Unmanned Maritime Autonomy Architecture (UMAA)
Maneuver Operations (MO)
Interface Control Document (ICD)
(UMAA-SPEC-MOICD)

MDE Version 5.0.1

UMAA Spec Commit 63fb9f9

Version 5.2.1
20 April 2022

Contents

List of Figures

List of Tables

1 Scope

1.1 Identification

This document defines a set of services as part of the Unmanned Maritime Autonomy Architecture (UMAA). The services and their corresponding interfaces covered in this ICD encompass the functionality to control and maneuver an Unmanned Maritime Vehicle (UMV) (surface or undersea). As such, it includes the commands and status to/from an unmanned vehicle's control systems for controlling all aspects of maneuvering and its associated dynamics. This includes both low-level controls such as heading and speed, as well as higher-level behaviors for traversing waypoints. This ICD also includes instructions for managing driving constraints such as setting bounds on desired speed range or setting a desired maximum turn rate. This document is generated automatically from data models that define its services and their interfaces as part of the Unmanned Systems (UxS) Control Segment (UCS) Architecture as extended by UMAA to provide autonomy services for unmanned vehicles.

To put each ICD in context of the UMAA Architecture Design Description (ADD), the UMAA functional decomposition mapping to UMAA ICDs is shown in Figure 1.

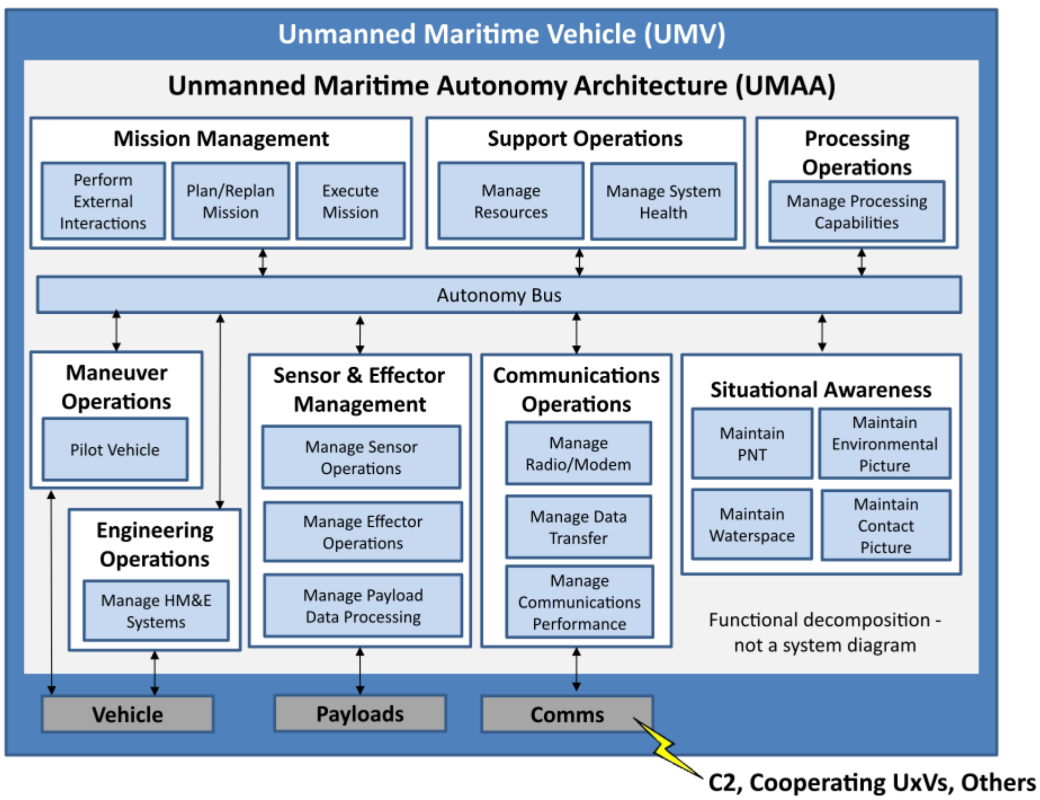

Figure 1: UMAA Functional Organization.

1.2 Overview

The fundamental purpose of UMAA is to promote the development of common, modular, and scalable software for unmanned vehicles that is independent of a particular autonomy implementation. Unmanned Maritime Systems (UMSs) consist of Command and Control (C2), one or more unmanned vehicles, and support equipment and software (e.g. recovery system, Post Mission Analysis applications). The scope of UMAA is focused on the autonomy that resides on-board the unmanned vehicle. This includes the autonomy for all classes of unmanned vehicles and must support varying levels of communication in mission (i.e., constant, intermittent, or none) with external systems. To enable modular development and upgrade of the functional capabilities of the on-board autonomy, UMAA defines eight high-level functions. These core functions include: Communications Operations, Engineering Operations, Maneuver Operations, Mission Management, Processing Operations, Sensor and Effector Operations, Situational Awareness, and Support Operations. In each of these areas, it is anticipated that new capabilities will be required to satisfy evolving Navy missions over time. UMAA seeks to define standard interfaces for these functions so that individual programs can leverage capabilities developed to these standard interfaces across programs that meet the standard interface specifications. Individual programs may group services and interfaces into components in different ways to serve their particular vehicle's needs. However, the entire interface defined by UMAA will be required as defined in the ICDs for all services that are included in a component. This requirement is what enables autonomy software to be ported between heterogeneous UMAA-compliant vehicles with their disparate vendor-defined vehicle control interfaces without recoding to a vehicle-specific interface.

Maneuver Operations defines the services required to drive an unmanned vehicle. Figure 2 depicts an example of various levels of maneuvering behaviors in relation to navigation sensing and Hull, Mechanical, & Electrical (HM&E) control services provided in separate ICDs. Figure 2 depicts an example of possible component service groupings (designated by dashed lines).

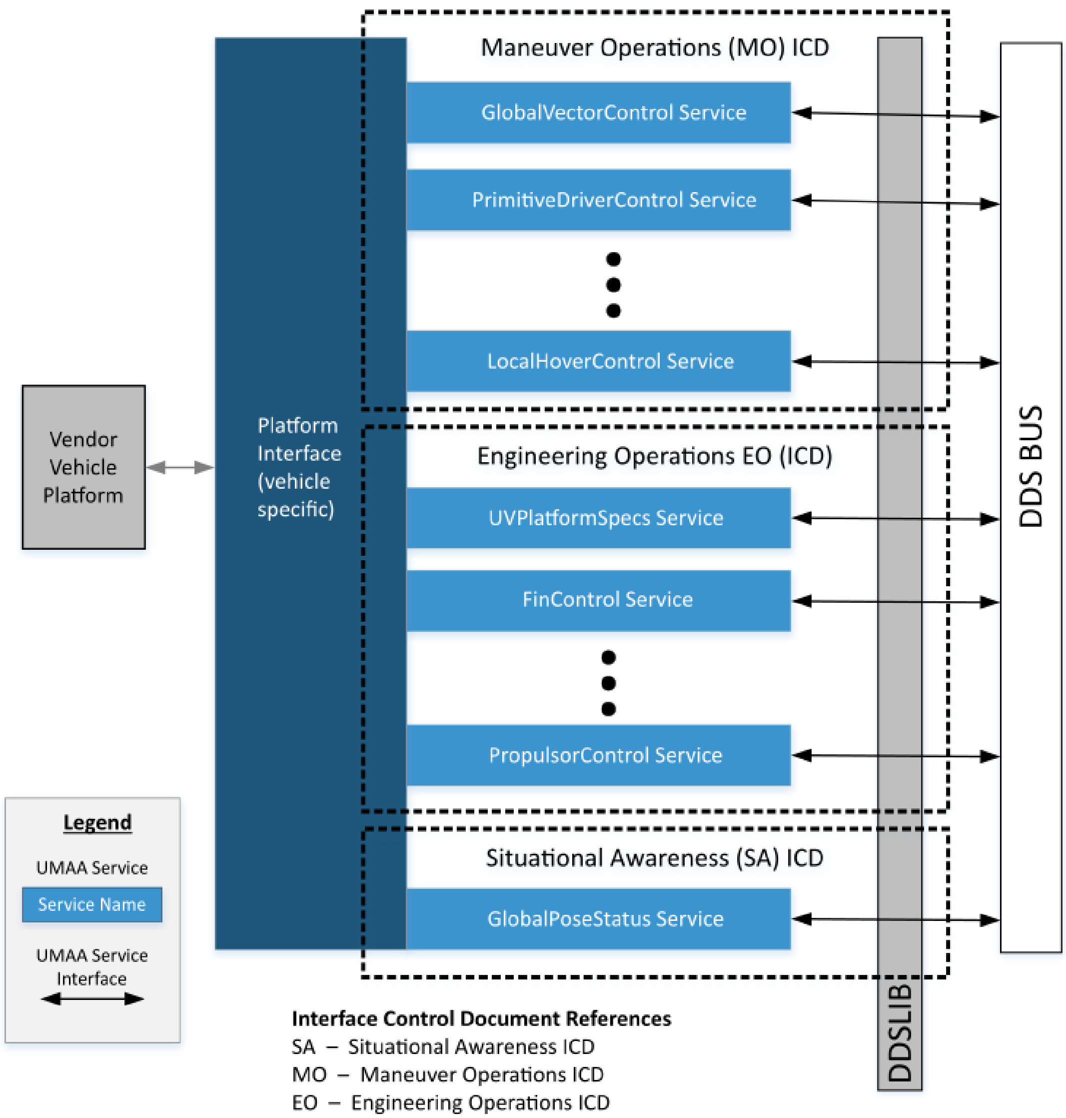

Figure 2: UMAA Services and Interfaces Example.

1.3 Document Organization

This interface control document is organized as follows:

Section 1 – Scope: A brief purview of this document

Section 2 – Referenced Documents: A listing of associated of government and non-government documents and standards

Section 3 – Introduction to Data Model, Services, and Interfaces: A description of the common data model across all services and interfaces

Section 4 – Introduction to Coordinate Reference Frames and Position Model: An overview of the reference frame model used by UMAA

Section 5 – Flow Control: A description of different flow control patterns used throughout UMAA

Section 6 – Maneuver Operations (MO) Services and Interfaces: A description of specific services and interfaces for this ICD

2 Referenced Documents

The documents in the following table were used in the creation of the UMAA interface design documents. Not all references may be applicable to this particular document.

Table 1: Standards Documents

Title	Release Date
A Universally Unique IDentifier (UUID) URN Namespace	July 2005
Data Distribution Service for Real-Time Systems Specification, Version 1.4	March 2015
Data Distribution Service Interoperability Wire Protocol (DDSI-RTPS), Version 2.3	April 2019
Object Management Group Interface Definition Language Specification (IDL)	March 2018
Extensible and Dynamic Topic Types for DDS, Version 1.3	February 2020
UAS Control Segment (UCS) Architecture, Architecture Description, Version 2.4	27 March 2015
UCS Architecture, Conformance Specification, Version 2.2	27 September 2014
UCS-SPEC-MODEL v3.4 Enterprise Architect Model	27 March 2015
UCS Architecture, Architecture Technical Governance, Version 2.5	27 March 2015
System Modeling Language Specification, Version 1.5	May 2017
Unified Modeling Language Specification, Version 2.5.1	December 2017
Interface Definition Language (IDL), Version 4.2	March 2018
U.S. Department Of Homeland Security, United States Coast Guard "Navigation Rules International-Inland" COMDTINST M16672.2D	March 1999
IEEE 1003.1-2017 - IEEE Standard for Information Technology–Portable Operating System Interface (POSIX(R)) Base Specifications, Issue 7	December 2017
Guard, U. C. (2018). Navigation Rules and Regulations Handbook: International—Inland. Simon and Schuster.	June 2018
Department of Defense Interface Standard: Joint Military Symbology (MIL-STD-2525D Appendix A)	10 June 2014
DOD Dictionary of Military and Associated Terms	August 2018

Table 2: Government Documents

Title	Release Date
Unmanned Maritime Autonomy Architecture (UMAA) Architecture Design Description (ADD), Version 1.0	January 2019
Manual for the Submission of Oceanographic Data Collected by Unmanned Undersea Vehicles (UUVs)	October 2018

3 Introduction to Data Model, Services, and Interfaces

3.1 Data Model

A common data model is at the heart of UMAA. The common data model describes the entities that represent system state data, the attributes of those entities and relationships between those entities. This is a "data at rest" view of system-level information. It also contains data classes that define types of messages that will be produced by components, or a "data in motion" view of system-level information.

The common data model and coordinated service interfaces are described in a Unified Modeling Language (UML^{TM}) modeling tool and are represented as UML^{TM} class diagrams. Interface definition source code for messages/topics and other interface definition products and documentation will be automatically generated from the common data model so that they are consistent with the data model and to ensure that delivered software matches its interface specification.

The data model is maintained as a Multi-Domain Extension (MDE) to the UCS Architecture and will be maintained under configuration control by the UMAA Board as UCSMDE and will be incrementally integrated into the core UCS standard. Section 6 content is automatically generated from this data model, as are other automated products such as IDL that are used for automated code generation.

3.2 Definitions

UMAA ICDs follow the UCS terminology definitions found in the UCS Architecture Description v2.4. The normative (required) implementation to satisfy the requirements of a UMAA ICD is to provide service and interface specification compliance. Components may group services and required interfaces in any manner so long as every service meets its interface specifications. Figure 3 shows a particular grouping of services into components. The interfaces are represented by the blue and green lines and may equate to one or more independent input and output interfaces for each service. The implementation of the service into software components is left up to the individual system development. Given this context, section 6 correspondingly defines services with their interfaces and not components.

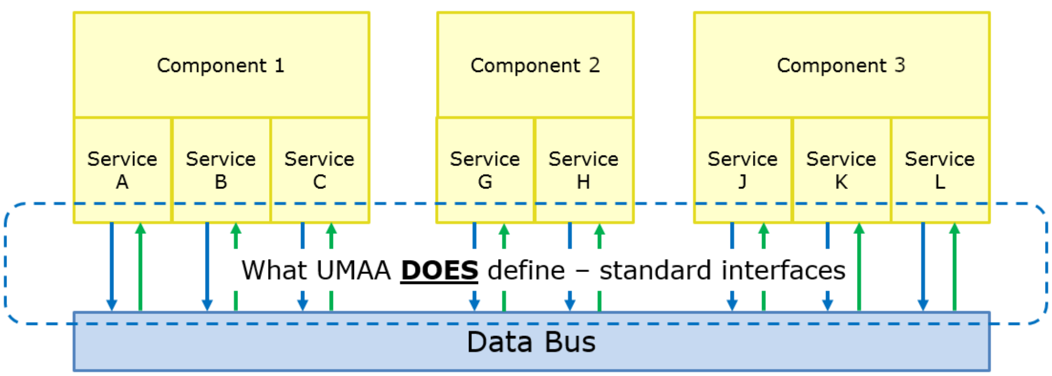

Figure 3: Services and Interfaces Exposed on the UMAA Data Bus.

Services may use other services within this ICD, or in other UMAA defined ICDs, to provide their capability. Additionally, components for acquisition and development may span multiple ICDs. An example of this would be a commercial radar that provides both status and control of the unit via the radar's software Application Programming Interface (API).

3.3 Data Distribution Service (DDS^{TM})

The data bus supporting autonomy messaging (as seen in Figure 3) is implemented via DDS^{TM}. DDS is a middleware protocol and API standard for data-centric connectivity from the Object Management Group (OMG). It integrates the components of a system together, providing low-latency data connectivity, extreme reliability, and a scalable architecture. In a distributed system, middleware is the software layer that lies between the operating system and applications. It enables the various system components to more easily communicate and share data. It simplifies the development of distributed systems by letting software developers focus on the specific purpose of their applications rather than the mechanics of passing information between applications and systems. The DDS specification is fully described in free reference material on the OMG website and there are both open source and commercially available implementations.

3.4 Naming Conventions

UMAA services are modeled within the UCS Architecture under the Multi-Domain Extension (MDE). The UCS Architecture uses SoaML concepts of participant, serviceInterface, service port, and request port to describe the interfaces that make up a service and show how the service is used. Each service defines the capability it provides as well as required interfaces. Each interface consists of an operation that accepts a single message (A SoaML MessageType). In SoaML, a MessageType is defined as a unit of information exchanged between participant Request and Service ports via ServiceInterfaces. Instances of a MessageType are passed as parameters in ServiceInterface operations. (Reference: UCS Architecture, Architecture Technical Governance)

To promote commonality across service definitions, a common way of naming services and their sets of operations and messages has been adopted for defining services within UCS-MDE. The convention uses the Service Base Name <SBN> and an optional Function Name [FN] to derive all service names and their associated operations and messages. As this is meant to be a guide, services might not include all of the defined operations and messages and their names might not follow the convention where a more appropriate name adds clarity.

Furthermore, services in UMAA are not required to be defined as indicated in Table 3 when all parts of the service capabilities are required for the service to be meaningful (such as ResourceAllocation).

Additionally, note that for UMAA not all operations defined in UCS-MDE result in a message being published to the DDS bus, e.g., since DDS uses publish/subscribe, most query operations result in a subscription to a topic and do not actually publish the associated request message. In the case of cancel commands, there is no associated implementation of the cancel<SBN>[FN]CommandStatus as it is just the intrinsic response of the DDS dispose function; so, it is essentially a NOOP (no operation) in implementation. The conventions used to define UCS-MDE services are as follows:

Service Name
<SBN>[FN]Config
<SBN>[FN]Control
<SBN>[FN]Specs
<SBN>[FN]Status OR Report

where the SBN should be descriptive of the task or information provided by the service. Note that the FN is optional and only included if needed to clarify the function of the service. The suffixes Status and Report are interchangeable. If a "Report" is a more appropriate description of the service, it can be used in lieu of "Status".

Table 3: Service Requests and Associated Responses

	Service Requests (Inputs)	**Service Responses (Outputs)**
Config	set<SBN>[FN]Config	report<SBN>[FN]ConfigCommandStatus
	query<SBN>[FN]ConfigAck	report<SBN>[FN]ConfigAck
	query<SBN>[FN]Config	report<SBN>[FN]Config
	cancel<SBN>[FN]Config	report<SBN>[FN]CancelConfigCommandStatus
	query<SBN>[FN]ConfigExecutionStatus	report<SBN>[FN]ConfigExecutionStatus
Control	set<SBN>[FN]	report<SBN>[FN]CommandStatus
	query<SBN>[FN]CommandAck	report<SBN>[FN]CommandAck
	cancel<SBN>[FN]Command	report<SBN>[FN]CancelCommandStatus
	query<SBN>[FN]ExecutionStatus	report<SBN>[FN]ExecutionStatus
Specs	query<SBN>[FN]Specs	report<SBN>[FN]Specs
Status OR Report	query<SBN>[FN]	report<SBN>[FN]

Service Requests (operation:message)
set<SBN>[FN]Config:<SBN>[FN]ConfigCommandType

query<SBN>[FN]Config:<SBN>[FN]ConfigRequestType[1]
set<SBN>[FN]:<SBN>[FN]CommandType
query<SBN>[FN]CommandAck:<SBN>[FN]CommandAckRequestType[1]
cancel<SBN>[FN]Command:<SBN>[FN]CancelCommandType[1]
cancel<SBN>[FN]Config:<SBN>[FN]CancelConfigType[1]
query<SBN>[FN]ExecutionStatus:<SBN>[FN]ExecutionStatusRequestType[1]
query<SBN>[FN]ConfigExecutionStatus:<SBN>[FN]ConfigExecutionStatusRequestType[1]
query<SBN>[FN]ConfigAck:<SBN>[FN]ConfigAckRequestType[1]
query<SBN>[FN]Specs:<SBN>[FN]SpecsRequestType[1]
query<SBN>[FN]:<SBN>[FN]RequestType [1] [2]

Service Responses (operation:message)
report<SBN>[FN]ConfigCommandStatus:<SBN>[FN]ConfigCommandStatusType
report<SBN>[FN]Config:<SBN>[FN]ConfigReportType
report<SBN>[FN]ConfigAck:<SBN>[FN]ConfigAckReportType
report<SBN>[FN]CommandStatus:<SBN>[FN]CommandStatusType
report<SBN>[FN]CommandAck:<SBN>[FN]CommandAckReportType
report<SBN>[FN]CancelCommandStatus:<SBN>[FN]CancelCommandStatusType[1]
report<SBN>[FN]CancelConfigCommandStatus:<SBN>[FN]CancelConfigCommandStatusType[1]
report<SBN>[FN]ExecutionStatus:<SBN>[FN]ExecutionStatusReportType
report<SBN>[FN]ConfigExecutionStatus:<SBN>[FN]ConfigExecutionStatusReportType
report<SBN>[FN]Specs:<SBN>[FN]SpecsReportType
report<SBN>[FN]:<SBN>[FN]ReportType

where,

- Config (Configuration) Command/Report − This is the setup of a resource for operation of a particular task. Attributes may be static or variable. Examples include: maximum RPM allowed, operational sonar frequency range allowed, and maximum allowable radio transmit power.
- Command Status − This is the current state of a particular command (either control or configuration).
- Command − This is the ability to influence or direct the behavior of a resource during operation of a particular task. Attributes are variable. Examples include a vehicle's speed, engine RPM, antenna raising/lowering, and controlling a light or gong.
- Command Ack (Acknowledgement) Report − This is the command currently being executed.
- Cancel − This is the ability to cancel a particular command that has been issued.
- Execution Status Report − This is the status related to executing a particular command. Examples associated with a waypoint command include cross track error, time to achieve, and distance remaining.
- Specs (Specifications) Report − Provides a detailed description of a resource and/or its capabilities and constraints. Attributes are static. Examples include: maximum RPM of a motor, minimum frequency of a passive sonar sensor, length of the unmanned vehicle, and cycle time of a radar.
- Report − This is the current information being provided by a resource. Examples include vehicle speed, rudder angle, current waypoint, and contact bearing.

3.5 Namespace Conventions

Each UMAA service and the messages under the service can be accessed through their appropriate UMAA namespace. The namespace reflects the mapping of a specific service to its parent ICD, and the parent ICD's mapping to the overall UMAA Design Description. For example:

Access the Primitive Driver Control service under Maneuver Operations:
UMAA::MO::PrimitiveDriverControl
Access the ContactReport Service under Situational Awareness:

[1] These message types are required for UCS model rules of construction, but are not implemented as messages in the UMAA specification.
[2] At this time, there are no Requests in the specification. This will be the message format when Requests have been added.

UMAA::SA::ContactReport

The UMAA model uses common data types that are re-used through the model to define service interface topics, interface topics, and other common data topics. These data types are not intended to be directly utilized but, for reference, they can be accessed in the same manner:

Access the common UMAA Status Message Fields:
UMAA::UMAAStatus
Access the common UMAA GeoPosition2D (i.e., latitude and longitude) structure:
UMAA::Common::Measurement::GeoPosition2D

3.6 Cybersecurity

The UMAA standard addressed in this ICD is independent from defining specific measures to achieve Cybersecurity compliance. This UMAA ICD does not preclude the incorporation of security measures, nor does it imply or guarantee any level of Cybersecurity within a system. Cybersecurity compliance will be performed on a program-specific basis and compliance testing is outside the scope of UMAA.

3.7 GUID algorithm

The UMAA standard utilizes the Globally Unique IDentifier (GUID), conforming to the variant defined in RFC 4122 (variant value of 2). Generators of GUIDs may generate GUIDs of any valid, RFC 4122-defined version that is appropriate for their specific use case and requirements. (Reference: A Universally Unique IDentifier (UUID) URN Namespace)

3.8 Large Collections

The UMAA standard defines Large Collections, which are collections of decoupled but related data. Large Collections provide the ability to update one or more elements of the collection without republishing the entire collection to the DDS bus. This avoids two problems related to using an unbounded sequence type in a DDS message: 1) resource consumption growing as the collection is appended to or updated, and 2) DDS implementation-specific limitations on unbounded sequences. There are two implementations of a Large Collection: the Large Set (unordered) and the Large List (ordered).

In both Large Collection implementations, there are two important abstractions: the collection metadata and collection element type. Because Large Collections are specific to the UMAA PSM, the type definitions for the collection metadata and collection element are not part of MDE, and the IDL definitions of these types are generated separately. A particular UMAA message that has a Large Collection attribute will reference the metadata type (LargeSetMetadata or LargeListMetadata). The collection element type is defined under the same namespace as the message that uses it, and follows the naming pattern <parent message name><attribute name><collection type>Element. Each element of the collection is published as a separate message on the DDS bus, and can be tracked back to their related collection using the setID or listID. Users can also trace an element in a set to the source attribute (a NumericGUID) of the Service Provider that generated the report with this set using the collection metadata.

3.8.1 Necessary QoS

To achieve the Large Collection consistency in the update process described below, ordering of samples on the collection element type topic is necessary. Therefore, publishers and subscribers to the collection element type topic must use the `PRESENTATION` QoS policy with an `access_scope` of `DDS_TOPIC_PRESENTATION_QOS` and `ordered_access`.

3.8.2 Updating Large Collections

When elements of the collection are updated, the metadata must be updated as well to signal a change in the set. The `updateElementID` is updated to match the elementID of the element whose reception signals the end of the atomic update of the collection. Because of the requirement of an ordered topic described above, this will be the element that is updated last chronologically. The metadata `updateElementTimestamp` must be updated to the timestamp of the same element that signals the end of the update.

The set can be updated as a batch (multiple elements in a single "update cycle," as determined by the provider). This allows for a coarse synchronization: data elements that do not match the metadata `updateElementID` and `updateElementTimestamp` can be assumed to be part of an in-progress update cycle. Consumers can choose to immediately act on those data individually or wait until the matching element is received to signal that the complete update cycle has finished and consider the set as a whole. Note that the coarseness of synchronization is service-dependent: in some cases an intermediate view of a collection update may be logically incorrect to act upon.

3.8.3 Specifying an Empty Large Collection

A particular Large Collection can be empty during initial creation. This is indicated by publishing metadata with a `size` of zero and an `updateElementID` set to the Nil UUID. As specified in section 4.1.7 of the referenced document "A Universally Unique IDentifier (UUID) URN Namespace", this is a "special form of UUID that is specified to have all 128 bits set to zero".

3.8.4 Large Set Types

The following details the LargeSetMetadata structure:

Table 4: LargeSetMetadata Structure Definition

Attribute Name	Attribute Type	Attribute Description
setID	NumericGUID	Identifies the Large Set instance this metadata relates to.
updateElementID	NumericGUID	This field references the element ID of the set element whose reception signals the end of an atomic update to this set. This elementID must be used in conjunction with the updateElementTimestamp below to fully identify when the atomic update has completed and the set is stable.
updateElementTimestamp†	DateTime	This field identifies the elementTimestamp of the element, referenced above by updateElementID, that signals the end of an atomic update to this set. This field will be empty in the event that the element update results from a DDS dispose.
size	LargeCollectionSize	Indicates the number of elements associated with this set after the atomic update is complete.

An example element type is shown below, where a `FooReportType` message has a Large Set attribute called "items" whose type is `BarType`

Table 5: Example FooReportTypeItemsSetElement Structure Definition

Attribute Name	Attribute Type	Attribute Description
element	BarType	The value of the set element.
setID	NumericGUID	Identifies the Large Set instance this element relates to.
elementID*	NumericGUID	Uniquely identifies this element within the set and across all large collection elements that currently exist on the DDS bus.
elementTimestamp	DateTime	The timestamp of this element.

3.8.5 Large List Types

The following details the LargeListMetadata structure:

Table 6: LargeListMetadata Structure Definition

Attribute Name	Attribute Type	Attribute Description
listID	NumericGUID	Identifies the Large List instance this metadata relates to.
updateElementID	NumericGUID	This field references the element ID of the list element whose reception signals the end of an atomic update to this list. This elementID must be used in conjunction with the updateElementTimestamp below to fully identify when the atomic update has completed and the list is stable.
updateElementTimestamp†	DateTime	This field identifies the elementTimestamp of the element, referenced above by updateElementID, that signals the end of an atomic update to this list. This field will be empty in the event that the element update results from a DDS dispose.
startingElementID	NumericGUID	This field identifies the list element, tying to its elementID, that is sequentially first in the list. This is provided for convenience when iterating through the linked list using the nextElementID field.
size	LargeCollectionSize	Indicates the number of elements associated with this set after the atomic update is complete.

An example element type is shown below, where a `FooReportType` message has a Large List attribute called "items" whose type is `BarType`

Table 7: Example FooReportTypeItemsListElement Structure Definition

Attribute Name	Attribute Type	Attribute Description
element	BarType	The value of the list element.
listID	NumericGUID	Identifies the Large List instance this element relates to.
elementID*	NumericGUID	Uniquely identifies this element within the list and across all large collection elements that currently exist on the DDS bus.
elementTimestamp	DateTime	The timestamp of this element.
nextElementID†	NumericGUID	This field references to the elementID of the element that logically follows this element in the linked list. This is empty if this element is sequentially last.

4 Introduction to Coordinate Reference Frames and Position Model

4.1 Vehicle Reference Frame

In the following Service Definitions, we use the parameters yaw, pitch, and roll to define the vehicle orientation with respect to the specified reference frame. Each parameter is described as a rotation around a given axis: Yaw about the Z axis. Pitch about the Y axis. Roll about the X axis. A UUV is shown in the diagrams because it has more degrees for freedom for its pose and motion, however, the terminology equally applies to both USVs and UUVs.

The axes are defined as:

- X - Positive in the forward direction, negative in the aft.
- Y - Positive in the starboard direction, negative in the port.
- Z - Positive in the down direction, negative in the up.

Additionally, rotations about all axes follow the right-hand rule.

4.2 Earth-Centered Earth-Fixed Frame

The Earth-Centered Earth-Fixed (ECEF) frame is a global reference frame with its origin at the center of the ellipsoid modeling the Earth's surface (Figure 4). The Z-axis points along the Earth's axis of rotation through the North Pole. The X-axis points from the origin to the intersection of the equator with the prime meridian, which defines 0° longitude. The Y-axis completes the right-handed orthogonal system, intersecting the equator at the 90° east meridian.

4.3 North-East-Down Frame

The North-East-Down (NED) frame is defined with an origin at the object described by the navigation solution. The Down axis is defined as normal to the surface of the reference ellipsoid in the direction pointing towards the interior of the Earth. The North axis is the projection of the line from the object to the north pole onto the plane orthogonal to the Down axis. The East axis completes the right-handed orthogonal system and points in the East direction.

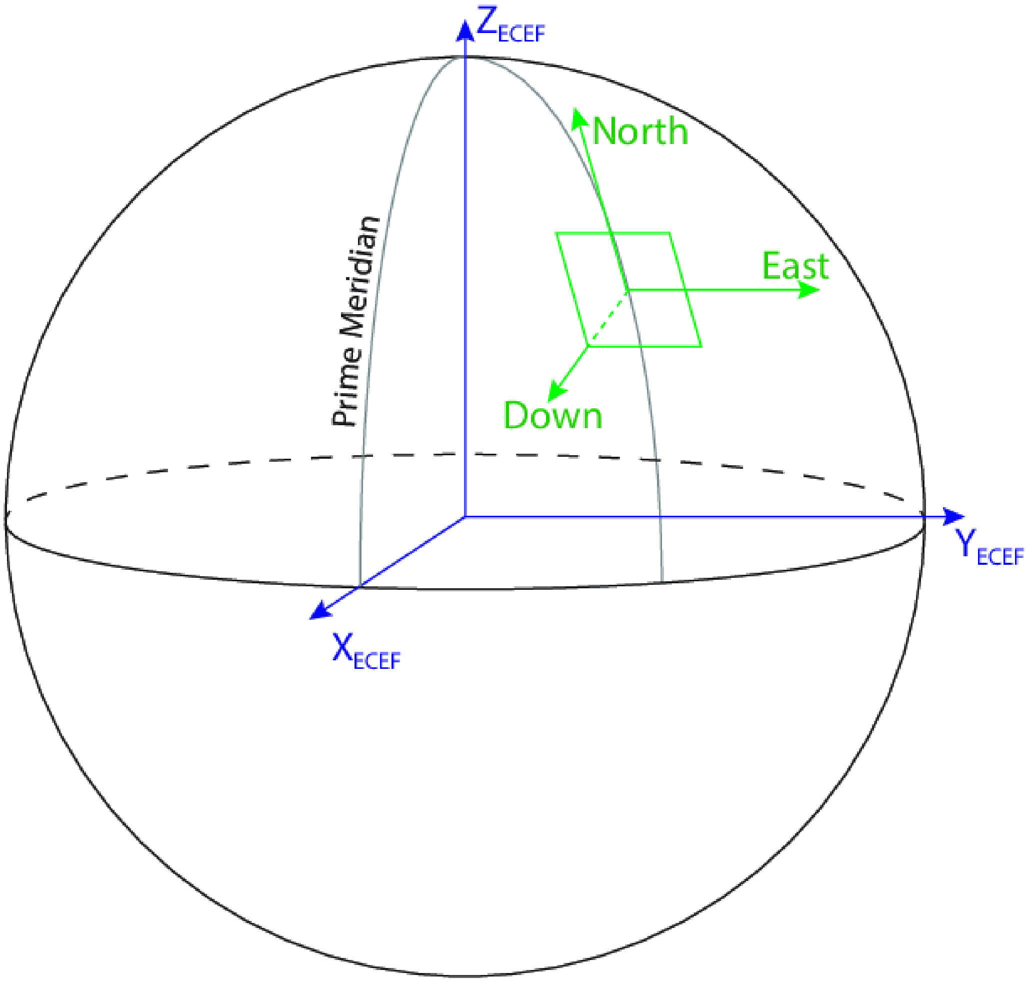

Figure 4: Origins and axes of the Earth-Centered Earth-Fixed (ECEF) and North-East-Down (NED) frames.

4.4 WGS 84

The World Geodetic System (WGS) 1984 defines a standard coordinate system for the Earth. It represents the Earth as an oblate spheroid, and defines the mapping between latitude-longitude-altitude (LLA) coordinates and Cartesian ECEF coordinates. GPS reports positions in WGS 84 LLA coordinates. It has become the standard datum for navigation.

While the UMAA services typically make use of the coordinate systems defined by WGS 84, it also defines an Earth Gravity Model (EGM) and a World Magnetic Model (WMM) which are updated regularly.

4.5 Vehicle Orientation

Determining the orientation of the vehicle (Figure 5) with respect to any reference frame is carried out via the following procedure (Figure 6).

1. Align the vehicle's longitudinal or X axis with the reference frame X axis. In the global reference frame, this is the north direction.
2. Align the vehicle's down or Z axis with the reference frame's Z axis. In the global reference frame, this is the gravity direction.
3. Ensure that the vehicle's transverse or Y axis is aligned with the reference frame's Y axis. In the global reference frame, this is the east direction.
4. Rotate the vehicle about the vehicle's Z axis by the yaw angle (Figure 7).
5. Rotate the vehicle about the vehicle's newly oriented Y axis by the pitch angle (Figure 8).

6. Rotate the vehicle about the vehicle's newly oriented X axis by the roll angle (Figure 9).

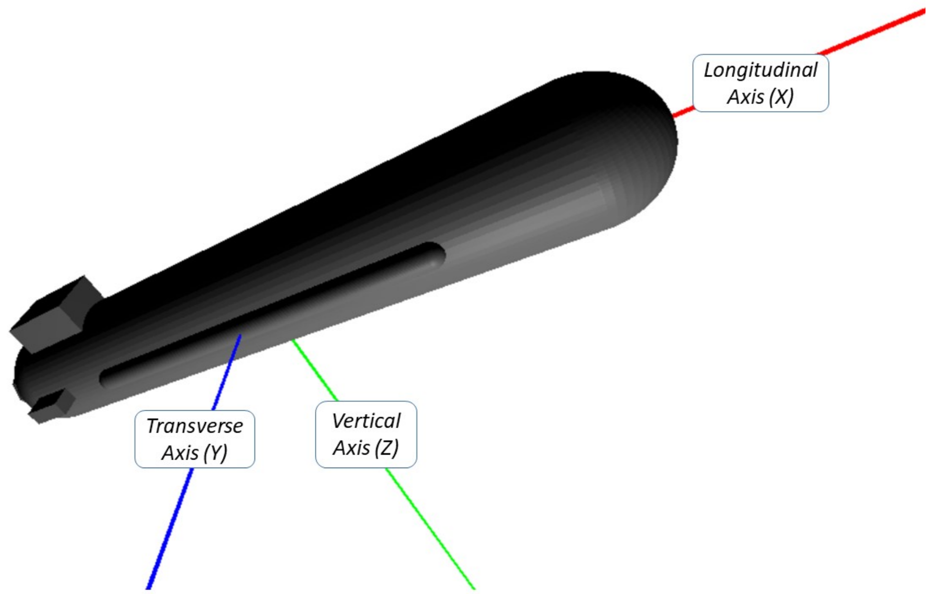

Figure 5: Define the Vehicle Coordinate System

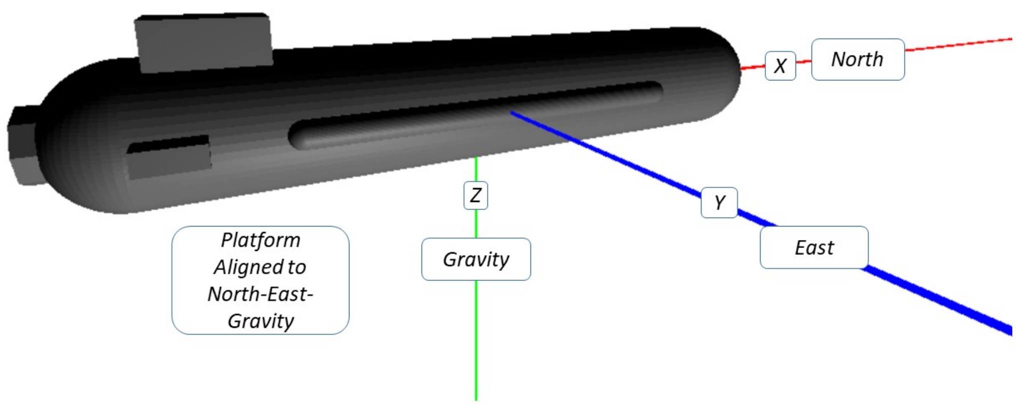

Figure 6: Align the Vehicle with the Reference Frame Axes.

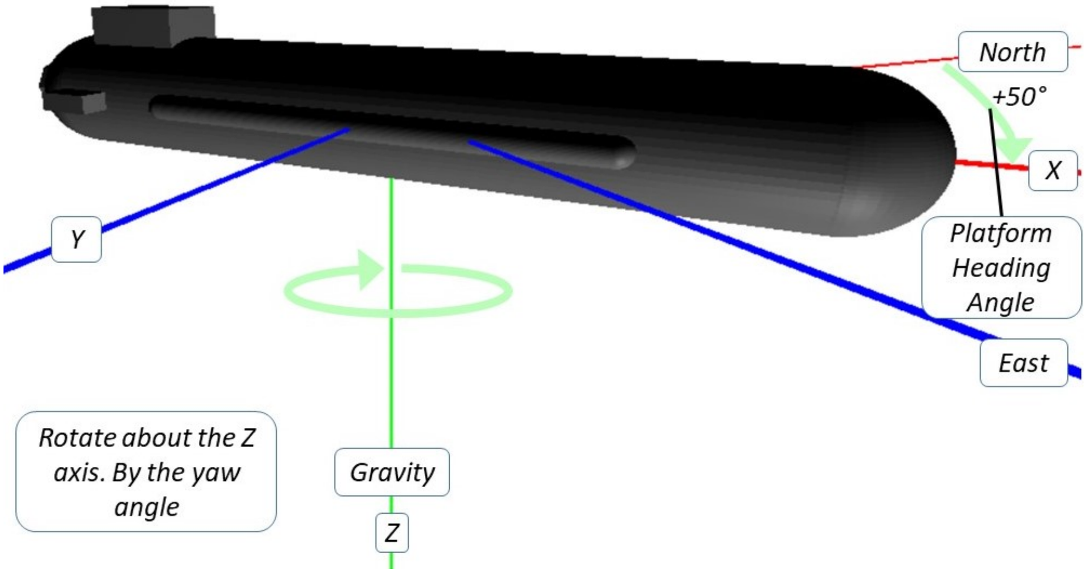

Figure 7: Rotate the Vehicle by the Yaw Angle.

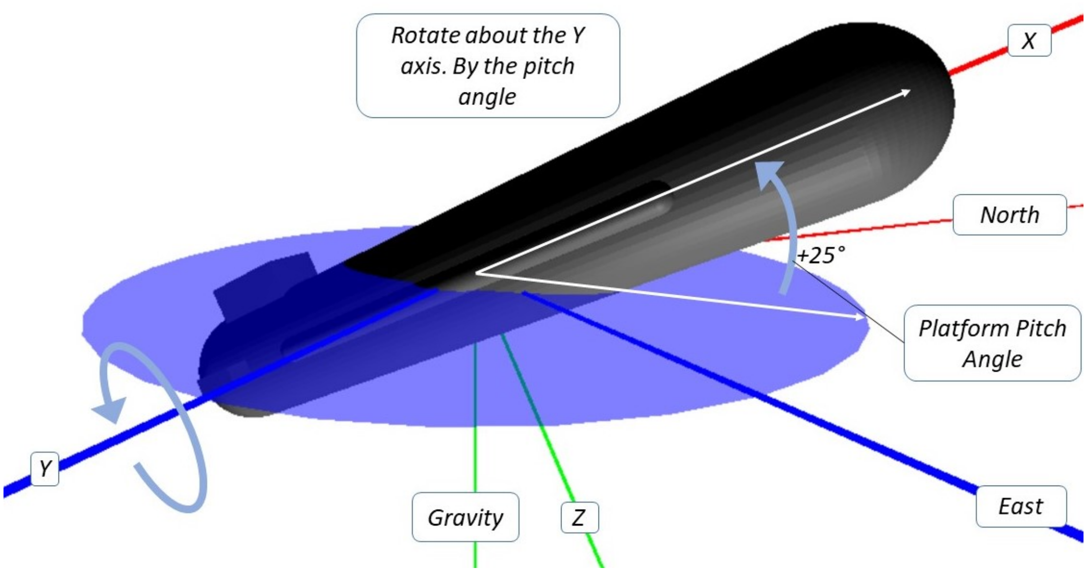

Figure 8: Rotate the Vehicle by the Pitch Angle.

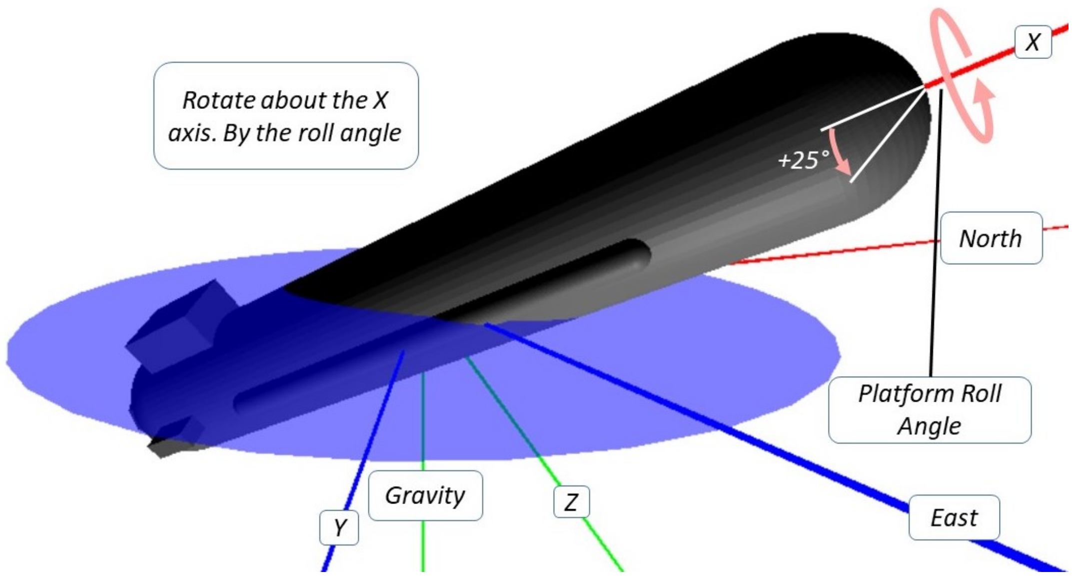

Figure 9: Rotate the Vehicle by the Roll Angle.

4.6 Vehicle Coordinate Reference Frame Origin

UMAA does not specify a required origin for the vehicle coordinate reference frame. However, certain applications may benefit from defining a specific origin such as the registration of multiple sensors with associated offsets for data fusion. Possible origins include the keel/transom intersection, bow/waterline intersection, center of gravity, center of buoyancy and location of INS. A few examples follow.

Definitions

- Keel Transom Intersection
  - Beam at Waterline (BWL) - The maximum distance of the vehicle at the waterline, the distance along the Y axis of the widest point of the hull where it meets the waterline.
  - Design Waterline (DWL) - The line representing the waterline on the vehicle at designed load in summer temperature.
  - Keel - The principal fore-and-aft component of a ship's framing, located along the centerline of the bottom and connected to the stem and stern frames.
  - Length at Waterline (LWL) - The measured distance of the vehicle at the level where it sits in the water, measured along the X axis.
  - Transom - The aftermost transverse flat or shaped plating enclosing the hull.

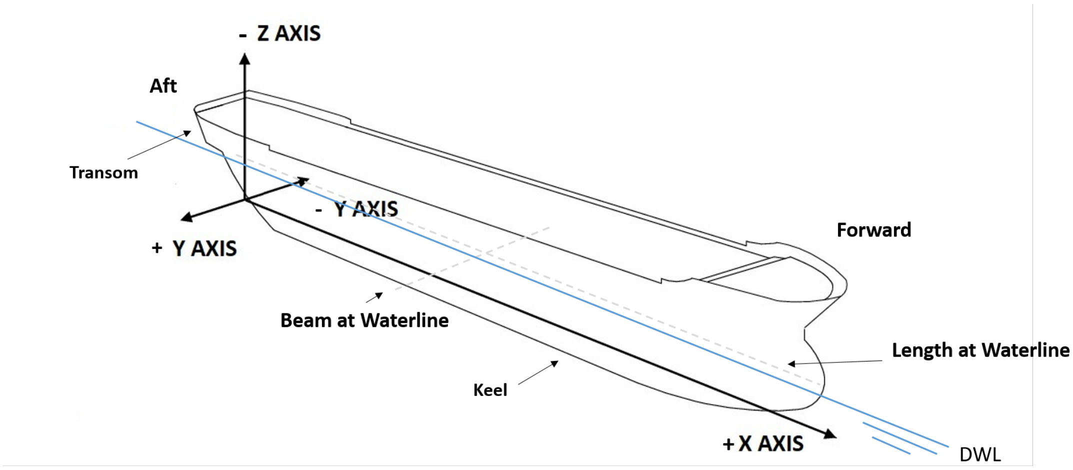

Figure 10: Keel Transom Intersection Origin Location on a USV as Example

- Center of Buoyancy
  - X - The Longitudinal Center of Buoyancy (LCB) when fully submerged.
  - Y - The symmetrical centerline.
  - Z - The Vertical Center of Buoyancy (VCB) when fully submerged.

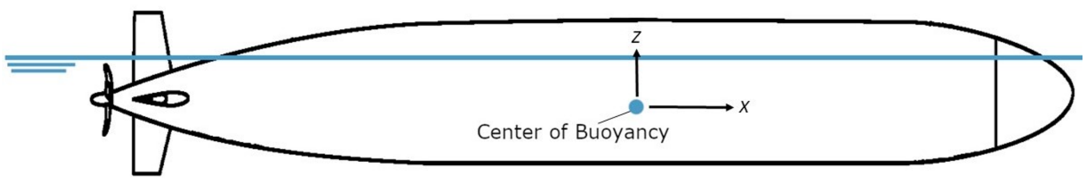

Figure 11: Center of Buoyancy Origin Location on a UUV as Example.

5 Flow Control

5.1 Command / Response

This section defines the flow of control for command/response over the DDS bus. A command/response controls a specific service. While the exact names and processes will depend on the specific service and command being executed, all command/responses in UMAA follow a similar pattern. A notional "Function" command `FunctionCommand` is used in the following examples. As will be described in subsequent paragraphs, DDS publish/subscribe methods are used in implementations to issue commands and responses.

To direct a `FunctionCommand` at a specific Service Provider, UMAA includes a `destination` GUID in all commands. A Service Provider is required to respond to all `FunctionCommand`s where the `destination` is the same as the Service Provider's ID. The Service Consumer will also create a `sessionID` for the command when commanded. The `sessionID` is used to track the command execution as a key into other command-related messages. The `sessionID` must be unique across all `FunctionCommand` instances that are active (i.e. currently on the DDS bus), otherwise the Service Provider will consider the `FunctionCommand` to be a command update (see Section 5.1.4.2). Once a `FunctionCommand` is removed from the DDS bus as part of the Command Cleanup process (see Section 5.1.5), its `sessionID` may be reused for future commands without triggering a command update; therefore it is not necessary for a Service Provider to maintain a complete history of `sessionID`s.

Service Provider and Service Consumer terminology in the following sections is adopted from the OMG Service-oriented architecture Modeling Language (SoaML).

To initialize, a Service Provider (controllable resource) subscribes to the `FunctionCommand` DDS topic. At startup or right before issuing a command, the Service Consumer (controlling resource) subscribes to the `FunctionCommandStatus` DDS topic. Optionally, the Service Consumer may also subscribe to the `FunctionCommandAckReport` to monitor which command is currently being executed, and the `FunctionExecutionStatusReport` (if defined for the Function service) that provides reporting on function-specific data status.

Both Service Providers and Service Consumers are required to recover or clean up any previous persisted commands on the bus during initialization.

To execute a command, the Service Consumer publishes a `FunctionCommandType` to the DDS bus. The Service Provider will be notified and will begin processing the request. During each phase of processing, the Service Provider will provide updates to the Service Consumer via published updates to a related `FunctionCommandStatus` topic. Command responses are correlated to their originating command via the `sessionID`. If a command with a duplicate `sessionID` is received, the Service Provider will regard this as a command update, and follow the flow control detailed in Section 5.1.4.2. Command status updates are provided in the command responses via the `commandStatus` field with additional details included in the `commandStatusReason` field. The Service Provider will also publish the current executing command to the `FunctionCommandAckReport` topic. When defined for the Function service, the Service Provider must also publish the `FunctionExecutionStatusReport` topic and update it as appropriate throughout the execution of the command.

The required state transitions for the `commandStatus` field are shown in Figure 12. Commands may complete normally, or they may terminate early due to failure (Section 5.1.4.4) or cancellation (Section 5.1.4.5). The state machine for a command can also be reset to `ISSUED` via a command update (Section 5.1.4.2). If there is not a self-transition indicated in the diagram, you cannot republish that state in a message. Every command must transition through the states as defined. For example, it is a violation to transition from `ISSUED` to `EXECUTING` without transitioning through `COMMANDED`. Even in the case where there is no logic executing between the `ISSUED` and `EXECUTING` states, the Service Provider is required to transition through `COMMANDED`. This ensures consistent behavior across different Service Providers, including those that do require the `COMMANDED` state.

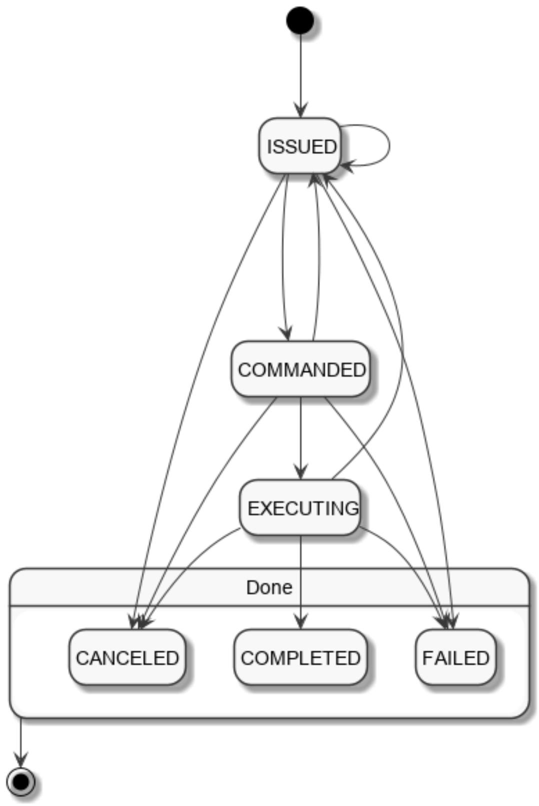

Figure 12: State transitions of the `commandStatus` as commands are processed.

As described above, each time a command transitions to a new state, a `FunctionCommandStatus` message is published containing the updated `commandStatus` and a `commandStatusReason` that indicates why the state transition happened. The table below shows all valid `commandStatusReason` values for each `commandStatus` transition.

	Ending State					
Starting State	ISSUED	COMMANDED	EXECUTING	COMPLETED	FAILED	CANCELED
Initial State	SUCCEEDED	—	—	—	—	—
ISSUED	UPDATED	SUCCEEDED	—	—	VALIDATION_FAILED RESOURCE_FAILED INTERRUPTED TIMEOUT SERVICE_FAILED	CANCELED
COMMANDED	UPDATED	—	SUCCEEDED	—	RESOURCE_REJECTED INTERRUPTED TIMEOUT SERVICE_FAILED	CANCELED
EXECUTING	UPDATED	—	—	SUCCEEDED	OBJECTIVE_FAILED RESOURCE_FAILED INTERRUPTED TIMEOUT SERVICE_FAILED	CANCELED
COMPLETED	—	—	—	—	—	—
FAILED	—	—	—	—	—	—
CANCELED	—	—	—	—	—	—

Figure 13: Valid `commandStatusReason` values for each `commandStatus` state transition. Entries marked with a (—) indicate that the state transition is invalid.

In the following sections, the sequence diagrams demonstrate different exchanges between a Service Consumer and Service

Provider. Within the diagrams, the dashed arrows represent implementation-specific communications that are outside of UMAA's scope. These sequence diagrams are just an example of one possible implementation. Other implementations may have different communication patterns between the Service Provider and the Resource or be implemented completely within the Service Provider process itself (no dependency on an external Resource). Likewise, the interactions between the User and Service Consumer may follow similar or different patterns. However, the UMAA-defined exchanges with the DDS bus between the Service Consumer and Service Provider must happen in the order shown within the sequence diagrams.

5.1.1 High-Level Flow

The high-level flow of a command sequence is shown in Figure 14 and can be described as follows:

1. The Command Startup Sequence is performed.
2. For each command to be executed:
   (a) The Command Start Sequence is performed.
   (b) The command is executed (sequence depends on the execution path, i.e., success, failure, or cancel).
   (c) The Command Cleanup Sequence is performed.
3. The Command Shutdown Sequence is performed.

The `ref` blocks will be defined in later sequence diagrams. Note that the duration of the system execution for any particular `FunctionCommandType` is defined by the combination of the Service Provider(s) and Service Consumer(s) in the system and may not be identical to the overall system execution duration. For example, providers may only be available to execute certain commands during specific mission phases or when certain hardware is in specific configurations. This Command Startup Sequence is not required to happen during a system startup phase. The only requirement is that it must be completed by at least one Service Provider and one Service Consumer before any `FunctionCommandType` commands can be fully executed. Likewise, the Command Shutdown sequence may occur at any time the `FunctionCommandType` will no longer be supported. There is no requirement stating that the Command Shutdown Sequence only be performed during a system shutdown phase.

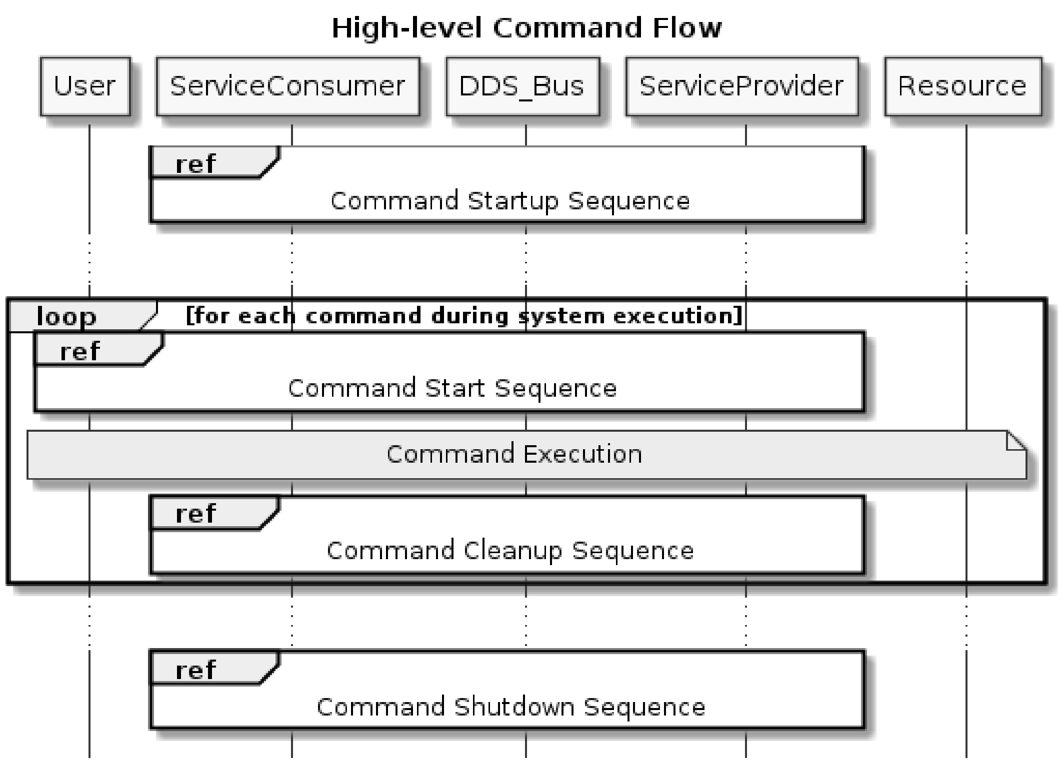

Figure 14: Sequence Diagram for the High-Level Description of a Command Execution.

5.1.2 Command Startup Sequence

As part of initialization both the Service Provider and Service Consumer are required to perform a startup sequence. This startup prepares the Service Provider to execute commands and the Service Consumer to request commands and monitor the progress of those requested commands.

The Service Provider and Service Consumer can initialize in any order. Commands will not be completely executed until both have completed their initialization. The sequence diagram is shown in Figure 15.

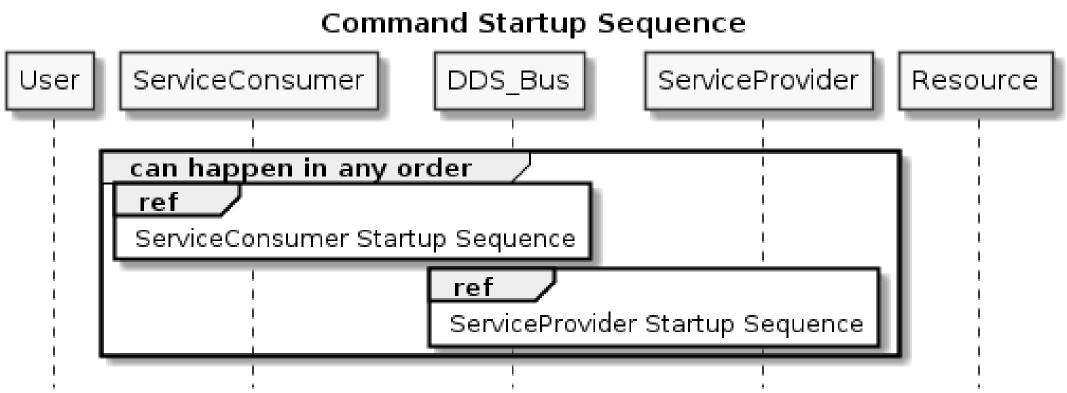

Figure 15: Sequence Diagram for Command Startup.

5.1.2.1 Service Provider Startup Sequence During startup, the Service Provider is required to register as a publisher to the `FunctionCommandStatus`, `FunctionCommandAckReport`, and (if defined for the Function service) the `FunctionExecutionStatusReport` topics.

The Service Provider is also required to subscribe to the `FunctionCommand` topic to be notified when new commands are published.

Finally, the Service Provider is required to handle any existing `FunctionCommandType` commands persisted on the DDS bus with the Service Provider's ID. For each command, if the Service Provider can and wishes to recover, it can continue to execute the command. To obtain the last published state of the command, the Service Provider must subscribe to the `FunctionCommandStatusType`. The Service Provider will continue following the normal status update sequence, picking up from the last status on the bus. If the Service Provider cannot or chooses not to continue processing the command, it must fail the command by publishing a `FunctionCommandStatus` with a `commandStatus` of `FAILED` and a `reason` of `SERVICE_FAILED`.

The Service Provider Startup sequence is shown in Figure 16.

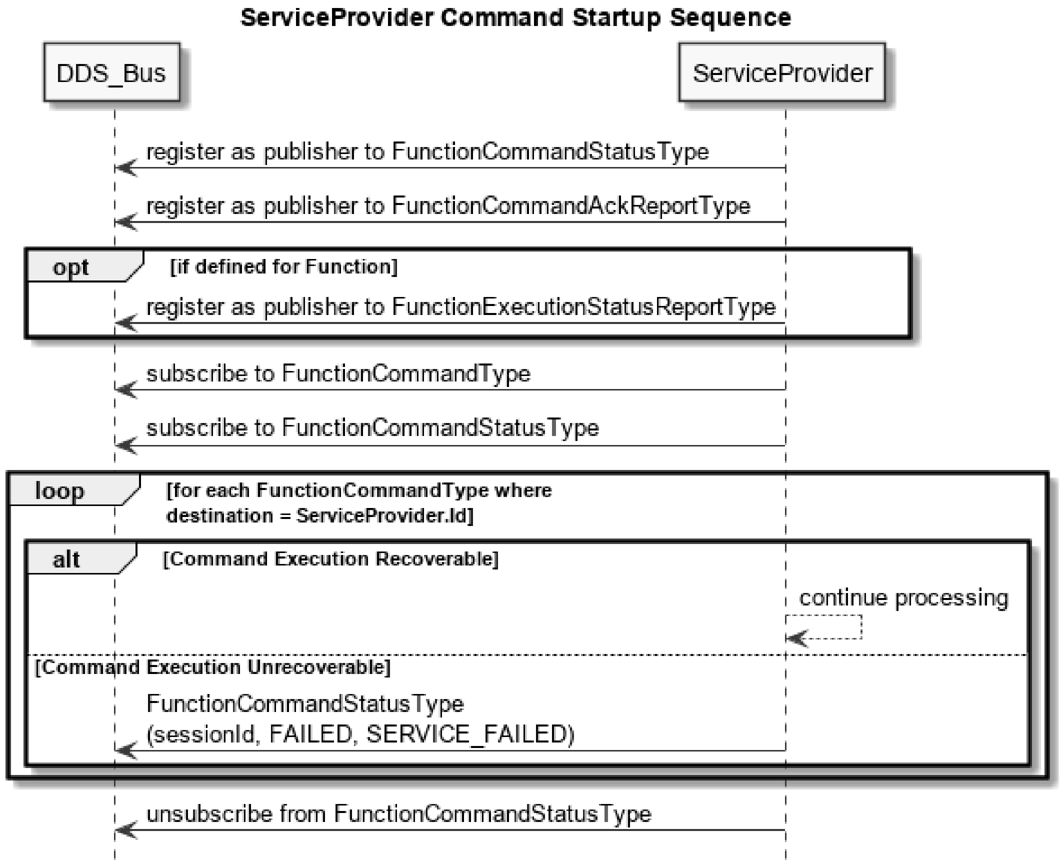

Figure 16: Sequence Diagram for Command Startup for Service Providers.

5.1.2.2 Service Consumer Startup Sequence

During startup, the Service Consumer is required to register as a publisher of the `FunctionCommandType`.

The Service Consumer is also required to subscribe to the `FunctionCommandStatusType` to monitor the execution of any published commands. The Service Consumer can optionally register for the `FunctionCommandAckReportType` and, if defined for the Function service, the `FunctionExecutionStatusReportType` if it desires to track additional status of the execution of commands.

Finally, the Service Consumer is required to handle any existing `FunctionCommandType` commands persisted on the DDS bus with this Service Consumer's ID. To find existing `FunctionCommandType`s on the bus, it must first subscribe to the topic. If the Service Consumer can and wishes to recover, it can continue to monitor the execution of the command. If the Service Consumer cannot or chooses not to continue the execution of the command, it must cancel the command via the normal command cancel method.

The Service Consumer Startup sequence is shown in Figure 17.

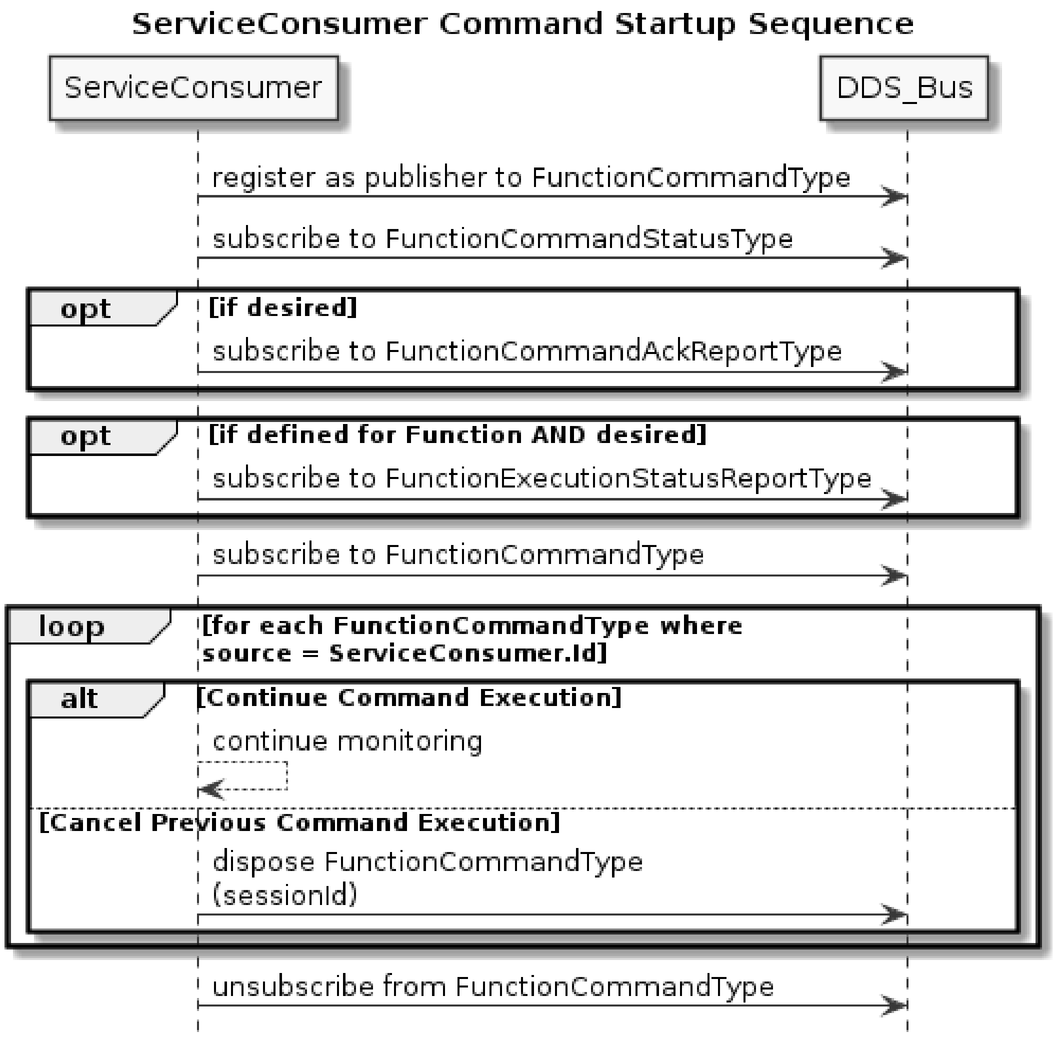

Figure 17: Sequence Diagram for Command Startup for Service Consumers.

5.1.3 Command Execution Sequences

Once both the Service Provider and Service Consumer have performed the startup sequence, the system is ready to begin issuing and executing commands.

5.1.4 Command Start Sequence

The initial start sequence to execute a single new command follows this pattern:

1. The User of the Service Consumer issues a request for a command to be executed.
2. The Service Consumer publishes the `FunctionCommandType` with a unique session ID, the source ID of the Service Consumer, and the destination ID of the desired Service Provider.
3. The Service Provider, upon notification of the new `FunctionCommandType`, publishes a new `FunctionCommandStatusType` with (1) the same session ID as the new `FunctionCommandType`, (2) the status of `ISSUED` and (3) the reason of `SUCCEEDED` to notify the Service Consumer it has received the new command.

The Command Start Sequence for a new command is shown in Figure 18. This pattern will be repeated each time a new command is requested. Note that the Command Start Sequence differs if the `FunctionCommandType` has a `sessionID` that matches another `FunctionCommandType` that currently exists on the DDS bus. This is considered a command update and detailed in Section 5.1.4.2.

After the Command Start Sequence, the sequence can take different paths depending on the actual execution of the command, detailed from Section 5.1.4.1 to Section 5.1.4.5, but they do not enumerate all of the possible execution paths. Other paths (e.g., an objective failing) will follow a similar pattern to other failures; all are required to follow the state diagram shown in Figure 12 and eventually end with the Command Cleanup Sequence (shown in Figure 25).

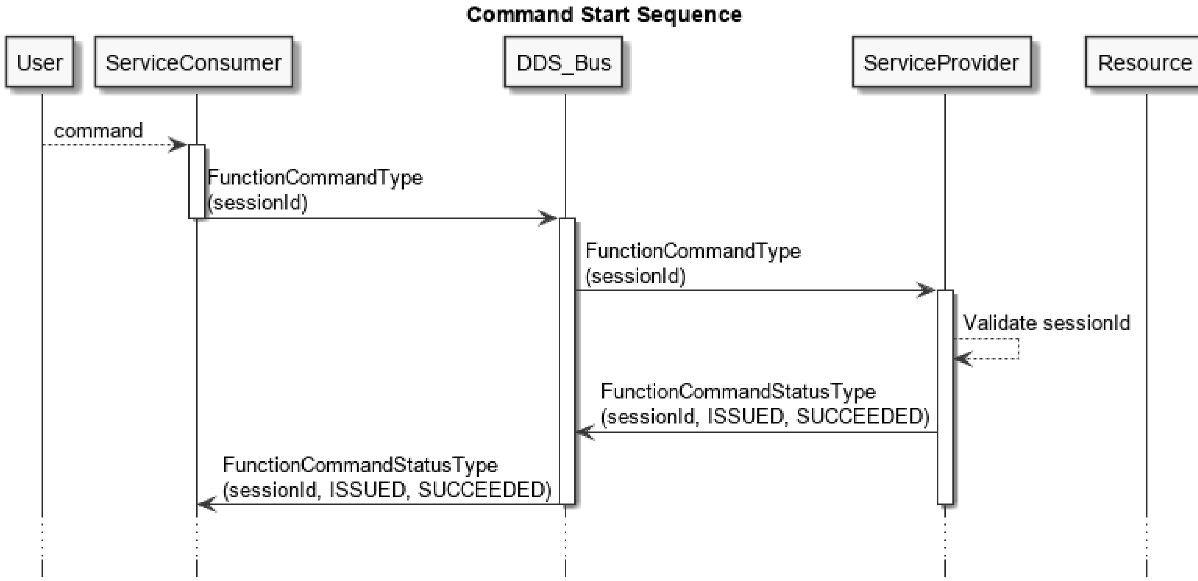

Figure 18: Sequence Diagram for the Start of a Command Execution.

5.1.4.1 Command Execution Once a Service Provider starts to process a command, the Command Execution sequence is:

1. The Service Provider publishes a `FunctionCommandAckReportType` with matching session ID and parameters as the `FunctionCommandType` it is starting to process.
2. The Service Provider performs any validation and negotiation with backing resources as necessary. Once the command is ready to be executed, the Service Provider publishes a `FunctionCommandStatusType` with a status `COMMANDED` and reason `SUCCEEDED` to notify the Service Consumer that the command has been validated and commanded to start execution.
3. Once the command has begun executing, the Service Provider publishes a `FunctionCommandStatusType` with a status `EXECUTING` and reason `SUCCEEDED` to notify the Service Consumer that the command has been validated and commanded to start.
4. If the Function has a defined `FunctionExecutionStatusReportType`, the Service Provider must publish a new instance with matching session ID as the associated `FunctionCommandType`. The `FunctionExecutionStatusReportType` must be updated by the Service Provider throughout the execution as dictated by the definitions of the command-specific attributes in the execution status report.

The command execution sequence is shown in Figure 19. This sequence holds until the command completes execution.

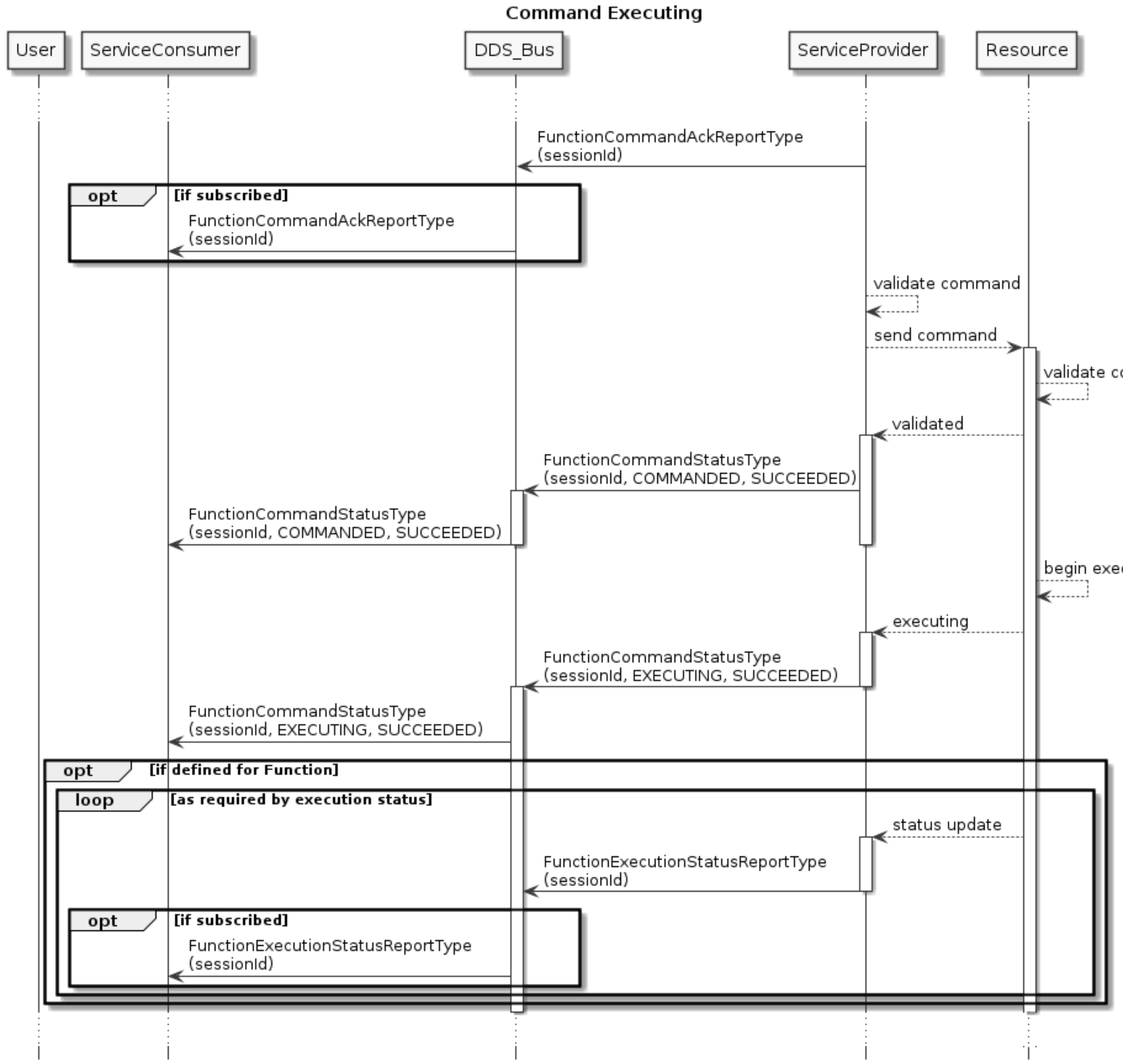

Figure 19: Beginning Sequence Diagram for a Command Execution.

The normal successful conclusion of a command being executed in some cases is initiated by the Service Consumer (an endless GlobalVector command concluded by canceling it) and in other cases is initiated by the Service Provider (a GlobalWaypoint commanded concluded by reaching the last waypoint). Unless otherwise explicitly stated, it is assumed the Service Provider will be able to identify the successful conclusion of a command. In the cases where commands are defined to be indeterminate the Service Consumer must cancel the command when the Service Consumer no longer desires the command to be executed.

5.1.4.2 Updating a Command An updated command is defined as a command with a source ID and session ID identical to the current command being processed by the Service Provider, but whose timestamp is newer than the current command. Only a command that is in a non-terminal state may be updated - otherwise, the Service Consumer must follow the normal command cleanup process and issue a new command with an updated unique session ID. When the Service Provider receives an updated command, it is required to take one of two possible actions:

1. If the current command is in a non-terminal state (`ISSUED`, `COMMANDED`, or `EXECUTING`), then the Service Provider publishes a `FunctionCommandStatusType` with a status `ISSUED` and reason `UPDATED`. The state machine then restarts and proceeds through the normal command flow detailed in 5.1.4. The Service Provider must consider the updated command as an entirely new command, resetting any internal state related to the command (e.g. a timer that keeps track of command timeout).

2. If the current command is in a terminal state (`COMPLETED`, `CANCELED`, or `FAILED`), then the updated command cannot be processed, and the Service Provider must publish a `FunctionCommandStatusType` with a status `FAILED` and follow the normal command cleanup process.

The flow control for command update is detailed below:

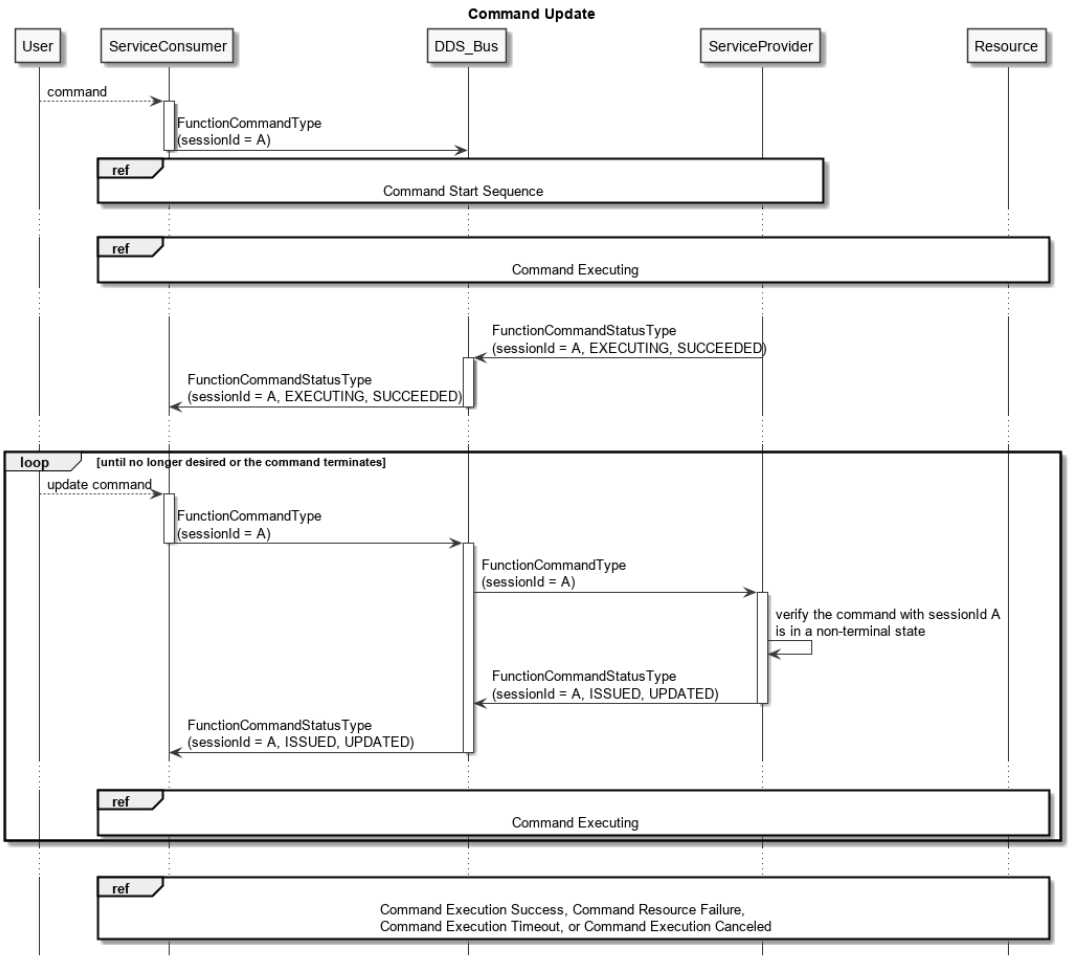

Figure 20: Sequence Diagram for Command Update.

5.1.4.3 Command Execution Success When the Service Provider determines a command has successfully completed, it must update the associated `FunctionCommandStatusType` with as status of `COMPLETED` and reason of `SUCCEEDED`. This signals to the Service Consumer that the command has completed successfully.

The Command Execution Success sequence is shown in Figure 21.

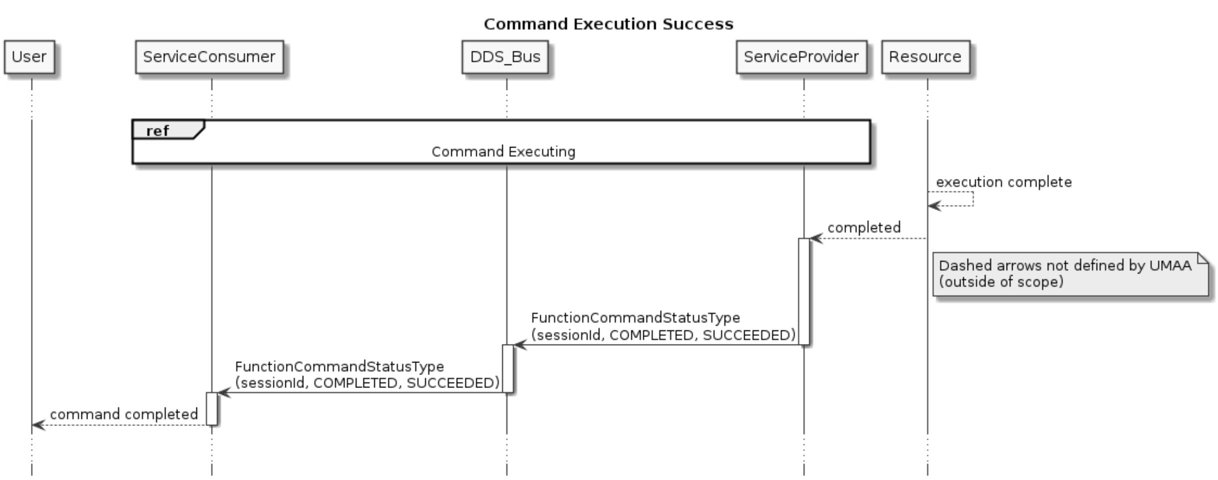

Figure 21: Sequence Diagram for a Command That Completes Successfully.

5.1.4.4 Command Execution Failure The command may fail to complete for any number of reasons including software errors, hardware failures, or unfavorable environmental conditions. The Service Provider may also reject a command for a number of reasons including inability to perform the task, malformed or out of range requests, or a command being interrupted by a higher priority process. In all cases, the Service Provider must publish a `FunctionCommandStatusType` with an identical `sessionID` as the originating `FunctionCommandType` with a status of `FAILED` and the reason that reflects the cause of the failure (`VALIDATION_FAILED, SERVICE_FAILED, OBJECTIVE_FAILED,` etc).

Figure 22 and Figure 23 provide examples where a command has failed.

In the first example, the backing Resource failed and the Service Provider is unable to communicate with it. In this case, the Service Provider will report a `FunctionCommandStatusType` with a status of `FAILED` and a reason of `RESOURCE_FAILED`. This is shown in Figure 22.

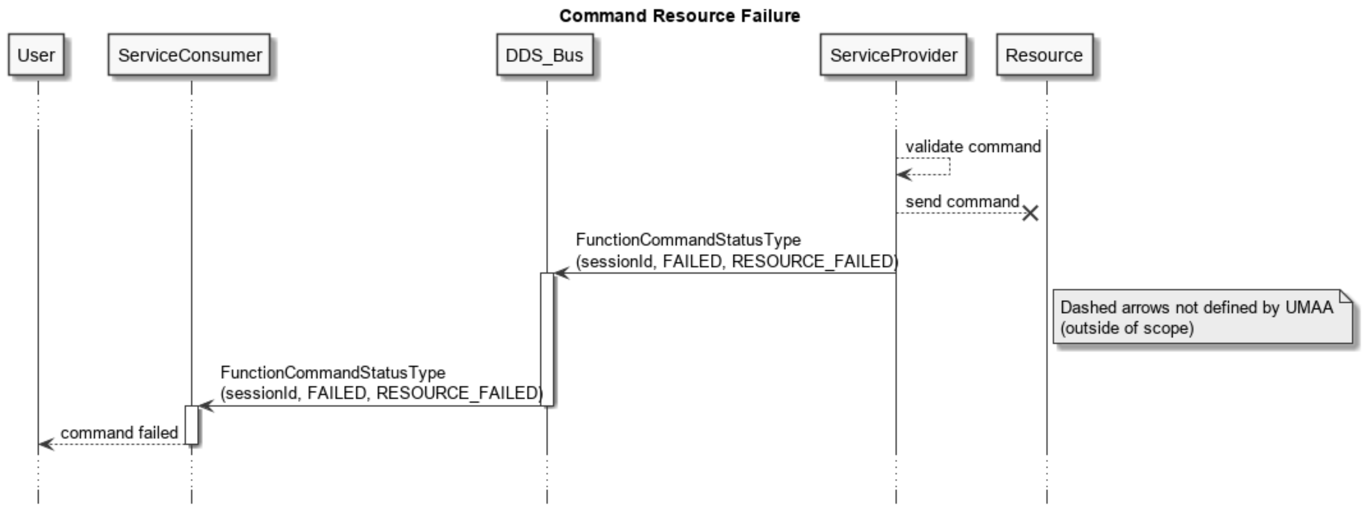

Figure 22: Sequence Diagram for a Command That Fails due to Resource Failure.

In the second example, the Resource takes too long to respond, so the Service Provider cancels the request and reports a `FunctionCommandStatusType` with a status of `FAILED` and a reason of `TIMEOUT`. This is shown in Figure 23.

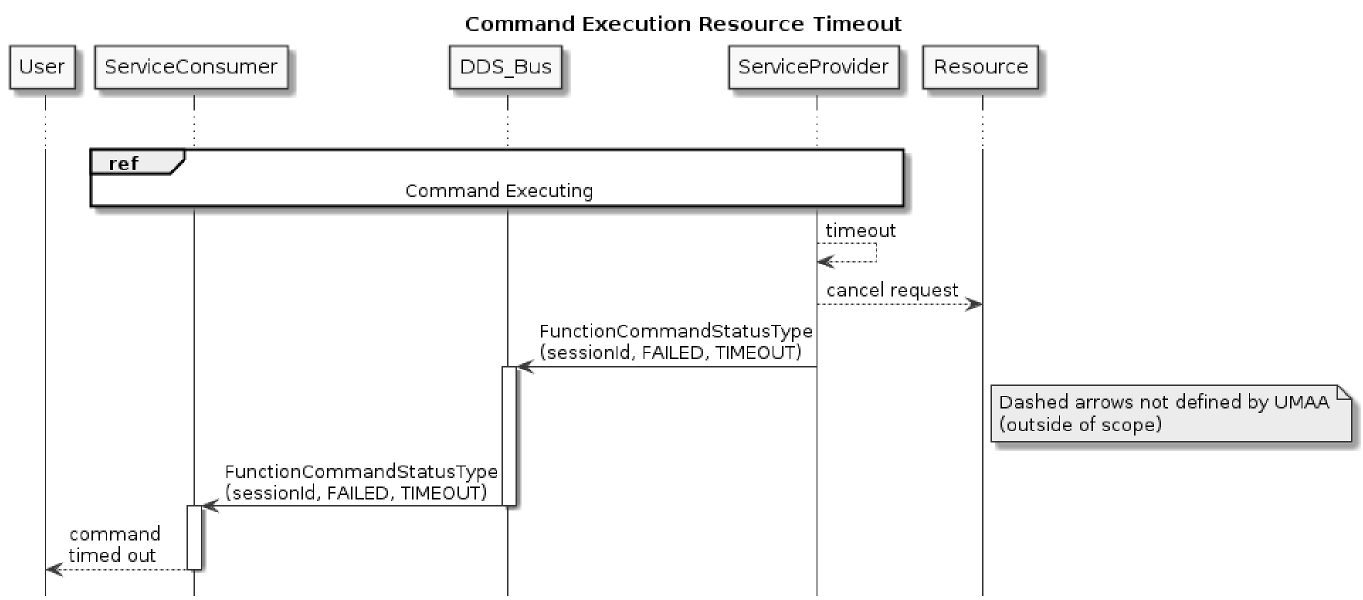

Figure 23: Sequence Diagram for a Command That Times Out Before Completing.

Other failure conditions will follow a similar pattern: when the failure is recognized, the Service Provider will publish a

FunctionCommandStatusType with a status of FAILED and a reason that reflect the cause of the failure.

5.1.4.5 Command Canceled The Service Consumer may decide to cancel the command before processing is finished. To signal a desire to cancel a command, the Service Consumer disposes of the existing FunctionCommandType from the DDS bus before the execution is complete. When notified of the command disposal, and if the Service Provider is able to cancel the command, it should respond to the Service Consumer with a FunctionCommandStatusType with both the status and reason as CANCELED. At this point, the DDS bus should dispose of the FunctionCommandStatusType, the FunctionCommandAckReportType and, (if defined for the Function service) the FunctionExecutionStatusReportType. This is shown in Figure 24. If the command cannot be canceled, then the Service Provider can continue to update the command status until the execution is completed. Reporting will include FunctionCommandStatusType with a status of COMPLETED and a reason of SUCCEEDED. Then, the DDS bus should dispose of the FunctionCommandStatusType, the FunctionCommandAckReportType, and (if defined for the Function service) the FunctionExecutionStatusReportType.

There is no new, unique, or specific status message response to a cancel command from the Service Provider. The cancel command status can be inferred through the corresponding FunctionCommandStatusType status and reason updates.

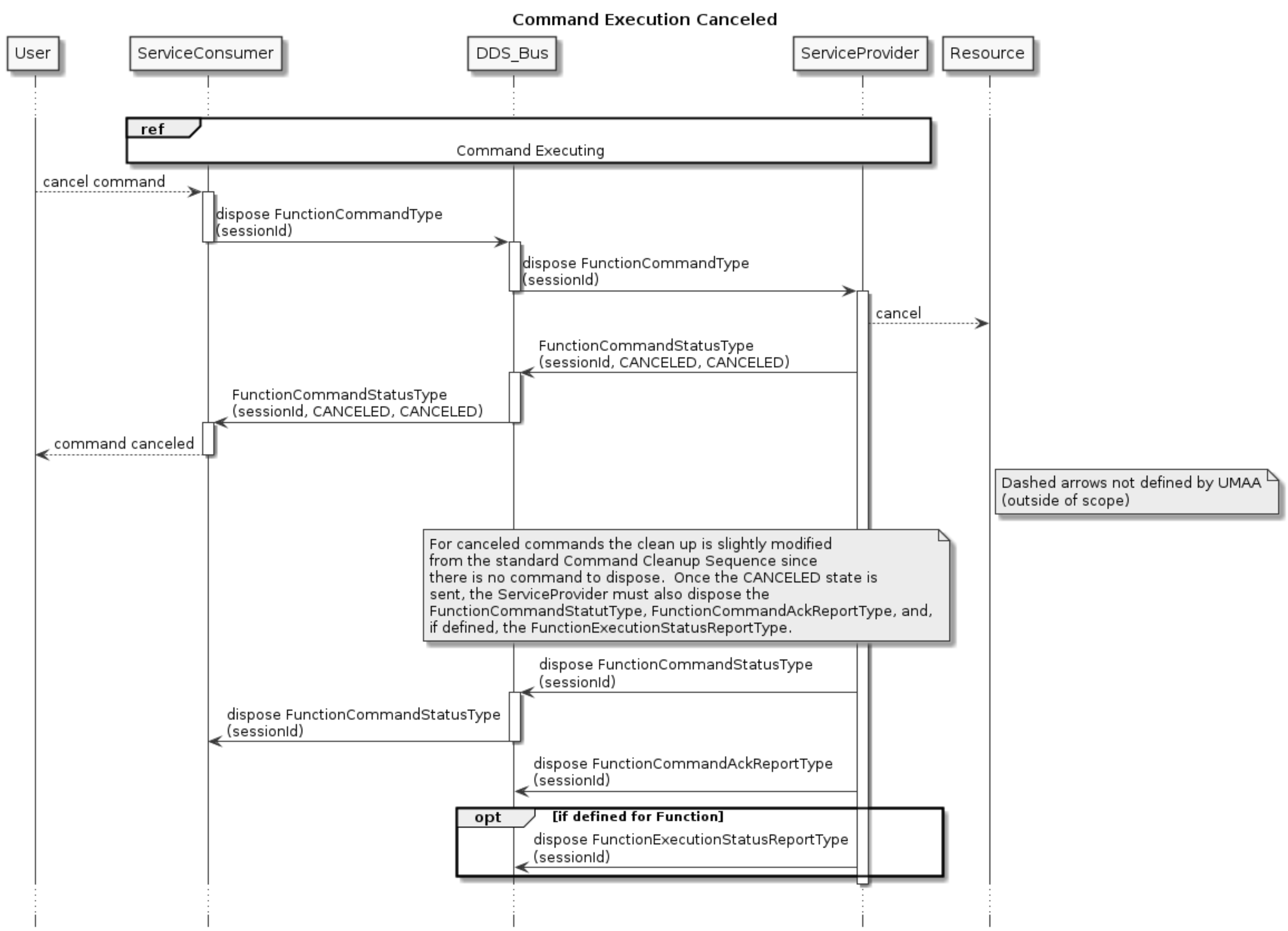

Figure 24: Sequence Diagram for a Command That is Canceled by the Service Consumer Before the Service Provider can Complete It.

5.1.5 Command Cleanup

The Service Consumer and Service Provider are responsible for disposing of corresponding data that is published to the DDS bus when the command is no longer active. With the exception of a canceled command, the signal that a FunctionCommandType can be disposed is when the FunctionCommandStatusType reports a terminal state (COMPLETED or FAILED)[3]. In turn, the

[3]While CANCELED is also a terminal state, the CANCELED command cleanup is handled specially as part of the cancelling sequence and, as such, does not need to be handled here.

signal that a FunctionCommandStatusType, FunctionCommandAckReportType, and (if defined for the Function service) the FunctionExecutionStatusReportType can be disposed is when the corresponding FunctionCommandType has been disposed. This is shown in Figure 25.

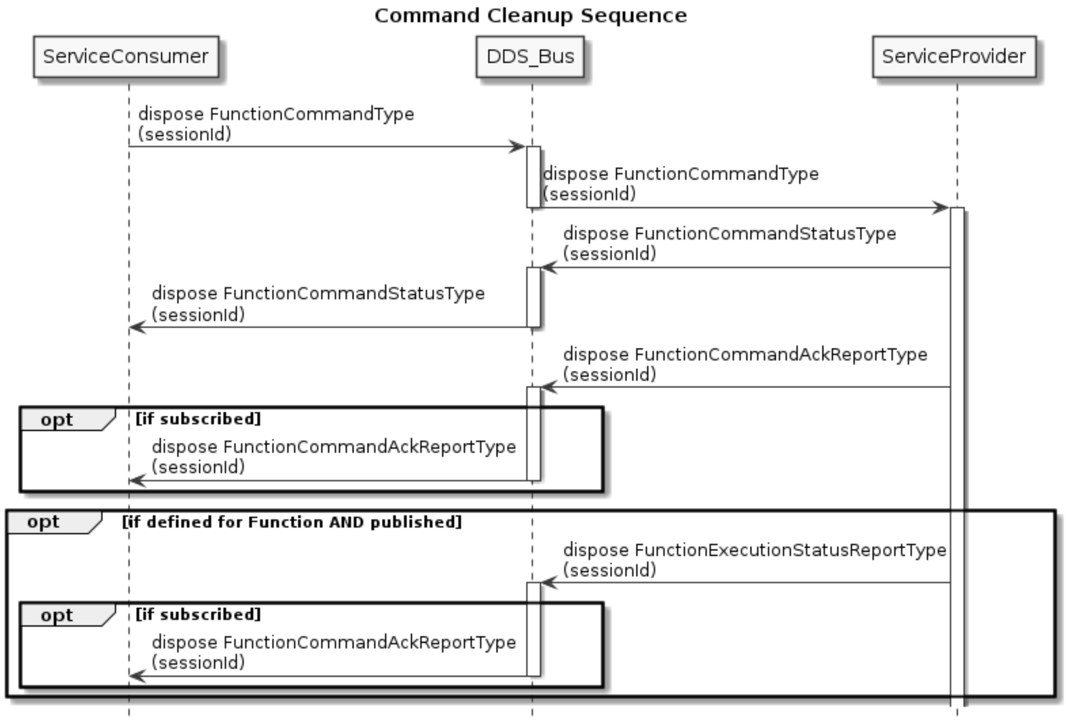

Figure 25: Sequence Diagram Showing Cleanup of the Bus When a Command Has Been Completed and the Service Consumer No Longer Wishes to Maintain the Commanded State.

5.1.6 Command Shutdown Sequence

As part of shutdown, both the Service Provider and Service Consumer are required to perform a shutdown sequence. This shutdown cleans up resources on the DDS bus and informs the system that the Service Provider and Service Consumer are no longer available.

The Service Provider and Service Consumer can shut down in any order. The sequence diagram is shown in Figure 26.

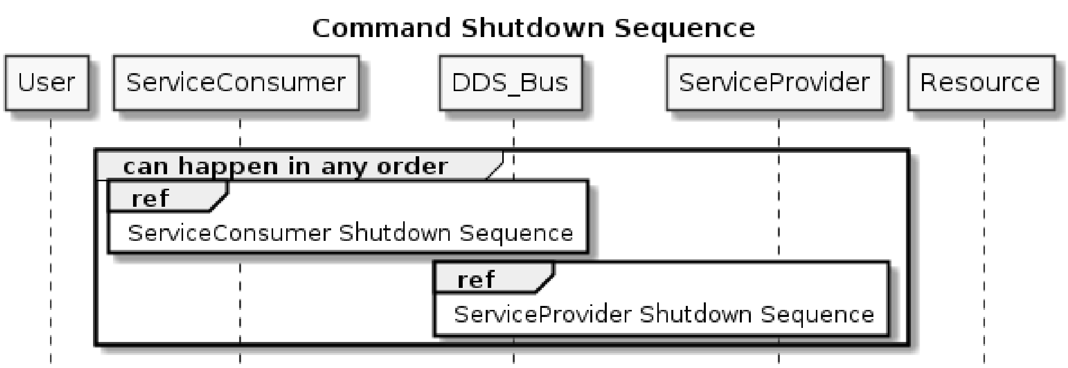

Figure 26: Sequence Diagram for Command Shutdown.

5.1.6.1 Service Provider Shutdown Sequence During shutdown, the Service Provider is required to fail any incomplete requests and then unregisters as a publisher of the FunctionCommandStatusType, FunctionCommandAckReportType, and (if defined for the Function service) the FunctionExecutionStatusReportType.

The Service Provider is also required to unsubscribe from the `FunctionCommandType`.

The Service Provider Shutdown sequence is shown in Figure 27.

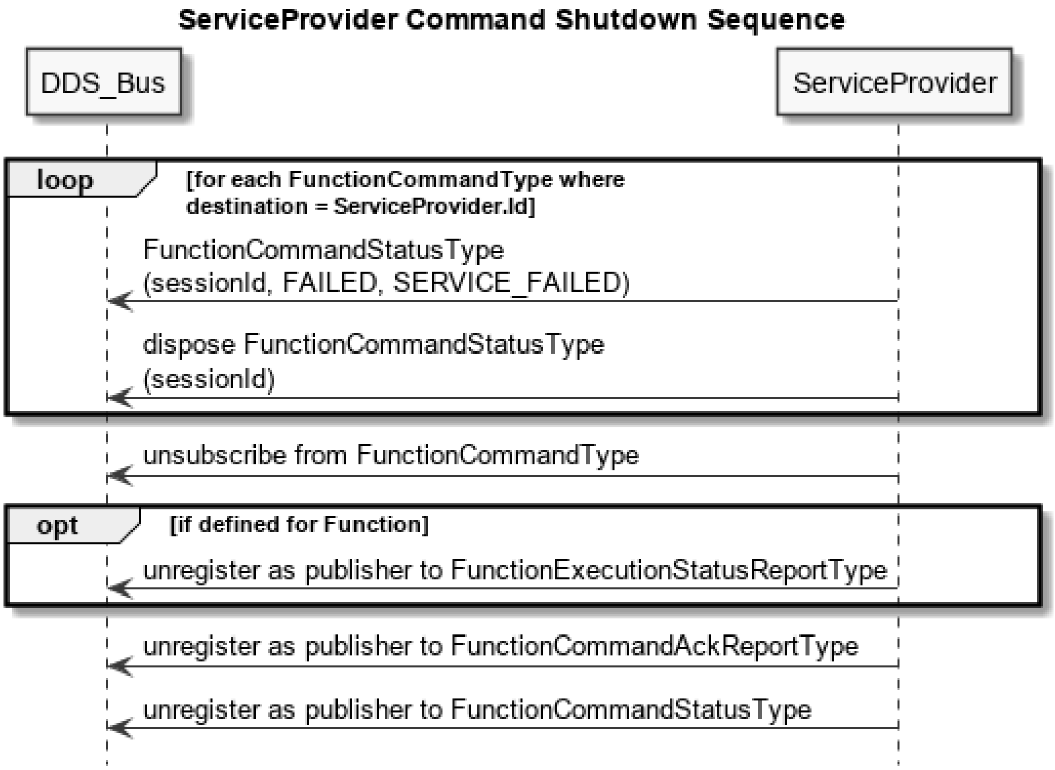

Figure 27: Sequence Diagram for Command Shutdown for Service Providers.

5.1.6.2 Service Consumer Shutdown Sequence During shutdown, the Service Consumer is required to cancel any incomplete requests and then unregister as a publisher of the `FunctionCommandType`.

The Service Consumer is also required to unsubscribe from the `FunctionCommandStatusType`, the `FunctionCommandAckReportType` if subscribed, and the `FunctionExecutionStatusReportType` if defined for the Function service and subscribed.

The Service Consumer Shutdown sequence is shown in Figure 28.

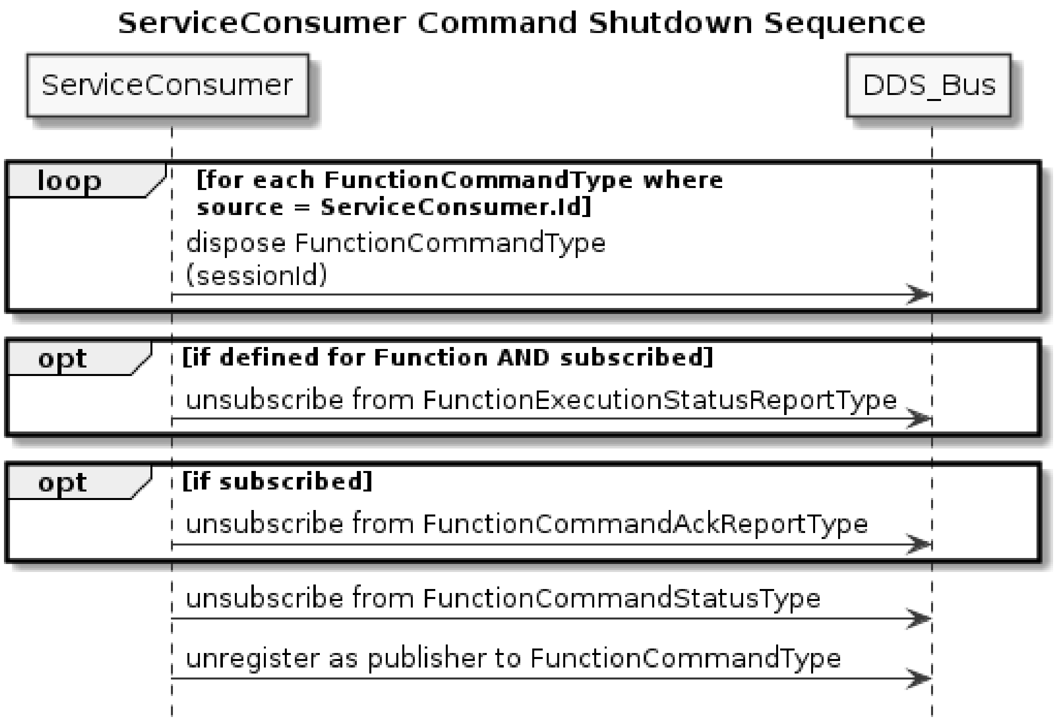

Figure 28: Sequence Diagram for Command Shutdown for Service Consumers.

5.2 Request / Reply

This section defines the flow of control for request/reply over the DDS bus. A request/reply is used to obtain data or status from a specific Service Provider.

A Service Provider is required to reply to all requests it receives. In the case of requests with no query data, this is accomplished via a DDS subscribe. In the case of a request with associated query data, a message with the query data must be published by the requester. To direct a request at a specific Service Provider or set of services, UMAA defines a `destination` GUID as part of requests.

The sequence diagrams in Sections 29 through 33 demonstrate different exchanges between a Service Consumer and Service Provider. Within the diagrams, the dashed arrows represent implementation-specific communications that are outside of UMAA's scope. Additionally, these sequence diagrams are examples of one possible implementation. Other implementations may have different communication patterns between the Service Provider and the Resource, or be implemented completely within the Service Provider process itself (no external Resource). However, in all implementations, UMAA-defined exchanges with the DDS bus between the Service Consumer and Service Provider must happen in the order shown within the sequence diagrams.

5.2.1 Request/Reply without Query Data

Figure 29 shows the sequence of exchanges in the case where there is no specific query data (i.e., the service is always just providing the current data to the bus).

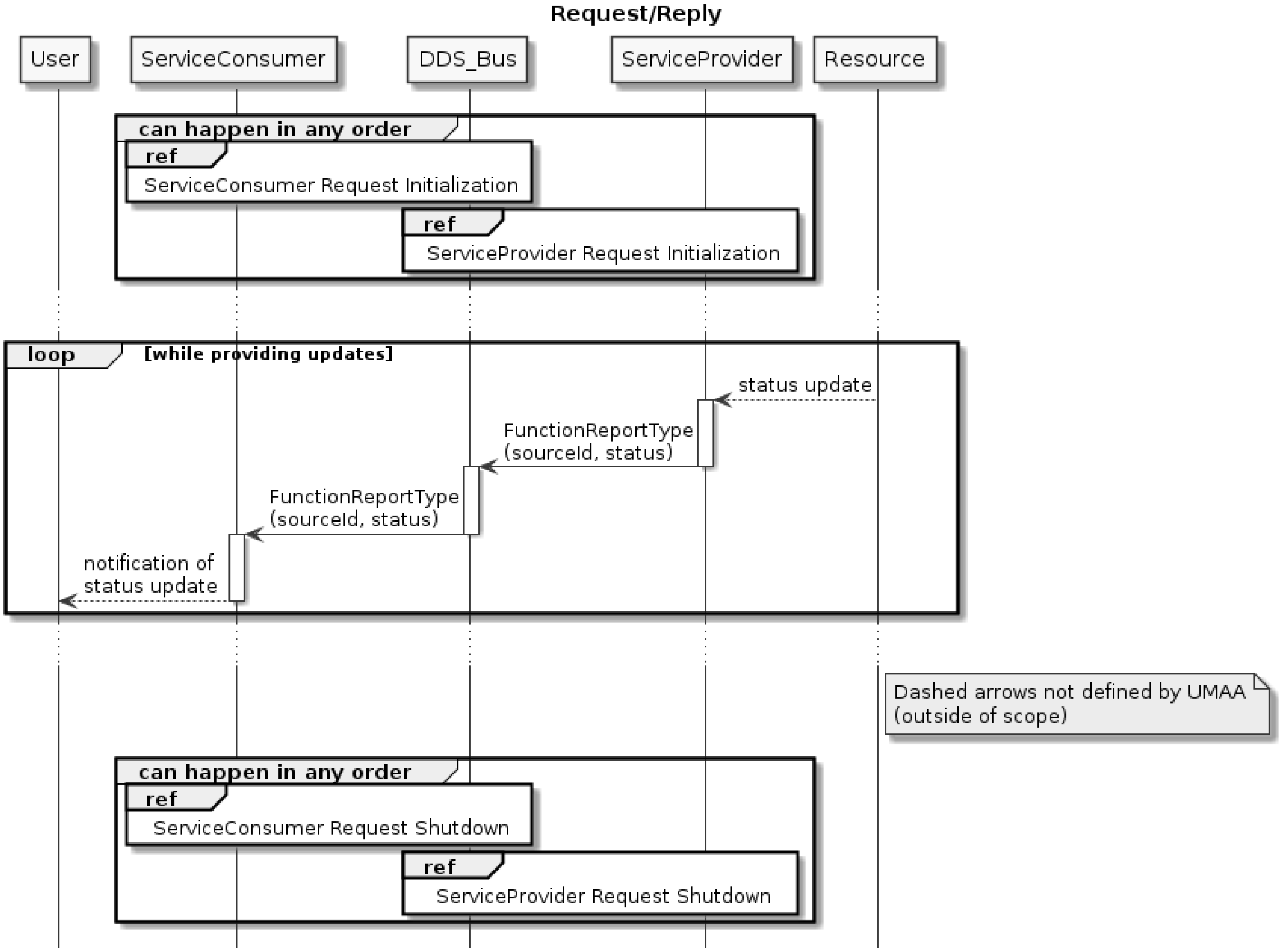

Figure 29: Sequence Diagram for a Request/Reply for Report Data That Does Not Require any Specific Query Data.

5.2.1.1 Service Provider Startup Sequence The Service Provider registers as a publisher of `FunctionReportType`s to be able to respond to requests. The Service Provider must also handle reports that exist on the bus from a previous instantiation, either by providing an immediate update or, if the status is unrecoverable, disposing of the old `FunctionReportType`. This is shown in Figure 30.

As `FunctionReportType` updates are required (either through event-driven changes or periodic updates), the Service Provider publishes the updated data. The DDS bus will deliver the updates to the Service Consumer.

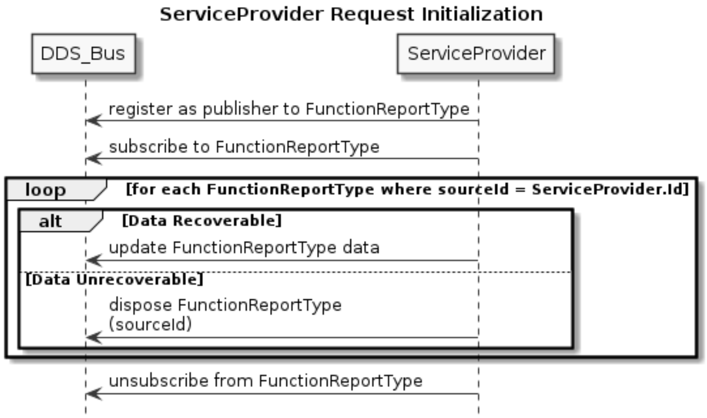

Figure 30: Sequence Diagram for Initialization of a Service Provider to Provide `FunctionReportType`s.

5.2.1.2 Service Consumer Startup Sequence The Service Consumer subscribes to the `FunctionReportType` to signal an outstanding request for updates. This is shown in Figure 31.

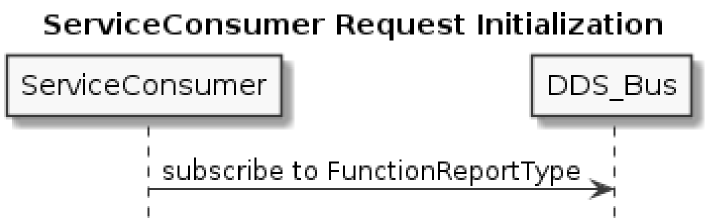

Figure 31: Sequence Diagram for Initialization of a Service Consumer to Request `FunctionReportType`s.

5.2.1.3 Service Provider Shutdown To no longer provide `FunctionReportType`s, the Service Provider disposes of the `FunctionReportType` and unregisters as a publisher of the data (shown in Figure 32).

Figure 32: Sequence Diagram for Shutdown of a Service Provider.

5.2.1.4 Service Consumer Shutdown To no longer request `FunctionReportType`s, the Service Consumer unsubscribes from `FunctionReportType` (shown in Figure 33).

Figure 33: Sequence Diagram for Shutdown of a Service Consumer.

5.2.2 Request/Reply with Query Data

Currently, UMAA does not define any request/reply interactions with query data, but it is expected that some will be defined. When defined, this section will be expanded to describe how they must be used.

6 Maneuver Operations (MO) Services and Interfaces

6.1 Services and Interfaces

The interfaces in the following subsections describe how each UCS-UMAA topic is defined by listing the name, namespace, and member attributes. The "name" corresponds with the message name of a given service interface. The "namespace" defines the scope of the "name" where similar commands are grouped together. The "member attributes" are fields that can be populated with differing data types, e.g. a generic "depth" attribute could be populated with a double data value. Note that using a UCS-UMAA "Topic Name" requires using the fully-qualified namespace plus the topic name.

Each interface topic is referenced by a UMAA service and is defined as either an input or output interface.

Attributes ending in one or more asterisk(s) denote the following:
* = Key (annotated with @key in IDL file; vendors may use different notation to indicate a key field)
† = Optional (annotated with @optional in IDL file; vendors may use different notation to indicate an optional field)

Optional fields should be handled as described in the UMAA Compliance Specification.

Commands issued on the DDS bus must be treated as if they are immutable in UMAA and, therefore, if updated (treated incorrectly as mutable), the resulting service actions are indeterminate and flow control protocols are no longer guaranteed.

A standard feature of the maneuver operations driver services is that a new driving control command to a service overrides the previous driver command to that service.

Operations without DDS Topics

The following operations are all handled directly by DDS. They are marked in the operations tables with a ⊕.

query<...> - All query operations are used to retrieve the correlated report message. For UMAA, this operation is accomplished through subscribing to the appropriate DDS topic.

cancel<...> - All cancel operations are used to nullify the current command. For UMAA, this operation is accomplished through the DDS dispose action on the publisher.

report<...>CancelCommandStatus - All cancel reports are included here to show completeness of the MDE model mapping to UMAA. For UMAA, this operation is not used. Instead, the cancel status is inferred from the associated command status. If the cancel command is successful, the corresponding command will fail with a command status and reason of CANCELED. If the corresponding command status reports COMPLETED, then this cancel command has failed.

6.1.1 ContactManeuverInfluenceStatus

The purpose of this service is to provide the influence of contacts on the maneuvering a vehicle.

Table 8: ContactManeuverInfluenceStatus Operations

Service Requests (Inputs)	**Service Responses (Outputs)**
queryContactManeuverInfluence⊕	reportContactManeuverInfluence

See Section 6.1 for an explanation of the inputs and outputs marked with a ⊕.

6.1.1.1 reportContactManeuverInfluence

Description: This operation is a response to retrieve the current influence of contacts on the maneuvering of the vehicle.

Namespace: UMAA::MO::ContactManeuverInfluenceStatus

Topic: ContactManeuverInfluenceReport

Data Type: ContactManeuverInfluenceReportType

Table 9: ContactManeuverInfluenceReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAAStatus		
contactID	NumericGUID	An identifier of the contact.
maneuver	ContactManeuverInfluenceEnumType	Specifies the maneuvers the vehicle is taking in response to a contact.

6.1.2 CoordinationSituationalSignalStatus

The purpose of this service is to provide the current coordination status of this vessel with respect to situational signals (coordinating with other vehicles).

Table 10: CoordinationSituationalSignalStatus Operations

Service Requests (Inputs)	Service Responses (Outputs)
queryCoordinationSituationalSignal⊕	reportCoordinationSituationalSignal

See Section 6.1 for an explanation of the inputs and outputs marked with a ⊕.

6.1.2.1 reportCoordinationSituationalSignal

Description: This operation is used to report the current coordination status of this vessel with respect to situational signals (coordinating with other vehicles).

Namespace: UMAA::MO::CoordinationSituationalSignalStatus

Topic: CoordinationSituationalSignalReport

Data Type: CoordinationSituationalSignalReportType

Table 11: CoordinationSituationalSignalReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAAStatus		
currentSituation	CoordinationSituationalSignalEnumType	The current coordination status of this vessel with respect to situational signals (coordinating with other vehicles).

6.1.3 DriverConfig

This service provides the ability to dynamically report bounds on various driving attributes. Whereas specification services are inherent limitations of the vehicle system and component capabilities, driving constraints can set limits based on mission profile. For example, setting a maximum speed below the vehicle's inherent capabilities in order to preserve power, or turn rate limitations to enable a sensor to continue to operate effectively through a slow turn. Nothing prohibits multiple implementations of this service to be implemented as needed. A consumer of the constraints service may use the information to limit how it commands the Maneuver Operations driving services. It is intended as a means of specifying safety bounds, optimizing performance (e.g. speed x.x gives optimal efficiency), allowing the payload to affect driving to enable sensor performance (e.g. turn rate limit y.y prevents the towed array from being tangled), and limiting any other use of a vehicles full performance capabilities. In addition, multiple constraints services can be instantiated by different components within a system. It is intended to be informative to higher level of reasoning (e.g. mission management) which may then determine to use the constraint(s) or ignore based on other considerations such as weighing mission importance versus optimizing operation.

Table 12: DriverConfig Operations

Service Requests (Inputs)	Service Responses (Outputs)
queryDriverConfig⊕	reportDriverConfig

See Section 6.1 for an explanation of the inputs and outputs marked with a ⊕.

6.1.3.1 reportDriverConfig

Description: This operation is used to report the driving configuration parameters.

Namespace: UMAA::MO::DriverConfig

Topic: DriverConfigReport

Data Type: DriverConfigReportType

Table 13: DriverConfigReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAAStatus		
achievementTimeout†	DurationSeconds	The amount of time that any boolean achievement execution status attribute that was achieved (true) may be lost (false) for driving commands without failing the command due to short-term temporal conditions. This timeout enables commanded tolerances to be violated for the specified time, allowing the driving service provider to reestablish the achievement (true). If not provided, the driving command will fail immediately upon tolerance violation.
minWaterDepth†	DistanceBSL	The smallest distance from the water surface to the sea floor required by the vehicle to operate.
surfaceBounds	SurfaceCapabilityBoundsType	The bounds of the vehicle while operating on the surface.
underwaterBounds†	UnderwaterCapabilityBoundsType	The bounds of the vehicle while operating submerged.

6.1.4 GlobalDriftControl

The purpose of this service is to maintain a position within the global reference frame and within a defined drift radius. See figure for reference.

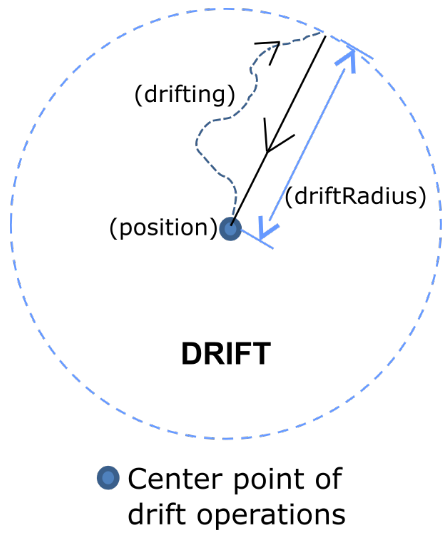

Figure 34: Example Drift Pattern

Table 14: GlobalDriftControl Operations

Service Requests (Inputs)	Service Responses (Outputs)
setGlobalDrift	reportGlobalDriftCommandStatus
queryGlobalDriftCommandAck⊕	reportGlobalDriftCommandAck
queryGlobalDriftExecutionStatus⊕	reportGlobalDriftExecutionStatus
cancelGlobalDriftCommand⊕	reportGlobalDriftCancelCommandStatus⊕

See Section 6.1 for an explanation of the inputs and outputs marked with a ⊕.

6.1.4.1 reportGlobalDriftCommandAck

Description: This operation is used to report the commanded values of the position and global drift and/or time that were commanded to the vehicle in the global coordinate system.

Namespace: UMAA::MO::GlobalDriftControl

Topic: GlobalDriftCommandAckReport

Data Type: GlobalDriftCommandAckReportType

Table 15: GlobalDriftCommandAckReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
command	GlobalDriftCommandType	The source command.

6.1.4.2 reportGlobalDriftCommandStatus

Description: This operation is used to report the status of the global drift command.

Namespace: UMAA::MO::GlobalDriftControl

Topic: GlobalDriftCommandStatus

Data Type: GlobalDriftCommandStatusType

Table 16: GlobalDriftCommandStatusType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatus		

6.1.4.3 reportGlobalDriftExecutionStatus

Description: This operation is used to report the current state of the vehicle drift in the global coordinate system.

Namespace: UMAA::MO::GlobalDriftControl

Topic: GlobalDriftExecutionStatusReport

Data Type: GlobalDriftExecutionStatusReportType

Table 17: GlobalDriftExecutionStatusReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
distanceFromReference	Distance	Defines the distance from the reference position.
globalDriftState	GlobalDriftStateType	Defines the state of the global drift.
timeDriftAchieved	DateTime	Defines the absolute time at which drift is estimated to be achieved or was actually first achieved.
timeDriftCompleted†	DateTime	Defines the absolute time at which the drift is estimated to be completed (optional in case duration is forever).

6.1.4.4 setGlobalDrift

Description: This operation is used to set the desired position in the global coordinate system given the specified global drift and/or time. If the command attributes do not specify a determinate end of execution, the consumer must perform a "cancel" of the command to initiate the end of command execution.

Namespace: UMAA::MO::GlobalDriftControl

Topic: GlobalDriftCommand

Data Type: GlobalDriftCommandType

Table 18: GlobalDriftCommandType Message Definition

Attribute Name	Attribute Type	Attribute Description
	Additional fields included from UMAA::UMAACommand	
driftTolerance	Distance	Defines the drift radius that specifies the maximum distance from the reference position the vehicle is allowed to drift.
elevation	ElevationType	Defines the elevation to maintain when within the driftTolerance of the drift position.
endTime†	DateTimeRequirement	Specifies the end of the command execution time period for the drift operation; if not specified runs indefinitely until command is changed externally.
position	GeoPosition2DRequirement	Defines the reference position for drifting.
speed	SpeedControlType	The desired speed to return to the drift position when the drift tolerance is exceeded.
transitElevation	ElevationType	The elevation used when driving back to get within the driftTolerance of the drift position.
transitSpeed	SpeedControlType	The speed at which one drives to get within the driftTolerance of the drift position.

6.1.5 GlobalFigure8Control

Intended to command the vehicle about a desired position in the global coordinate frame using a figure 8 pattern. See figure for reference.

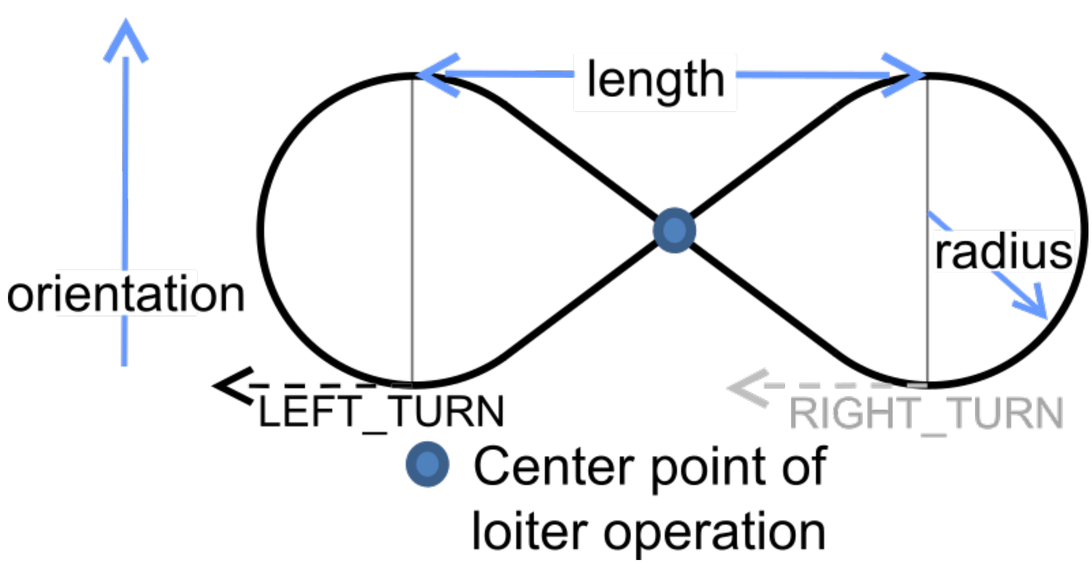

Figure 35: Example Figure 8 Pattern

Table 19: GlobalFigure8Control Operations

Service Requests (Inputs)	Service Responses (Outputs)
setGlobalFigure8	reportGlobalFigure8CommandStatus
queryGlobalFigure8CommandAck⊕	reportGlobalFigure8CommandAck
queryGlobalFigure8ExecutionStatus⊕	reportGlobalFigure8ExecutionStatus
cancelGlobalFigure8Command⊕	reportGlobalFigure8CancelCommandStatus⊕

See Section 6.1 for an explanation of the inputs and outputs marked with a ⊕.

6.1.5.1 reportGlobalFigure8CommandAck

Description: This operation is used to report the commanded values of the position and pattern and/or time that were commanded to the vehicle in the global coordinate system.

Namespace: UMAA::MO::GlobalFigure8Control

Topic: GlobalFigure8CommandAckReport

Data Type: GlobalFigure8CommandAckReportType

Table 20: GlobalFigure8CommandAckReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
command	GlobalFigure8CommandType	The source command.

6.1.5.2 reportGlobalFigure8CommandStatus

Description: This operation is used to report the status of the global figure 8 command.

Namespace: UMAA::MO::GlobalFigure8Control

Topic: GlobalFigure8CommandStatus

Data Type: GlobalFigure8CommandStatusType

Table 21: GlobalFigure8CommandStatusType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatus		

6.1.5.3 reportGlobalFigure8ExecutionStatus

Description: This operation is used to report the current position and pattern and/or time of the vehicle based in the global coordinate system.

Namespace: UMAA::MO::GlobalFigure8Control

Topic: GlobalFigure8ExecutionStatusReport

Data Type: GlobalFigure8ExecutionStatusReportType

Table 22: GlobalFigure8ExecutionStatusReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
globalFigure8State	GlobalFigure8StateType	Defines the state of the global figure 8.
timePatternAchieved	DateTime	The absolute time at which the figure 8 pattern is estimated to be achieved or was actually first achieved.
timePatternCompleted†	DateTime	The absolute time at which the figure 8 pattern is estimated to be completed.

6.1.5.4 setGlobalFigure8

Description: This operation is used to set the desired position in the global coordinate system given the specified figure 8 pattern and/or time. If the command attributes do not specify a determinate end of execution, the consumer must perform a "cancel" of the command to initiate the end of command execution.

Namespace: UMAA::MO::GlobalFigure8Control

Topic: GlobalFigure8Command

Data Type: GlobalFigure8CommandType

Table 23: GlobalFigure8CommandType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommand		
crossTrackTolerance	Distance	The amount of error in position allowed from the pattern being executed.
elevation	ElevationType	The elevation used for the vehicle. This value is 0 for USVs.
endTime†	DateTimeRequirement	Specifies the end of the command execution time period for the pattern; if not specified runs indefinitely until command is changed externally.
length	Distance	Describes the length between the semicircles at either end of the figure 8 the vehicle should stay in.
orientation	DirectionRequirementType	Defines the orientation of the figure 8, measured perpendicular to the length axis.

Attribute Name	Attribute Type	Attribute Description
position	GeoPosition2D	The desired figure 8 position of the vehicle in the global coordinate system.
radius	Distance	Describes the radius of the semicircles at either end of the figure 8 the vehicle should stay in.
speed	SpeedControlType	The desired speed of the vehicle.
transitElevation	ElevationType	The elevation used while driving to the figure 8 track. Surface-based vehicles must specify this value as 0.
transitSpeed	SpeedControlType	The speed at which one drives to the figure 8 track.
turnDirection	WaterTurnDirectionEnumType	The desired turn direction for the figure 8 pattern of the vehicle.

6.1.6 GlobalHoverControl

The function of this service is to command the vehicle to hover in a desired position in the global coordinate frame.

Table 24: GlobalHoverControl Operations

Service Requests (Inputs)	Service Responses (Outputs)
setGlobalHover	reportGlobalHoverCommandStatus
queryGlobalHoverCommandAck⊕	reportGlobalHoverCommandAck
queryGlobalHoverExecutionStatus⊕	reportGlobalHoverExecutionStatus
cancelGlobalHoverCommand⊕	reportGlobalHoverCancelCommandStatus⊕

See Section 6.1 for an explanation of the inputs and outputs marked with a ⊕.

6.1.6.1 reportGlobalHoverCommandAck

Description: This operation is used to report the commanded values of the position or time that was commanded to the vehicle in the global coordinate system.

Namespace: UMAA::MO::GlobalHoverControl

Topic: GlobalHoverCommandAckReport

Data Type: GlobalHoverCommandAckReportType

Table 25: GlobalHoverCommandAckReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
command	GlobalHoverCommandType	The source command.

6.1.6.2 reportGlobalHoverCommandStatus

Description: This operation is used to report the status of the global hover command.

Namespace: UMAA::MO::GlobalHoverControl

Topic: GlobalHoverCommandStatus

Data Type: GlobalHoverCommandStatusType

Table 26: GlobalHoverCommandStatusType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatus		

6.1.6.3 reportGlobalHoverExecutionStatus

Description: This operation is used to report the current position or time that the vehicle was hovering based in the global coordinate system.

Namespace: UMAA::MO::GlobalHoverControl

Topic: GlobalHoverExecutionStatusReport

Data Type: GlobalHoverExecutionStatusReportType

Table 27: GlobalHoverExecutionStatusReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
globalHoverState	GlobalHoverStateType	Defines the state of the global hover.
timeHoverAchieved	DateTime	The absolute time at which hover is estimated to be achieved or was actually first achieved.
timeHoverCompleted†	DateTime	The absolute time at which the hover is estimated to be completed (optional in case duration is forever).

6.1.6.4 setGlobalHover

Description: This operation is used to set the desired hover position in the global coordinate system given the desired location and/or time. If the command attributes do not specify a determinate end of execution, the consumer must perform a "cancel" of the command to initiate the end of command execution.

Namespace: UMAA::MO::GlobalHoverControl

Topic: GlobalHoverCommand

Data Type: GlobalHoverCommandType

Table 28: GlobalHoverCommandType Message Definition

Attribute Name	Attribute Type	Attribute Description
	Additional fields included from UMAA::UMAACommand	
controlPriority	HoverKindEnumType	The desired priority to hover at the specified point.
elevation	ElevationType	The elevation used for the vehicle. This value is 0 for USVs.
endTime†	DateTimeRequirement	Specifies the end of the command execution time period for the hover; if not specified runs indefinitely until command is changed externally.
heading†	DirectionRequirementType	Defines the heading that the vehicle must maintain for hovering. When not specified, the system will determine the best heading (e.g. current heading, into the wind/current, etc.)
position	GeoPosition2DRequirement	The desired hover position of the vehicle in the global coordinate system.
transitElevation	ElevationType	The elevation used while driving to the hover location. Surface-based vehicles must specify this value as 0.
transitSpeed	VariableSpeedControlType	The speed at which one drives to the hover location.

6.1.7 GlobalRacetrackControl

The purpose of this service is to command the vehicle into a racetrack pattern about a desired position in the global coordinate frame. The start location on the Racetrack and the path to this start location is system dependent.
See figure for reference.

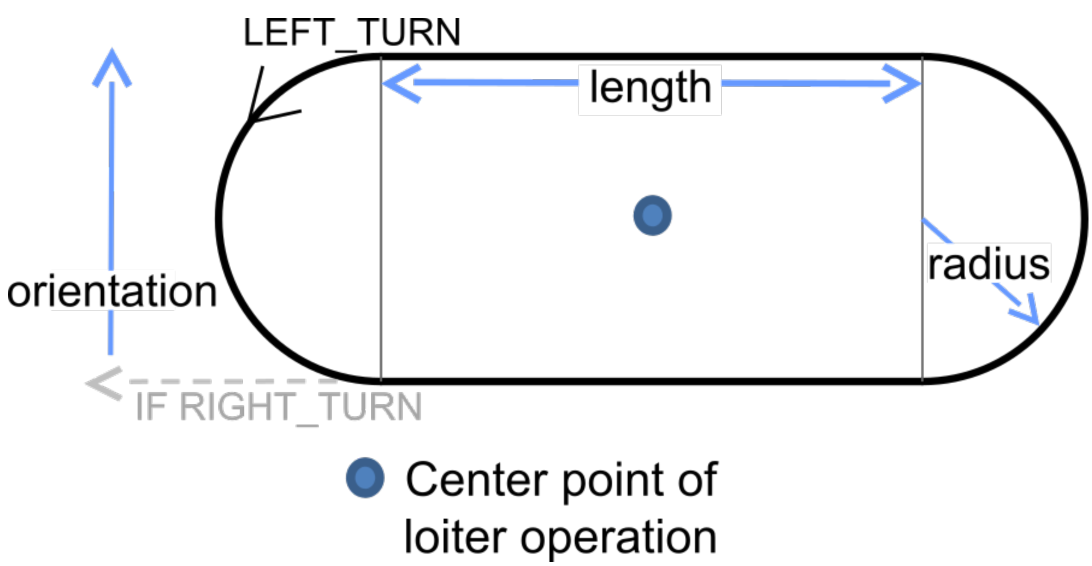

Figure 36: Example Racetrack Pattern

Table 29: GlobalRacetrackControl Operations

Service Requests (Inputs)	Service Responses (Outputs)
setGlobalRacetrack	reportGlobalRacetrackCommandStatus
queryGlobalRacetrackCommandAck⊕	reportGlobalRacetrackCommandAck
queryGlobalRacetrackExecutionStatus⊕	reportGlobalRacetrackExecutionStatus
cancelGlobalRacetrackCommand⊕	reportGlobalRacetrackCancelCommandStatus⊕

See Section 6.1 for an explanation of the inputs and outputs marked with a ⊕.

6.1.7.1 reportGlobalRacetrackCommandAck

Description: This operation is used to report the parameters of the racetrack pattern that was commanded to the vehicle in the global coordinate system.

Namespace: UMAA::MO::GlobalRacetrackControl

Topic: GlobalRacetrackCommandAckReport

Data Type: GlobalRacetrackCommandAckReportType

Table 30: GlobalRacetrackCommandAckReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
command	GlobalRacetrackCommandType	The source command.

6.1.7.2 reportGlobalRacetrackCommandStatus

Description: This operation is used to report the status of the global racetrack command.

Namespace: UMAA::MO::GlobalRacetrackControl

Topic: GlobalRacetrackCommandStatus

Data Type: GlobalRacetrackCommandStatusType

Table 31: GlobalRacetrackCommandStatusType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatus		

6.1.7.3 reportGlobalRacetrackExecutionStatus

Description: This operation is used to report the current racetrack pattern execution status based on the global coordinate system.

Namespace: UMAA::MO::GlobalRacetrackControl

Topic: GlobalRacetrackExecutionStatusReport

Data Type: GlobalRacetrackExecutionStatusReportType

Table 32: GlobalRacetrackExecutionStatusReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
globalRacetrackState	GlobalRacetrackStateType	Defines the state of the global racetrack.
timePatternAchieved	DateTime	The absolute time at which the racetrack pattern is estimated to be achieved or was actually first achieved.
timePatternCompleted†	DateTime	The absolute time at which the racetrack pattern is estimated to be completed.

6.1.7.4 setGlobalRacetrack

Description: This operation is used to initiate the racetrack pattern based on the specified position in the global coordinate system. If the command attributes do not specify a determinate end of execution, the consumer must perform a "cancel" of the command to initiate the end of command execution.

Namespace: UMAA::MO::GlobalRacetrackControl

Topic: GlobalRacetrackCommand

Data Type: GlobalRacetrackCommandType

Table 33: GlobalRacetrackCommandType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommand		
crossTrackTolerance	Distance	The amount of error in position allowed from the pattern being executed.
elevation	ElevationType	The elevation used for the vehicle. This value is 0 for USVs.
endTime†	DateTimeRequirement	Specifies the end of the command execution time period for the pattern; if not specified runs indefinitely until command is changed externally.
length	Distance	Describes the length between the semicircles at either end of the racetrack the vehicle should stay in.
orientation	DirectionRequirementType	Defines the orientation of the racetrack, measured perpendicular to the length axis.

Attribute Name	Attribute Type	Attribute Description
position	GeoPosition2D	The desired racetrack position of the vehicle in the global coordinate system.
radius	Distance	Describes the radius of the semicircles at either end of the racetrack the vehicle should stay in.
speed	SpeedControlType	The desired speed of the vehicle.
transitElevation	ElevationType	The elevation used while driving to the racetrack. Surface-based vehicles must specify this value as 0.
transitSpeed	SpeedControlType	The speed at which one drives to the racetrack.
turnDirection	WaterTurnDirectionEnumType	The desired turn direction for the racetrack pattern of the vehicle.

6.1.8 GlobalRegularPolygonControl

Intended to command the vehicle about a desired position in the global coordinate frame using a regular polygon pattern circumscribed on a circle. The start location on the RegularPolygon and the path to this start location is system dependent. See figure for reference.

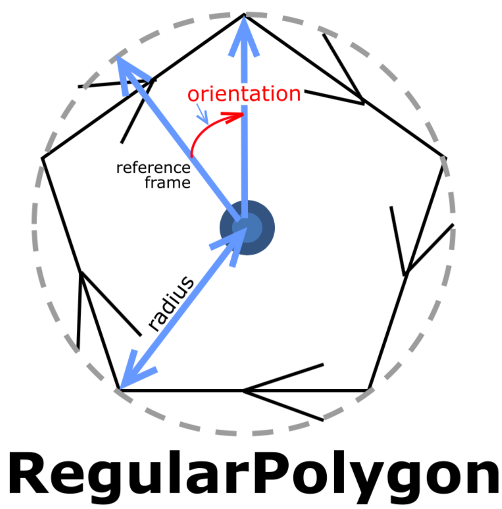

Figure 37: Example Polygon Pattern

Table 34: GlobalRegularPolygonControl Operations

Service Requests (Inputs)	Service Responses (Outputs)
setGlobalRegularPolygon	reportGlobalRegularPolygonCommandStatus
queryGlobalRegularPolygonCommandAck⊕	reportGlobalRegularPolygonCommandAck
queryGlobalRegularPolygonExecutionStatus⊕	reportGlobalRegularPolygonExecutionStatus
cancelGlobalRegularPolygonCommand⊕	reportGlobalRegularPolygonCancelCommandStatus⊕

See Section 6.1 for an explanation of the inputs and outputs marked with a ⊕.

6.1.8.1 reportGlobalRegularPolygonCommandAck

Description: This operation is used to report the commanded values of the position and pattern and/or time that were commanded to the vehicle in the global coordinate system.

Namespace: UMAA::MO::GlobalRegularPolygonControl

Topic: GlobalRegularPolygonCommandAckReport

Data Type: GlobalRegularPolygonCommandAckReportType

Table 35: GlobalRegularPolygonCommandAckReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
command	GlobalRegularPolygonCommandType	The source command.

6.1.8.2 reportGlobalRegularPolygonCommandStatus

Description: This operation is used to report the status of the global regular polygon command.

Namespace: UMAA::MO::GlobalRegularPolygonControl

Topic: GlobalRegularPolygonCommandStatus

Data Type: GlobalRegularPolygonCommandStatusType

Table 36: GlobalRegularPolygonCommandStatusType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatus		

6.1.8.3 reportGlobalRegularPolygonExecutionStatus

Description: This operation is used to report the current position and pattern and/or time of the vehicle based in the global coordinate system.

Namespace: UMAA::MO::GlobalRegularPolygonControl

Topic: GlobalRegularPolygonExecutionStatusReport

Data Type: GlobalRegularPolygonExecutionStatusReportType

Table 37: GlobalRegularPolygonExecutionStatusReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
globalRegularPolygonState	GlobalRegularPolygonStateType	Defines the state of the global regular polygon.
timePatternAchieved	DateTime	The absolute time at which the polygon pattern is estimated to be achieved or was actually first achieved.
timePatternCompleted†	DateTime	The absolute time at which the polygon pattern is estimated to be completed.

6.1.8.4 setGlobalRegularPolygon

Description: This operation is used to set the desired position in the global coordinate system given the specified regular polygon pattern and/or time. If the command attributes do not specify a determinate end of execution, the consumer must perform a "cancel" of the command to initiate the end of command execution.

Namespace: UMAA::MO::GlobalRegularPolygonControl

Topic: GlobalRegularPolygonCommand

Data Type: GlobalRegularPolygonCommandType

Table 38: GlobalRegularPolygonCommandType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommand		
crossTrackTolerance	Distance	The amount of error in position allowed from the pattern being executed.
diameter	Distance	The diameter of a circumscribed circle around the polygon.
elevation	ElevationType	The elevation used for the vehicle. This value is 0 for USVs.
endTime†	DateTimeRequirement	Specifies the end of the command execution time period for the pattern; if not specified runs indefinitely until command is changed externally.
numberSides	SidesCount	The number of sides on the polygon.
orientation	DirectionRequirementType	Defines the orientation from the reference position of the polygon to one point on the polygon.
position	GeoPosition2D	The desired polygon position of the vehicle in the global coordinate system.
speed	SpeedControlType	The desired speed of the vehicle.
transitElevation	ElevationType	The elevation used while driving to the polygon track. Surface-based vehicles must specify this value as 0.
transitSpeed	SpeedControlType	The speed at which one drives to the polygon track.
turnDirection	WaterTurnDirectionEnumType	The desired turn direction for the polygon pattern of the vehicle.

6.1.9 GlobalVectorControl

The purpose of this service is to command the vehicle to maintain a provided speed and altitude or depth (if supported).

Table 39: GlobalVectorControl Operations

Service Requests (Inputs)	Service Responses (Outputs)
setGlobalVector	reportGlobalVectorCommandStatus
queryGlobalVectorCommandAck⊕	reportGlobalVectorCommandAck
queryGlobalVectorExecutionStatus⊕	reportGlobalVectorExecutionStatus
cancelGlobalVectorCommand⊕	reportGlobalVectorCancelCommandStatus⊕

See Section 6.1 for an explanation of the inputs and outputs marked with a ⊕.

6.1.9.1 reportGlobalVectorCommandAck

Description: This operation is used to report the current commanded values of the speed and depth or altitude to a vehicle in the global coordinate system.

Namespace: UMAA::MO::GlobalVectorControl

Topic: GlobalVectorCommandAckReport

Data Type: GlobalVectorCommandAckReportType

Table 40: GlobalVectorCommandAckReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
command	GlobalVectorCommandType	The source command.

6.1.9.2 reportGlobalVectorCommandStatus

Description: This operation is used to report the status of the global vector command.

Namespace: UMAA::MO::GlobalVectorControl

Topic: GlobalVectorCommandStatus

Data Type: GlobalVectorCommandStatusType

Table 41: GlobalVectorCommandStatusType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatus		

6.1.9.3 reportGlobalVectorExecutionStatus

Description: This operation is used to report the current vector data based in the global coordinate system.

Namespace: UMAA::MO::GlobalVectorControl

Topic: GlobalVectorExecutionStatusReport

Data Type: GlobalVectorExecutionStatusReportType

Table 42: GlobalVectorExecutionStatusReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
elevationAchieved	boolean	When the vector is executing, this indicates that the elevation requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
speedAchieved	boolean	When the vector is executing, this indicates that the speed requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.

6.1.9.4 setGlobalVector

Description: This operation is used to command the speed and altitude or depth of a vehicle in the global coordinate system. If the command attributes do not specify a determinate end of execution, the consumer must perform a "cancel" of the command to initiate the end of command execution.

Namespace: UMAA::MO::GlobalVectorControl

Topic: GlobalVectorCommand

Data Type: GlobalVectorCommandType

Table 43: GlobalVectorCommandType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommand		
depthChangePitch†	PitchYNEDRequirement	The desired angle of the vehicle when traversing to the requested elevation for UUVs.

Attribute Name	Attribute Type	Attribute Description
direction	DirectionRequirementType	The direction the vehicle is traveling regardless of its attitude.
directionMode	DirectionModeEnumType	Specifies the vehicle direction mode.
elevation	ElevationType	Specifies the elevation of the vector. This value is 0 for USVs.
endTime†	DateTimeRequirement	Specifies the end of the command execution time period for the vector; if not specified runs indefinitely until command is changed externally or another terminating condition occurs.
speed	SpeedControlType	The desired speed of the vehicle.

6.1.10 GlobalWaypointControl

The purpose of this service is to command the vehicle to traverse through a series of waypoints, each with a desired speed and the option to maintain track.

Table 44: GlobalWaypointControl Operations

Service Requests (Inputs)	Service Responses (Outputs)
setGlobalWaypoint	reportGlobalWaypointCommandStatus
queryGlobalWaypointCommandAck⊕	reportGlobalWaypointCommandAck
queryGlobalWaypointExecutionStatus⊕	reportGlobalWaypointExecutionStatus
cancelGlobalWaypointCommand⊕	reportGlobalWaypointCancelCommandStatus⊕

See Section 6.1 for an explanation of the inputs and outputs marked with a ⊕.

6.1.10.1 reportGlobalWaypointCommandAck

Description: This operation is used to report the commanded values of the waypoint data based in the global coordinate system.

Namespace: UMAA::MO::GlobalWaypointControl

Topic: GlobalWaypointCommandAckReport

Data Type: GlobalWaypointCommandAckReportType

Table 45: GlobalWaypointCommandAckReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
command	GlobalWaypointCommandType	The source command.

6.1.10.2 reportGlobalWaypointCommandStatus

Description: This operation is used to report the status of the global waypoint command.

Namespace: UMAA::MO::GlobalWaypointControl

Topic: GlobalWaypointCommandStatus

Data Type: GlobalWaypointCommandStatusType

Table 46: GlobalWaypointCommandStatusType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatus		

6.1.10.3 reportGlobalWaypointExecutionStatus

Description: This operation is used to report execution status details related to a commanded series of waypoint data based in the global coordinate system.

Namespace: UMAA::MO::GlobalWaypointControl

Topic: GlobalWaypointExecutionStatusReport

Data Type: GlobalWaypointExecutionStatusReportType

Table 47: GlobalWaypointExecutionStatusReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
arrivalTime	DateTime	The arrival time of the end of the route.
crossTrackError	Distance	Defines the current cross track error (only valid if trackTolerance is defined).
cumulativeDistance	Distance	Defines the ground distance travel from the start of the route to this point.
distanceRemaining	Distance	Defines the amount of distance remaining from a point to the end of the route.
distanceToWaypoint	Distance	Defines the remaining distance to the current waypoint.
elevationAchieved	boolean	When the waypoint is executing, this indicates that the elevation requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
speedAchieved	boolean	When the waypoint is executing, this indicates that the speed requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
timeToWaypoint	DateTime	The absolute time at which the waypoint is estimated to be achieved or was actually first achieved.

Attribute Name	Attribute Type	Attribute Description
trackLineAchieved	boolean	When the waypoint is executing, this indicates that the track line requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
waypointsRemaining	Count	Defines the remaining number of waypoints, which includes the current waypoint.
waypointID*	NumericGUID	Defines the current waypoint ID.

6.1.10.4 setGlobalWaypoint

Description: This operation is used to command a series of waypoint data based in the global coordinate system.

Namespace: UMAA::MO::GlobalWaypointControl

Topic: GlobalWaypointCommand

Data Type: GlobalWaypointCommandType

Table 48: GlobalWaypointCommandType Message Definition

Attribute Name	Attribute Type	Attribute Description
	Additional fields included from UMAA::UMAACommand	
waypoints→listID	LargeList<GlobalWaypointType>	The desired series of waypoints in the global coordinate system. This attribute is implemented as a large list, see subsection 3.8 for an explanation. The associated topic is UMAA::MO::GlobalWaypointControl::GlobalWaypointCommandWaypointsListElement.

6.1.11 PrimitiveDriverControl

This service provides mobility in six degrees of freedom using a percent of available effort in each direction. Additionally, no power plant is implied and the service functions strictly in an open loop manner, i.e., a velocity is not commanded or held since that requires a speed sensor. The service definition makes no assertion about the preventative actions that must be taken to avoid unintended consequences, such as losing positive control when given a zero propulsive effort. This service uses "effort" as a relative measure of the amount of drive power. This measure is intentionally kept agnostic of the underlying control system for portability across hardware types. As a result, the implementation of an "effort" driver may map the request to a percent of maximum current of an electric motor, fluid pressure of a hydraulic system, duty-cycle of a pulse-width modulated controller, or position of a control lever. These examples are meant to be illustrative; the actual mapping is not restricted so long as it can be expressed as a percent of some maximum.

Table 49: PrimitiveDriverControl Operations

Service Requests (Inputs)	Service Responses (Outputs)
setPrimitiveDriver	reportPrimitiveDriverCommandStatus
queryPrimitiveDriverCommandAck⊕	reportPrimitiveDriverCommandAck
queryPrimitiveDriverExecutionStatus⊕	reportPrimitiveDriverExecutionStatus

Service Requests (Inputs)	Service Responses (Outputs)
cancelPrimitiveDriverCommand⊕	reportPrimitiveDriverCancelCommandStatus⊕

See Section 6.1 for an explanation of the inputs and outputs marked with a ⊕.

6.1.11.1 reportPrimitiveDriverCommandAck

Description: This operation is used to report the current effort command.

Namespace: UMAA::MO::PrimitiveDriverControl

Topic: PrimitiveDriverCommandAckReport

Data Type: PrimitiveDriverCommandAckReportType

Table 50: PrimitiveDriverCommandAckReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
command	PrimitiveDriverCommandType	The source command.

6.1.11.2 reportPrimitiveDriverCommandStatus

Description: This operation is used to report the status of the effort command.

Namespace: UMAA::MO::PrimitiveDriverControl

Topic: PrimitiveDriverCommandStatus

Data Type: PrimitiveDriverCommandStatusType

Table 51: PrimitiveDriverCommandStatusType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatus		

6.1.11.3 reportPrimitiveDriverExecutionStatus

Description: This operation is used to report the current mobility in the vehicle coordinate frame.

Namespace: UMAA::MO::PrimitiveDriverControl

Topic: PrimitiveDriverExecutionStatusReport

Data Type: PrimitiveDriverExecutionStatusReportType

Table 52: PrimitiveDriverExecutionStatusReportType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
propulsiveLinearEffort	LinearEffort	The current propulsive linear effort (X, Y, Z) in percent.
propulsiveRotationalEffort	RotationalEffort	The current propulsive rotational effort (X, Y, Z) in percent.
resistiveLinearEffort	LinearEffort	The current resistive linear effort (X, Y, Z) in percent.
resistiveRotationalEffort	RotationalEffort	The current resistive rotational effort (X, Y, Z) in percent.

6.1.11.4 setPrimitiveDriver

Description: This operation is used to set the mobility of the vehicle using the effort. The consumer must perform a "cancel" of the command to initiate the end of command execution as this command has no determinate end of execution.

Namespace: UMAA::MO::PrimitiveDriverControl

Topic: PrimitiveDriverCommand

Data Type: PrimitiveDriverCommandType

Table 53: PrimitiveDriverCommandType Message Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommand		
propulsiveLinearEffort	LinearEffort	The desired propulsive linear effort (X, Y, Z) in percent. Propulsive linear effort represents an action that results in a linear motion along the respective axis.
propulsiveRotationalEffort	RotationalEffort	The desired propulsive rotational effort (X, Y, Z) in percent. Propulsive rotational effort represents an action that results in a rotational motion about the respective axis.
resistiveLinearEffort	LinearEffort	The desired resistive linear effort (X, Y, Z) in percent. Resistive linear effort represents an action that impedes linear motion along the respective axis.
resistiveRotationalEffort	RotationalEffort	The desired resistive rotational effort (X, Y, Z) in percent. Resistive rotational effort represents an action that impedes rotational motion about the respective axis.

6.2 Common Data Types

Common data types define DDS types that are referenced throughout the UMAA model. These DDS types are considered common because they can be re-used as the data type for many attributes defined in service interface topics, interface topics, and other common data types. These data types are not intended to be directly published to/subscribed as DDS topics.

6.2.1 UCSMDEInterfaceSet

Namespace: UMAA::UCSMDEInterfaceSet

Description: Defines the common UCSMDE Interface Set Message Fields.

Table 54: UCSMDEInterfaceSet Structure Definition

Attribute Name	Attribute Type	Attribute Description
timeStamp	DateTime	The time at which the data is valid.

6.2.2 UMAACommand

Namespace: UMAA::UMAACommand

Description: Defines the common UMAA Command Message Fields.

Table 55: UMAACommand Structure Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UCSMDEInterfaceSet		
source*	NumericGUID	The unique identifier of the originating source of the command interface.
destination*	NumericGUID	The unique identifier of the destination of the command interface.
sessionID*	NumericGUID	The identifier of the session.

6.2.3 UMAAStatus

Namespace: UMAA::UMAAStatus

Description: Defines the common UMAA Status Message Fields.

Table 56: UMAAStatus Structure Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UCSMDEInterfaceSet		
source*	NumericGUID	The unique identifier of the originating source of the status interface.

6.2.4 UMAACommandStatusBase

Namespace: UMAA::UMAACommandStatusBase

Description: Defines the common UMAA Command Status Base Message Fields.

Table 57: UMAACommandStatusBase Structure Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UCSMDEInterfaceSet		
source*	NumericGUID	The unique identifier of the originating source of the command status interface.
sessionID*	NumericGUID	The identifier of the session.

6.2.5 UMAACommandStatus

Namespace: UMAA::UMAACommandStatus

Description: Defines the common UMAA Command Status Message Fields.

Table 58: UMAACommandStatus Structure Definition

Attribute Name	Attribute Type	Attribute Description
Additional fields included from UMAA::UMAACommandStatusBase		
commandStatus	CommandStatusEnumType	The status of the command.
commandStatusReason	CommandStatusReasonEnumType	The reason for the status of the command.
logMessage	StringLongDescription	Human-readable description related to response. Systems should not parse or use any information from this for processing purposes.

6.2.6 DateTime

Namespace: UMAA::Measurement::DateTime

Description: Describes an absolute time. Conforms with POSIX time standard (IEEE Std 1003.1-2017) epoch reference point of January 1st, 1970 00:00:00 UTC.

Table 59: DateTime Structure Definition

Attribute Name	Attribute Type	Attribute Description
seconds	DateTimeSeconds	The number of seconds offset from the standard POSIX (IEEE Std 1003.1-2017) epoch reference point of January 1st, 1970 00:00:00 UTC.
nanoseconds	DateTimeNanoSeconds	The number of nanoseconds elapsed within the current DateTimeSecond.

6.2.7 Acceleration3DPlatformXYZ

Namespace: UMAA::Common::Measurement::Acceleration3DPlatformXYZ

Description: Specifies the platform's rate of change of velocity with respect to time in the x, y, and z axes.

Table 60: Acceleration3DPlatformXYZ Structure Definition

Attribute Name	Attribute Type	Attribute Description
xAccel	AccelerationScalar	Specifies the vehicle's rate of change of velocity with respect to time in the x-axis.
yAccel	AccelerationScalar	Specifies the vehicle's rate of change of velocity with respect to time in the y-axis.
zAccel	AccelerationScalar	Specifies the vehicle's rate of change of velocity with respect to time in the z-axis.

6.2.8 AirSpeedRequirement

Namespace: UMAA::Common::Speed::AirSpeedRequirement

Description: Defines the speed through air.

Table 61: AirSpeedRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
speed	IndicatedAirspeed	Specifies speed through air.
speedTolerance	AirSpeedTolerance	Specifies the tolerance for a speed through air.

6.2.9 AirSpeedTolerance

Namespace: UMAA::Common::Speed::AirSpeedTolerance

Description: Defines the speed through air tolerance.

Table 62: AirSpeedTolerance Structure Definition

Attribute Name	Attribute Type	Attribute Description
lowerlimit	IndicatedAirspeed	Specifies the lower limit of allowable values for the air speed.
upperlimit	IndicatedAirspeed	Specifies the upper limit of allowable values for the air speed.

6.2.10 AltitudeAGLType

Namespace: UMAA::Common::Measurement::AltitudeAGLType

Description: The height above ground level.

Table 63: AltitudeAGLType Structure Definition

Attribute Name	Attribute Type	Attribute Description
altitudeAGL	DistanceAGL	Specifies the distance above ground level.

6.2.11 AltitudeASFType

Namespace: UMAA::Common::Measurement::AltitudeASFType

Description: The height above sea floor.

Table 64: AltitudeASFType Structure Definition

Attribute Name	Attribute Type	Attribute Description
altitudeASF	DistanceASF	The height above the sea floor.

6.2.12 AltitudeGeodeticType

Namespace: UMAA::Common::Measurement::AltitudeGeodeticType

Description: The geodetic height above the ellipsoid.

Table 65: AltitudeGeodeticType Structure Definition

Attribute Name	Attribute Type	Attribute Description
altitudeGeodetic	GeodeticAltitude	The altitude above the reference ellipsoid.

6.2.13 AltitudeMSLType

Namespace: UMAA::Common::Measurement::AltitudeMSLType

Description: The orthometric height above the Geoid (Mean Sea Level).

Table 66: AltitudeMSLType Structure Definition

Attribute Name	Attribute Type	Attribute Description
altitudeMSL	MSLAltitude	The orthometric height above the Geoid (Mean Sea Level).

6.2.14 DateTimeRequirement

Namespace: UMAA::Common::Requirements::DateTimeRequirement

Description: Realizes TimeRequirementType: a Requirement that specifies the type of time.

Table 67: DateTimeRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
time	DateTime	Describes the required time value.
timeTolerance	DateTimeTolerance	Describes the time tolerance.

6.2.15 DateTimeTolerance

Namespace: UMAA::Common::Measurement::DateTimeTolerance

Description: Realizes TimeToleranceType: an ObservableTolerance that specifies the range of allowable values for a time attribute.

Table 68: DateTimeTolerance Structure Definition

Attribute Name	Attribute Type	Attribute Description
lowerLimit	DateTime	specifies the minimum value of the time point.
upperLimit	DateTime	specifies the maximum value of time point.

6.2.16 DepthType

Namespace: UMAA::Common::Measurement::DepthType

Description: Defines the depth below sea level.

Table 69: DepthType Structure Definition

Attribute Name	Attribute Type	Attribute Description
depth	DistanceBSL	The depth below sea level.

6.2.17 DirectionCurrentRequirement

Namespace: UMAA::Common::Orientation::DirectionCurrentRequirement

Description: A requirement that specifies the direction with respect to the current.

Table 70: DirectionCurrentRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
direction	HeadingCurrentDirection	Specifies the heading offset angle relative to the current.
directionTolerance	DirectionToleranceType	Specifies the heading reference angle tolerance relative to the current.

6.2.18 DirectionMagneticNorthRequirement

Namespace: UMAA::Common::Orientation::DirectionMagneticNorthRequirement

Description: A requirement that specifies the direction with respect to magnetic north.

Table 71: DirectionMagneticNorthRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
direction	HeadingMagneticNorth	Specifies the heading reference angle relative to magnetic north.
directionTolerance	DirectionToleranceType	Specifies the heading reference angle tolerance relative to magnetic north.

6.2.19 DirectionRequirementType

Namespace: UMAA::Common::Orientation::DirectionRequirementType

Description: Union Type. Direction of the vehicle motion or pattern being performed.

Table 72: DirectionRequirementType Union(s)

Type Name	Type Description
DirectionCurrentRequirement	A requirement that specifies the direction with respect to the current.
DirectionMagneticNorthRequirement	A requirement that specifies the direction with respect to magnetic north.
DirectionTrueNorthRequirement	A requirement that specifies the direction with respect to true north.
DirectionWindRequirement	A requirement that specifies the direction with respect to the direction of the wind.

6.2.20 DirectionToleranceType

Namespace: UMAA::Common::Orientation::DirectionToleranceType

Description: An angle tolerance associated with a direction.

Table 73: DirectionToleranceType Structure Definition

Attribute Name	Attribute Type	Attribute Description
lowerlimit	Angle	Describes the direction bound counterclockwise from the specified direction.
upperlimit	Angle	Describes the direction bound clockwise from the specified direction.

6.2.21 DirectionTrueNorthRequirement

Namespace: UMAA::Common::Orientation::DirectionTrueNorthRequirement

Description: A requirement that specifies the direction with respect to true north.

Table 74: DirectionTrueNorthRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
direction	HeadingTrueNorthAngle	Specifies the heading reference angle relative to true north.
directionTolerance	DirectionToleranceType	Specifies the heading reference angle tolerance relative to true north.

6.2.22 DirectionWindRequirement

Namespace: UMAA::Common::Orientation::DirectionWindRequirement

Description: A requirement that specifies the direction with respect to the direction of the wind.

Table 75: DirectionWindRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
direction	HeadingWindDirection	Specifies the heading reference angle relative to the wind direction.
directionTolerance	DirectionToleranceType	Specifies the heading reference angle tolerance relative to the wind direction.

6.2.23 ElevationType

Namespace: UMAA::Common::Measurement::ElevationType

Description: Union Type. Elevation in either altitude from sea floor or depth from surface (other altitude options support above ground and sea level for potential hybrid vehicles).

Table 76: ElevationType Union(s)

Type Name	Type Description
AltitudeAGLType	The height above ground level.
AltitudeASFType	The height above sea floor.
AltitudeGeodeticType	The geodetic height above the ellipsoid.
AltitudeMSLType	The orthometric height above the Geoid (Mean Sea Level).
DepthType	Defines the depth below sea level.

6.2.24 EngineRPM

Namespace: UMAA::Common::Speed::EngineRPM

Description: Defines the engine RPM.

Table 77: EngineRPM Structure Definition

Attribute Name	Attribute Type	Attribute Description
rpm	EngineRPMSpeedRequirement	Specifies engine rpm.

6.2.25 EngineRPMSpeedRequirement

Namespace: UMAA::Common::Speed::EngineRPMSpeedRequirement

Description: Defines the engine rpm.

Table 78: EngineRPMSpeedRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
speed	FrequencyRPM	Specifies speed via engine rpm.
speedTolerance	EngineRPMSpeedTolerance	Specifies the tolerance for an engine rpm.

6.2.26 EngineRPMSpeedTolerance

Namespace: UMAA::Common::Speed::EngineRPMSpeedTolerance

Description: Defines the speed through engine rpm.

Table 79: EngineRPMSpeedTolerance Structure Definition

Attribute Name	Attribute Type	Attribute Description
lowerlimit	FrequencyRPM	Specifies the lower limit of allowable values for the engine rpm.
upperlimit	FrequencyRPM	Specifies the upper limit of allowable values for the engine rpm.

6.2.27 GeoPosition2D

Namespace: UMAA::Common::Measurement::GeoPosition2D

Description: Specifies a location on the surface of the Earth.

Table 80: GeoPosition2D Structure Definition

Attribute Name	Attribute Type	Attribute Description
geodeticLatitude	GeodeticLatitude	Specifies the north-south coordinate of the position.
geodeticLongitude	GeodeticLongitude	Specifies the east-west coordinate of the position.

6.2.28 GeoPosition2DRequirement

Namespace: UMAA::Common::Position::GeoPosition2DRequirement

Description: Defines a position requirement.

Table 81: GeoPosition2DRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
tolerance	GeoPosition2DTolerance	Specifies the required position tolerance.
value	GeoPosition2D	Specifies the required position.

6.2.29 GeoPosition2DTolerance

Namespace: UMAA::Common::Position::GeoPosition2DTolerance

Description: Defines a position tolerance.

Table 82: GeoPosition2DTolerance Structure Definition

Attribute Name	Attribute Type	Attribute Description
limit	Distance	Specifies the limit of the tolerance.

6.2.30 GlobalDriftStateType

Namespace: UMAA::MO::GlobalDriftState::GlobalDriftStateType

Description: Union Type. State of the global drift. While first transiting to the drift position, the selector will be GlobalTransitDriftType until the position and elevation are first achieved within their respective tolerances. Once achieved, the union selector will change to GlobalRegionDriftType. The selector will not change as a result of any of the GlobalRegionDriftType achievements states being lost and regained as a result of tolerance settings being violated. This is true until the service determines that the elevation or drift tolerances are violated by a sufficient margin that it is more effective for the vehicle to return to transiting to the drift location. In that case, the GlobalRegionDriftType reverts to the GlobalTransitDriftType selector and those transit achievements then are actively set.

Table 83: GlobalDriftStateType Union(s)

Type Name	Type Description
GlobalRegionDriftType	Indicates that the vehicle is in the global drift region.
GlobalTransitDriftType	Indicates that vehicle is in transit to the global drift region.

6.2.31 GlobalFigure8PatternType

Namespace: UMAA::MO::GlobalFigure8State::GlobalFigure8PatternType

Description: Indicates that the global figure 8 pattern is currently executing.

Table 84: GlobalFigure8PatternType Structure Definition

Attribute Name	Attribute Type	Attribute Description
elevationAchieved	boolean	Indicates that the elevation requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
positionAchieved	boolean	When the pattern is executing, this indicates that the position requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
speedAchieved	boolean	When the pattern is executing, this indicates that the speed requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.

6.2.32 GlobalFigure8StateType

Namespace: UMAA::MO::GlobalFigure8State::GlobalFigure8StateType

Description: Union Type. State of the global figure 8 pattern being executed. While first transiting to the figure 8 pattern to be performed, the selector will be GlobalFigure8TransitType until the pattern position, speed, and elevation are first achieved within their respective tolerances. Once achieved, the union selector will change to GlobalFigure8PatternType. The selector will not change as a result of any of the GlobalFigure8PatternType achievements states being lost and regained as a result of tolerance settings being violated. The service is expected to make driving adjustments to attempt to keep all achievement states satisfied. This is true until the service determines tolerance(s) are violated by a sufficient margin that it is more effective for the vehicle to return to transiting to the pattern location. In that case, the GlobalFigure8StateType reverts to the GlobalFigure8TransitType selector and those transit achievements are then set.

Table 85: GlobalFigure8StateType Union(s)

Type Name	Type Description
GlobalFigure8PatternType	Indicates that the global figure 8 pattern is currently executing.
GlobalFigure8TransitType	Indicates that the vehicle is in transit to where the global figure 8 pattern is to be performed.

6.2.33 GlobalFigure8TransitType

Namespace: UMAA::MO::GlobalFigure8State::GlobalFigure8TransitType

Description: Indicates that the vehicle is in transit to where the global figure 8 pattern is to be performed.

Table 86: GlobalFigure8TransitType Structure Definition

Attribute Name	Attribute Type	Attribute Description
transitElevationAchieved	boolean	Indicates that the transit elevation requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.

Attribute Name	Attribute Type	Attribute Description
transitSpeedAchieved	boolean	Indicates that the transit speed requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.

6.2.34 GlobalHoverStateType

Namespace: UMAA::MO::GlobalHoverState::GlobalHoverStateType

Description: Union Type. State of the global hover. While first transiting to the hover location, the selector will be GlobalTransitHoverType until the position, heading, and elevation are first achieved within their respective tolerances. Once achieved, the union selector will change to GlobalHoveringHoverType. The selector will not change as a result of any of the GlobalHoveringHoverType achievements states being lost and regained as a result of tolerance settings being violated. The service is expected to make driving adjustments to attempt to keep all achievement states satisfied. This is true until the service determines that tolerance(s) are violated by a sufficient margin that it is more effective for the vehicle to return to transiting to the hover location. In that case, the GlobalHoverStateType reverts to the GlobalTransitHoverType selector and those transit achievements then are actively set.

Table 87: GlobalHoverStateType Union(s)

Type Name	Type Description
GlobalHoveringHoverType	Indicates that the global hover is currently executing.
GlobalTransitHoverType	Indicates that the vehicle is in transit to where the global hover is to be performed.

6.2.35 GlobalHoveringHoverType

Namespace: UMAA::MO::GlobalHoverState::GlobalHoveringHoverType

Description: Indicates that the global hover is currently executing.

Table 88: GlobalHoveringHoverType Structure Definition

Attribute Name	Attribute Type	Attribute Description
elevationAchieved	boolean	Indicates that the elevation requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
positionAchieved	boolean	Indicates that the position requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.

6.2.36 GlobalRacetrackPatternType

Namespace: UMAA::MO::GlobalRacetrackState::GlobalRacetrackPatternType

Description: Indicates that the global racetrack pattern is currently executing.

Table 89: GlobalRacetrackPatternType Structure Definition

Attribute Name	Attribute Type	Attribute Description
elevationAchieved	boolean	Indicates that the elevation requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
positionAchieved	boolean	Indicates that the position requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
speedAchieved	boolean	Indicates that the speed requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.

6.2.37 GlobalRacetrackStateType

Namespace: UMAA::MO::GlobalRacetrackState::GlobalRacetrackStateType

Description: Union Type. State of the global racetrack pattern being executed. While first transiting to the racetrack location to be performed, the selector will be GlobalRacetrackTransitType until the pattern position, speed, and elevation are first achieved within their respective tolerances. Once achieved, the union selector will change to GlobalRacetrackPatternType. The selector will not change as a result of any of the GlobalRacetrackPatternType achievements states being lost and regained as a result of tolerance settings being violated. The service is expected to make driving adjustments to attempt to keep all achievement states satisfied. This is true until the service determines that tolerance(s) are violated by a sufficient margin that it is more effective for the vehicle to return to transiting to the racetrack location. In that case, the GlobalRacetrackStateType reverts to the GlobalRacetrackTransitType selector and those transit achievements are then set.

Table 90: GlobalRacetrackStateType Union(s)

Type Name	Type Description
GlobalRacetrackPatternType	Indicates that the global racetrack pattern is currently executing.
GlobalRacetrackTransitType	Indicates that the vehicle is in transit to where the global racetrack pattern is to be performed.

6.2.38 GlobalRacetrackTransitType

Namespace: UMAA::MO::GlobalRacetrackState::GlobalRacetrackTransitType

Description: Indicates that the vehicle is in transit to where the global racetrack pattern is to be performed.

Table 91: GlobalRacetrackTransitType Structure Definition

Attribute Name	Attribute Type	Attribute Description
transitElevationAchieved	boolean	Indicates that the transit elevation requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
transitSpeedAchieved	boolean	Indicates that the transit speed requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.

6.2.39 GlobalRegionDriftType

Namespace: UMAA::MO::GlobalDriftState::GlobalRegionDriftType

Description: Indicates that the vehicle is in the global drift region.

Table 92: GlobalRegionDriftType Structure Definition

Attribute Name	Attribute Type	Attribute Description
elevationAchieved	boolean	Indicates that the elevation requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
positionAchieved	boolean	Indicates that the position requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.

6.2.40 GlobalRegularPolygonPatternType

Namespace: UMAA::MO::GlobalRegularPolygonState::GlobalRegularPolygonPatternType

Description: Indicates that the global regular polygon pattern is currently executing.

Table 93: GlobalRegularPolygonPatternType Structure Definition

Attribute Name	Attribute Type	Attribute Description
elevationAchieved	boolean	Indicates that the elevation requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
positionAchieved	boolean	Indicates that the position requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
speedAchieved	boolean	Indicates that the speed requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.

6.2.41 GlobalRegularPolygonStateType

Namespace: UMAA::MO::GlobalRegularPolygonState::GlobalRegularPolygonStateType

Description: Union Type. State of the global regular polygon pattern being executed. While first transiting to the regular polygon location to be performed, the selector will be GlobalRegularPolygonTransitType until the pattern position, speed, and elevation are first achieved within their respective tolerances. Once achieved, the union selector will change to GlobalRegularPolygonPatternType. The selector will not change as a result of any of the GlobalRegularPolygonPatternType achievements states being lost and regained as a result of tolerance settings being violated. The service is expected to make driving adjustments to attempt to keep all achievement states satisfied. This is true until the service determines that tolerance(s) are violated by a sufficient margin that it is more effective for the vehicle to return to transiting to regular polygon pattern location. In that case, the GlobalRegularPolygonStateType reverts to the GlobalRegularPolygonTransitType selector and those transit achievements are then set.

Table 94: GlobalRegularPolygonStateType Union(s)

Type Name	Type Description
GlobalRegularPolygonPatternType	Indicates that the global regular polygon pattern is currently executing.
GlobalRegularPolygonTransitType	Indicates that the vehicle is in transit to where the global regular polygon pattern is to be performed.

6.2.42 GlobalRegularPolygonTransitType

Namespace: UMAA::MO::GlobalRegularPolygonState::GlobalRegularPolygonTransitType

Description: Indicates that the vehicle is in transit to where the global regular polygon pattern is to be performed.

Table 95: GlobalRegularPolygonTransitType Structure Definition

Attribute Name	Attribute Type	Attribute Description
transitElevationAchieved	boolean	Indicates that the transit elevation requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
transitSpeedAchieved	boolean	Indicates that the transit speed requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.

6.2.43 GlobalTransitDriftType

Namespace: UMAA::MO::GlobalDriftState::GlobalTransitDriftType

Description: Indicates that vehicle is in transit to the global drift region.

Table 96: GlobalTransitDriftType Structure Definition

Attribute Name	Attribute Type	Attribute Description
transitElevationAchieved	boolean	Indicates that the transit elevation requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
transitSpeedAchieved	boolean	Indicates that the transit speed requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.

6.2.44 GlobalTransitHoverType

Namespace: UMAA::MO::GlobalHoverState::GlobalTransitHoverType

Description: Indicates that the vehicle is in transit to where the global hover is to be performed.

Table 97: GlobalTransitHoverType Structure Definition

Attribute Name	Attribute Type	Attribute Description
elevationAchieved	boolean	Indicates that the transit elevation requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.
speedAchieved	boolean	Indicates that the transit speed requested is within the commanded tolerance. Achievement may be lost and regained resulting in multiple changes to this attribute.

6.2.45 GlobalWaypointType

Namespace: UMAA::MO::GlobalWaypointControl::GlobalWaypointType

Description: The structure is used to describe a waypoint in a global reference frame.

Table 98: GlobalWaypointType Structure Definition

Attribute Name	Attribute Type	Attribute Description
attitude†	Orientation3DNEDRequirement	The desired orientation (roll, pitch, yaw) of the vehicle as arriving at the waypoint.
elevation	ElevationType	The desired elevation used for the vehicle. This value is 0 for USVs.
name†	StringShortDescription	A short name for the waypoint.
position	GeoPosition2DRequirement	The desired waypoint position in the global coordinate system.
speed	VariableSpeedControlType	The desired waypoint travel speed of the vehicle with reference to the medium, the ground, the air, RPM, or true speed.
trackTolerance†	Distance	The desired tolerance of the path measured by distance. If defined, the vehicle must maintain track; if not defined, there is no need to maintain track. Use the vehicle position at time of command to define the track for the first waypoint.
waypointID	NumericGUID	The desired id to keep track of the waypoint.

6.2.46 GroundSpeedRequirement

Namespace: UMAA::Common::Speed::GroundSpeedRequirement

Description: Defines the speed over ground.

Table 99: GroundSpeedRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
speed	GroundSpeed	Specifies speed over ground.
speedTolerance	GroundSpeedTolerance	Specifies the tolerance for a speed over ground.

6.2.47 GroundSpeedTolerance

Namespace: UMAA::Common::Speed::GroundSpeedTolerance

Description: Defines the speed over ground tolerance.

Table 100: GroundSpeedTolerance Structure Definition

Attribute Name	Attribute Type	Attribute Description
lowerlimit	GroundSpeed	Specifies the lower limit of allowable values for the ground speed.
upperlimit	GroundSpeed	Specifies the upper limit of allowable values for the ground speed.

6.2.48 LinearEffort

Namespace: UMAA::Common::Measurement::LinearEffort

Description: Defines the along-axes efforts as a percentage.

Table 101: LinearEffort Structure Definition

Attribute Name	Attribute Type	Attribute Description
xAxis	Effort	Linear effort along the x-axis, expressed as a percentage.
yAxis	Effort	Linear effort along the y-axis, expressed as a percentage.
zAxis	Effort	Linear effort along the z-axis, expressed as a percentage.

6.2.49 Orientation3DNEDRequirement

Namespace: UMAA::Common::Orientation::Orientation3DNEDRequirement

Description: A requirement that describes a desired 3D orientation in a NED coordinate system.

Table 102: Orientation3DNEDRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
pitchY†	PitchYNEDRequirement	Defines a pitch relative to the NED coordinate system.
rollX†	RollXNEDRequirement	Defines a roll relative to the NED coordinate system.
yawZ	YawZNEDRequirement	Defines a yaw relative to the NED coordinate system.

6.2.50 OrientationAcceleration3D

Namespace: UMAA::Common::Measurement::OrientationAcceleration3D

Description: OrientationAcceleration3D specifies the acceleration for each axis of an Orientation.

Table 103: OrientationAcceleration3D Structure Definition

Attribute Name	Attribute Type	Attribute Description
pitchAccelY	PitchAcceleration	pitchAccelY specifies the acceleration of the platform's rotation about the lateral axis (e.g. the axis parallel to the wings) in a locally level, XYZ coordinate system centered on the platform.
rollAccelX	RollAcceleration	rollAccelX specifies the acceleration of the platform's rotation about the longitudinal axis (e.g. the axis through the body of an aircraft from tail to nose) in a locally level, XYZ coordinate system centered on the platform.
yawAccelZ	YawAcceleration	yawAccelZ specifies the acceleration of the platform's rotation about the vertical axis (e.g. the axis from top to bottom through an aircraft) in a locally level, XYZ coordinate system centered on the platform.

6.2.51 PitchYNEDRequirement

Namespace: UMAA::Common::Orientation::PitchYNEDRequirement

Description: A requirement that specifies a pitch relative to the NED coordinate system.

Table 104: PitchYNEDRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
pitch	PitchYNEDType	Defines a pitch relative to the NED system.
pitchTolerance	PitchYNEDTolerance	Describes the pitch bounding limits.

6.2.52 PitchYNEDTolerance

Namespace: UMAA::Common::Orientation::PitchYNEDTolerance

Description: A down or up angle tolerance.

Table 105: PitchYNEDTolerance Structure Definition

Attribute Name	Attribute Type	Attribute Description
lowerlimit	PitchYNEDType	Defines the steepest downangle allowed.
upperlimit	PitchYNEDType	Defines the steepest upangle allowed.

6.2.53 PitchYNEDType

Namespace: UMAA::Common::Orientation::PitchYNEDType

Description: A requirement that specifies a pitch relative to the NED coordinate system.

Table 106: PitchYNEDType Structure Definition

Attribute Name	Attribute Type	Attribute Description
pitch	PitchHalfAngle	Defines a pitch relative to the NED coordinate system.

6.2.54 Quaternion

Namespace: BasicTypes::Quaternion

Description: Defines a four-element vector that can be used to encode any rotation in a 3D coordinate system.

Table 107: Quaternion Structure Definition

Attribute Name	Attribute Type	Attribute Description
a		Real number a.
b		Real number b.
c		Real number c.
d		Real number d.

6.2.55 RecommendedSpeedControl

Namespace: UMAA::Common::Speed::RecommendedSpeedControl

Description: Defines the recommended speed mode.

Table 108: RecommendedSpeedControl Structure Definition

Attribute Name	Attribute Type	Attribute Description
recommendedSpeedControl	SpeedControlType	Specifies the recommended speed mode.

6.2.56 RequiredSpeedControl

Namespace: UMAA::Common::Speed::RequiredSpeedControl

Description: Defines the required speed mode.

Table 109: RequiredSpeedControl Structure Definition

Attribute Name	Attribute Type	Attribute Description
requiredSpeedControl	SpeedControlType	Specifies the required speed mode.

6.2.57 RollXNEDRequirement

Namespace: UMAA::Common::Orientation::RollXNEDRequirement

Description: A requirement that specifies a roll relative to the NED coordinate system.

Table 110: RollXNEDRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
roll	RollXNEDType	Defines a roll relative to the NED system.
rollTolerance†	RollXNEDTolerance	Describes the roll bounding limits.

6.2.58 RollXNEDTolerance

Namespace: UMAA::Common::Orientation::RollXNEDTolerance

Description: A rotational tolerance.

Table 111: RollXNEDTolerance Structure Definition

Attribute Name	Attribute Type	Attribute Description
lowerlimit	RollXNEDType	Defines the lower bound.
upperlimit	RollXNEDType	Defines the lower bound.

6.2.59 RollXNEDType

Namespace: UMAA::Common::Orientation::RollXNEDType

Description: A requirement that specifies a roll relative to the NED coordinate system.

Table 112: RollXNEDType Structure Definition

Attribute Name	Attribute Type	Attribute Description
roll	RollAngle	Defines a roll relative to the NED coordinate system.

6.2.60 RotationalEffort

Namespace: UMAA::Common::Measurement::RotationalEffort

Description: Describes a set of efforts around each axis as a percentage, using the right-hand rule.

Table 113: RotationalEffort Structure Definition

Attribute Name	Attribute Type	Attribute Description
pitchEffort	Effort	Rotational effort around the y-axis, expressed as a percentage.
rollEffort	Effort	Rotational effort around the x-axis, expressed as a percentage.
yawEffort	Effort	Rotational effort around the z-axis, expressed as a percentage.

6.2.61 SpeedControlType

Namespace: UMAA::Common::Speed::SpeedControlType

Description: **Union Type**. Speed of the vehicle.

Table 114: SpeedControlType Union(s)

Type Name	Type Description
EngineRPM	Defines the engine RPM.
SpeedOverGround	Defines the speed over ground.
SpeedThroughAir	Defines the speed through air.
SpeedThroughWater	Defines the speed through water.
VehicleSpeedMode	Defines the speed mode.

6.2.62 SpeedOverGround

Namespace: UMAA::Common::Speed::SpeedOverGround

Description: Defines the speed over ground.

Table 115: SpeedOverGround Structure Definition

Attribute Name	Attribute Type	Attribute Description
speed	GroundSpeedRequirement	Specifies speed over ground.

6.2.63 SpeedThroughAir

Namespace: UMAA::Common::Speed::SpeedThroughAir

Description: Defines the speed through air.

Table 116: SpeedThroughAir Structure Definition

Attribute Name	Attribute Type	Attribute Description
speed	AirSpeedRequirement	Specifies speed through air.

6.2.64 SpeedThroughWater

Namespace: UMAA::Common::Speed::SpeedThroughWater

Description: Defines the speed through water.

Table 117: SpeedThroughWater Structure Definition

Attribute Name	Attribute Type	Attribute Description
speed	WaterSpeedRequirement	Specifies speed through water.

6.2.65 SurfaceCapabilityBoundsType

Namespace: UMAA::MO::DriverConfig::SurfaceCapabilityBoundsType

Description: This structure describes the capability limits for any vehicle operating on the surface.

Table 118: SurfaceCapabilityBoundsType Structure Definition

Attribute Name	Attribute Type	Attribute Description
maxAcceleration†	AccelerationLocalWaterMass	The highest rate of increase of vehicle linear velocity as a function of time.
maxDeceleration†	AccelerationLocalWaterMass	The highest rate of decrease of vehicle linear velocity as a function of time.
maxForwardSpeed†	SpeedLocalWaterMass	The largest possible forward horizontal displacement of the vehicle as a function of time.
maxReverseSpeed†	SpeedLocalWaterMass	The largest possible reverse horizontal displacement of the vehicle as a function of time.
maxTowingSpeed†	SpeedLocalWaterMass	The fastest linear velocity that is allowed when dragging a payload.
maxTowingTurnAcceleration†	AngleAcceleration	The highest rate of linear velocity change as a function of time when towing.
maxTowingTurnRate†	TurnRate	The fastest vehicle direction change rate when towing a payload.
maxTurnAcceleration†	AngleAcceleration	The highest rate of linear velocity change as a function of time.
maxTurnRate†	TurnRate	The fastest vehicle direction change rate.
minSpeedInMedium†	SpeedLocalWaterMass	The slowest linear velocity that is required to enable control surfaces to operate.
minTowingSpeed†	SpeedLocalWaterMass	The slowest linear velocity that is allowed when dragging a payload.
recommendedAcceleration†	Acceleration3DPlatformXYZ	The recommended value for the linear acceleration of the vehicle.
recommendedSpeed†	SpeedControlType	The recommended value for the linear speed of the vehicle.
recommendedTurnRate†	TurnRate	The recommended turn rate.

6.2.66 TimeWithSpeed

Namespace: UMAA::Common::Speed::TimeWithSpeed

Description: Defines the time window and the recommended speed of a vehicle.

Table 119: TimeWithSpeed Structure Definition

Attribute Name	Attribute Type	Attribute Description
arrivalTime	DateTimeRequirement	Specifies the arrival time of the waypoint.
recommendedSpeed†	SpeedControlType	Specifies the recommended speed of the waypoint.

6.2.67 UnderwaterCapabilityBoundsType

Namespace: UMAA::MO::DriverConfig::UnderwaterCapabilityBoundsType

Description: This structure describes the capability bounds for any vehicle operating under the surface.

Table 120: UnderwaterCapabilityBoundsType Structure Definition

Attribute Name	Attribute Type	Attribute Description
maxAcceleration†	AccelerationLocalWaterMass	The highest rate of increase of vehicle linear velocity as a function of time.
maxAttitudeAcceleration†	OrientationAcceleration3D	The highest rate of increase of vehicle rotational velocity as a function of time in three dimensions.
maxAttitudeDeceleration†	OrientationAcceleration3D	The highest rate of decrease of vehicle rotational velocity as a function of time in three dimensions.
maxDeceleration†	AccelerationLocalWaterMass	The highest rate of decrease of vehicle linear velocity as a function of time.
maxDepthAcceleration†	SpeedBSLAcceleration	The highest rate of vertical velocity change as a function of time.
maxDepthChangeRate†	SpeedBSL	The largest possible vertical displacement of the vehicle as a function of time.
maxForwardSpeed†	SpeedLocalWaterMass	The largest possible forward horizontal displacement of the vehicle as a function of time.
maxPitch†	Angle	The maximum down or up pitch angle allowed.
maxPitchRate†	PitchRate	The highest angular rate of change in the rotation of a vehicle about the transverse axis.
maxReverseSpeed†	SpeedLocalWaterMass	The largest possible reverse horizontal displacement of the vehicle as a function of time.
maxRoll†	Angle	The maximum roll angle allowed, either positive or negative.
maxTowingSpeed†	SpeedLocalWaterMass	The fastest linear velocity that is allowed when dragging a payload.
maxTowingTurnAcceleration†	AngleAcceleration	The highest rate of linear velocity change as a function of time when towing.
maxTowingTurnRate†	TurnRate	The fastest vehicle direction change rate when towing a payload.
maxTurnAcceleration†	AngleAcceleration	The highest rate of linear velocity change as a function of time.
maxTurnRate†	TurnRate	The fastest vehicle direction change rate.
maxVehicleDepth†	DistanceBSL	The largest vehicle operating distance below the water surface.
minSpeedInMedium†	SpeedLocalWaterMass	The slowest linear velocity that is required to enable control surfaces to operate.

Attribute Name	Attribute Type	Attribute Description
minTowingSpeed†	SpeedLocalWaterMass	The slowest linear velocity that is allowed when dragging a payload.
minVehicleDepth†	DistanceBSL	The smallest vehicle distance below the water surface to which the vehicle should dive to begin operations.
recommendedAcceleration†	Acceleration3DPlatformXYZ	The recommended value for the linear acceleration of the vehicle.
recommendedAttitudeAcceleration†	OrientationAcceleration3D	The rate of rotational velocity change of the vehicle as a function of time in three dimensions that should be used as nominal values when no other factors apply.
recommendedDepthAcceleration†	SpeedBSLAcceleration	The rate of vertical velocity change as a function of time used as the nominal value when no other factors apply.
recommendedDepthChangeRate†	SpeedBSL	The vertical displacement of the vehicle as a function of time used as the nominal value when no other factors apply.
recommendedPitchRate†	PitchRate	The angular rate of change in the rotation of a vehicle about the transverse axis that should be used as the nominal value when no other factors apply.
recommendedSpeed†	SpeedControlType	The recommended value for the linear speed of the vehicle.
recommendedTurnRate†	TurnRate	The recommended turn rate.
recommendedVehicleDepth†	DistanceBSL	The vehicle operational distance below the water surface that should be used as the nominal value when no other factors apply.

6.2.68 VariableSpeedControlType

Namespace: UMAA::Common::Speed::VariableSpeedControlType

Description: Union Type. Speed specifier for the vehicle which may be based on explicit speed, a recommended speed, a time window, or a time window with an optional recommended speed.

Table 121: VariableSpeedControlType Union(s)

Type Name	Type Description
RecommendedSpeedControl	Defines the recommended speed mode.
RequiredSpeedControl	Defines the required speed mode.
TimeWithSpeed	Defines the time window and the recommended speed of a vehicle.

6.2.69 VehicleSpeedMode

Namespace: UMAA::Common::Speed::VehicleSpeedMode

Description: Defines the speed mode.

Table 122: VehicleSpeedMode Structure Definition

Attribute Name	Attribute Type	Attribute Description
mode	VehicleSpeedModeEnumType	Specifies the speed mode.

6.2.70 WaterSpeedRequirement

Namespace: UMAA::Common::Speed::WaterSpeedRequirement

Description: Defines the speed through water.

Table 123: WaterSpeedRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
speed	SpeedLocalWaterMass	Specifies speed through water.
speedTolerance	WaterSpeedTolerance	Specifies the tolerance for a speed through water.

6.2.71 WaterSpeedTolerance

Namespace: UMAA::Common::Speed::WaterSpeedTolerance

Description: Defines the speed through water tolerance.

Table 124: WaterSpeedTolerance Structure Definition

Attribute Name	Attribute Type	Attribute Description
lowerlimit	SpeedLocalWaterMass	Specifies the lower limit of allowable values for the water speed.
upperlimit	SpeedLocalWaterMass	Specifies the upper limit of allowable values for the water speed.

6.2.72 YawZNEDRequirement

Namespace: UMAA::Common::Orientation::YawZNEDRequirement

Description: A requirement that specifies a yaw relative to the NED coordinate system.

Table 125: YawZNEDRequirement Structure Definition

Attribute Name	Attribute Type	Attribute Description
yaw	YawZNEDType	Defines a yaw relative to the NED system.
yawTolerance†	YawZNEDTolerance	Describes the yaw bounding limits.

6.2.73 YawZNEDTolerance

Namespace: UMAA::Common::Orientation::YawZNEDTolerance

Description: A directional tolerance.

Table 126: YawZNEDTolerance Structure Definition

Attribute Name	Attribute Type	Attribute Description
lowerlimit	YawZNEDType	Defines the lower bound.
upperlimit	YawZNEDType	Defines the lower bound.

6.2.74 YawZNEDType

Namespace: UMAA::Common::Orientation::YawZNEDType

Description: Specifies a yaw relative to the NED coordinate system.

Table 127: YawZNEDType Structure Definition

Attribute Name	Attribute Type	Attribute Description
yaw	YawPosAngle	Defines a yaw relative to the NED coordinate system.

6.3 Enumerations

Enumerations are used extensively throughout UMAA. This section lists the values associated with each enumeration defined in UCS-UMAA.

6.3.1 CommandStatusReasonEnumType

Namespace: UMAA::Common::MaritimeEnumeration::CommandStatusReasonEnumType

Description: Defines a mutually exclusive set of reasons why a command status state transition has occurred.

Table 128: CommandStatusReasonEnumType Enumeration

Enumeration Value	Description
CANCELED	Indicates a transition to the CANCELED state when the command is canceled successfully.
INTERRUPTED	Indicates a transition to the FAILED state when the command has been interrupted by a higher priority process.
OBJECTIVE_FAILED	Indicates a transition to the FAILED state when the commanded resource is unable to achieve the command's objective due to external factors.
RESOURCE_FAILED	Indicates a transition to the FAILED state when the commanded resource is unable to achieve the command's objective due to resource or platform failure.
RESOURCE_REJECTED	Indicates a transition to the FAILED state when the commanded resource rejects the command for some reason.
SERVICE_FAILED	Indicates a transition to the FAILED state when the commanded resource is unable to achieve the command's objective due to processing failure.
SUCCEEDED	Indicates the conditions to proceed to this state have been met and a normal state transition has occurred.
TIMEOUT	Indicates a transition to the FAILED state when the command is not acknowledged within some defined time bound.
UPDATED	Indicates a transition back to the ISSUED state from a non-terminal state when the command has been updated.
VALIDATION_FAILED	Indicates a transition to the FAILED state when the command contains missing, out-of-bounds, or otherwise invalid parameters.

6.3.2 ContactManeuverInfluenceEnumType

Namespace: UMAA::Common::MaritimeEnumeration::ContactManeuverInfluenceEnumType

Description: A mutually exclusive set of values that defines the maneuver of a vehicle in response to a contact.

Table 129: ContactManeuverInfluenceEnumType Enumeration

Enumeration Value	Description
BEING_OVERTAKEN_COMPLIANT	COLREGS being overtaken where the other vehicle is determined to be compliant
BEING_OVERTAKEN_NONCOMPLIANT	COLREGS being overtaken where the other vehicle is determined to be noncompliant
COLLISION_AVOIDANCE	Maneuvering to avoid a dynamic obstacle
CROSSING_LEFT_COMPLIANT	COLREGS crossing left where the other vehicle is determined to be compliant

Enumeration Value	Description
CROSSING_LEFT_NONCOMPLIANT	COLREGS crossing left where the other vehicle is determined to be non-compliant
CROSSING_RIGHT_COMPLIANT	COLREGS crossing right where the other vehicle is determined to be compliant
CROSSING_RIGHT_NONCOMPLIANT	COLREGS crossing right where the other vehicle is determined to be non-compliant
GUIDE	Contact is guiding or informing maneuvering (e.g., guide vehicle for Stationkeep, cooperating swarm member)
HEAD_ON_COMPLIANT	COLREGS head on where the other vehicle is determined to be compliant
HEAD_ON_NONCOMPLIANT	COLREGS head on where the other vehicle is determined to be non-compliant
IN_EXTREMIS	Determined in a situation where collision can no longer be avoided by one ship acting alone
NONE	The contact has been examined and it was determined it has no influence on the maneuvering
OBSTACLE_AVOIDANCE	Maneuvering to avoid a static obstacle
OVERTAKING_COMPLIANT	COLREGS overtaking where the other vehicle is determined to be compliant
OVERTAKING_NONCOMPLIANT	COLREGS overtaking where the other vehicle is determined to be non-compliant
PREEMPTIVE	Maneuvering to avoid a perceived future state but not in direct response to configured obstacle avoidance thresholds

6.3.3 CoordinationSituationalSignalEnumType

Namespace: UMAA::Common::MaritimeEnumeration::CoordinationSituationalSignalEnumType

Description: Defines a mutually exclusive set of values that define the current coordination status of this vessel with respect to situational signals (coordinating with other vehicles).

Table 130: CoordinationSituationalSignalEnumType Enumeration

Enumeration Value	Description
AGREE_TO_BE_OVERTAKEN	Agree to be overtaken signal active status.
ALTERING_COURSE_TO_PORT	Turning to port signal active status.
ALTERING_COURSE_TO_STARBOARD	Turning to starboard signal active status.
BLIND_BEND_SIGNAL	Blind bend signal active status.
DANGER_SIGNAL	Danger signal active status.
IN_DISTRESS_NEED_ASSISTANCE	In distress need assistance signal active status.
NONE	No signal active.
OPERATING_ASTERN_PROPULSION	Operating astern propulsion signal active status.
TO_OVERTAKE_LEAVE_VESSEL_TO_PORT	To overtake, leave vessel to port signal active status.
TO_OVERTAKE_LEAVE_VESSEL_TO_STARBOARD	To overtake, leave vessel to starboard signal active status.
VESSEL_LEAVING_DOCK	Vessel leaving dock signal active status.

Enumeration Value	Description
VISIBILITY_RESTRICTED_VEHICLE_STOPPED	Visibility restricted with vehicle stopped signal active status.
VISIBILITY_RESTRICTED_VEHICLE_UNDERWAY	Visibility restricted with vehicle underway signal active status.

6.3.4 DirectionModeEnumType

Namespace: UMAA::Common::MaritimeEnumeration::DirectionModeEnumType

Description: Specifies whether direction is a course or a based on the heading of the vehicle

Table 131: DirectionModeEnumType Enumeration

Enumeration Value	Description
COURSE	Specifies that direction is the course of the vehicle
HEADING	Specifies that direction is the heading of the vehicle

6.3.5 HoverKindEnumType

Namespace: UMAA::Common::MaritimeEnumeration::HoverKindEnumType

Description: A mutually exclusive set of values that defines the hover priority of the vehicle.

Table 132: HoverKindEnumType Enumeration

Enumeration Value	Description
LAT_LON_PRIORITY	Prioritize maintaining a latitude/longitude position
Z_PRIORITY	Prioritize maintaining an elevation

6.3.6 CommandStatusEnumType

Namespace: UMAA::Common::MaritimeEnumeration::CommandStatusEnumType

Description: Defines a mutually exclusive set of values that defines the states of a command as it progresses towards completion.

Table 133: CommandStatusEnumType Enumeration

Enumeration Value	Description
CANCELED	The command was canceled by the requestor before the command completed successfully.
COMMANDED	The command has been placed in the resource's command queue but has not yet been accepted.
COMPLETED	The command has been completed successfully.

Enumeration Value	Description
EXECUTING	The command is being performed by the resource and has not yet been completed.
FAILED	The command has been attempted, but was not successful.
ISSUED	The command has been issued to the resource (typically a sensor or streaming device), but processing has not yet commenced.

6.3.7 VehicleSpeedModeEnumType

Namespace: UMAA::Common::MaritimeEnumeration::VehicleSpeedModeEnumType

Description: A mutually exclusive set of values that defines the type of performance speed of the vehicle.

Table 134: VehicleSpeedModeEnumType Enumeration

Enumeration Value	Description
LRC	Long Range Cruise. A speed that optimizes time, distance and fuel consumption for a vehicle (definition of "optimized" is subjective. Example: for a planing hull, this is usually the minimum planing speed, even though lower speeds can achieve longer endurance or range.)
MEC	Maximum Endurance Cruise. The speed that maximizes the time a vehicle can travel.
MRC	Maximum Range Cruise. The speed that maximizes the distance a vehicle can travel.
SLOW	Slow speed. Minimum speed at which the vehicle can operate (definition of "operate" is subjective. Example: minimum speed to achieve maneuverability, engine idle speed/gear clutched in "idle ahead", etc.)
VEHICLE_SPECIFIC	Preset speed for the vehicle, that is in the range of speeds for the subject vehicle

6.3.8 WaterTurnDirectionEnumType

Namespace: UMAA::Common::MaritimeEnumeration::WaterTurnDirectionEnumType

Description: A mutually exclusive set of values that define the types of turn directions applied by the vehicle during turns.

Table 135: WaterTurnDirectionEnumType Enumeration

Enumeration Value	Description
LEFT_TURN	The vehicle will make left turns.
RIGHT_TURN	The vehicle will make right turns.

6.4 Type Definitions

This section describes the type definitions for UMAA. The table below lists how UMAA defined types are mapped to the DDS primitive types.

Table 136: Type Definitions

Type Name	Primitive Type	Range of Values	Description
AccelerationLocal WaterMass	double	fractionDigits=6 maxInclusive=299,792,458 minInclusive=-299,792,458 units=MeterPerSecondSquared	The change in velocity over time relative to local water mass.
AccelerationScalar	double	fractionDigits=3 maxInclusive=1310.68 minInclusive=-1310.68 units=MeterPerSecondSquared referenceFrame=Counting	This type stores acceleration in m/s/s.
Angle	double	fractionDigits=3 maxInclusive=3.141592653589 7932384626433832795 minInclusive=-3.141592653589 7931264626433832795 units=Radian referenceFrame=Counting	Specifies the amount of turning necessary to bring one ray, line or plane into coincidence with or parallel to another. The measurement is stated in radians between -pi and pi.
AngleAcceleration	double	fractionDigits=3 maxInclusive=N/A minInclusive=N/A units=RadianPerSecondSquare d referenceFrame=PlatformXYZ	Represents the rate of change of angular velocity measured in radians per second squared.
BooleanEnumTyp e	boolean		A mutually exclusive set of values that defines the truth values of logical algebra.
Count	long	referenceFrame=Counting units=N/A minInclusive=-2147483648 maxInclusive=2147483647 fractionDigits=0	Represents a whole (non-fractional) number that can be positive, negative or zero.
DateTimeNanosec onds	long	units=Nanoseconds minInclusive=0 maxInclusive=999999999 fractionDigits=0	number of nanoseconds elapsed within the current second.
DateTimeSeconds	longlong	units=Seconds minInclusive=-92233720368547 75807 maxInclusive=92233720368547 75807 fractionDigits=0	seconds offset from the standard POSIX (IEEE Std 1003.1-2017) epoch reference point of January 1st, 1970 00:00:00 UTC.
Distance	double	fractionDigits=3 maxInclusive=401056000 minInclusive=0 units=Meter referenceFrame=Counting	This type stores a distance in meters.

Type Name	Primitive Type	Range of Values	Description
DistanceAGL	double	fractionDigits=3 minInclusive=0.0 units=Meter referenceFrame=AGL	Describes the height above ground level of the vehicle.
DistanceASF	double	fractionDigits=3 maxInclusive=401056000 minInclusive=0 units=Meter referenceFrame=ASF	The altitude or distance above the sea floor in meters.
DistanceBSL	double	fractionDigits=3 maxInclusive=10000 minInclusive=0 units=Meter referenceFrame=BSL	The distance below sea level in meters.
DurationSeconds	double	fractionDigits=6 maxInclusive=37817280 minInclusive=0 units=Seconds referenceFrame=Counting	Represents a time duration in seconds.
Effort	double	fractionDigits=3 maxInclusive=100 minInclusive=-100 units=Percent referenceFrame=PlatformXYZ	Represents the level of effort measured in percent.
FrequencyRPM	long	fractionDigits=0 maxInclusive=100000 minInclusive=-100000 units=RevolutionsPerMinute referenceFrame=Counting	This type stores number of occurrences in revolutions per minute (RPM). Negative number is used for reverse RPM.
GeodeticAltitude	double	fractionDigits=6 maxInclusive=700000 minInclusive=-10000 units=Meter axisAbbrev=Altitude axisDirection=up axisUnit=Meter rangeMeaning=exact resolution=0.0000000001	Used for measuring position and increases in magnitude as position extends upward. Altitude measurements are expressed in meters.
GeodeticLatitude	double	axisAbbrev=Latitude axisDirection=north/south axisUnit=Degrees maximumValue=90.0 minimumValue=-90.0 rangeMeaning=exact resolution=0.0000000001	Used for measuring position and increases in magnitude as position extends from the south pole to the north pole. Latitude measurements are expressed in degrees.
GeodeticLongitude	double	axisAbbrev=Longitude axisDirection=east axisUnit=Degrees maximumValue=180.0 minimumValue=-180.0 rangeMeaning=wraparound resolution=0.0000000001	Used for measuring position and increases in magnitude as position extends eastward. Longitude measurements are expressed in degrees. Longitude measurements are periodic and whose limits (min and max), while mathematically discontinuous, represent a continuous range.

Type Name	Primitive Type	Range of Values	Description
GroundSpeed	double	fractionDigits=3 maxInclusive=299,792,458 minInclusive=-299,792,458 units=MeterPerSecond referenceFrame=TrueNorth	The magnitude of the horizontal velocity vector of an aircraft relative to the ground.
HeadingCurrentDirection	double	fractionDigits=3 maxInclusive=3.142 minInclusive=-3.142 units=Radian referenceFrame=CurrentDirection	Describes heading as a value between -pi and pi with respect to the current direction.
HeadingMagneticNorth	double	fractionDigits=3 maxInclusive=3.142 minInclusive=-3.142 units=Radian referenceFrame=MagneticNorth	Describes heading as a value between -pi and pi with respect to Magnetic North.
HeadingTrueNorthAngle	double	fractionDigits=3 maxInclusive=3.142 minInclusive=-3.142 units=Radian referenceFrame=TrueNorth	Describes heading as a value between -pi and pi with respect to True North.
HeadingWindDirection	double	fractionDigits=3 maxInclusive=3.142 minInclusive=-3.142 units=Radian referenceFrame=WindDirection	Describes heading as a value between -pi and pi with respect to the wind direction.
IndicatedAirspeed	double	fractionDigits=6 maxInclusive=299,792,458 minInclusive=0 units=MeterPerSecond referenceFrame=LocalAirMass	This type specifies the magnitude of an aircraft's velocity (the rate of change of its position). Indicated airspeed (IAS) is the airspeed read directly from the airspeed indicator on an aircraft, driven by the pitot-static system.
LargeCollectionSize	long	fractionDigits=0 maxInclusive=2147483647 minInclusive=0 units=N/A	Specifies the size of a Large Collection.
MSLAltitude	double	fractionDigits=3 minInclusive=0.0 units=Meter referenceFrame=Altitude	Describes the current orthometric height above the Geoid (Mean Sea Level).
NumericGUID	octet[16]	units=N/A minInclusive=0 maxInclusive=(2^128)-1 fractionDigits=0	Represents a 128-bit number according to RFC 4122 variant 2.
PitchAcceleration	double	fractionDigits=3 maxInclusive=10000 minInclusive=0 units=RadianPerSecondSquared referenceFrame=Counting	Specifies the platform's angular acceleration about the lateral axis in a locally level, North-East-Down coordinate system centered on the platform.

Type Name	Primitive Type	Range of Values	Description
PitchHalfAngle	double	fractionDigits=3 maxInclusive=1.571 minInclusive=-1.571 units=Radian referenceFrame=PlatformNED	Specifies the platform's rotation about the lateral axis (e.g. the axis parallel to the wings) in a locally level, North-East-Down coordinate system centered on the platform. Pitch is zero when the platform is "nose to tail level" in the North-East plane. The measurement is stated in radians between -0.5 pi and 0.5 pi.
PitchRate	double	fractionDigits=3 maxInclusive=32.767 minInclusive=0 units=RadianPerSecond referenceFrame=Counting	Specifies the rate of change of the platform's pitch angle relative to a NED frame centered at the platform location.
RollAcceleration	double	fractionDigits=3 maxInclusive=10000 minInclusive=0 units=RadianPerSecondSquared referenceFrame=Counting	Specifies the angular acceleration of the platform about the longitudinal axis (e.g. the axis through the body of the vehicle from tail to nose) in a locally level, North-East-Down coordinate system centered on the platform.
RollAngle	double	fractionDigits=3 maxInclusive=3.142 minInclusive=-3.142 units=Radian referenceFrame=PlatformNED	Specifies a platform's rotation about the longitudinal axis (e.g. the axis through the body of the vehicle from tail to nose) in a locally level, North-East-Down coordinate system centered on the vehicle. Roll is zero when the platform is "wing-tip to wing-tip" level in the North-East plane. The measurement is stated in radians between -pi and pi.
SidesCount	long	fractionDigits=0 maxInclusive=255 minInclusive=3 units=N/A	Represents the number of sides a polygon has using a positive integer.
SpeedBSL	double	fractionDigits=3 maxInclusive=299,792,458 minInclusive=-299,792,458 units=MeterPerSecond referenceFrame=BSL	This type stores speed in meters/s in a below sea level reference frame.
SpeedBSLAcceleration	double	fractionDigits=6 maxInclusive=299792458 minInclusive=-299792458 units=MeterPerSecondSquared	Describes change in velocity over time below sea level.
SpeedLocalWaterMass	double	fractionDigits=6 maxInclusive=299,792,458 minInclusive=0 units=MeterPerSecond referenceFrame=LocalWaterMass	This type stores speed in meters/s.
StringLongDescription	string	length=4095 units=N/A minInclusive=N/A maxInclusive=N/A	Represents a long format description.

Type Name	Primitive Type	Range of Values	Description
StringShortDescription	string	length=1023 units=N/A minInclusive=N/A maxInclusive=N/A	Represents a short format description.
TurnRate	double	fractionDigits=3 maxInclusive=32.767 minInclusive=0 units=RadianPerSecondreference referenceFrame=Counting	Specifies the rate of change of the heading angle of a platform.
YawAcceleration	double	fractionDigits=3 maxInclusive=N/A minInclusive=N/A units=RadianPerSecondSquared referenceFrame=Counting	Specifies the platform's angular acceleration about the vertical axis in the body coordinate system.
YawPosAngle	double	fractionDigits=3 maxInclusive=6.283185307179586364925286766559 minInclusive=0 units=Radian referenceFrame=PlatformNED	The yaw angle relative to the NED coordinate system centered at the platform location.

A Appendices

A.1 Glossary

Note: This glossary aims to define terms that are uncommon, or have a special meaning in the context of UMAA and/or the DoD. This glossary covers the complete UMAA specification. Not every word defined here appears in every ICD.

Almanac Data (GPS)	A navigation message that contains information about the time and status of the entire satellite constellation.
Coulomb	The SI unit of electric charge, equal to the quantity of electricity conveyed in one second by a current of one ampere.
Ephemeris Data (GPS)	A navigation message used to calculate the position of each satellite in orbit.
Glowplug or Glow Plug	A heating device used to aid in starting diesel engines.
Interoperability	1) The ability to act together coherently, effectively, and efficiently to achieve tactical, operational, and strategic objectives. 2) The condition achieved among communications-electronics systems or items of communications-electronics equipment when information or services can be exchanged directly and satisfactorily between them and/or their users.
Mean Sea Level	The average height of the surface of the sea for all stages of the tide; used as a reference for elevations.
Middleware	A type of computer software that provides services to software applications beyond those available from the operating system. Middleware makes it easier for software developers to implement communication and input/output, so they can focus on the specific purpose of their application.
SoaML	The Service oriented architecture Modeling Language (SoaML) specification that provides a metamodel and a UML profile for the specification and design of services within a service-oriented architecture. The specification is managed by the Object Management Group (OMG).

A.2 Acronyms

Note: This acronym list is included in every ICD and covers the complete UMAA specification. Not every acronym appears in every ICD.

ADD	Architecture Design Description
AGL	Above Sea Level
ASF	Above Sea Floor
BSL	Below Sea Level
BWL	Beam at Waterline
C2	Command and Control
CMD	Command
CO	Comms Operations
CPA	Closest Point of Approach
CTD	Conductivity, Temperature and Depth
DDS	Data Distribution Service
DTED	Digital Terrain Elevation Data
EGM	Earth Gravity Model
EO	Engineering Operations
FB	Feedback
GUID	Globally Unique Identifier
HM&E	Hull, Mechanical, & Electrical
ICD	Interface Control Document

ID	Identifier
IDL	Interface Definition Language Specification
IMO	International Maritime Organization
INU	Inertial Navigation Unit
LDM	Logical Data Model
LOA	Length Over All
LRC	Long Range Cruise
LWL	Length at Waterline
MDE	Maritime Domain Extensions
MEC	Maximum Endurance Cruise
MM	Mission Management
MMSI	Maritime Mobile Service Identity
MO	Maneuver Operations
MRC	Maximum Range Cruise
MSL	Mean Sea Level
OMG	Object Management Group
PIM	Platform Independent Model
PMC	Primary Mission Control
PNT	Precision Navigation and Timing
PO	Processing Operations
PSM	Platform Specific Model
RMS	Root-Mean-Square
RPM	Revolutions per minute
RTPS	Real Time Publish Subscribe
RTSP	Real Time Streaming Protocol
SA	Situational Awareness
SEM	Sensor and Effector Management
SO	Support Operations
SoaML	Service-oriented architecture Modeling Language
STP	Standard Temperature and Pressure
UCS	Unmanned Systems Control Segment
UMAA	Unmanned Maritime Autonomy Architecture
UML	Unified Modeling Language
UMS	Unmanned Maritime System
UMV	Unmanned Maritime Vehicle
UxS	Unmanned System
WGS84	Global Coordinate System
WMM	World Magnetic Model
WMO	World Meteorological Organization