Unmanned Maritime Autonomy Architecture (UMAA) Software Development Kit (SDK) Systems Model

Component Specification

For

UMAA SDK Autopilot

Component Model Element Server Id: eed71342-8465-4463-ac06-1ab22f96f106

Published on: November 05, 2024

by: Chris C. Olson, JHU/APL

Source Data

From Unmanned Maritime Systems Autonomy Model

Release branch version commit: 197

(NOTE: This document was not auto-generated from the data model)

CONTENTS

1 Scope

1.1 Identification

This document describes the Component Specification for the UMAA SDK Autopilot Component.

1.2 Unmanned Autonomy Architecture (UMAA) Component Overview

For an overview of UMAA, see UMAA-INF-ADD "Unmanned Maritime Autonomy Architecture (UMAA) Architecture Design Description (ADD)."

A Component is defined as a set of one or more Service Providers and/or Consumers along with any non-UMAA defined Interfaces that collectively meet the needs to perform a specific function or capability. Service Providers describe which UMAA Services must be implemented in accordance with the UMAA ICDs as part of the Component. Service Consumers describe software on the Component that utilizes a Service Provider either to obtain data or to effect change within a system utilizing interfaces described in the UMAA ICDs. In order to support an acquisition strategy that focuses on rapid development and integration of scoped software components, the Architecture Framework Working Group (AFWG), under the direction of the Chief Unmanned Autonomy Framework Architect (CUAFA), has established a reference architecture that defines several components for implementing functions within an autonomy system with standardized interfaces. The model-based reference architecture is contained within a SysML model hosted by Naval IME and maintained by PEO USC, PMS406.

The reference architecture describes Components in three abstraction layers: Component Definition, Component Specification, and Component Design. All three abstraction layers are represented in SysML, with the Component Definitions managed by the UMAA Board and the Component Specification and Design managed as part of the Autonomy Baseline under the CUAFA. Figure 1 summarizes the content described in each abstraction layer.

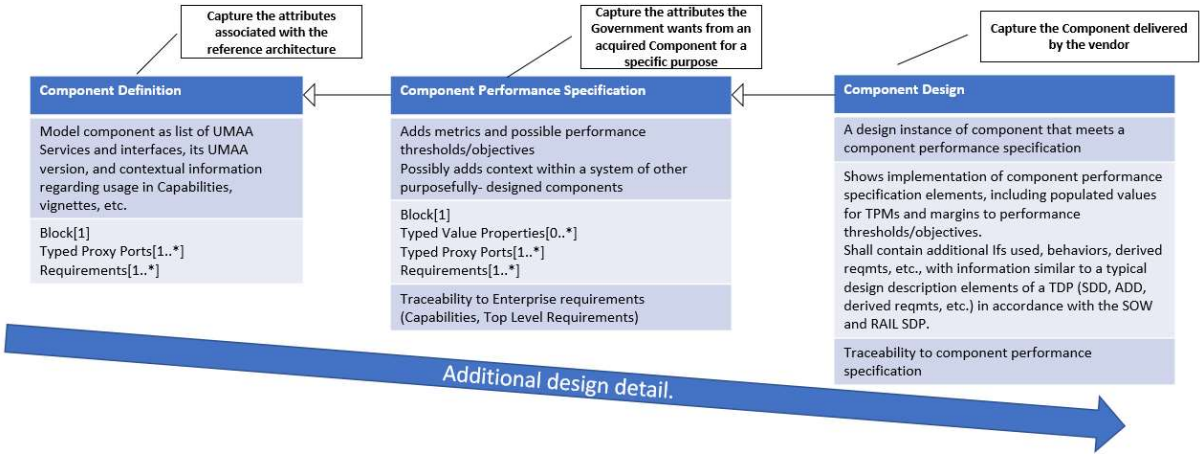

Figure 1. Model-based Components

The Component Definition is the Component's highest level of abstraction represented in the reference architecture. It provides context for the need of the Component, identifies the Service Providers and Service Consumers and their interfaces, requirements to conform with the reference architecture, and metrics that may be used in the Component Specification to monitor performance during the acquisition process.

The Component Specification is a representation of the Component that captures the Government's desired attributes for the Component in a specific acquisition. In addition to the attributes from the Component Definition, the Component Specification identifies any metrics the performer must report on during the contract and any additional requirements (threshold/objective) derived for the specific acquisition. Derived requirements show traceability to parent requirements contained in platform specifications or architecture elements in the PMS 406 UxV Portfolio Model.

The Component Design represents a Component that is implemented by a developer and delivered to the Government as part of the Autonomy Baseline. The Component Design shows implementation of the Component Specification, including traceability of design elements to the requirements, metrics, and Service Participant elements they satisfy. Component Designs provide similar information to Software Description documents and Algorithm Description Documents and may be utilized as evidence when assessing UMAA compliance. The "PMS 406 Autonomy Software Development Process" contains additional guidance in the development of Component Designs.

The three abstraction layers are purposely defined to support various phases of the acquisition process and for controlling and evaluating Components for re-use. The figure below illustrates one example for how each is used.

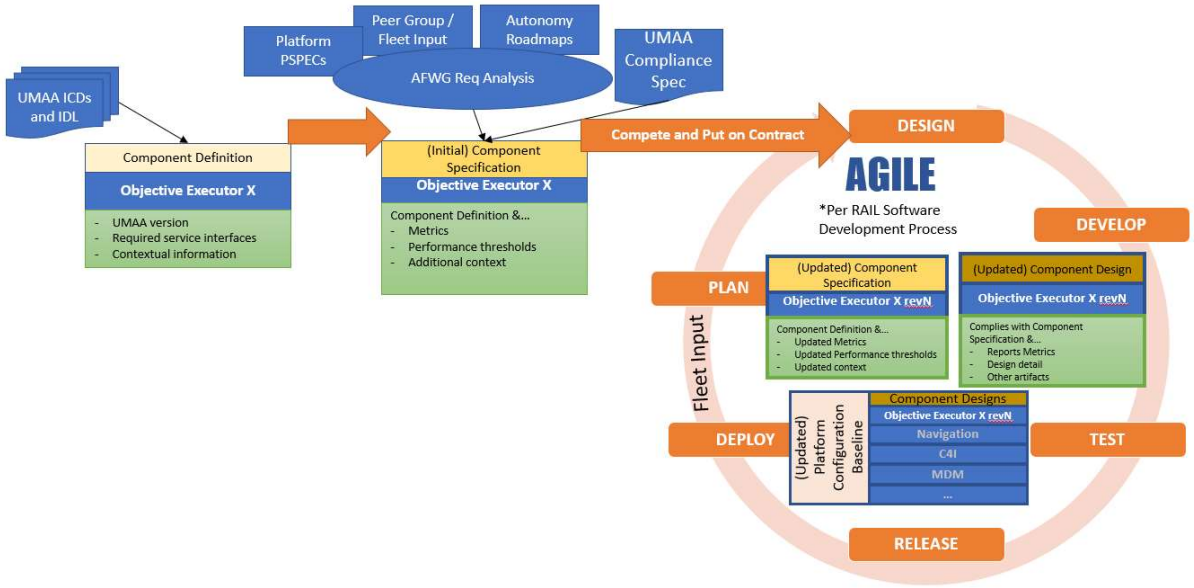

Figure 2. Example Use of Model-based Component Definition, Specification, and Design

1.2.1 Component Reference Architecture Overview

A set of UMAA Components form the basis of a reference architecture. Components satisfying the needs for new capabilities must be integrated into this reference architecture. Due to sometimes conflicting needs, there are two "notional" platform configurations that the AFWG uses to describe the reference architecture: one for unmanned surface vehicles and another for unmanned underwater vehicles. The figure below illustrates one aspect of the reference architecture and its componentization. Blocks in the diagram represent released Component Definitions. Clouds in the diagram represent areas of capability the AFWG is working to componentize at this time.

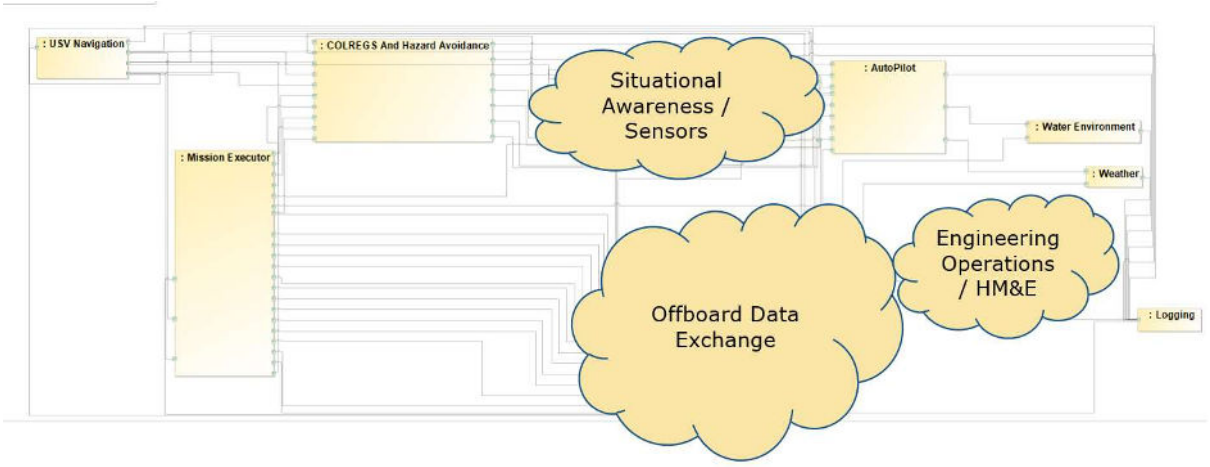

Figure 3. Component Reference Architecture

1.3 Document Overview

This document represents an export of the model-based Component Definition for this Component.

This document is organized in the following sections:

- **Scope** – Provides a brief overview of UMAA, definition of autonomy components, and the organization of this document.
- **References** – Lists the documents or other artifacts referenced in this document.
- **Component Description** – Provides context for this specific component and introduces its interfaces.
- **Component Requirements** – Identifies requirements associated with the Component Definition.
- **Component Metrics** – Identifies metrics that may be associated with the component in its Component Specification.
- **Component Interfaces** – Provides additional detail on the component's interfaces.

This document is auto-generated from a model and should not be altered in its document-based form. Source data for this published document is contained in the Unmanned Maritime Systems Autonomy Model hosted on Naval IME's Teamwork cloud server. Details on model version and component identification within the model are contained on the cover page of this document. For access to the model-based component specification contact NAVSEA, PEO USC, ATTN: PMS 406 via the contact information on the cover page of this document.

2 References

Number	Version	Title
UMAA-INF-ADD	1.1a	Unmanned Maritime Autonomy Architecture (UMAA) Architecture Design Description (ADD)
T0300-BE-IDS-010	1	Unmanned Maritime Autonomy Architecture (UMAA) Compliance Specification
UMAA-SPEC-EOICD UMAA-SPEC-MOICD UMAA-SPEC-MMICD UMAA-SPEC-SEMICD UMAA-SPEC-SAICD	6.0.0	UMAA Interface Control Documents (ICD)

Number	Version	Title
UMAA-SPEC-SOICD		
N/A	Draft (v0.3)	PMS 406 Autonomy Software Development Process
N/A	3.0	Autonomy Component to Simulated Procession of UxV Resources (SPUR) Version 2.0 ICD

3 Component Description

3.1 Component Summary

The Autopilot Component provides for vehicle maneuverability control. The UMAA SDK Autopilot Component must be capable of being commanded by the GlobalVectorControl service, and may optionally be controlled by the PrimitiveDriverControl service. The platform is not capable of hovering, so the GlobalHoverControl service is not necessary. This Component will provide vehicle control commands over the UDP transport protocol to the SPUR simulation tool, developed by ARL-PSU, according to the SPUR version 2.0 ICD to control the simulated maritime vehicle.

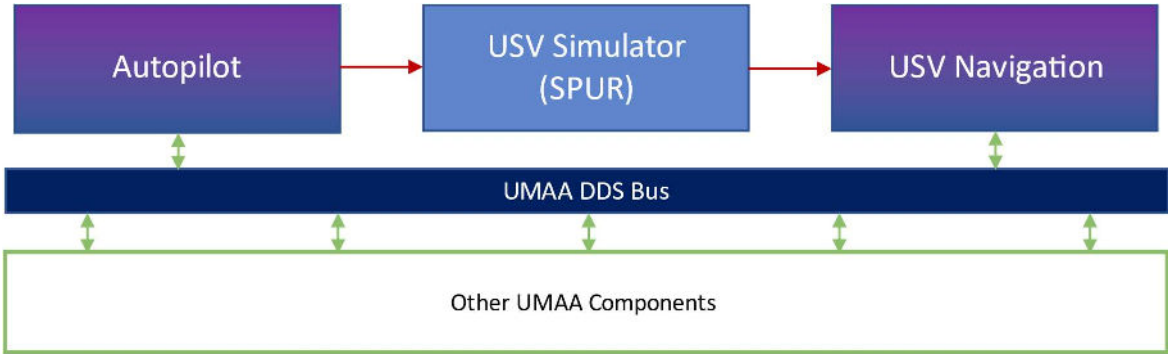

Figure 4. UMAA SDK Autopilot and USV Navigation Components with SPUR

The Autopilot Component provides UVPlatformSpecs in part to convey overall platform maneuvering limitations. Autopilot may also add constraints to the mission plan using MissionPlanConstraintControl and ConditionalControl given more dynamic vehicle control constraints. This Component may consume wind and water current information to assess the environmental conditions surrounding the vessel.

3.2 Component Interfaces

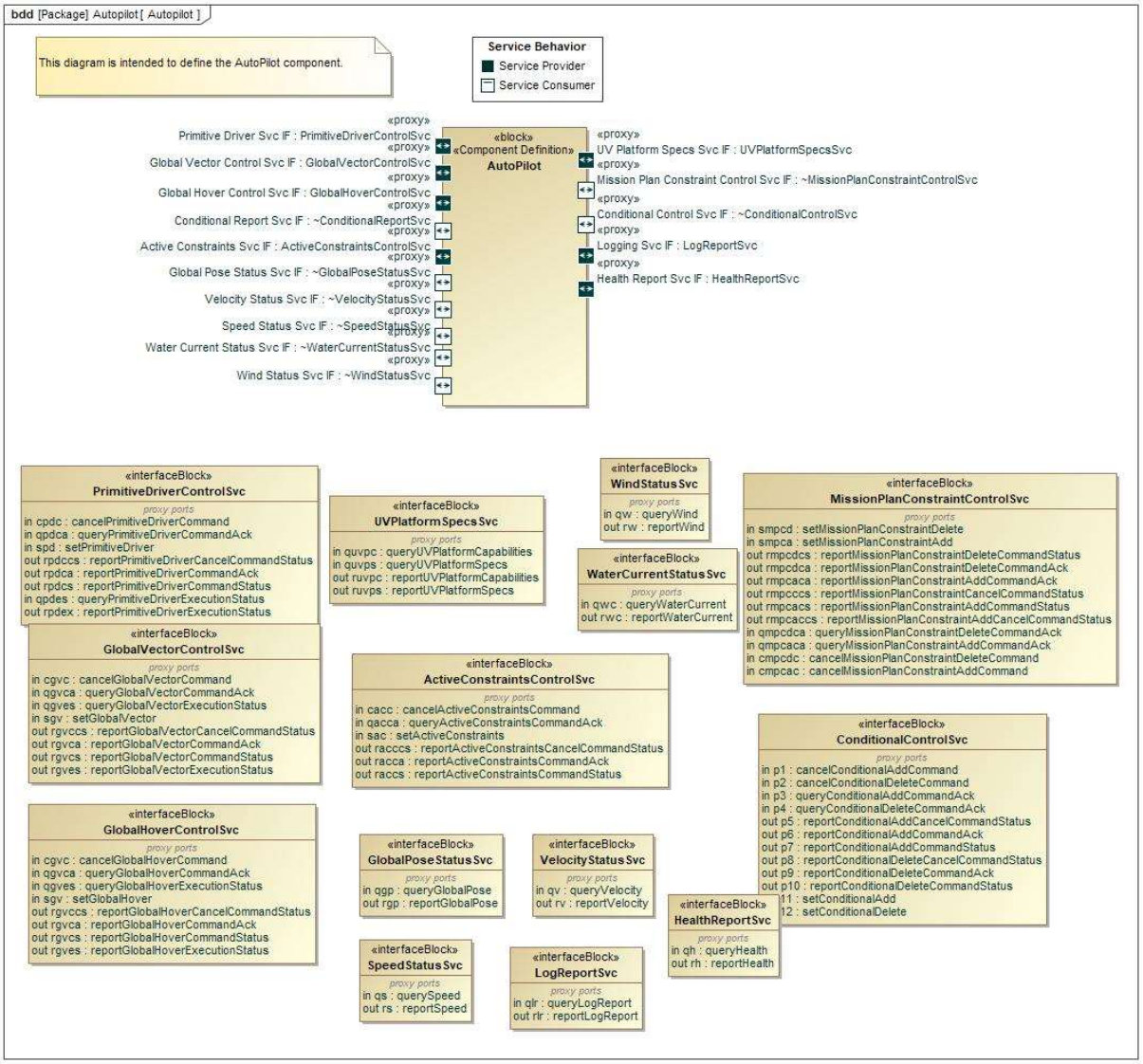

Figure 5. UMAA SDK Autopilot

The above block definition diagram summarizes the Component Specification. Ports on the Component identify the Service Providers and Service Consumers for the Component. Conjugated ports, annotated with "~" prior to the defined port Type, represent usage of the interface as a consumer. Otherwise, the port represents usage of the interface as a provider of the service. The usage of a Service for the particular component is also identified by its color per the legend. Each port is typed by an Interface Block representing a UMAA Service, with nested proxy ports representing the operations of the Service. Additional details on the interfaces are described in section 6.

Note: Conjugated proxy ports, annotated with "~" prior to the defined Type, represent usage of the interface as a consumer. This results in the flows defined in the typed InterfaceBlock to be reversed when used by the component. Proxy ports that are not conjugated represent usage of the interface as a provider and the flows defined in the Typed InterfaceBlock are not reversed. The Interface Blocks representing Services have nested proxy ports with directivity (in / out) defined from the perspective of a Service Provider. This is because the nested proxy ports' flow properties,

representing the Topics, are defined with directivity representing a message being published or subscribed by a Service Provider in relation to the DDS bus. This is why a conjugated port is used to represent a Service Consumer; the directivity of each flow property is reversed. This is consistent with the UMAA ICD definition of Services, with operations similarly categorized as Service Requests (inputs) and Service Responses (outputs).

3.3 Component Behavior

No additional behavior diagrams are included in this specification.

3.4 UMAA SDK Autopilot Component Requirements [UMAASDK-AUTO-1]

3.4.1 Functional Requirements [UMAASDK-AUTO-1.1]

3.4.1.1 Maneuver Platform [UMAASDK-AUTO-1.1.1]

The Autopilot shall maneuver the platform using the GlobalVectorControl service interface.

3.4.1.2 Consume Own-ship Navigation Information [UMAASDK-AUTO-1.1.2]

The Autopilot shall consume own-ship navigation information using the GlobalPoseStatus, SpeedStatus, and VelocityStatus service interfaces.

3.4.1.3 Health Reporting [UMAASDK-AUTO-1.1.3]

The Autopilot shall be capable of reporting on the health of the component and on any errors when they arise.

3.4.2 Interface Requirements [UMAASDK-AUTO-1.2]

The Autopilot is required to implement all interfaces described in Figure 4 of this specification in accordance with T0300-BE-IDS-010 "Unmanned Maritime Autonomy Architecture (UMAA) Compliance Specification." The subsections below include additional specification of select UMAA interface details.

3.4.2.1 GlobalVectorControl [UMAASDK-AUTO-1.2.1]

3.4.2.1.1 SpeedRequirementVariantType [UMAASDK-AUTO-1.2.1.1]

The Autopilot shall be capable of handling SpeedControlType GROUNDSPEEDREQUIREMENTVARIANT_D in GlobalVectorControl.

3.4.2.1.2 DirectionRequirementType [UMAASDK-AUTO-1.2.1.2]

The Autopilot shall be capable of handling DirectionRequirementType DIRECTIONTRUENORTHREQUIREMENTVARIANT_D in GlobalVectorControl.

3.4.2.1.3 ElevationType [UMAASDK-AUTO-1.2.1.3]

The Autopilot shall be capable of handling Elevation Type enumeration DEPTHREQUIREMENTVARIANT_D in GlobalVectorControl.

3.4.2.2 GlobalPoseStatus [UMAASDK-AUTO-1.2.2]

3.4.2.2.1 depth [UMAASDK-AUTO-1.2.2.1]

The Autopilot shall be capable of handling depth in GlobalPoseReport.

3.4.2.2.2 attitude [UMAASDK-AUTO-1.2.2.2]

The Autopilot shall be capable of handling attitude in GlobalPoseReport.

3.4.2.3 SpeedStatus [UMAASDK-AUTO-1.2.3]

3.4.2.3.1 speedOverGround [UMAASDK-AUTO-1.2.3.1]

The Autopilot shall be capable of handling speedOverGround in SpeedStatus.

3.4.3 Performance Requirements [UMAASDK-AUTO-1.3]

3.4.3.1 Minimize Delay of Publishing a SPUR Command [UMAASDK-AUTO-1.3.1]

The Autopilot shall minimize the delay between receiving a GlobalVectorCommand message and publishing a vehicle vector command to SPUR.

3.4.3.2 SPUR Command Receive Rate [UMAASDK-AUTO-1.3.2]

The Autopilot shall not exceed the rate in which SPUR can receive and process incoming vehicle vector commands.

3.4.4 Design Constraints [UMAASDK-AUTO-1.4]

3.4.4.1 UMAA v6.0.0 [UMAASDK-AUTO-1.4.1]

The Autopilot shall meet requirements of UMAA release v6.0.0.

3.4.4.2 Configurable QoS [UMAASDK-AUTO-1.4.2]

Quality of Service (QoS) for the Autopilot (domain participants), service interfaces (DDS endpoints), and Topics shall be configurable without changes to or recompiling of the implementation source code.

3.4.4.3 Configurable Domain ID and Partitions [UMAASDK-AUTO-1.4.3]

Domain ID and partitions for the Autopilot and its services shall be configurable without changes to or recompiling of the implementation source code.

3.4.4.4 Configurable Source and Destination GUIDs [UMAASDK-AUTO-1.4.4]

Source and destination IdentifierTypes shall be configurable using without changes to or recompiling of the implementation source code.

4 Component performance metrics

The following are metrics that must be measured for this component or the system that contains this component. Both component-level and system-level metrics must be captured during component development in accordance with the PMS 406 Autonomy Software Development Process. The system-level metrics are captured using the defined test bed / simulation tools as determined by the PAA (platform-specific system) or ABM (reference architecture system).

Metric Name	Metric Description
Form Messages Correctly	The Component forms UMAA service messages correctly. This is per the UMAA standard and per the Component Specification (e.g., specified VariantType, optionals, etc.).
Vehicle Command Publish Rate	The rate in which vehicle control commands are sent to SPUR.
Execution Status Messages	The Autopilot reports its command execution status at some rate, if supported by the control service.

5 Component interfaces

The sections below describe the interfaces used by the component. Each Service interface is modeled as an Interface Block with nested proxy ports for the Service's operations. Each nested proxy port has a flow property typed by a signal representing the Topic associated with the operation. The directivity of this flow property is set from the perspective of a Service Provider in relation to the DDS bus, by definition. Each signal representing a Topic has one attribute representing its Data Type. The Topic and Data Type is the finest detail for the interface described in the Unmanned Maritime Systems Autonomy Model. For further details on the interface data, such as message and structure definitions, refer to the UMAA ICDs. Example use of the Services as part of a system can also be found in either the UMAA ICDs or in the Unmanned Maritime Systems Autonomy Model.

5.1 Conditional Control Svc IF

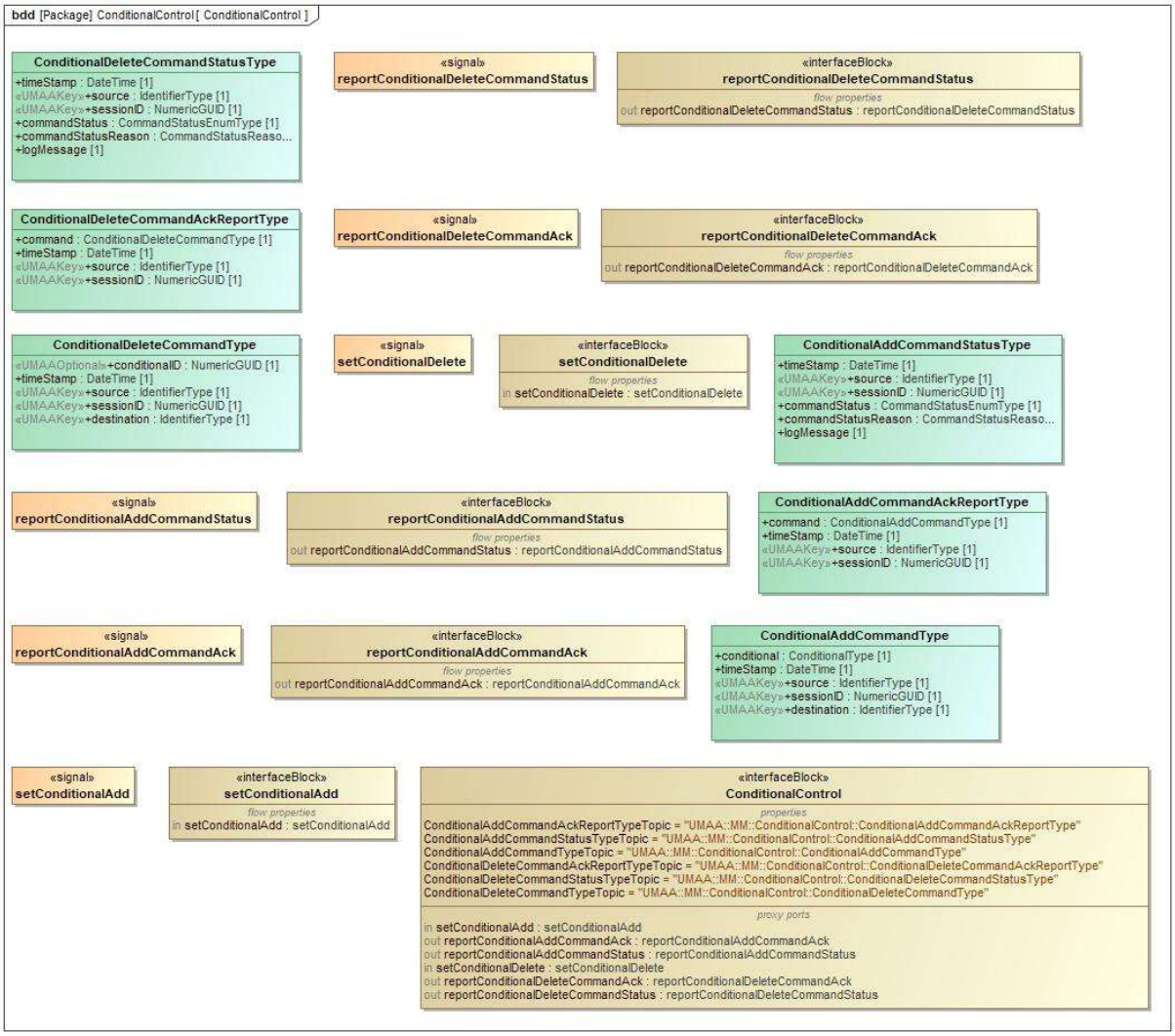

Figure 6. ConditionalControl

5.2 Mission Plan Constraint Control Svc IF

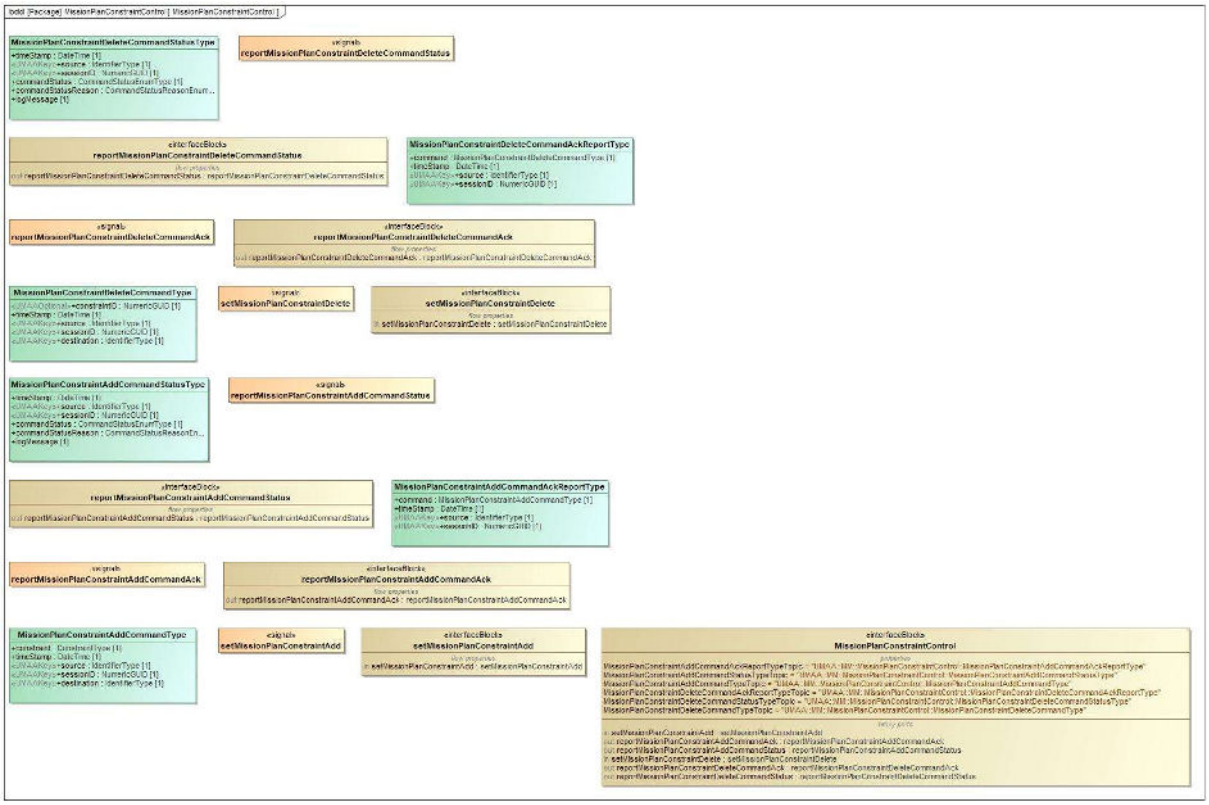

Figure 7. MissionPlanConstraintControl

5.3 Primitive Driver Control Svc IF

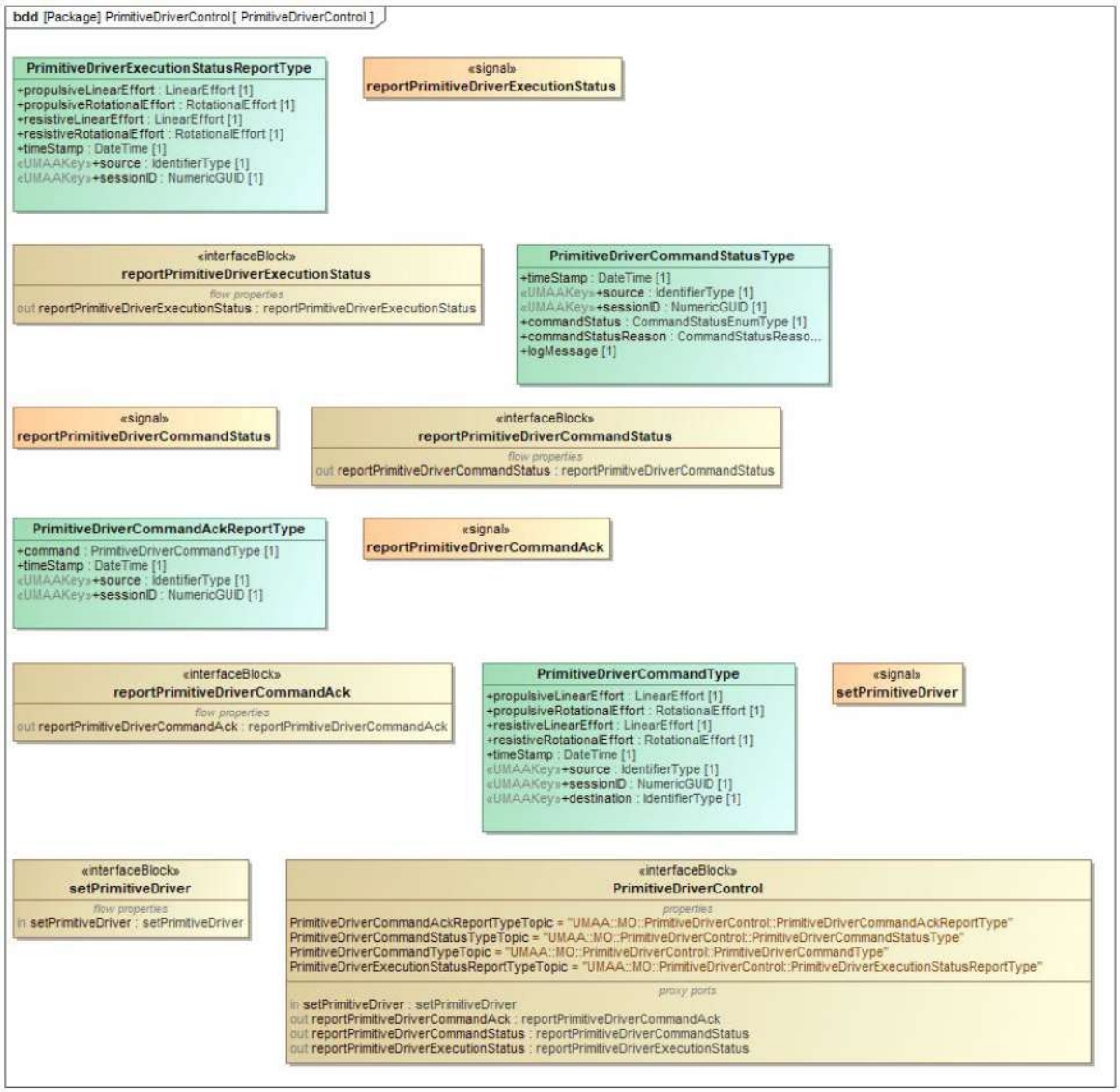

Figure 8. PrimitiveDriverControl

5.4 Log Report Svc IF

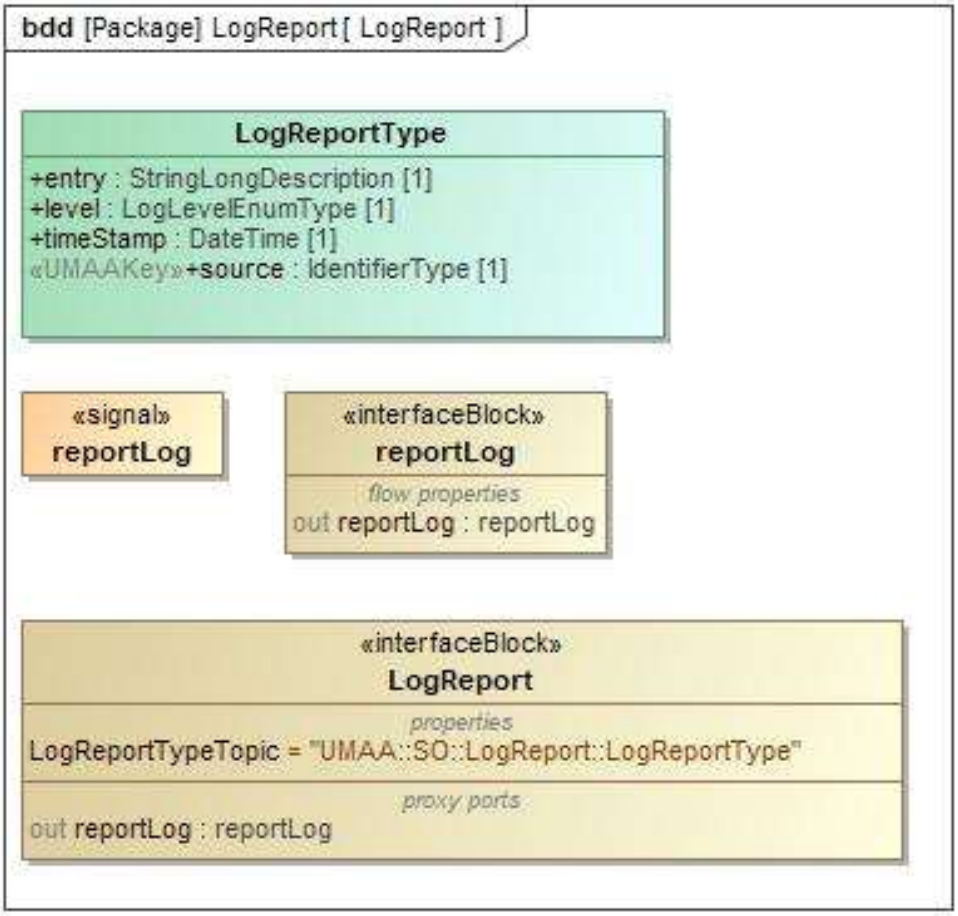

Figure 9. LogReport

5.5 Active Constraints Control Svc Consumer IF

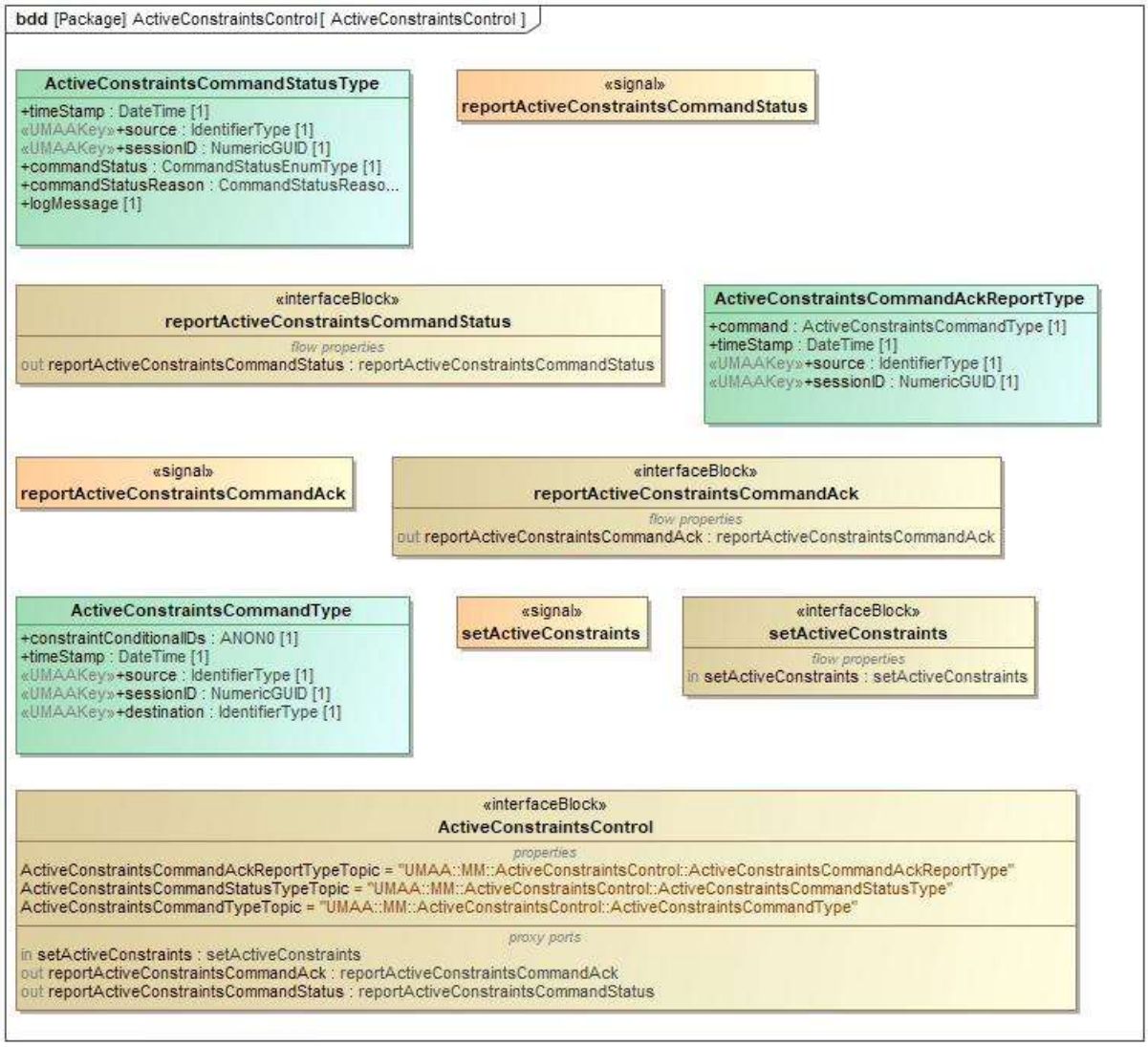

Figure 10. ActiveConstraintsControl

5.6 Conditional Report Svc IF

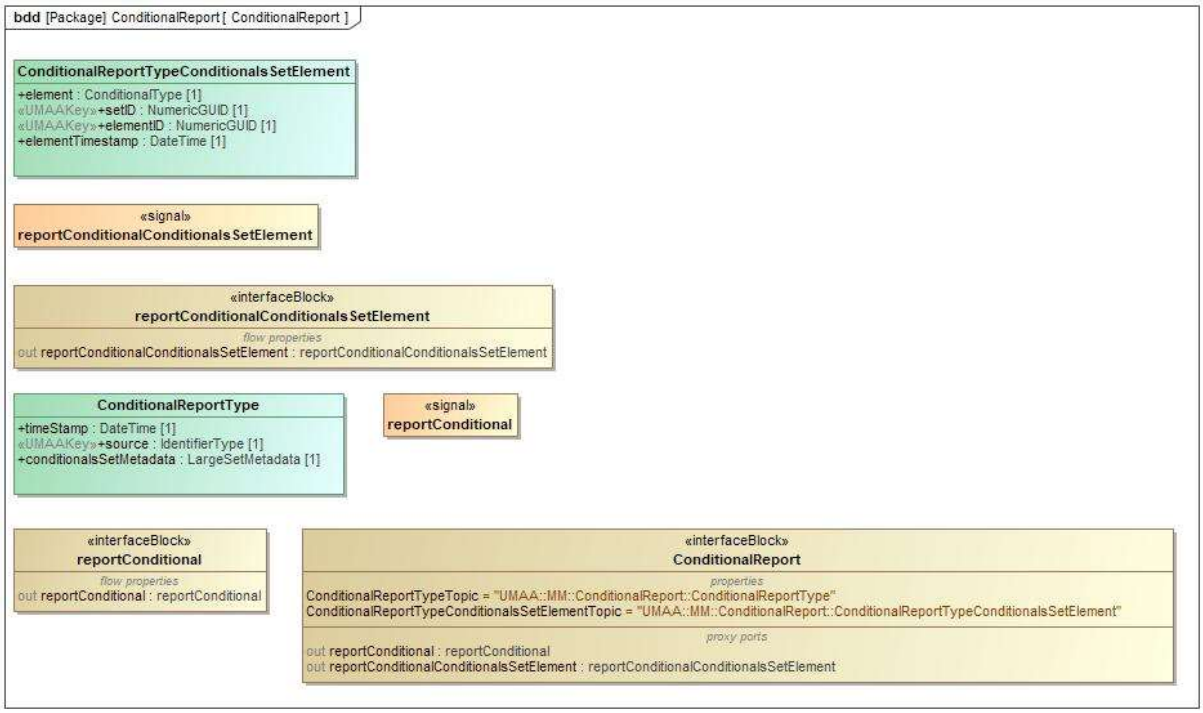

Figure 11. ConditionalReport

5.7 Health Report Svc IF

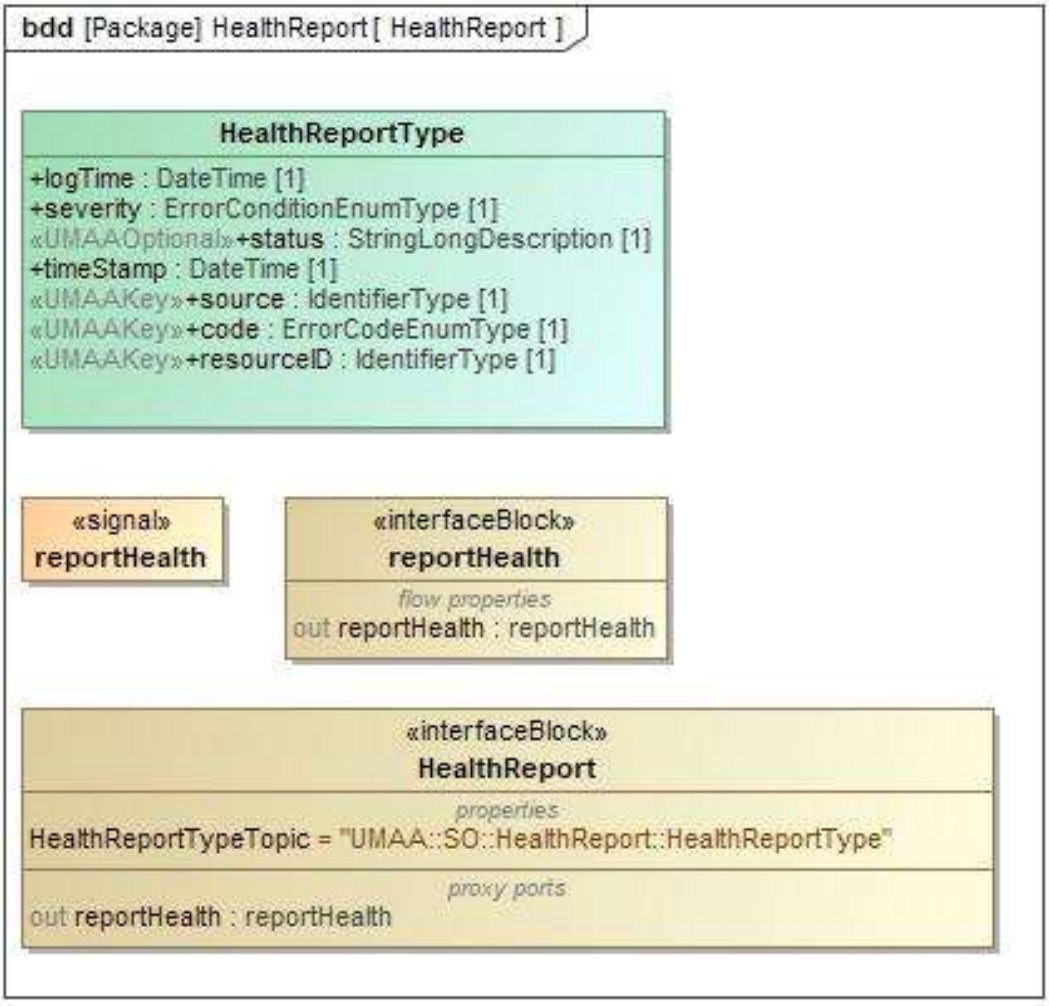

Figure 12. HealthReport

5.8 Global Pose Status Svc IF

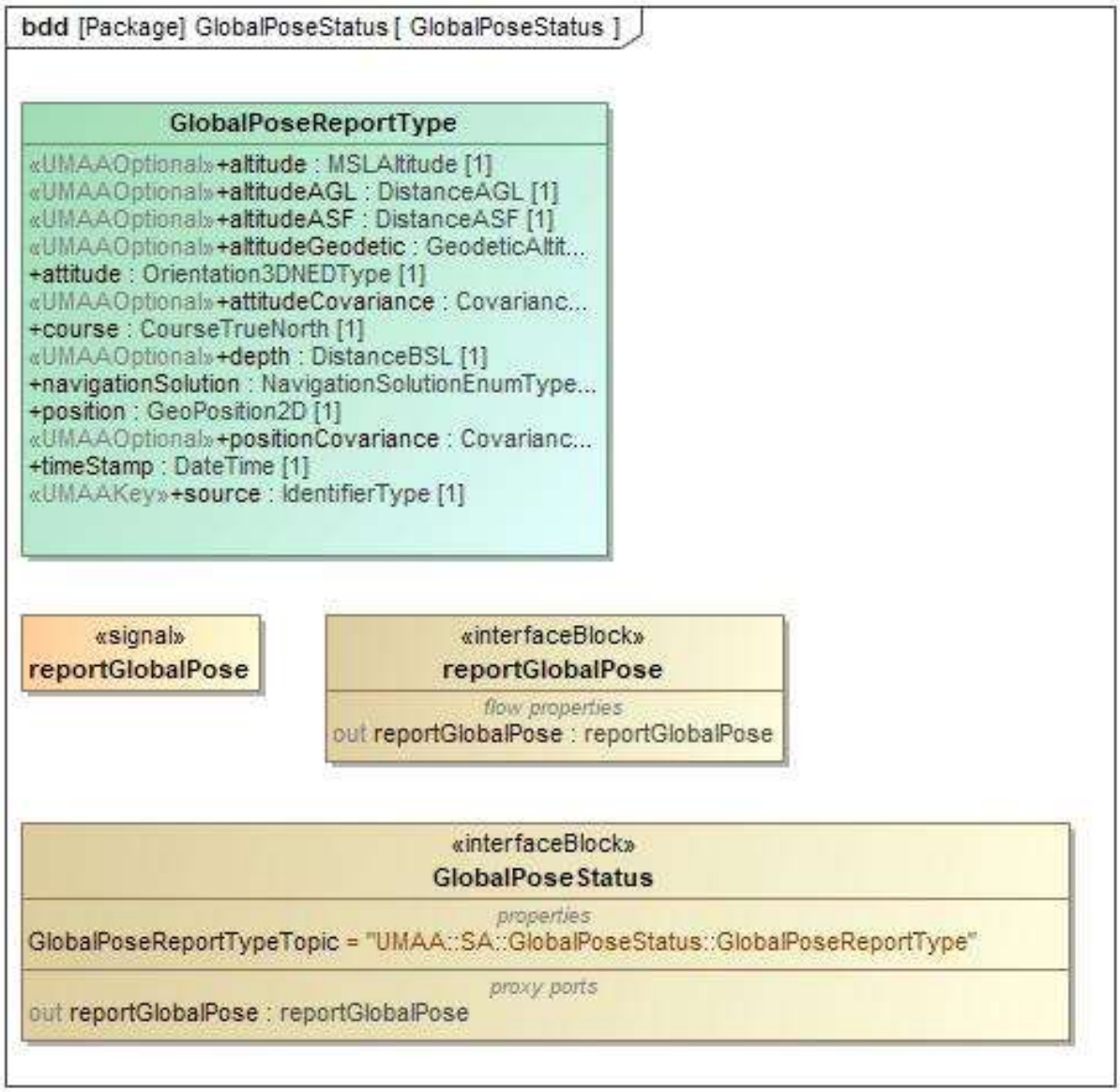

Figure 13. GlobalPoseStatus

5.9 Speed Status Svc IF

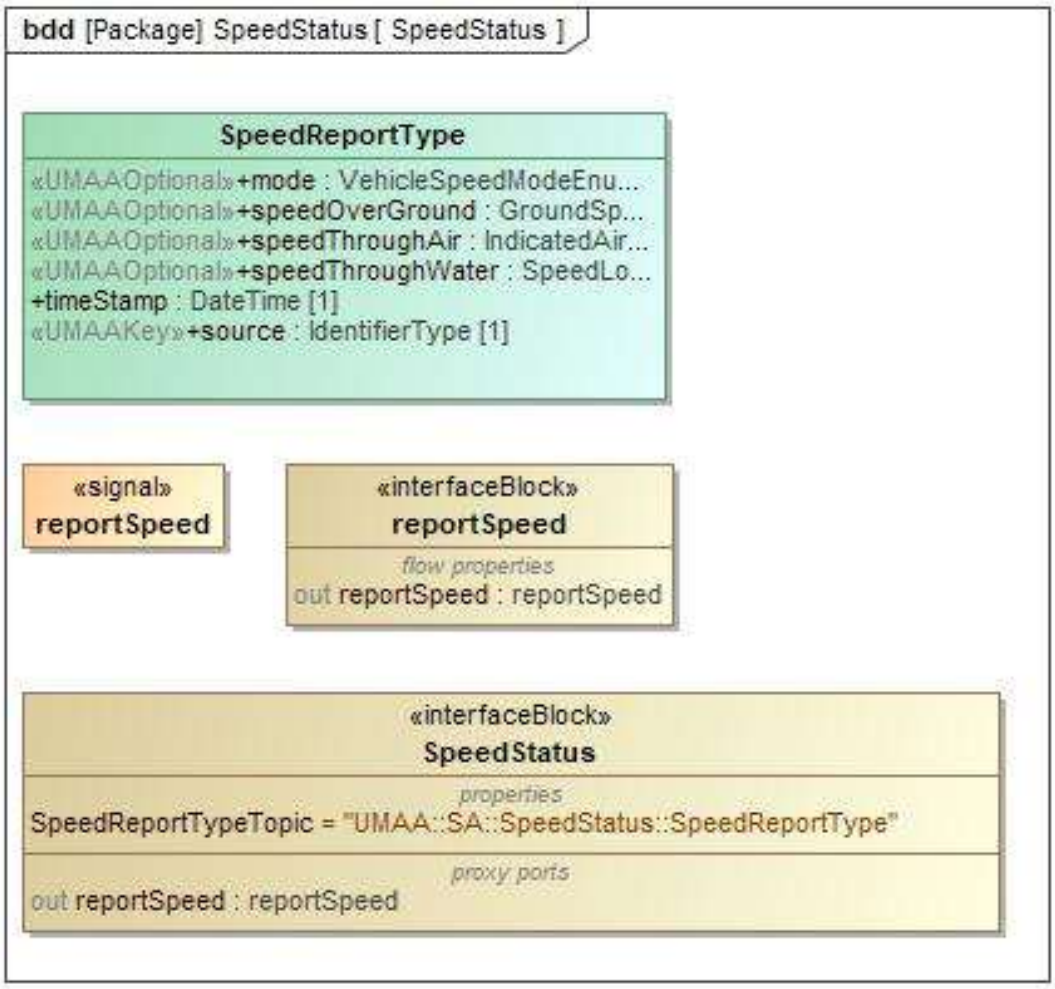

Figure 14. SpeedStatus

5.10 Global Vector Control Svc IF

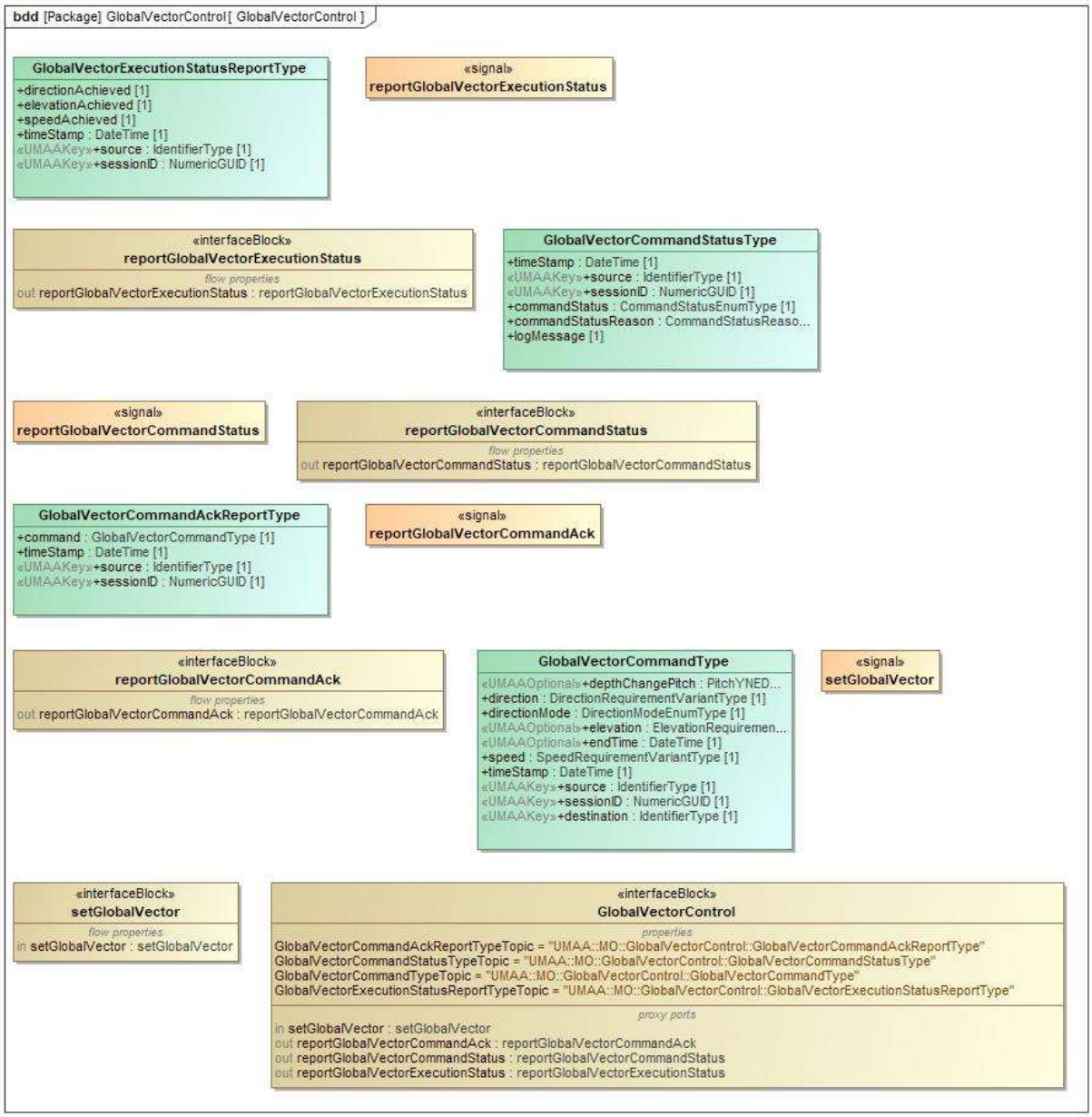

Figure 15. GlobalVectorControl

5.11 Global Hover Control Svc IF

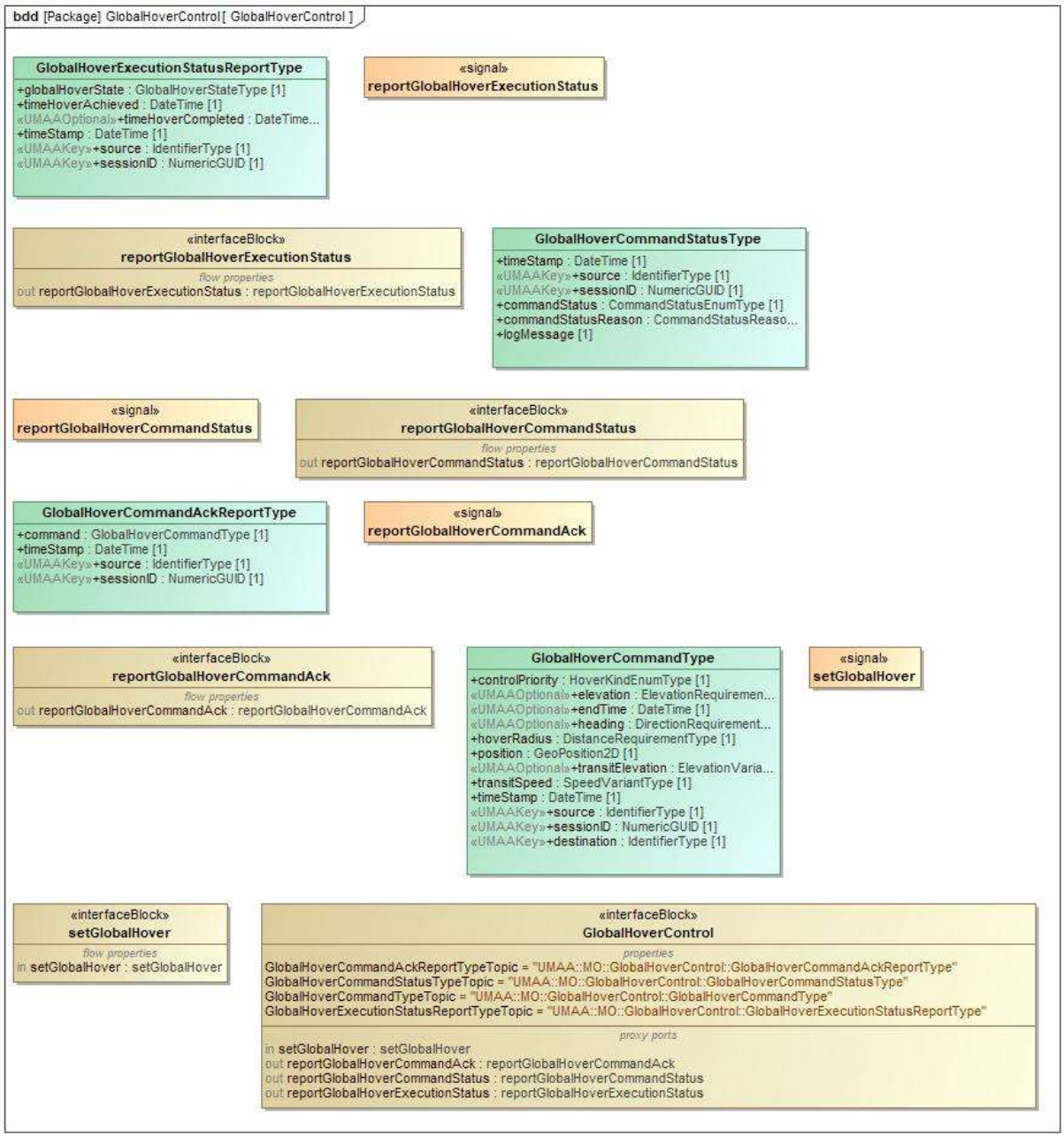

Figure 16. GlobalHoverControl

5.12 Velocity Status Svc IF

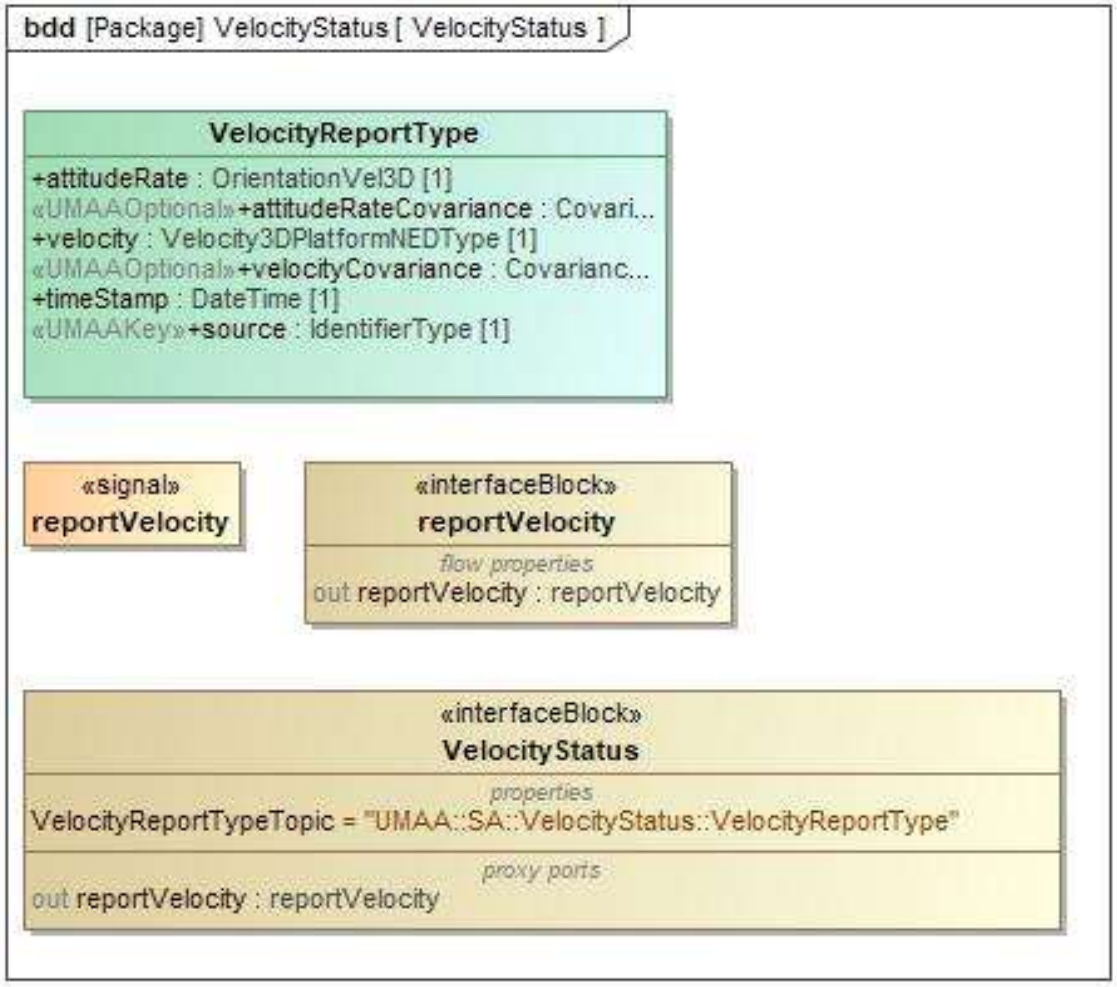

Figure 17. VelocityStatus

5.13 UV Platform Specs Svc IF

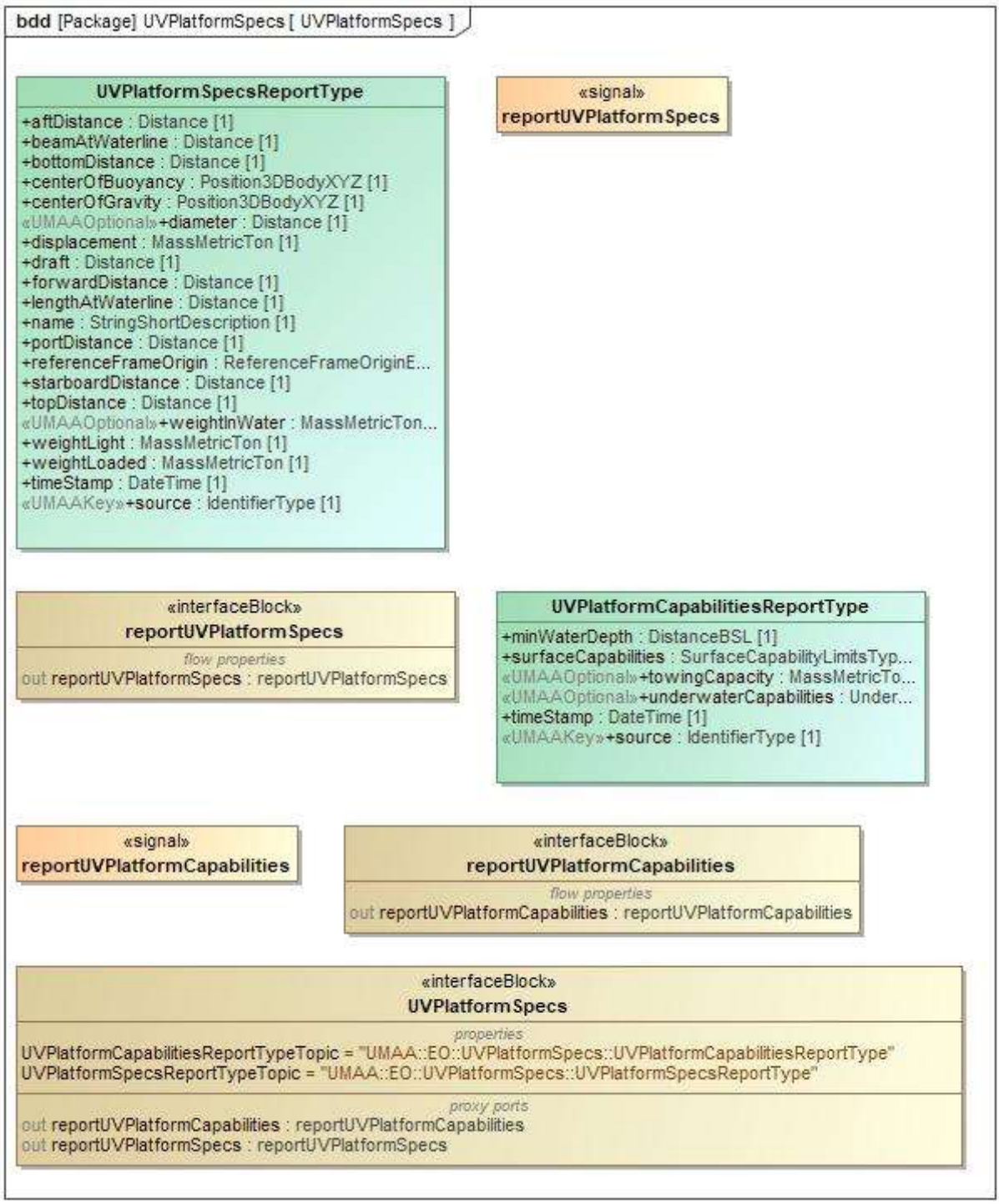

Figure 18. UVPlatformSpecs

5.14 Water Current Status Svc IF

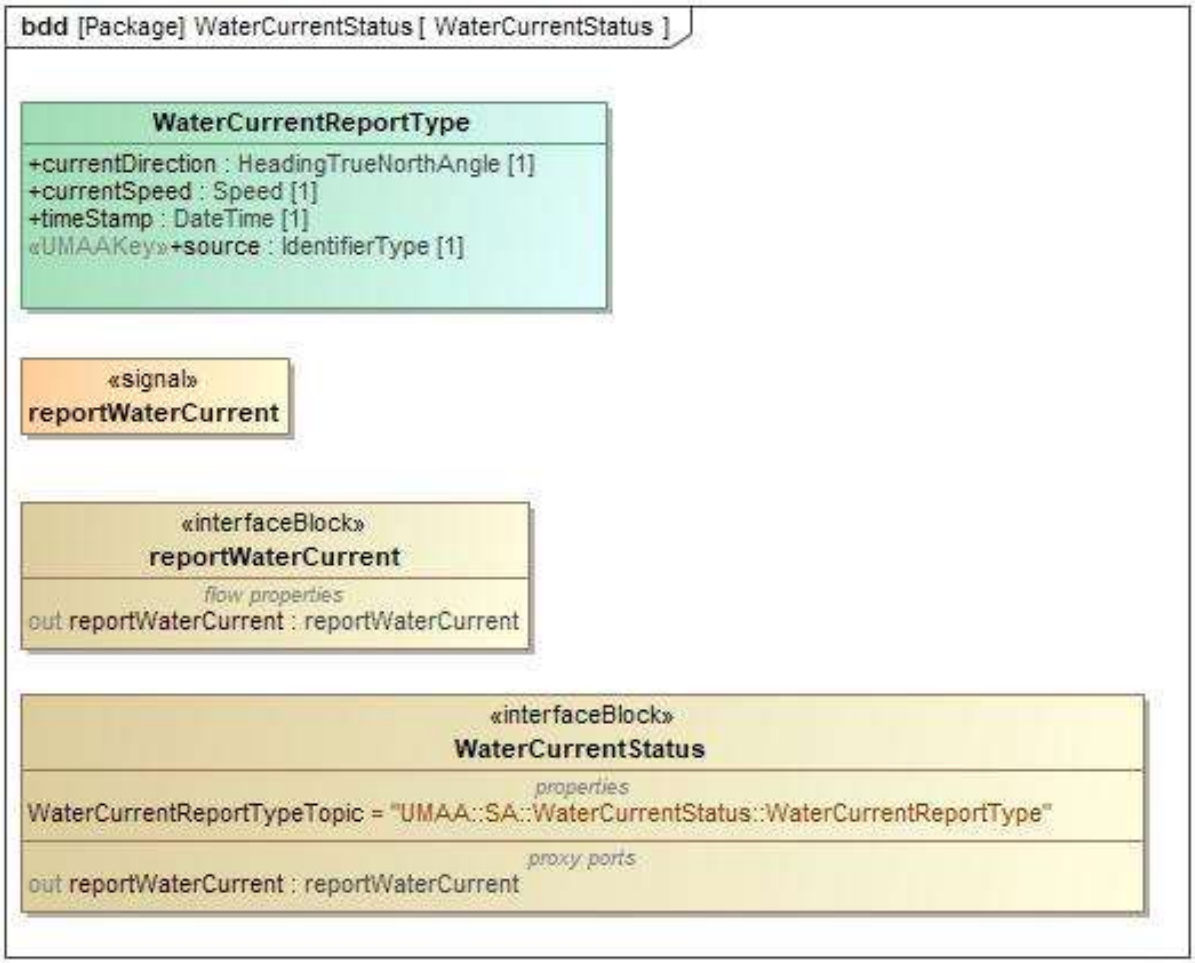

Figure 19. WaterCurrentStatus

5.15 Wind Status Svc IF

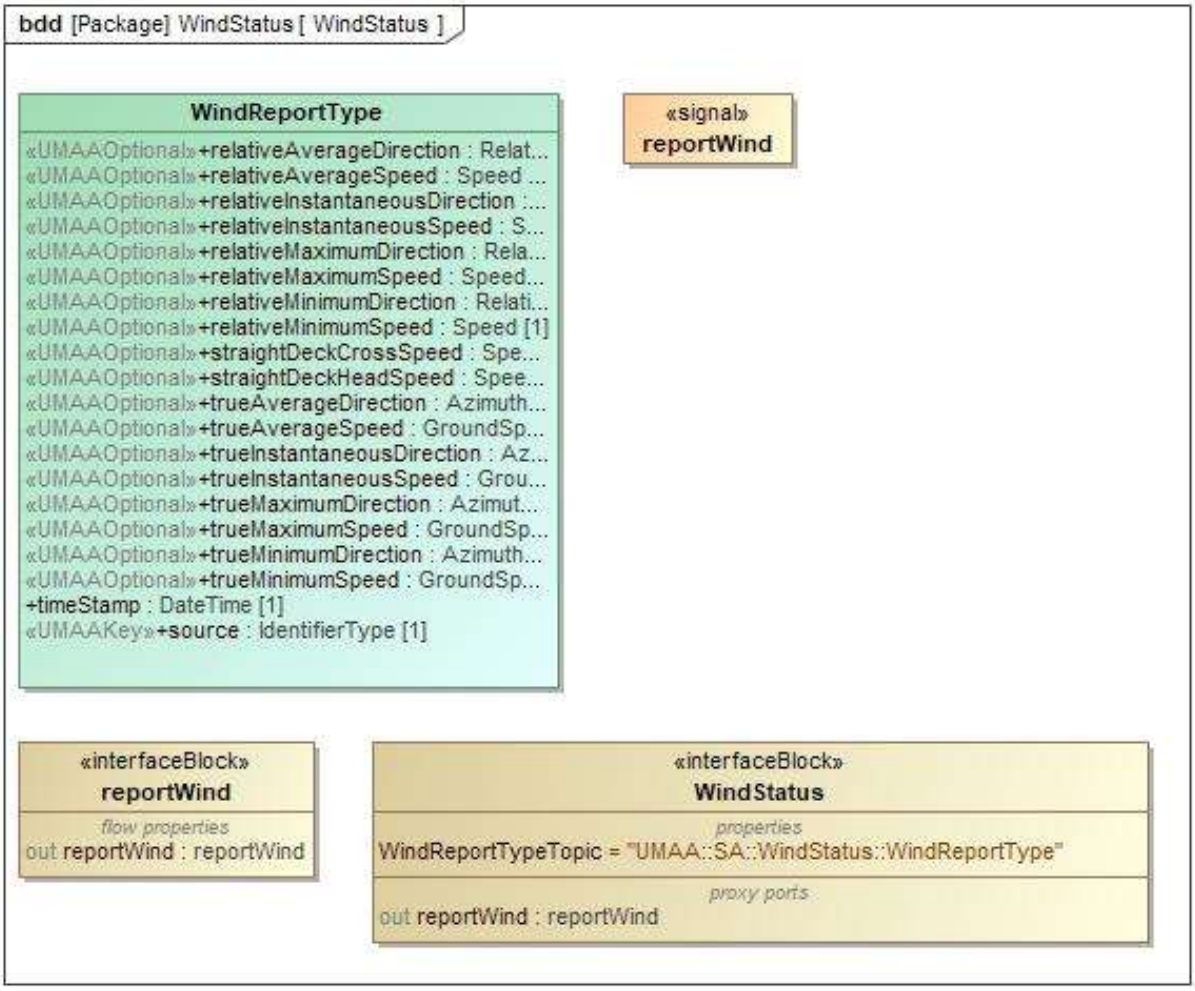

Figure 20. WindStatus

6 Traceability

Additional detail on traceability is available in the Unmanned Maritime Systems Autonomy Model.

END OF DOCUMENT