timestamp | M | UTC time of message |
| int | sensorID | M | Unique identifier of ASM or HLDMM to be tasked |
| int | taskID | M | 0= default task, See 7.4.1 |
| string | taskName | O | Optional name of task |
| string | taskDescription | O | Optional description of task |
| dateTime | taskStartTime | O | UTC time to start task |
| dateTime | taskEndTime | O | UTC time to end task |
| string | control | M | ‘Start’, ‘Stop’, ‘Pause’, ‘Default’ - See 7.4.2 |
| SensorTaskRegion[] | region | O List | Optional List of regions associated with the task. See 7.5 |
| SensorTaskCommand | command | O | Populated if this task is a command. See 7.6 |

#### 7.4.1 Sensor Task - Task ID

Control messages must include a task ID. The task ID is an integer number generated by the HLDMM to uniquely identify a task so that the acknowledgement message and any subsequent detection messages have a reference. Task 0 indicates the default task to be undertaken when there is no other task allocated to the ASM.

#### 7.4.2 Sensor Task – Control

This field indicates what to do with this task as follows:

- Start - Initialises and schedules or starts the task or restarts a paused task
- Stop - stop the task, remove its definition, revert to previous task
- Pause - stop the task but keep it defined for later restart, revert to previous task
- Default – (without other fields populated in the message) - start or revert to the default task, stopping all other tasks.
- Default – (plus other fields populated in the message) - defines the default task to be undertaken when no other tasks are defined.

NB. A stop message does not require a 'region' or 'command' section

## 7.5 SensorTask - Region

Region tasks are designed to control the geographical area within the coverage of steerable ASMs or field of view of fixed ASMs that the ASM should report detections from. Typically by default, the ASM will detect from its full field of view and so a region task will limit the area to report from.

In addition, if supported by the ASM, object type can be used to restrict the types of objects that will be reported.

Any task overrides the default task. If a second override task comes and the ASM can't support multiple tasks then it needs to report that in the acknowledgement message when tasked with the second task. If regions are to be removed from the default task then a default task message is sent with all remaining regions.

| Type | Field Name | Status | Description |
|---|---|---|---|
| String | type | M | Area of Interest, Ignore, Boundary. |
| Int | regionID | M | Region ID Number generated by the HLDMM |
| String | regionName | M | e.g. Car Park |
| locationList | locationList | M* | Cartesian Points should be in order around the boundary of the region |
| rangeBearingCone | rangeBearingCone | M* | Alternatively describe region as a cone |
| SensorTaskRegionClassFilter[] | classFilter | O List | List of class Filters to apply to region. If omitted then no filtering is applied. |
| SensorTaskRegionBehaviourFilter[] | behaviourFilter | O List | List of behaviour Filters to apply to region. If omitted then no filtering is applied. |

### 7.5.1 SensorTask – Region - Class Filter

| Type | Field Name | Status | Description |
|---|---|---|---|
| parameter | parameter | M | Typically filter on confidence value |
| subClassFilter[] | subClassFilter | O List | List of sub classes to filter on |
| string | type | O | Class to filter on |
| string | priority | O | Priority of this filter |

### 7.5.2 SensorTask – Region - Behaviour Filter

| Type | Field Name | Status | Description |
|---|---|---|---|
| parameter | parameter | M | Typically filter on confidence value |
| string | type | O | Behaviour to filter on |
| string | priority | O | Priority of this filter |

## 7.6 SensorTask – Command

This section of the message is populated for command based tasks. These tend to be short lived or instantaneous whereas Region ones tend to persist until another task is received. Only one command message should be issued at a time and so only one of the top level fields should be populated.

| Type | Field Name | Status | Description |
|---|---|---|---|
| string | request | M* | See 7.6.1 |
| string | detectionThreshold | M* | See 7.6.2 |
| string | detectionReportRate | M* | See 7.6.3 |
| string | classificationThreshold | M* | Placeholder for future development |
| string | mode | M* | See 7.6.4 |
| sensorTaskCommandLookAt | lookAt | M* | See 7.6.5 |
| sensorTaskCommandLookFor | lookFor | M* | Placeholder for future development |

### 7.6.1 Request Commands

The ‘Request: Registration’ command allows the HLDMM to request the ASM to send the initialisation message again.
The ‘Request: Heartbeat’ command allows the HLDMM to request a full heartbeat message with the current value of all fields supported by that ASM.
The ‘Request: Reset’ command allows the HLDMM to stop the ASM’s current task and return to its default state
The ‘Request: Stop’ command makes the ASM stop sending all detection messages. The ASM should continue sending heartbeat messages to maintain the connection to the data agent.

The 'Request: Start' command makes the ASM resume sending detection messages after a 'Request: Stop' command.

#### 7.6.2 SensorTask – Command – detectionThreshold

The individual ASMs are expected to exhibit a level of autonomy. Therefore the HLDMM is not expected to use a "long screwdriver" to set low-level parameters within an individual ASM. The HLDMM may however choose to request an ASM to adjust its internal thresholds to reduce its FAR, or increase its P(d), based for example on comparing the detection performance of one ASM with another with an overlapping field of view.

This is covered by the command 'DetectionThreshold'. This allows the HLDMM to adjust the ASMs detection threshold to set it 'Lower' or 'Higher' or return it to its own 'Auto' threshold.

#### 7.6.3 SensorTask – Command – detectionReportRate

The HLDMM may wish to request an individual ASM to alter the frequency at which it reports data, for example to throttle back the detections if they were supplied too frequently. This is covered by the command 'DetectionReportRate' which can be changed up or down by the HLDMM, or used to request a specific output rate. This allows the HLDMM to adjust the ASMs report rate to set it 'Lower' or 'Higher' or return it to its own 'Auto' rate.

#### 7.6.4 SensorTask – Command – mode

The 'mode' message forces the sensor to change to one of its operating modes defined in the Initialisation message. If a sensor only supports one mode, then this command can be ignored.

#### 7.6.5 SensorTask – Command – Look At

The 'lookAt' command tasks the ASM to suspend its current tasking, take a quick look at a specific location to see if it can see anything and then return to its on-going tasking. It will return a detection message if it detects an object at that location, or a 'SensorTaskACK' Acknowledgement message with the 'Status' field "Nothing found". Both will include the task ID of the 'lookAt' command.

The ASM will determine the most appropriate field of view to use in response to the 'lookAt' command. If this is not the default field of view, then this will be reported by a heartbeat message to accompany any detection message response to the 'lookAt' command.

| Type | Field Name | Status | Description |
|---|---|---|---|
| locationList | locationList | M* | Location or region to look at |
| rangeBearingCone | rangeBearingCone | M* | Spherical coordinate location to look at relative to the sensor location. |

#### 7.6.6 SensorTask – Command – Look For

This is a placeholder for future functionality.

| Type | Field Name | Status | Description |
|---|---|---|---|
| string | type | M | Type of object to look for |
| string | url | M | URL to source data. |

## 7.7 SensorTask Acknowledgement Message

These short messages complete the handshaking, and allow an ASM to accept or reject a task. If an ASM cannot comply with a control message, it sends a SensorTaskACK acknowledgement message with an appropriate error message.

For example, the following indicates that the ASM can perform the task:
<SensorTaskACK><timestamp>2014-03-14 14:00:00Z</timestamp>
<sensorID>1</sensorID><TaskID>1</TaskID>
<Status>**Accepted**</Status></SensorTaskACK>

The following indicates that the ASM can perform the task but with performance degraded by external conditions e.g. rain snow
<SensorTaskACK><timestamp>2014-03-14 14:00:00Z</timestamp>
<sensorID>1</sensorID><TaskID>1</TaskID>
<Status>**Accepted**</Status>
<Reason>**Performance Degraded**</Reason></SensorTaskACK>

The following indicates that the ASM is unable to perform the task due to its performance being severely affected by external conditions e.g. illumination:
<SensorTaskACK><timestamp>2014-03-14 14:00:00Z</timestamp>
<sensorID>1</sensorID><TaskID>1</TaskID>
<Status>**Rejected**</Status>
<Reason>**Performance Degraded**</Reason></SensorTaskACK>

The following indicates that the ASM has received a control message that it cannot interpret or contains errors:
<SensorTaskACK><timestamp>2014-03-14 14:00:00Z</timestamp>
<sensorID>1</sensorID><TaskID>1</TaskID>
<Status>**Rejected**</Status>
<Reason>**Message Error**</Reason></SensorTaskACK>

## 7.8 Error Messages

If an invalid message is received by the Data Agent from either an ASM or HLDMM client then the Data Agent will respond by sending an error message back to the source. This contains the time of the message the content of the invalid message and a message describing the error.

# 8 Example Messages

In order to help clarify the use of the xml schema, example messages are provided for a fictional generic EO sensor, and its interaction with the HLDMM. This sensor is a fixed field of view camera, which has the ability to detect and track objects, classify them as human or vehicle. Further, it can classify humans according to a number of activities being undertaken, and further discriminate vehicles into a number of vehicle types.

Examples are given of

- the sensor registration message,
- heartbeat messages, including as the situation changes,
- a detection message
- control and acknowledgement messages, showing a dialogue between the ASM and HLDMM

## 8.1 Example Registration Message

```
<?xml version="1.0"?>
<SensorRegistration>
<timestamp>2014-02-26T11:31:25.000Z</timestamp> <!-- UTC (Zulu) time message sent -
->
<sensorID></sensorID>      <!-- Included if the ASM already has a sensorID
associated with it. If omitted, the Sapient system will assign a sensorID for the
ASM. -->
<sensorType>ACME Generic EO Sensor</sensorType>  <!-- Unique Type identifier -->

<!-- This section describes the format of Heartbeat Messages -->

<heartbeatDefinition>
<heartbeatInterval units="seconds" value="10" /> <!-- time between heartbeats -->

      <fieldOfViewDefinition>
      <locationType units="decimal degrees-metres" datum="WGS84" zone="30U"
north="Grid" >GPS</locationType>
      </fieldOfViewDefinition>

      <obscurationDefinition>
      <locationType units="decimal degrees-metres" datum="WGS84" zone="30U"
north="Grid" >GPS</locationType>
      </obscurationDefinition>

<!-- list of all reported parameters - this list can be extended -->
<heartbeatReport category="sensor" type="sensorLocation" units="" onChange="true"
/>
<heartbeatReport category="sensor" type="fieldOfView" units="locationList"
onChange="true" />
<heartbeatReport category="sensor" type="obscurations" units="locationList"
onChange="true" />
<heartbeatReport category="power" type="status" units="OK, Fault" onChange="true"
/>
<heartbeatReport category="status" type="InternalFault" units="" onChange="true" />
<heartbeatReport category="status" type="ExternalFault" units="" onChange="true" />
<heartbeatReport category="status" type="Illumination" units="Bright, Dark, Normal"
onChange="true" />
```

```xml
</heartbeatDefinition>
<modeDefinition type = "Permanent">
<modeName>Default</modeName> <!-- mandatory name to associate with this mode -->
<modeDescription>Normal Operation</modeDescription> <!-- optional description to
associate with this mode -->
<settleTime units="seconds" value="10"/>
<maximumLatency units="seconds" value="5"/> <!-- maximum detection latency -->
<scanType>Fixed</scanType>

<trackingType>Tracklet</trackingType>

<!-- This section describes the format of Detection Messages -->
<detectionDefinition>

      <locationType units="decimal degrees-metres" datum="WGS84"
north="Grid">GPS</locationType>

      <!-- list of location error characterisations -->
      <geometricError type = "Standard Deviation" units="metres"
variationType="Linear with Range">
            <performanceValue type = "eRmin" value="0.1" />
            <performanceValue type = "eRmax" value="0.5" />
      </geometricError>

      <!-- list of all reported detection parameters - this list can be extended -->
      <detectionReport category="detection" type="confidence" units="probability" />
      <detectionReport category="track" type="confidence" units="probability" />
      <detectionReport category="track" type="speed" units="m-s" />
      <detectionReport category="track" type="az" units="degrees" />

      <detectionReport category="object" type="majorLength" units="metres" />
      <detectionReport category="object" type="majorAxisAz" units="degrees" />
      <detectionReport category="object" type="minorLength" units="metres" />
      <detectionReport category="object" type="height" units="metres" />
      <detectionReport category="object" type="colour" units="none" />
      <detectionReport category="associatedFile" type="image" units="none" />

<detectionClassDefinition>
      <confidenceDefinition>Multiple Class</confidenceDefinition>

      <classPerformance type = "FAR" units ="Per Period" unitValue="1"
variationType="Linear with Range">
            <performanceValue type = "eRmin" value="0.1" />
            <performanceValue type = "eRmax" value="0.5" />
      </classPerformance>

      <!-- list all the object classification types the ASM can detect -->
      <classDefinition type = "Human"><confidence units="probability" />
      </classDefinition>

      <classDefinition type = "Vehicle">
      <subClassDefinition level = "1" type="Vehicle Class" values = "4 Wheeled">
      <confidence units="probability" />
            <subClassDefinition level = "2" type="Size" values = "Light">
            <confidence units="probability" />
                  <subClassDefinition level = "2" type="Vehicle Type" values = "Car">
                  <confidence units="probability" />
                  </subClassDefinition>
            </subClassDefinition>
      </subClassDefinition>
      </classDefinition>
```

```
</detectionClassDefinition>

<behaviourDefinition type = "Walking">
<confidence units="probability" /></behaviourDefinition>

<behaviourDefinition type = "Running">
<confidence units="probability" /></behaviourDefinition>

<behaviourDefinition type = "Crawling">
<confidence units="probability" /></behaviourDefinition>

</detectionDefinition>

<!-- This section describes the format of Control Messages -->
<taskDefinition>
<concurrentTasks>1</concurrentTasks>
      <regionDefinition>

      <!-- list all types of region the ASM can support - can be omitted if there
are none -->
            <regionType>Area of Interest</regionType>
            <regionType>Ignore</regionType>

            <settleTime units="seconds" value="1"/> <!-- time to settle to normal
      performance -->

            <locationType units="decimal degrees" datum="WGS84" north="Grid"
>GPS</locationType>

      <classFilterDefinition type = "Human">
      <!-- list of class parameters the ASM can filter on -->
      <filterParameter name="confidence" operators="All" />
      </classFilterDefinition>

      <classFilterDefinition type = "Vehicle">
      <!-- list of class parameters the ASM can filter on -->
      <filterParameter name="confidence" operators="Greater Than" />

      <subClassFilterDefinition level = "1" type="Vehicle Class">
      <!-- list of sub-class parameters the ASM can filter on -->
      <filterParameter name="confidence" operators="Less Than" />

      <subClassFilterDefinition level = "2" type="Size">
      <!-- list of sub-class parameters the ASM can filter on -->
      <filterParameter name="confidence" operators="All" />

      <subClassFilterDefinition level = "3" type="Vehicle Type">
      <!-- list of sub-class parameters the ASM can filter on -->
      <filterParameter name="confidence" operators="All" />
      </subClassFilterDefinition>

      </subClassFilterDefinition>
      </subClassFilterDefinition>
</classFilterDefinition>

<behaviourFilterDefinition type = "Walking">
      <filterParameter name="confidence" operators="All, Greater Than, Less Than,
Equals, None" />
      </behaviourFilterDefinition>
<behaviourFilterDefinition type = "Running">
```

```
      <filterParameter name="confidence" operators="All, Greater Than, Less Than,
Equals, None" />
      </behaviourFilterDefinition>

      <behaviourFilterDefinition type = "Crawling">
      <filterParameter name="confidence" operators="All, Greater Than, Less Than,
Equals, None" />
      </behaviourFilterDefinition>

</regionDefinition>

<command name="Request" units="Registration, Reset, Heartbeat, Sensor Time, Stop,
Start, Take Snapshot" completionTime="10" completionTimeUnits="seconds" />
<command name="DetectionThreshold" units="Auto, Low, Medium, High"
completionTime="1" completionTimeUnits="seconds" />
<command name="DetectionReportRate" units="Auto, Low, Medium, High"
completionTime="1" completionTimeUnits="seconds" />
<command name="LookAt" units="locationList, rangeBearingCone" completionTime="30"
completionTimeUnits="seconds" />

</taskDefinition>
</modeDefinition>
</SensorRegistration>
```


## 8.2 Example Heartbeat Messages

### 8.2.1 Initial heartbeat message


```
<?xml version="1.0"?><StatusReport>
      <timestamp>2014-02-27T13:06:43.100Z</timestamp>
      <sourceID>5</sourceID>
      <reportID>67</reportID>
      <system>OK</system>
      <info>New</info>
      <activeTaskID>0</activeTaskID>
      <power source = "Mains" status = "OK" />

      <fieldOfView>
            <locationList>
            <location><X>123450</X><Y>345670</Y><eX>2</eX> <eY>3</eY></location>
            <location><X>123480</X><Y>345670</Y><eX>2</eX> <eY>3</eY></location>
            <location><X>123480</X><Y>345690</Y><eX>2</eX> <eY>3</eY></location>
            <location><X>123450</X><Y>345690</Y><eX>2</eX> <eY>3</eY></location>
            <location><X>123450</X><Y>345670</Y><eX>2</eX> <eY>3</eY></location>
            </locationList>
      </fieldOfView>

      <obscuration>
            <locationList>
      <location><X>123475</X><Y>345680</Y><eX>2</eX> <eY>3</eY></location>
      <location><X>123470</X><Y>345690</Y><eX>2</eX> <eY>3</eY></location>
      <location><X>123477</X><Y>345690</Y><eX>2</eX> <eY>3</eY></location>
            </locationList>
      </obscuration>

      <status level = "Information" type = "Illumination" value = "Dark" />
      <status level = "Sensor" type = "Exposure" value = "High" />
</StatusReport>
```

### 8.2.2 Additional Obscuration Detected, Illumination Level changed

```xml
<?xml version="1.0"?><StatusReport> <!-- These are the heartbeat messages-->
      <timestamp>2014-02-27T13:06:53.200Z</timestamp>
      <sourceID>5</sourceID>
      <reportID>68</reportID>
      <system>OK</system>
      <info>Additional</info>
      <activeTaskID>0</activeTaskID>

      <obscuration>
            <locationList>
            <location><X>123465</X><Y>345680</Y><eX>2</eX> <eY>3</eY></location>
            <location><X>123460</X><Y>345690</Y><eX>2</eX> <eY>3</eY></location>
            <location><X>123467</X><Y>345690</Y><eX>2</eX> <eY>3</eY></location>
            </locationList>
      </obscuration>

      <status level = "Information" type = "Illumination" value = "Medium" />
</StatusReport>
```

### 8.2.3 Status Unchanged

```xml
<?xml version="1.0"?><StatusReport> <!-- These are the heartbeat messages-->
      <timestamp>2014-02-27T13:07:03Z</timestamp>
      <sourceID>5</sourceID>
      <reportID>69</reportID>
      <system>OK</system>
      <info>Unchanged</info>
</StatusReport>
```

## 8.3 Example Detection Messages

### 8.3.1 Example Detection

```xml
<?xml version="1.0"?><DetectionReport>
      <timestamp>2014-02-27T14:08:25.365Z</timestamp>
      <sourceID>5</sourceID>
      <reportID>1</reportID>
      <objectID>34</objectID>
      <taskID>0</taskID>

      <!--provide location or range bearing -->
      <location>
            <X>123465</X><Y>345681</Y><Z>1.5</Z>
            <eX>1</eX><eY>0.5</eY><eZ>0.25</eZ>
      </location>

      <detectionConfidence>0.93</detectionConfidence>

      <trackInfo type="confidence" value = "0.9" e = "0.01"/>
      <trackInfo type="speed" value = "2.0" e = "0.5"/>
      <trackInfo type="az" value = "130.0" e = "5.0"/>

      <objectInfo type="majorLength" value = "1.3" e = "0.1" />
      <objectInfo type="majorAxisAz" value = "270.0" e = "5.0" />
      <objectInfo type="minorLength" value = "1.4" e = "0.1" />
      <objectInfo type="height" value = "1.8" e = "0.1" />

      <colour>red</colour>
```

```xml
        <class type="Human"><confidence>0.7</confidence></class>
        <class type="Vehicle"><confidence>0.2</confidence></class>

        <behaviour type="Walking"><confidence>0.5</confidence></behaviour>

    </DetectionReport>
```

### 8.3.2 Track Lost from View

```xml
    <?xml version="1.0"?><DetectionReport>
        <timestamp>2014-02-27T14:08:25.365Z</timestamp>
        <sourceID>5</sourceID>
        <reportID>2</reportID>
        <objectID>34</objectID>
        <taskID>0</taskID>
        <state>lostFromView</state>
    <!-- optional field to allow reporting loss of track from field of view -->
    </DetectionReport>
```

### 8.3.3 Track Lost in View

```xml
    <?xml version="1.0"?><DetectionReport>
        <timestamp>2014-02-27T14:08:25.365Z</timestamp>
        <sourceID>5</sourceID>
        <reportID>3</reportID>
        <objectID>34</objectID>
        <taskID>0</taskID>
        <state>lostInView</state>
    <!-- optional field to allow reporting loss of track within field of view -->
    </DetectionReport>
```

## 8.4 Example Control and Acknowledgement Messages

### 8.4.1 Default Tasking

```xml
<?xml version="1.0"?><SensorTask>
<!-- Define the default tasking to be used when no other tasking -->
<timestamp>2014-02-27T12:00:00.000Z</timestamp> <!-- UTC (Zulu) time message sent -
->
<sensorID>5</sensorID> <!-- unique numerical identifier of the ASM to task -->
<taskID>0</taskID> <!-- generated by the HLDMM, 0 for default task -->
<taskName>Default</taskName> <!-- optional name to associate with this task ID -->
<taskDescription>Default task - detect intruders in car park</taskDescription>
<control>Default</control> <!-- Default task to undertake when no other tasks -->

<region><type>Area of Interest</type>
<regionID>1</regionID><!-- Region ID Number generated by the HLDMM -->
<regionName>Car Park</regionName>

<!-- Points should be in order around the boundary of the region -->
<locationList>
<location><X>123450</X><Y>345670</Y></location>
<location><X>123480</X><Y>345670</Y></location>
<location><X>123480</X><Y>345690</Y></location>
<location><X>123450</X><Y>345690</Y></location>
<location><X>123450</X><Y>345670</Y></location>
</locationList>

<!-- class omitted so all classes detected -->
</region>
</SensorTask>
```

### 8.4.2 Acknowledgement Message for Default Task

```
<?xml version="1.0"?><SensorTaskACK>
<timestamp>2014-02-27T12:00:01.000Z</timestamp> <!-- UTC (Zulu) time message sent -
->
<sensorID>5</sensorID>
<taskID>0</taskID>
<status>Accepted</status>
</SensorTaskACK>
```

### 8.4.3 Task 1 to over-ride Default tasking

```
<?xml version="1.0"?><SensorTask>

<!-- start alternate task -->
<timestamp>2014-02-27T13:00:00.000Z</timestamp> <!-- UTC (Zulu) time message sent -
->
<sensorID>5</sensorID> <!-- unique numerical identifier of the ASM to task -->
<taskID>1</taskID> <!-- generated by the HLDMM, 0 for default task -->
<control>Start</control> <!-- What to do with this task. Default sets the default
task to undertake when no other tasks are given. -->

<region>
<type>Area of Interest</type>
<regionID>2</regionID><!-- Region ID Number generated by the HLDMM -->
<regionName>Trial Area</regionName>

<!-- Points should be in order around the boundary of the region -->
<locationList>
<location><X>123550</X><Y>345670</Y></location>
<location><X>123580</X><Y>345670</Y></location>
<location><X>123580</X><Y>345690</Y></location>
<location><X>123550</X><Y>345690</Y></location>
<location><X>123550</X><Y>345670</Y></location>
</locationList>

<classFilter type="Human">
<parameter name="confidence" operator="Greater Than" value="0.5" />
</classFilter>

<classFilter type="Vehicle"> <!-- ignore all vehicles -->
<parameter name="confidence" operator="None" value="0" />
</classFilter>

<behaviourFilter type = "Walking">
<parameter name="confidence" operator="Greater Than" value="0.5" />
</behaviourFilter>

<behaviourFilter type = "Running">
<parameter name="confidence" operator="Greater Than" value="0.3" />
</behaviourFilter>
</region>
</SensorTask>
```

### 8.4.4 Acknowledgement Message for Task 1

```
<?xml version="1.0"?><SensorTaskACK>
<timestamp>2014-02-27T13:00:01.000Z</timestamp> <!-- UTC time message sent -->
<sensorID>5</sensorID>
<taskID>1</taskID>
<status>Accepted</status>
</SensorTaskACK>
```

### 8.4.5 Stop Task 1 and return to default task

```
<?xml version="1.0"?><SensorTask>
<!-- stop alternate task and revert to default task -->

<timestamp>2014-02-27T14:00:00.000Z</timestamp> <!-- UTC time message sent -->
<sensorID>5</sensorID> <!-- unique numerical identifier of the ASM to task -->
<taskID>1</taskID> <!-- generated by the HLDMM, 0 for default task -->
<control>Stop</control> <!-- Stopping this task will revert to default task -->
</SensorTask>
```

### 8.4.6 Acknowledgment of Stop Message

```
<?xml version="1.0"?><SensorTaskACK>
<timestamp>2014-02-27T14:00:01.000Z</timestamp> <!-- UTC time message sent -->
<sensorID>5</sensorID>
<taskID>1</taskID>
<status>Complete</status>
</SensorTaskACK>
```

### 8.4.7 Set Detection Threshold High Command

```
<?xml version="1.0"?><SensorTask>
<timestamp>2014-02-27T12:30:00.000Z</timestamp> <!-- UTC time message sent -->
<sensorID>5</sensorID>
<taskID>0</taskID> <!-- generated by the HLDMM, 0 for default task -->
<control>Start</control>
<command>
      <detectionThreshold>High</detectionThreshold>
</command>
</SensorTask>
```

### 8.4.8 Acknowledgement of Successful Threshold High Command

```
<?xml version="1.0"?><SensorTaskACK>
<timestamp>2014-02-27T12:30:01.000Z</timestamp> <!-- UTC time message sent -->
<sensorID>5</sensorID>
<taskID>0</taskID>
<status>Accepted</status>
</SensorTaskACK>
```

### 8.4.9 Acknowledgement of Rejected Threshold High Command

```
<?xml version="1.0"?><SensorTaskACK>
<timestamp>2014-02-27T12:30:01.000Z</timestamp> <!-- UTC time message sent -->
<sensorID>5</sensorID>
<taskID>0</taskID>
<status>Rejected</status>
<reason>Not Supported</reason>
</SensorTaskACK>
```

### 8.4.10 Request Reset Command

```
<?xml version="1.0"?><SensorTask>
<timestamp>2014-02-27T14:30:00.000Z</timestamp> <!-- UTC  time message sent -->
<sensorID>5</sensorID>
<taskID>0</taskID>
<control>Start</control>
<command><request>Reset</request></command>
</SensorTask>
```

#### 8.4.11 LookAt Command

The 'lookAt' command tasks the ASM to suspend its current tasking, take a quick look at a specific location to see if it can see anything and then return to its on-going tasking. The actual time spent undertaking the lookAt command is at the discretion of the ASM.

```
<?xml version="1.0"?><SensorTask>
<timestamp>2014-02-28T12:45:00.000Z</timestamp> <!-- UTC time message sent -->
<sensorID>5</sensorID>
<taskID>10</taskID> <!-- generated by the HLDMM, 0 for default task -->
<taskName>Look At Gate</taskName> <!-- optional name for this task ID -->
<taskDescription>Have a quick look at the gate</taskDescription>
<control>Start</control> <command>
     <lookAt>
          <!-- provide location list or range bearing -->
          <locationList>
          <location><X>123550</X><Y>345670</Y></location>
          </locationList>
     </lookAt>
</command>
</SensorTask>
```

#### 8.4.12 Detection Report response to LookAt Command

```
<?xml version="1.0"?><DetectionReport>
     <timestamp>2014-02-28T12:45:05.365Z</timestamp> <!-- UTC time of detection -->
     <sourceID>5</sourceID>
     <reportID>123</reportID>
     <objectID>84</objectID>
     <taskID>10</taskID> <!-- task that this detection is generated from; -->

     <location>
          <X>123550</X><Y>345670</Y><Z>1.0</Z>
          <eX>1</eX><eY>0.5</eY><eZ>0.25</eZ>
     </location>

     <detectionConfidence>0.93</detectionConfidence>

     <objectInfo type="majorLength" value = "1.3" e = "0.1" />
     <objectInfo type="height" value = "1.6" e = "0.1" />
     <colour>red</colour>

     <class type="Human"><confidence>0.7</confidence></class>
     <class type="Vehicle"><confidence>0.01</confidence></class>
</DetectionReport>
```

#### 8.4.13 Null Response to LookAt Command

```
<?xml version="1.0"?><SensorTaskACK>
<timestamp>2014-02-28T12:45:05.000</timestamp> <!-- UTC time message sent -->
<sensorID>5</sensorID>
<taskID>10</taskID>
<status>Complete</status>
<reason>Nothing Found</reason>
</SensorTaskACK>
```

# 9 HLDMM – ASM Database Interaction

## 9.1 ASM – HLDMM Message Protocol Summary

This section summarises the message protocol between the ASMs and the HLDMM.

In general, all messages from the ASM to the HLDMM will be stored in a database with each message type having one or more database tables assigned to it.

### 9.1.1 Initialisation

As described in section 4.3.2. The Registration Message shall be sent by the ASM to the Data Agent before any other messages.

The Data Agent shall forward the message to the HLDMM Data Agent where if required, a sensor ID will be allocated to this ASM and the acknowledgement message sent back to the ASM via the Data Agent.

The Initialisation message will be forwarded on to the HLDMM by the HLDMM Data Agent. It will also be stored in the database for future reference by the HDA.

### 9.1.2 Ongoing Messages from ASMs during normal operation

As described in section 4.5.1, ongoing messages from ASMs, (Detections, Heartbeats and Alerts) shall be stored in the database by the Data Agent for the HLDMM to read.
If an alert message requires an immediate response, such as when an ASM wishes to autonomously change mode, but would like permission from the HLDMM, the message will be additionally forwarded on to the HLDMM via the HLDMM Data Agent.

### 9.1.3 Control Messages

As described in section 4.6.1, control messages are sent from the HLDMM to the HLDMM Data Agent which stores the message in the database. They are then routed to the Data Agent for the ASM specified in the message which forwards the message on to the ASM. The ASM then shall respond with an acknowledgement message to accept or reject the task in the control message. The acknowledgement message is sent back to the Data Agent by the ASM which forwards the message on to the HLDMM Data Agent. The HLDMM Data Agent forwards the acknowledgement on to the HLDMM and updates the message record in the database with the acknowledgement information.

## 9.2 Initialisation

The initialisation message will be stored in the database table ‘sensor_registration’.

### 9.2.1 Sensor_Registration Table

It contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| sensor_id | Bigint | ASM sensor identifier as provided in registration message or allocated by HLDMM Data Agent |
| sensor_type | Text | ASM type description from registration message |
| message_time | Timestamp without time zone | Timestamp of registration message |
| text | Xml | Full XML text of registration message |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| key_id | Bigserial | Primary key for table |

Example Query:

SELECT "time", sensor_id, sensor_type, text

FROM sensor_registration

## 9.3 Heartbeat Message

The heartbeat message will be split over a number of database tables for efficiency of performance and storage. A primary table will be provided for the HLDMM to poll to check that messages are being regularly received. The sensor location and power status is included in this table. Regional information such as sensor field of view, coverage and areas of obscuration are stored in either the region or range_bearing tables as appropriate to each ASM. Additional status information is stored as one or more entries in the messages table.

### 9.3.1 Status_Report Table

This is the primary table for heartbeat messages; it contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| report_id | Bigint | Unique identifier of this heartbeat report – always populated |
| source_id | Bigint | ASM sensor identifier – always populated |
| message_ time | Timestamp without time zone | Timestamp of report – always populated |
| active_task_id | Text | Identifier(s) of current tasks associated with this ASM – always populated |
| system | Text | ‘system’ element of message (Null if not populated) |
| info | Text | ‘info’ element of message (Null if not populated) |

| | | |
|---|---|---|
| mode | Text | ‘mode’ element of message (Null if not populated) |
| power_source | Text | ‘source’ attribute of ‘power’ element (Null if not populated) |
| power_status | Text | ‘status’ attribute of ‘power’ element (Null if not populated) |
| power_level | Int | ‘level’ attribute of ‘power’ element (Null if not populated) |
| x | Double | ASM location X (Eastings) coordinate (Null if not populated) |
| y | Double | ASM location Y (Northings) coordinate (Null if not populated) |
| z | Double | ASM location Altitude (Null if not populated) |
| ex | Double | X (Eastings) coordinate error (Null if not populated) |
| ey | Double | Y (Northings) coordinate error (Null if not populated) |
| ez | Double | Altitude error (Null if not populated) |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| status_id | Bigint | Primary key for table / Common key for status tables |

Example Query:

SELECT source_id, message_time, report_id, system, info, active_task_id, mode, power_source, power_status, power_level, x, y, z, ex, ey, ez

FROM status_report;

#### 9.3.2 Status_Report_Region Table

This table holds ASM field of view, coverage and areas of obscuration data in Cartesian coordinates systems. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| report_id | Bigint | Unique identifier of this heartbeat report |
| source_id | Bigint | ASM sensor identifier |
| message_<br>time | Timestamp without time zone | Timestamp of report |
| type | Text | Type of region e.g.’fieldOfView’,’coverage’ |
| region_id | Int | Unique identifier where there are multiple regions of the same type from |

| | | |
|---|---|---|
| | | the same ASM e.g. areas of obscuration |
| location | Polygon | Location polygon |
| location_e | Polygon | Error values for location polygon |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| status_id | Bigint | Common key for status tables |
| key_id | Bigserial | Primary key for table |

Example Query:

SELECT source_id, message_time, report_id, type, region_id, location, location_e

FROM status_report_region;

### 9.3.3 Status_Report_Range_Bearing Table

This table holds ASM field of view, coverage and areas of obscuration data in Range-Bearing coordinate system. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| report_id | Bigint | Unique identifier of this heartbeat report |
| source_id | Bigint | ASM sensor identifier |
| message_ time | Timestamp without time zone | Timestamp of report |
| type | Text | Type of region e.g.'fieldOfView','coverage' |
| r | Double | Range coordinate |
| az | Double | Azimuth angle of centre of region |
| ele | Double | Elevation angle of centre of region |
| hextent | Double | Horizontal angle of extent of region |
| vextent | Double | Vertical angle of extent of region |
| er | Double | Range coordinate error |
| eaz | Double | Azimuth angle of centre of region error |
| eele | Double | Elevation angle of centre of region error |
| ehextent | Double | Horizontal angle of extent of region error |
| evextent | Double | Vertical angle of extent of region error |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| status_id | Bigint | Common key for status tables |
| key_id | Bigserial | Primary key for table |

Example Query:

SELECT source_id, message_time, report_id, type, r, az, ele, hextent, vextent, er, eaz, eele, ehextent, evextent

FROM status_report_range_bearing;

### 9.3.4 Status_Report_Messages Table

This table holds one or more status elements from the message. This is populated when one or more ‘status’ elements is present in the heartbeat message. Where a heartbeat message contains multiple values, multiple records will be stored. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| report_id | Bigint | Unique identifier of this heartbeat report |
| source_id | Bigint | ASM sensor identifier |
| message_ time | Timestamp without time zone | Timestamp of report |
| level | Text | Severity of information as per ‘level’ attribute of ‘status’ element. |
| type | Text | Type of information as per ‘type’ attribute of ‘status’ element. |
| value | Text | ‘value’ attribute of ‘status’ element. |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| status_id | Bigint | Common key for status tables |
| key_id | Bigserial | Primary key for table |

Example Query:

SELECT source_id, message_time, report_id, level, type, value

FROM status_report_messages;

## 9.4 Detection Message

The detection message will be split over a number of database tables for efficiency of performance and storage. A primary table will be provided for detections in Cartesian coordinates and a primary table will be provided for detections in Range-Bearing coordinates. Supplementary information such as object type will be stored in separate tables. The primary tables should be polled for new detections and then the other tables queried as required.

### 9.4.1 Detection_Report_Location Table

This is the primary table for detection messages in Cartesian coordinates systems.

The table containsthe following fields:

| Field | SQL type | Description |
|---|---|---|
| report_id | Bigint | Unique identifier of this detection report – always populated |
| source_id | Bigint | ASM sensor identifier – always populated |
| object_id | Bigint | Unique identifier (for this ASM) of the object in this detection report – always populated |
| detection_ time | Timestamp without time zone | Timestamp of detection – always populated |
| task_id | Bigint | Identifier of any task associated with this detection (Null if not populated) |
| state | Text | 'State' element of message (Null if not populated) |
| detection_ confidence | Double | Detection confidence value (Null if not populated) |
| x | Double | X (Eastings) coordinate – always populated |
| y | Double | Y (Northings) coordinate – always populated |
| z | Double | Altitude (Null if not populated) |
| ex | Double | X (Eastings) coordinate error (Null if not populated) |
| ey | Double | Y (Northings) coordinate error (Null if not populated) |
| ez | Double | Altitude error (Null if not populated) |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| detection_id | Bigint | Primary key for table / Common key for detection tables |

Example Query:

SELECT source_id, detection_time, report_id, object_id, task_id, state, detection_confidence, x, y, z, ex, ey, ez

FROM detection_report_location;

### 9.4.2 Detection_Report_Range_Bearing Table

This is the primary table for detection messages in Range-Bearing coordinate system. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| report_id | Bigint | Unique identifier of this detection report – always populated |
| source_id | Bigint | ASM sensor identifier – always populated |
| object_id | Bigint | Unique identifier (for this ASM) of the object in this detection report – always populated |

| | | |
|---|---|---|
| detection _time | Timestamp without time zone | Timestamp of detection – always populated |
| task_id | Bigint | Identifier of any task associated with this detection (Null if not populated) |
| state | Text | ‘State’ element of message (Null if not populated) |
| detection_ confidence | Double | Detection confidence value (Null if not populated) |
| r | Double | Range coordinate |
| az | Double | Azimuth coordinate |
| ele | Double | Elevation Angle coordinate (Null if not populated) |
| z | Double | Altitude (Null if not populated) |
| er | Double | Range coordinate error (Null if not populated) |
| eaz | Double | Azimuth coordinate error (Null if not populated) |
| eele | Double | Elevation coordinate error (Null if not populated) |
| ez | Double | Altitude error (Null if not populated) |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| detection_id | Bigint | Primary key for table / Common key for detection tables |

Example Query:

```
SELECT source_id, detection_time, report_id, object_id, task_id, state, detection_confidence, r, az, ele, z, er, eaz, eele, ez
  FROM detection_report_range_bearing;
```

### 9.4.3 Detection_Report_Track_Info Table

This table contains coordinates system independent supplementary detection data. This is populated when one or more ‘trackInfo’ or ‘objectInfo’ elements is present in the detection message. Where a detection message contains multiple values, multiple records will be stored. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| report_id | Bigint | Unique identifier of this detection report |
| source_id | Bigint | ASM sensor identifier |
| object_id | Bigint | Unique identifier (for this ASM) of the object in this detection report |

| | | |
|---|---|---|
| detection _time | Timestamp without time zone | Timestamp of detection |
| info_type | Text | Whether a 'trackInfo' or 'objectInfo' element |
| type | Text | 'Type' attribute of trackInfo' or 'objectInfo' element |
| value | Double | 'Value' attribute of trackInfo' or 'objectInfo' element |
| e | Double | 'e' attribute of trackInfo' or 'objectInfo' element |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| detection_id | Bigint | Common key for detection tables |
| key_id | Bigserial | Primary key for table |

Example Query:

SELECT source_id, detection_time, report_id, object_id, info_type, type, value, e
  FROM detection_report_track_info;

### 9.4.4 Detection_Report_Class Table

This table contains the top-level object type classification data. This is only populated when one or more 'class' elements is present in the detection message. Where a detection message contains multiple 'class' elements; multiple records will be stored. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| report_id | Bigint | Unique identifier of this detection report |
| source_id | Bigint | ASM sensor identifier |
| object_id | Bigint | Unique identifier (for this ASM) of the object in this detection report |
| detection _time | Timestamp without time zone | Timestamp of detection |
| type | Text | 'Type' attribute of 'class' element |
| confidence | Double | Value of 'confidence' element |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| detection_id | Bigint | Common key for detection tables |
| key_id | Bigserial | Primary key for table |

Example Query:

SELECT source_id, detection_time, report_id, object_id, type, confidence
  FROM detection_report_class;

9.4.5 Detection_Report_Subclass Table

This table contains the lower-level object type sub-classification data. This is only populated when one or more 'subClass' elements is present in the detection message. Where a detection message contains multiple 'subClass' elements; multiple records will be stored. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| report_id | Bigint | Unique identifier of this detection report |
| source_id | Bigint | ASM sensor identifier |
| object_id | Bigint | Unique identifier (for this ASM) of the object in this detection report |
| detection _time | Timestamp without time zone | Timestamp of detection |
| class | Text | Parent 'class' type of this sub class |
| level | Integer | 'level' attribute of 'subClass' element |
| type | Text | 'type' attribute of 'subClass' element |
| value | Text | 'value' attribute of 'subClass' element |
| confidence | Double | Value of 'confidence' element |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| detection_id | Bigint | Common key for detection tables |
| key_id | Bigserial | Primary key for table |
| subclass_id | Bigint | ID of this record in sub class tree |
| parent_subclass_id | Bigint | ID of parent class in sub class tree |

Example Query:

```
SELECT source_id, detection_time, report_id, object_id, class, level, type, value, confidence
  FROM detection_report_subclass;
```

9.4.6 Detection_Report_Behaviour Table

This table contains the object behaviour data. This is only populated when one or more 'behaviour' elements are present in the detection message. Where a detection message contains multiple 'behaviour' elements, multiple records will be stored. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| report_id | Bigint | Unique identifier of this detection report |
| source_id | Bigint | ASM sensor identifier |
| object_id | Bigint | Unique identifier (for this ASM) of the object in this detection report |

| | | |
|---|---|---|
| detection _time | Timestamp without time zone | Timestamp of detection |
| type | Text | ‘Type’ attribute of ‘behaviour’ element |
| confidence | Double | Value of ‘confidence’ element |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| detection_id | Bigint | Common key for detection tables |
| key_id | Bigserial | Primary key for table |

Example Query:

```
SELECT source_id, detection_time, report_id, object_id, type, confidence
  FROM detection_report_behaviour;
```

### 9.4.7 Detection_Report_Assoc_File Table

This table contains any associated file data associated with the detection message. This is only populated when the ‘associatedFile’ element is present in the detection message. Where a detection message contains multiple ‘associatedFile’ elements, multiple records will be stored. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| report_id | Bigint | Unique identifier of this detection report |
| source_id | Bigint | ASM sensor identifier |
| object_id | Bigint | Unique identifier (for this ASM) of the object in this detection report |
| detection _time | Timestamp without time zone | Timestamp of detection |
| type | Text | ‘type’ attribute of ‘associatedFile’ element |
| url | Text | ‘url’ attribute of ‘associatedFile’ element |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| detection_id | Bigint | Common key for detection tables |
| key_id | Bigserial | Primary key for table |

Example Query:

```
SELECT source_id, detection_time, report_id, object_id, type, url
  FROM detection_report_assoc_file;
```

### 9.4.8 Ground Truth Table

This table is similar to the detection_report_location table and can be used where a ground truth feed is available such as a GPS location from a target drone used for testing. Such a feed can be provided to the middleware using detection messages in Cartesian coordinates systems.

The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| report_id | Bigint | Unique identifier of this detection report – always populated |
| source_id | Bigint | Ground Truth ASM sensor identifier – always populated |
| object_id | Bigint | Unique identifier (for this ASM) of the object in this detection report – always populated |
| report_ time | Timestamp without time zone | Timestamp of ground truth data – always populated |
| x | Double | X (Eastings) coordinate – always populated |
| y | Double | Y (Northings) coordinate – always populated |
| z | Double | Altitude (Null if not populated) |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| detection_id | Bigint | Primary |

Example Query:

SELECT source_id, report_time, update_time, report_id, object_id, x, y, z

FROM ground_truth;

## 9.5 Control Message

The control message will be passed directly between modules to prevent delay. It will be stored in the database for diagnostics in the sensor_task table. It could be used to restore the system state after a failure.

### 9.5.1 Sensor_Task Table

The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| task_id | Bigint | Unique identifier of this task – always populated |
| sensor_id | Bigint | ASM sensor identifier – always populated |
| taskxml | Text | XML text of this message – always populated |
| message_time | Timestamp without time zone | Timestamp of this message – always populated |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| key_id | Bigserial | Primary key for table |

Example Query:

```
SELECT sensor_id, message_time, task_id, taskxml
  FROM sensor_task;
```

### 9.5.2 Sensor_TaskAck Table

The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| task_id | Bigint | Unique identifier of this task – always populated |
| sensor_id | Bigint | ASM sensor identifier – always populated |
| ack_timestamp | Timestamp without time zone | Timestamp of acknowledgement message to this message (Null if not populated) |
| ack_status | Text | Text from 'status' element of acknowledgement message (Null if not populated) |
| ack_reason | Text | Text from 'reason' element of acknowledgement message (Null if not populated) |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| task_key_id | Bigint | Key in sensor_task table for this task |
| file_type | Text | Optional attached file type |
| file_url | Text | Optional attached file url |
| key_id | Bigserial | Primary key for table |

Example Query:

```
SELECT sensor_id, ack_timestamp, task_id, ack_status, ack_reason,
     task_key_id, file_type, file_url
  FROM taskack;
```

## 9.6 Alert Message

The control message will be split over a number of database tables for efficiency of performance and storage. A primary table will be provided with the key text fields and supplementary tables will be used where location information is provided, one for Cartesian coordinates, one for Range-Bearing coordinates.

9.6.1 Alert Table

This is the primary table for alert messages. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| alert_id | Bigint | Unique identifier of this alert message – always populated |
| source_id | Bigint | ASM sensor identifier – always populated |
| alert_time | Timestamp without time zone | Timestamp of alert event – always populated |
| alert_type | Text | Text from 'alertType' element of message – always populated |
| status | Text | Text from 'status' element of message – always populated |
| description | Text | Text from 'description' element of message – always populated |
| priority | Text | Low/Medium/High alert priority |
| ranking | Double | Alert Ranking (Higher number more important) |
| response_timestamp | Timestamp without time zone | Timestamp of latest response message to this alert (Null if not populated) |
| response_reason | Text | Text from 'reason' element of alert response message (Null if not populated) |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| key_id | Bigint | Primary key for table / Common key for alert tables |

Example Query:

```
SELECT source_id, alert_time, alert_id, alert_type, status, description,
    response_timestamp, response_status, response_reason
  FROM alert;
```

9.6.2 Alert_Location Table

This is the table for alert messages that include locations in Cartesian coordinates systems. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| alert_id | Bigint | Unique identifier of this alert message – always populated |
| source_id | Bigint | ASM sensor identifier – always populated |

| | | |
|---|---|---|
| alert_time | Timestamp without time zone | Timestamp of alert event – always populated |
| x | Double | X (Eastings) coordinate – always populated |
| y | Double | Y (Northings) coordinate – always populated |
| z | Double | Altitude(Null if not populated) |
| ex | Double | X (Eastings) coordinate error (Null if not populated) |
| ey | Double | Y (Northings) coordinate error (Null if not populated) |
| ez | Double | Altitude error (Null if not populated) |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| key_id | Bigint | Common key for alert tables |

Example Query:

```
SELECT source_id, alert_time, alert_id, x, y, z, ex, ey, ez
  FROM alert_location;
```

9.6.3 Alert_Range_Bearing Table

This is the table for alert messages that include locations in Range-Bearing coordinate system. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| alert_id | Bigint | Unique identifier of this alert message – always populated |
| source_id | Bigint | ASM sensor identifier – always populated |
| alert_time | Timestamp without time zone | Timestamp of alert event – always populated |
| r | Double | Range coordinate – always populated |
| az | Double | Azimuth coordinate – always populated |
| ele | Double | Elevation Angle Coordinate (Null if not populated) |
| z | Double | Altitude (Null if not populated) |
| er | Double | Range coordinate error (Null if not populated) |
| eaz | Double | Azimuth coordinate error (Null if not populated) |
| eele | Double | Elevation Angle Coordinate error (Null if |

| | | |
|---|---|---|
| | | not populated) |
| ez | Double | Altitude error (Null if not populated) |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| key_id | Bigint | Common key for alert tables |

Example Query:

SELECT source_id, alert_time, alert_id, r, az, ele, z, er, eaz, eele, ez

FROM alert_range_bearing;

9.6.4 Alert_Assoc_File Table

This table contains any associated file data associated with the alert message. This is only populated when the 'associatedFile' element is present in the alert message. Where an alert message contains multiple 'associatedFile' elements, multiple records will be stored. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| alert_id | Bigint | Unique identifier of this alert message – always populated |
| source_id | Bigint | ASM sensor identifier – always populated |
| alert_time | Timestamp without time zone | Timestamp of alert event – always populated |
| type | Text | 'type' attribute of 'associatedFile' element |
| url | Text | 'url' attribute of 'associatedFile' element |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| key_id | Bigint | Common key for alert tables |
| table_key_id | Bigserial | Primary key for table |

Example Query:

SELECT source_id, alert_time, alert_id, type, url FROM alert_assoc_file;

9.6.5 Alert_Assoc_Detection Table

This table contains any detections associated with the alert message. This is only populated when the 'associatedDetection' element is present in the alert message. Where an alert message contains multiple 'associatedDetection' elements, multiple records will be stored. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| alert_id | Bigint | Unique identifier of this alert message – always populated |
| source_id | Bigint | ASM sensor identifier – always populated |
| alert_time | Timestamp without time zone | Timestamp of alert event – always populated |

| | | |
|---|---|---|
| Object_id | Bigint | Unique identifier (for this ASM) of the detection object |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| key_id | Bigint | Common key for alert tables |
| table_key_id | Bigserial | Primary key for table |

Example Query:

```
SELECT source_id, alert_time, alert_id, object_id
  FROM alert_assoc_detection;
```

# 10 Interface between HLDMM and GUI

The following data will be passed between the HLDMM and the GUI.

| Data Type | Direction |
|---|---|
| Tracks and Detection Points | To GUI |
| Alerts | To GUI |
| Sensor Field of View and Coverage | To GUI |
| New High-Level Tasking / Regions of Interest | From GUI |
| Operator Response to alerts | From GUI |

## 10.1 HLDMM – GUI Database

A second set of the database tables defined in section 9 is used for passing data between the HLDMM and the SAPIENT Middleware GUI as follows:

### 10.1.1 Initialisation

Currently there is no requirement for a registration message to be passed between the HLDMM and the GUI and so there is no database table for this.

### 10.1.2 Heartbeat Message

A single table is used to store status reports (heartbeat messages) from the HLDMM. It holds region data in Cartesian coordinates systems. The table contains the following fields:

| Field | SQL type | Description |
|---|---|---|
| report_id | Bigint | Unique identifier of this heartbeat report |
| source_id | Bigint | Always zero (source is HLDMM) |
| message_ time | Timestamp without time zone | Timestamp of report |
| type | Text | Type of region e.g.'fieldOfView','coverage' |
| region_id | Int | Unique identifier where there are multiple regions of the same type from the same ASM e.g. areas of obscuration |
| location | Polygon | Location polygon |
| location_e | Polygon | Error values for location polygon |
| region_name | Text | Name for region |
| status | Text | Region status description |
| description | Text | Description of region |
| update_time | Timestamp without time zone | Timestamp of last update to table |
| key_id | Bigserial | Primary key for table |

Example Query:

SELECT source_id, message_time, report_id, type, region_id, location, location_e, region_name, status, description

FROM hl_status_report_region;

#### 10.1.3 Detection Messages

A second set of detection tables is used to store HLDMM fused detections for display on the GUI. These tables have the prefix 'hl_'. The HLDMM will always provide locations in Cartesian coordinates and so the range-bearing coordinates table is obsolete.

#### 10.1.4 Control Messages

A second set of control tables is used to store GUI generated Sensor Tasks; these are sent to the HDA which then writes the message to the database. See section 10.5 for details.

#### 10.1.5 Alert Messages

A second set of alert tables is used to store HLDMM generated alerts for display on the GUI. See section 10.3 for details.

### 10.2 Tracks and Detection Points

The HLDMM will provide these by using the Detection Message as defined in section 7.2. The HLDMM Data Agent will then put these in the correct 'hl_detection' database tables for the GUI to read.

### 10.3 Alerts

The HLDMM will provide these by using the Alert Message as defined in section 7.3. The HLDMM Data Agent will then put these in the 'hl_alert' tables in the database for the GUI to read.

### 10.4 Sensor Field of View and Coverage

The GUI will read these from the database tables populated by the ASM Data Agents. Where the HLDMM wishes to provide additional field of view information, it will provide a heartbeat message to the HLDMM Data Agent with the region element populated as defined in section 7.1.

### 10.5 High-Level Tasking and Regions of Interest

Tasking from the GUI will be passed to the HLDMM using control messages in the same format as from the HLDMM to the ASMs. The High-Level Tasking messages will be stored in the database. New regions of interest will be defined as new records in the 'hl_regions' table in the database. This table is used by the GUI for displaying regions of interest.

## 10.6 Operator Response to Alerts

The GUI will allow the operator to give the system a response to alerts by sending AlertResponse messages to the Middleware. These can then be forwarded on to HLDMM. Examples of how this might be used are:

- To remove old or unimportant alerts from the display so that others are not missed. i.e. Clear alert
- To give the system feedback - to build up a picture of false alarms as follows:
    - Acknowledge alert – this is a genuine target, classified correctly
    - Reject alert – the system has not detected a genuine target or has classified it incorrectly
    - Ignore alert – the operator does not wish to be made aware of this or similar alerts
- To instruct the HLDMM to perform an action e.g. prioritise an alert
- To inform the system of the operator's intent

# A XML Message Structure

This Appendix contains blank versions of the XML messages. The xsd schemas accompanying this ICD should be considered as the master definitive reference but the XML is included here as a quick reference for a module developer.

## A.1 Registration Message

```xml
<?xml version="1.0"?>
<SensorRegistration>
  <timestamp></timestamp>
  <sensorID></sensorID>
  <sensorType> </sensorType>

  <!-- This section describes the format of Heartbeat Messages -->
  <heartbeatDefinition>
    <heartbeatInterval units="" value="" />

    <sensorLocationDefinition>
      <locationType units="" datum="" zone="" north="" ></locationType>
    </sensorLocationDefinition>

    <fieldOfViewDefinition>
      <locationType units="" datum="" zone="" north="" ></locationType>
    </fieldOfViewDefinition>

    <coverageDefinition>
      <locationType units="" datum="" zone="" north="" ></locationType>
    </coverageDefinition>

    <obscurationDefinition>
      <locationType units="" datum="" zone="" north="" ></locationType>
    </obscurationDefinition>

    <!-- list of all reported parameters -->
    <heartbeatReport category="" type="" units="" onChange="" />
  </heartbeatDefinition>

  <!-- This section describes the behaviour of each operating mode including the
format of Detection messages produced and Control messages accepted -->
  <modeDefinition type = "">
    <modeName/>
    <modeDescription/>
    <settleTime units="" value=""/>
    <maximumLatency units="" value=""/>
    <scanType/>
    <trackingType/>
    <duration units="" value=""/>
    <modeParameter type="" value="" />

    <!-- This section describes the format of Detection Messages for this mode -->
    <detectionDefinition>
      <locationType />
      <geometricError />
      <performanceValue/>
      </geometricError>
      <detectionReport category="" type="" units="" />
```

```
      <detectionPerformance />
      <performanceValue/>
      </detectionPerformance>

      <!-- List of all the object classification types and sub-class types the ASM
can detect -->
      <detectionClassDefinition>
        <confidenceDefinition/>
        <confusionMatrix type="" />
        <classPerformance />
        <performanceValue/>
        </classPerformance>

        <classDefinition type = "">
          <subClassDefinition level = "1" type="" values = "">
            <confidence units="" />
            <subClassDefinition level = "2" type="" values = "">
              <confidence units="" />
            </subClassDefinition>
          </subClassDefinition>
        </classDefinition>
      </detectionClassDefinition>

      <!-- List of all the behaviours the ASM can detect -->
      <behaviourDefinition type = "">
        <confidence units="" />
      </behaviourDefinition>
    </detectionDefinition>

    <!-- This section describes the format of Control Messages in this mode -->
    <taskDefinition>
      <concurrentTasks/>

       <!-- List all the types of region tasks the ASM can support in this mode -->
      <regionDefinition>
        <regionType/>
        <settleTime units="" value=""/>
        <locationType units="" datum="" zone="" north="" ></locationType>

        <classFilterDefinition type = "">
          <filterParameter name="" operators="" />
          <subClassFilterDefinition level = "1" type="">
            <filterParameter name="" operators="" />
            <subClassFilterDefinition level = "2" type="">
              <filterParameter name="" operators="" />
            </subClassFilterDefinition>
          </subClassFilterDefinition>
        </classFilterDefinition>

        <behaviourFilterDefinition type = "">
          <filterParameter name="" operators="" />
        </behaviourFilterDefinition>
      </regionDefinition>

      <!-- List of all the commands the ASM can support in this mode -->
      <command name="Request" units="" completionTime="" completionTimeUnits="" />
    </taskDefinition>

  </modeDefinition>
</SensorRegistration>
```

## A.2 Status Report Message

```
<?xml version="1.0"?>
<StatusReport>
<timestamp> </timestamp>
<sourceID></sourceID>
<reportID></reportID>
<system> </system>
<info> </info>
<activeTaskID></activeTaskID>
<power source = "" status = "" level="" />
<mode> </mode>

<sensorLocation>
  <location>
      <X></X> <!-- Eastings -->
      <Y></Y> <!-- Northings -->
      <Z></Z> <!-- altitude -->
      <eX></eX> <!-- GPS error -->
      <eY></eY> <!-- GPS error -->
      <eZ></eZ> <!-- Altitude error -->
    </location>
</sensorLocation>

<fieldOfView>
<!-- provide location list or range bearing -->
<locationList>
  <location> </location>
</locationList>
<!-- alternatively describe field of view as a cone -->
<rangeBearingCone>
     <R></R>
      <Az></Az>
      <Ele></Ele>
      <hExtent> </hExtent>
      <vExtent> </vExtent>
      <eR> </eR>
      <eAz> </eAz>
      <eEle> </eEle>
      <ehExtent> </ehExtent>
      <evExtent> </evExtent>
 </rangeBearingCone>
</fieldOfView>

<coverage>
<!-- provide location list or range bearing -->
<locationList>  </locationList>
<rangeBearingCone> </rangeBearingCone>
</coverage>

<!-- list of obscuration regions -->
<obscuration>
<!-- provide location list or range bearing -->
<locationList>  </locationList>
<rangeBearingCone> </rangeBearingCone>
</obscuration>

<status level = "" type = "" value = "" />

</StatusReport>
```

## A.3 Detection Message

```xml
<?xml version="1.0"?>
<DetectionReport>
<timestamp> </timestamp>
<sourceID></sourceID>
<reportID></reportID>
<objectID></objectID>
<taskID></taskID>
<state> </state>

<!-- provide location or range bearing -->
<location>
<X></X> <Y></Y> <Z></Z> <eX></eX> <eY></eY> <eZ></eZ>
</location>

<rangeBearing>
<R></R> <Az></Az> <Ele></Ele> <Z></Z>

<eR></eR> <eAz></eAz> <eEle></eEle> <eZ></eZ
</rangeBearing>

<detectionConfidence> </detectionConfidence>

<trackInfo type=" " value = " " e = " "/>

<predictedLocation>
     <!-- provide location or range bearing -->
     <location>  </location>
     <rangeBearing> </rangeBearing>
     <predictionTimestamp> </predictionTimestamp>
</predictedLocation>

<objectInfo type="  " value = " " e = " " />

<colour>  </colour>

<class type=" ">
     <confidence> </confidence>

     <subClass level = "1" type=" " value = " ">
          <confidence> </confidence>
          <subClass level = "2" type=" " value = " ">
               <confidence> </confidence>
               <subClass level = "3" type=" " value = " ">
               <confidence> </confidence>
               </subClass>
          </subClass>
     </subClass>
</class>

<behaviour type="">
  <confidence> </confidence>
 </behaviour>

<associatedFile type=" " url=""/>
</DetectionReport>
```

## A.4 Task Message

```xml
<?xml version="1.0"?>
<SensorTask>
<timestamp> </timestamp>
<sensorID></sensorID>
<taskID></taskID>
<taskName> </taskName>
<taskDescription></taskDescription>
<taskStartTime> </taskStartTime>
<taskEndTime> </taskEndTime>
<control> </control>

<region>
  <type> </type>
  <regionID></regionID>
  <regionName> </regionName>

  <!-- provide location list or range bearing cone-->
  <locationList></locationList>
  <rangeBearingCone></rangeBearingCone>

  <!-- list of class Filters to apply to region -->
  <!-- if omitted then no filtering is applied -->
  <classFilter type=" " priority=" ">
  <parameter name=" " operator=" " value="" />
     <subClassFilter level = "1" type=" " value=" " priority=" ">
     <parameter name=" " operator=" " value="" />
          <subClassFilter level = "2" type=" " value = "" priority=" ">
          <parameter name=" " operator=" " value="" />
               <subClassFilter level = "3" type=" " value = "">
               <parameter name=" " operator=" " value="" />
               </subClassFilter>
          </subClassFilter>
     </subClassFilter>
  </classFilter>

  <behaviourFilter type = "" priority=" ">
  <parameter name="e" operator=" " value="" />
  </behaviourFilter>
</region>

<command>
     <request> </request>
     <detectionThreshold> </detectionThreshold>
     <detectionReportRate> </detectionReportRate>
     <classificationThreshold> </classificationThreshold>
     <mode></mode>

     <lookAt>
          <!-- provide location list or range bearing cone-->
          <locationList></locationList>
          <rangeBearingCone></rangeBearingCone>
     </lookAt>
     <lookFor type="" url=" " />
</command>

</SensorTask>
```

## A.5 Alert Message

```xml
<?xml version="1.0"?>
<Alert>
<timestamp> </timestamp>
<sourceID></sourceID>
<alertID></alertID>
<alertType> </alertType>
<status> </status>
<description> </description>
<location>  <X></X> <Y></Y> </location>
<rangeBearing> <R></R> <Az></Az></rangeBearing>
<regionID></regionID>
<priority></priority>
<ranking></ranking>
<confidence></confidence>

  <associatedFile type="" url="" />
  <associatedDetection timestamp=" " sourceID="" objectID="" />
</Alert>
```

## A.6 Registration Acknowledgement Message

```xml
<?xml version="1.0"?>
<SensorRegistrationACK>
<sensorID></sensorID>
</SensorRegistrationACK>
```

## A.7 Task Acknowledgement Message

```xml
<?xml version="1.0"?>
<SensorTaskACK>
<timestamp> </timestamp>
<sensorID></sensorID>
<taskID></taskID>
<status> </status>
<reason> </reason>
<associatedFile type="" url="" />
</SensorTaskACK>
```

## A.8 Alert Response Message

```xml
<?xml version="1.0"?>
<AlertResponse>
<timestamp> </timestamp>
<sourceID></sourceID>
<alertID></alertID>
<status> </status>
<reason></reason>
</AlertResponse>
```

## A.9 Error Message

```xml
<?xml version="1.0"?>
<Error>
<timestamp> </timestamp>
<packet> </packet>
<errorMessage> </errorMessage>
</Error>
```

# B Cross Reference Table for XML fields

For completeness, this appendix contains all the valid xml fields, arranged in tabular form; with how many times each field should be used. The final column gives the field in the registration message required if the field is to be supplied in other messages.
Key:
M* = either location or rangeBearing must be supported. Supporting both is unnecessary in this mandatory field
M** = these fields are mandatory if the region element is supported
M*** = default commands that all ASMs should support
O* = either location or rangeBearing must be supported. Supporting both is unnecessary in this optional field

| **DetectionReport** | **Type** | **Units** | **Number of Occurances** | **Required field in Registration message** |
|---|---|---|---|---|
| timestamp | xs:dateTime | time | 1 | |
| sourceID | xs:int | none | 1 | |
| reportID | xs:long | none | 1 | |
| objectID | xs:long | none | 1 | |
| taskID | xs:int | none | 1 | |
| | | | | |
| state | xs:string | none | 0-1 | detectionReport category="object" type="state" |
| location: x,y,z,eX,eY,eZ | xs:doubles | metres | 0-1, M* | locationType |
| rangeBearing: R, Az, Ele, Z, eR, eAz, eEle, eZ | xs:doubles | decimal degrees-metres | 0-1, M* | locationType |
| detectionConfidence | xs:float | probability | 0-1 | detectionReport category="detection" type="confidence" |
| trackInfo:confidence | xs:float | probability | 0-1 | detectionReport category="track" type="confidence" |
| trackInfo:speed | xs:float | m-s | 0-1 | detectionReport category="track" type="speed" |
| trackInfo:az | xs:float | degrees | 0-1 | detectionReport category="track" type="az" |
| trackInfo:dR | xs:float | metres | 0-1 | detectionReport category="track" type="dR" |
| trackInfo:dAz | xs:float | degrees | 0-1 | detectionReport category="track" |

| | | | | |
|---|---|---|---|---|
| | | | | type="dAz" |
| predictedLocation: location: x,y,z,eX,eY,eZ | xs:doubles | metres | 0-1, O* | detectionReport category="track" type="predictedLocation" |
| predictedLocation: rangeBearing: R, Az, Ele, Z, eR, eAz, eEle, eZ | xs:doubles | decimal degrees-metres | 0-1, O* | detectionReport category="track" type="predictedLocation" |
| predictedLocation: predictionTimestamp | xs:dateTime | time | 0-1 | detectionReport category="track" type="predictionTimestamp" |
| objectInfo: dopplerSpeed | xs:float | m-s | 0-1 | detectionReport category="object" type="dopplerSpeed" |
| objectInfo: dopplerAz | xs:float | degrees | 0-1 | detectionReport category="object" type="dopplerAz" |
| objectInfo:majorLength | xs:float | metres | 0-1 | detectionReport category="object" type="majorLength" |
| objectInfo:majorAxisAz | xs:float | degrees | 0-1 | detectionReport category="object" type="majorAxisAz" |
| objectInfo:minorLength | xs:float | metres | 0-1 | detectionReport category="object" type="minorLength" |
| objectInfo:height | xs:float | metres | 0-1 | detectionReport category="object" type="height" |
| colour | xs:string | none | 0-1 | detectionReport category="object" type="colour" |
| **class** | | | 0-unbounded | **detectionClassDefinition: classDefinition** |
| type | xs:string | none | 1 | type |
| confidence | xs:float | probability | 1 | confidence |
| **subclass** | | | 0-unbounded | **detectionClassDefinition: subClassDefinition** |
| level | xs:int | none | 1 | level |
| type | xs:string | none | 1 | type |
| value | xs:string | none | 1 | values |
| confidence | xs:float | probability | 1 | confidence |
| **behaviour** | | | 0-unbounded | **behaviourDefinition** |
| type | xs:string | none | 1 | type |

| | | | | |
|---|---|---|---|---|
| confidence | xs:float | probability | 1 | confidence |
| **associatedFile** | | | 0-unbounded | detectionReport category="associatedFile" type="image" |
| type | xs:string | none | 1 | |
| url | xs:string | none | 1 | |
| | | | | |
| **StatusReport** | | | | |
| timestamp | xs:dateTime | time | 1 | |
| sourceID | xs:int | none | 1 | |
| reportID | xs:long | none | 1 | |
| system | xs:string | none | 0-1 | |
| info | xs:string | none | 0-1 | |
| activeTask | xs:string | none | 0-1 | |
| **power** | | | 0-1 | |
| source | xs:string | none | 1 | heartbeatReport category="power" type="status" |
| status | xs:string | none | 1 | heartbeatReport category="power" type="status" |
| level | xs:int | percentage | 0-1 | heartbeatReport category="power" type="level" |
| mode | xs:string | none | 0-1 | heartbeatReport category="mode" |
| sensorLocation: location: x,y,z,eX,eY,eZ | xs:doubles | metres | 0-1 | heartbeatReport category="sensor" type="sensorLocation" |
| **fieldOfView** | | | 0-1 | heartbeatReport category="sensor" type="fieldOfView" |
| locationList: location: x,y,z,eX,eY,eZ | xs:doubles | metres | 0-1, O* | |
| rangeBearingCone: R,Az,Ele,hExtent,vExtent,minR,eR,eAz,eEle,ehExtent,evExtent,eminR | xs:doubles | decimal degrees-metres | 0-1, O* | |
| **coverage** | | | 0-1 | heartbeatReport category="sensor" type="coverage" |
| locationList: location: x,y,z,eX,eY,eZ | xs:doubles | metres | 0-1, O* | |

| | | | | |
|---|---|---|---|---|
| rangeBearingCone: R,Az,Ele,hExtent,vExtent,minR,eR,eAz,eEle,ehExtent,evExtent.eminR | xs:doubles | decimal degrees-metres | 0-1, O* | |
| | | | | |
| **obscuration** | | | 0-unbounded | heartbeatReport category="sensor" type="obscurations" |
| locationList: X,Y,Z | xs:doubles | metres | 0-1, O* | |
| rangeBearingCone: R Az,Ele, hExtent, vExtent,minR | xs:doubles | decimal degrees-metres | 0-1, O* | |
| **status** | | | 0-unbounded | heartbeatReport category="status" |
| level | xs:string | none | | |
| type | xs:string | none | | |
| value | xs:string | none | | |
| | | | | |
| **SensorTask** | | | | |
| timestamp | xs:dateTime | time | 1 | |
| sensorID | xs:int | none | 1 | |
| taskID | xs:int | none | 1 | |
| taskName | xs:string | none | 0-1 | |
| taskDescription | xs:string | none | 0-1 | |
| taskStartTime | xs:dateTime | time | 0-1 | |
| taskEndTime | xs:dateTime | time | 0-1 | |
| control | xs:string | none | 1 | |
| **region** | | | 0-unbounded | |
| type | xs:string | none | 1, M** | member of regionType list |
| regionID | xs:int | none | 1, M** | |
| regionName | xs:string | none | 0-1 | |
| locationList: location: x,y,z,eX,eY,eZ | xs:doubles | metres | 0-1, O* | |
| rangeBearingCone: R,Az,Ele,hExtent,vExtent,minR | xs:doubles | decimal degrees-metres | 0-1, O* | |
| **classFilter** | | | 0-unbounded | **classFilterDefinition** |

| | | | | |
|---|---|---|---|---|
| Type | xs:string | none | 1 | type |
| **Parameter** | | | 1 | **filterParameter** |
| Name | xs:string | none | 1 | name |
| Operator | xs:string | none | 1 | operators |
| Value | float | type dependent | 1 | |
| **subClassFilter** | | | 0-unbounded | **subClassFilterDefinition** |
| level | xs:int | none | 1 | level |
| type | xs:string | none | 1 | type |
| value | xs:string | none | 1 | |
| **parameter** | | | 1 | **filterParameter** |
| name | xs:string | None | 1 | name |
| operator | xs:string | None | 1 | operators |
| value | xs:string | type dependent | 1 | |
| **behaviourFilter** | | | 0-unbounded | **behaviourFilterDefinition** |
| type | xs:string | None | 1 | type |
| **parameter** | | | 1 | **filterParameter** |
| name | xs:string | None | 1 | name |
| operator | xs:string | None | 1 | operators |
| value | float | type dependent | 1 | |
| **command** | | | 0-1 | |
| request | xs:string | None | M*** | command name="Request" |
| detectionThreshold | xs:string | None | M*** | command name="DetectionThreshold" |
| detectionReportRate | xs:string | None | M*** | command name="DetectionReportRate" |
| classificationThreshold | xs:string | None | | command name="ClassificationThreshold" |
| mode | xs:string | None | | command name="Mode" |
| **lookAt** | | | | command name="LookAt" |
| locationList: location: x,y,z,eX,eY,eZ | xs:doubles | Metres | 0-1, O* | |

| | | | | |
|---|---|---|---|---|
| rangeBearingCone: R,Az,Ele,hExtent,vExtent,minR,eR,eAz,eEle,ehExtent,evExtent,eminR | xs:doubles | decimal degrees-metres | 0-1, O* | |
| **lookFor** | | | 0-1 | command name="LookFor" |
| Type | xs:string | None | 1 | |
| url | xs:string | None | 1 | |

# C SAPIENT Demonstration System Implementation -Specific Information

## C.1 SAPIENT Demonstration System -Specific Architecture

This section describes the architecture deployed on the SAPIENT Integration events, trials and demonstrations to date.

### C.1.1 Physical Connection

A single 100BaseT, IPv4 Ethernet connection is provided for each ASM onto the Lead Systems Integrator (LSI) network. 1000BaseT equipment may also be used if available. The LSI will assign a single IP address for each ASM to use on the LSI-supplied network.
The use of Fibre and Wireless Ethernet communication has also been demonstrated.

### C.1.2 Firewalls and Network Security

Following discussion with the Dstl Security Accreditor, as long as the SAPIENT network is not connected to other MOD systems and is a stand-alone network i.e. not connected to the internet, then standard good IT practice i.e. use of Firewall and Antivirus software is sufficient security measures. Each instantiation of the SAPIENT system will be approved by the Security Accreditor.

For accreditation to higher classifications, more secure network architecture will be implemented.

### C.1.3 Data Interface

The main data interface between the ASMs and the LSI-supplied Data Agent will be a TCP socket. The Data Agent will be the server, the ASM the client. The only valid data will be XML messages as defined in this document. The only exceptions are certain additional data files that are shared on a File Share described below. It is the ASM's responsibility to monitor the connection and attempt to reconnect if it detects a loss of connection. If the ASM is unable to connect to the Data Agent, then it should attempt to retry every 10 seconds until it is successful.

A separate instance of the Data Agent is deployed for each instance of an ASM. This will allow the Data Agents to be spread across multiple servers if required.

The IP address and port of the Data Agent will be provided by the LSI at integration time. DHCP may be used as appropriate.

A suggested TCP port range is 14001-14100 for the ASM Data Agents to listen on. This range is not expected to conflict with other applications, but if a conflict is found an alternative will be assigned as required.

A single listening port is used for each Data Agent. Ephemeral Ports as assigned by the server and client operating systems are used to complete the connections.

It is recommended that the TCP_NO_DELAY flag is set for all socket connections to minimise the latency through the system.

### C.1.4 File Share

The File Share was implemented in the SAPIENT project using a specific approach discussed in Appendix C.5.

### C.1.5 Database

Postgres 9.x has been used for all the SAPIENT trials and demonstrations to date.

### C.1.6 Sensor ID Allocation

Whilst the system can support dynamic sensor ID allocation, in practice system deployment, data management and testing is simplified by pre-allocating sensor IDs and IP ports to the ASMs connected to the system.

### C.1.7 Sensor Location

For sensors without GPS, the LSI manually provides a location to the installer of the ASM to provide in the ASM heart beat message. The location may be derived from physical survey or from the Map GUI.

### C.1.8 System Clock

The LSI provides a Network Time Protocol (NTP) Server on the SAPIENT network. All modules will be expected to synchronise with this at regular intervals during system operation. ASMs are expected to synchronise with the NTP server as often as required in order to maintain a clock drift of no more than 100ms. This should be possible assuming the network devices can support it. It is anticipated that an interval of between 15 and 60 minutes between synchronisation requests for each module should be sufficient to maintain time accuracy. Given the variable clock-drift of different devices, this interval should be chosen as appropriate by the ASM supplier.

### C.1.9 Power Supplies

Individual modules may be either battery powered (batteries to be supplied by and remain the responsibility of the module supplier) or mains powered. If mains powered, it is recommended dual voltage compatible (230V / 110V) power supplies are used.

## C.2 System Architecture

The LSI has responsibility for deploying and configuring the network infrastructure so that the ASM and HLDMM components can connect to the wider system.

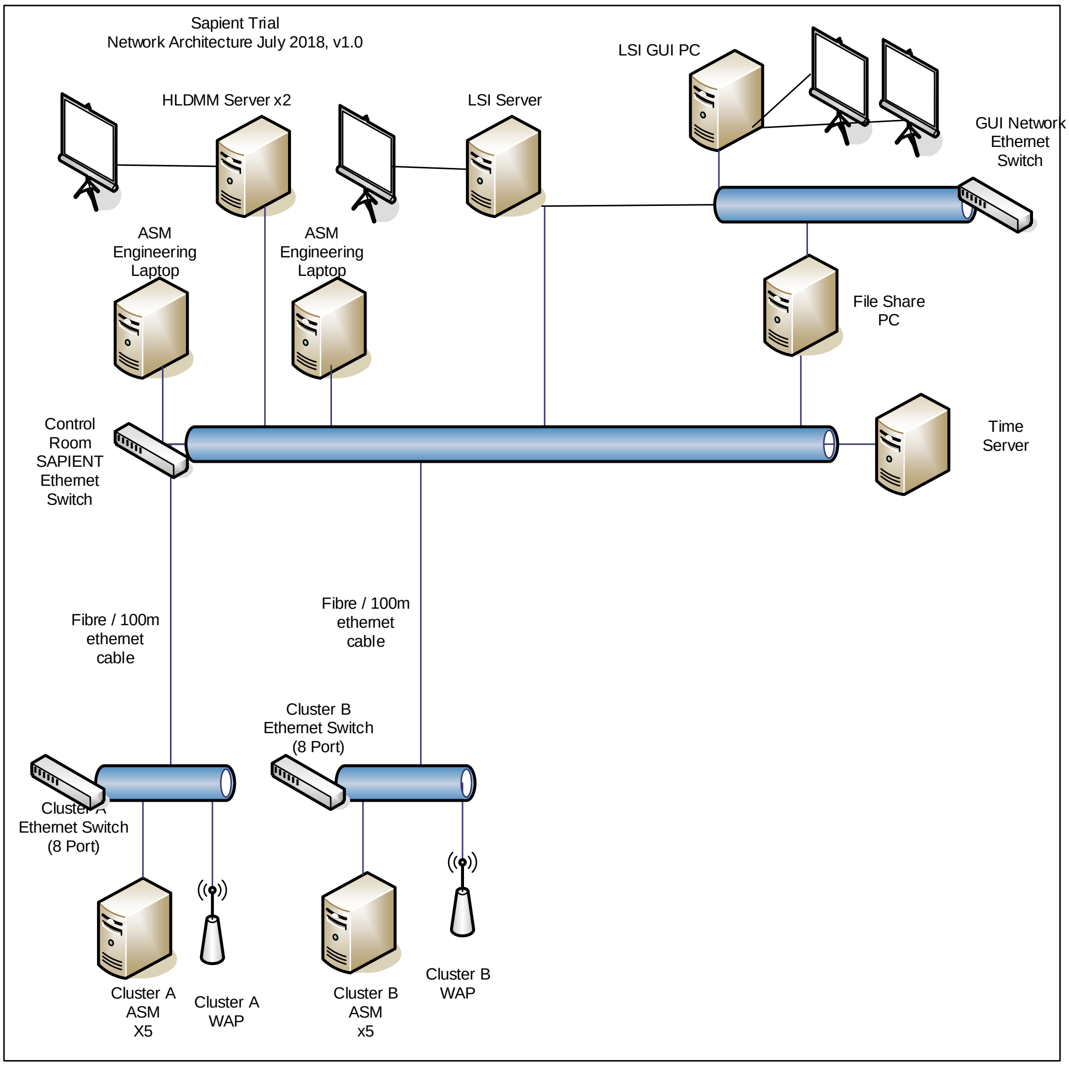


*Figure 4: SAPIENT Trial System Architecture*

Figure 4 shows the system architecture of the SAPIENT demonstration system.

## C.2.1 LSI Server

This is a PC for hosting the SAPIENT Middleware i.e. the Data Agent software and database for several or all of the ASMs. If required for performance reasons, the Data Agents can be split across multiple VMs.

## C.2.2 ASM Engineering PCs

These are PCs required for the ASMs for set up and configuration

## C.2.3 HLDMM Server

This is a PC for hosting the HLDMM software.

### C.2.4 GUI PC

This is a Workstation PC with dual displays. The first display will show a map-based situational awareness picture overlaid with sensor detections and alerts. The second display will show an alert list and text messages from the HLDMM.

## C.3 Implementation Specific requirements

### C.3.1 Sensor Locations

These are preferred in UTM coordinates although GPS coordinates are also acceptable.

The current HLDMM and GUI require sensor location to be reported in all heartbeat messages.

### C.3.2 Sensor Coverage and Field Of View

The current HLDMM and GUI require one of these to be reported by the ASM in the heartbeat message. The sensor region field part of the message is currently ignored.

## C.4 ASM Database Configuration

### C.4.1 Demonstration Configuration

For the SAPIENT Phase 2 Demonstration, a PostgreSQL database has been used. This provides a cost-effective solution for demonstration systems requiring a database.

Using an alternative database for future deployments would not have any effect on the interfaces to the ASMs. The only interface change would be for the HLDMM querying the ASM data.

### C.4.2 Database Server Installation

PostgreSQL should be installed following the instructions included with the installation media. Typically this will include setting up the database server to listen on TCP port 5432 and setting up a database administration user called ‘postgres’.

### C.4.3 Database Creation

Running an instance of the Data Agent for the first time will create the required database tables for the ASM information.

### C.4.4 HLDMM Database access

An alternative user account with read-only access to the database should be created for the HLDMM to use.

The HLDMM should then be able to read the ASM data from the database as a number of standard SQL queries as outlined below.

## C.5 File Share

The LSI provides one or more file shares for sending additional data files from the ASMs to the GUI. This is intended solely to provide the system operator with additional information such as image snapshots that they would need to respond correctly to an alert generated by the system. It is intended that the autonomous behaviour of the system should be encapsulated within the XML messages of the main data interface.

To enable flexible and well-known file sharing across different platforms, file sharing is implemented using Samba on an Ubuntu Linux machine. This enables the ASMs that are running Linux based operating systems and the HLDMM and GUI that are running Windows operating systems to easily share files.

Currently File Sharing is only used for sharing image snapshots from the ASMs to the GUI. A shared folder called ‘Images’ is set up on a networked PC visible to both the ASMs and the GUI. Within this folder there is a sub-folder for each ASM named ‘sensor_N’ where N is the sensor identifier of the ASM. The use of sub-folders is to allow simpler file management than one big folder. The sub-folder name shall be included in any urls sent in detection or other messages so that the url can be re-used in fused messages that may not be associated with the original ASM.

## C.6 Cubica HLDMM Specific Constraints

### C.6.1 Reporting Classes, Sub-classes and Behaviour

The Cubica HLDMM prefers ASM suppliers to provide multiple class probabilities, rather than single class as described in section 6.4.5.

### C.6.2 Status Report Messages

The Cubica HLDMM expects ASMs to send a Status Report with a status of ‘New’ following registration; an ‘Unchanged’ status without a previous ‘New’ would be effectively meaningless.

### C.6.3 Bearings Only Detection Work around

The advice given in Appendix E should be followed for bearings-only sensors.

# D SAPIENT Taxonomy

The successful operation of a SAPIENT Middleware system relies on defining a taxonomy of object classes, behaviours and other characteristics. The taxonomy shall define a set of descriptive nouns such that they are mutually exclusive for the purposes of fusion.

## D.1 Object Classification Taxonomy

It is up to each ASM supplier to define the list of class types that they can classify.

The following is the current core list:

- Human
- Vehicle
- Air Vehicle
- Animal
- Static Object
- Unknown

Unknown means 'I am certain this is a class I don't recognise' rather than 'I can't tell what class this is'.

Vehicle can be used when the ASM cannot discriminate between different types of vehicle. Alternatively it can be used as part of a class hierarchy if multiple vehicle types can be discriminated.

### D.1.1 Human Sub Class Types

The Human class can have the following sub class types:

- Male
- Female
- Person ID (e.g. Name or Staff Pass Number)

### D.1.2 Vehicle Sub Class Types

The Vehicle class can have the following sub classes:

| Sub Class Level 1 | Sub Class Level 2 | Sub Class Level 3 |
|---|---|---|
| 4 Wheeled | Heavy | Artic, Truck, Bus, Tracked Vehicle, Military Vehicle |
| | Medium | Van, People Carrier, 4X4 |
| | Light | Car, Unmanned Ground Vehicle (UGV) |
| 2 Wheeled | | Motorbike |
| | | Bicycle |

### D.1.3 Air Vehicle Sub Class Types

The Air Vehicle related classes can have the following sub classes:

| Class | Sub Class Level 1 | Sub Class Level 2 | Sub Class Level 3 |
|---|---|---|---|
| Air Vehicle | Fixed Wing | Airliner, Jet, Light | |
| | Rotor | Helicopter, Chinook | |

| Class | Sub Class Level 1 | Sub Class Level 2 | Sub Class Level 3 |
|---|---|---|---|
| | Fixed Wing UAV | | |
| | Rotor UAV | Quadcopter, Hexcopter | DJI Phantom 3, 4 etc. |
| Ground station | | | |

D.1.4 RF Communications Sub Class Types

The following, non-exhaustive list of sub-types for RF Communications sensors have been used:

| Class | Sub Class Level 1 | Sub Class Level 2 |
|---|---|---|
| RF Comms | SignalType: Video,ControlOr Telemetry | SignalSource:UAS, Controller |

## D.2 Object Behaviour Taxonomy

Where object behaviour can be detected, this is reported independently of object type using the behaviour types. Any confidence values associated with the behaviour should be expected to relate to the object class type with the highest confidence. E.g. If an object is detected as person with a probability of 0.6 and animal with a probability of 0.3, any behaviour probability is expected to be in relation to it being a person.

The following is the current core list of behaviours. This list can be easily extended in future releases of the ICD. Not all of these behaviours will necessarily be supported on a given system. This is likely to be implementation specific.

- Walking
- Running
- Crawling
- Climbing
- Digging
- Throwing
- Loitering
- Active
- Passive

# E Line of Bearing Sensors

Some sensors are capable of providing a Line of Bearing (LoB) to a target but are unable to provide the distance to the target. These are potentially valuable observations that may otherwise be ignored because the sensor cannot fully describe the location of the target. In these situations it is recommended that, as a default, the target is reported as being at half the maximum distance the sensor could possibly recognise the target, with an error of half the maximum distance. For example, if the maximum distance a sensor could possibly recognise a person was 1000 meters, the person should be reported as being at a distance of 500 meters, with an error on the observation of 500 meters. Obviously where ASMs have an approach which refines this estimate in some way, they should use that.

# Report documentation page

| | | | |
|---|---|---|---|
| Originator's Report Number | QINETIQ/TIS/C4ISR/ICD1400047 | | |
| Originator's Name and Location | David Faulkner, QinetiQ Ltd, Malvern Technology Centre, St Andrews Road, Malvern, Worcs, WR14 3PS | | |
| Customer Contract Number and Period Covered | | | |
| Customer Sponsor's Post/Name and Location | P. Thomas, S Colley, Dstl Porton | | |
| Report Protective Marking and any other markings<br>OFFICIAL | Date of issue<br>11th May 2020 | Pagination<br>Cover + 101 | |
| Report Title<br>SAPIENT Interface Control Document | | | |
| Translation / Conference details (if translation give foreign title / if part of conference then give conference particulars)<br>N/A | | | |
| Title Protective Marking | N/A | | |
| Authors | Dr G.F. Marshall, D.A.A Faulkner | | |
| Downgrading Statement | Enter downgrading statement | | |
| Secondary Release Limitations | Enter secondary release limitations | | |
| Announcement Limitations | Enter announcement limitations | | |
| Keywords / Descriptors | SAPIENT, SAPIENT, ICD, interface description | | |
| Abstract<br><br>This document is the Interface Control Document (ICD) for the SAPIENT project. It defines the interfaces between the High Level Decision Making Module (HLDMM), the Autonomous Sensor Modules (ASM) and the SAPIENT Middleware. It also defines how a Graphical User Interface (GUI) such as that provided by the Lead Systems Integrator for the initial SAPIENT demonstration may ingest the output of the system. The purpose of this document is to enable individual module developers to build component modules compatible with the overall system. For this issue of the ICD, the document has been re-organised and rewritten based on 5 years of experience of running SAPIENT and feedback from suppliers with the intention that it is clearer and easier to maintain by the SAPIENT Interface Management Panel (SIMP). | | | |
| Abstract Protective Marking: | OFFICIAL | | |

This form meets DRIC-SPEC 1000 issue 7

**Blank page**