Unmanned Maritime Systems Autonomy Component Model

Component Definitions

Version 1.0

CONTENTS

1 Scope

1.1 Identification

This document describes the Component Definitions released for use with UMAA 6.0 versioned services.

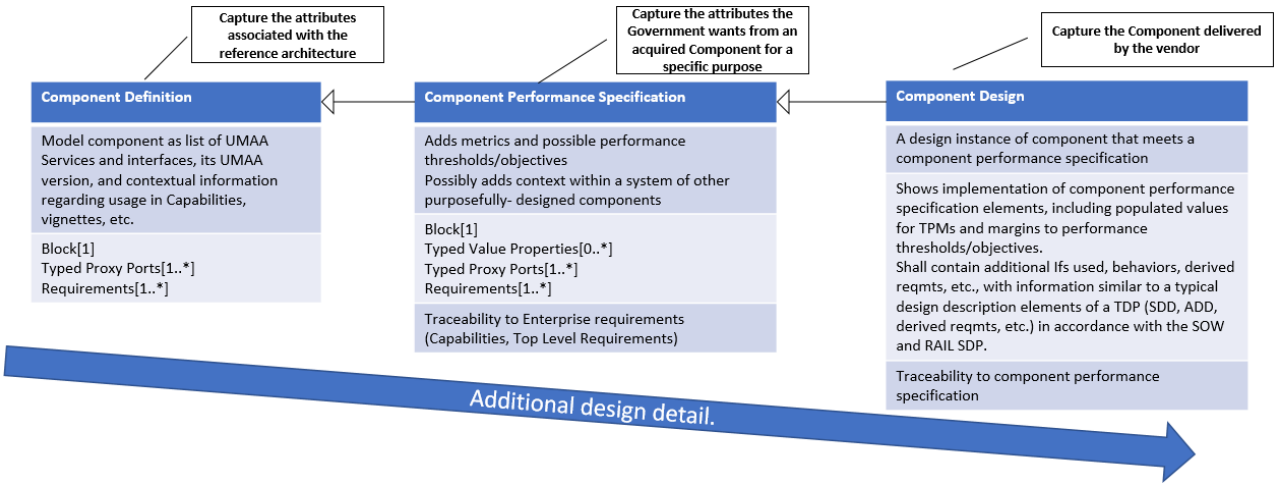
1.2 Overview

1.2.1 Unmanned Maritime Autonomy Architecture (UMAA) Component Overview

For an overview of UMAA, see "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. Service Providers describe which UMAA Services must be implemented in accordance with the UMAA ICDs as part of the Component. Service Consumers describe software 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 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. The figure below summarizes the content described in each abstraction layer.

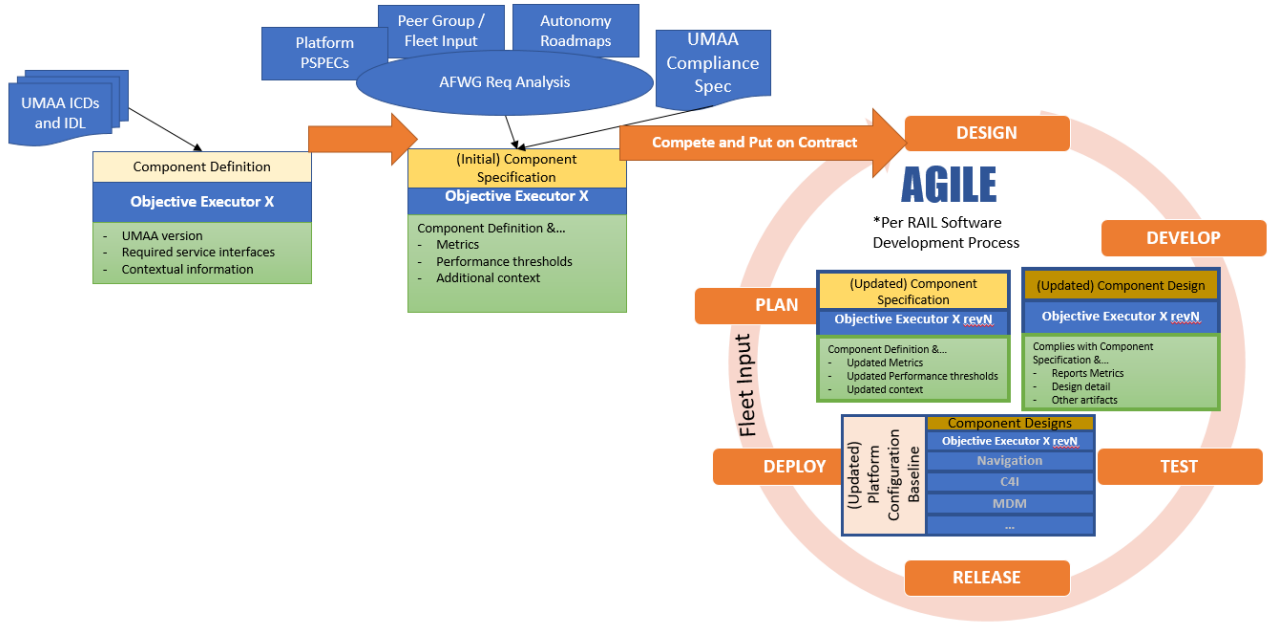


The Component Definition is the Component's highest level of abstraction represented in the reference architecture. It provides context for the need of the Component, identifies the Service Providers and Service Consumers and their interfaces, requirements that must be copied into any Component Specification 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 portfolio.

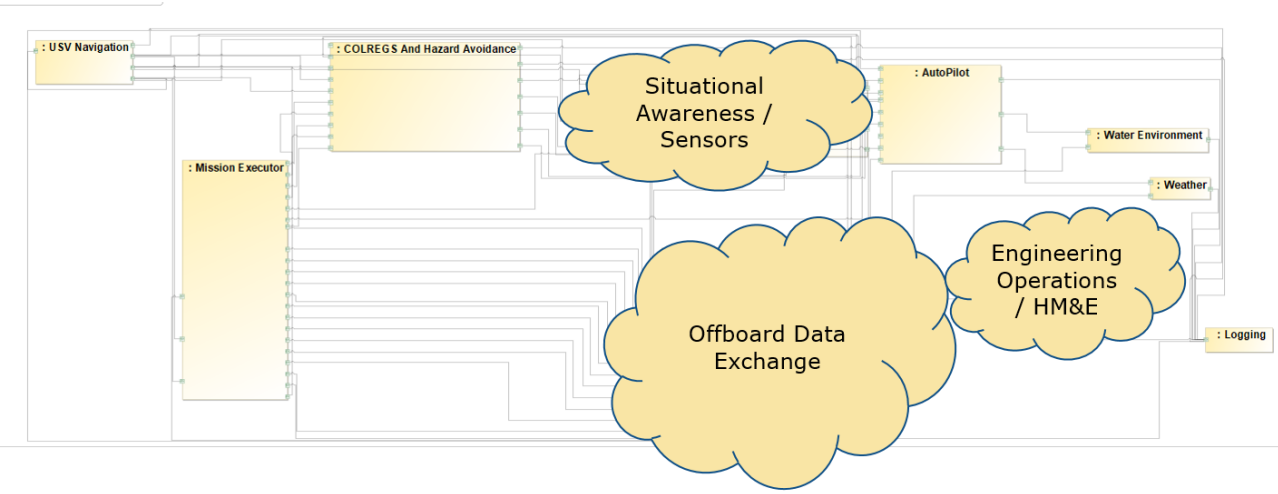
The Component Design represents a Component that is implemented by the vendor and delivered to the Government as part of the Autonomy Baseline. The Component Design shows implementation of the Component Specification, including traceability of design elements to the requirements, metrics, and Service Participant elements they satisfy. The modeled Component Design is in accordance with the acquisition's statement of work and RAIL SDP. Informally, this is the same information that is typically contained in a Software Description Document, Algorithm Description Document, etc., and can be used in part to show UMAA compliance.

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.



1.2.2 Component Reference Architecture Overview

The 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 as 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.



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 Definitions** – Provides the Component Definitions including component name, description, and list of services.

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 definition contact NAVSEA, PEO USC, ATTN: PMS 406 via the contact information on the cover page of this document.

This is not a requirements document. Requirements are to be inserted into Component Specifications or Platform Specifications to control, in part, implementation of the component in accordance with the UMAA Compliance Specification.

2 References

Number	Version	Title	Source
UMAA-INF-ADD	1.1a	Unmanned Maritime Autonomy Architecture (UMAA) Architecture Design Description (ADD)	AUVSI.org
T0300-BD-EDS-010	1	Unmanned Maritime Autonomy Architecture (UMAA) Compliance Specification	PMS 406
N/A	UMAA release 6.0	UMAA Interface Control Documents (ICD)	artifact.nswc.navy.mil

3 Component Definitions

The sections below identify the Component Definitions released by the AFWG for use with 6.0 versioned UMAA services. This includes the Component's name, description, and list of services associated with the Component. In the list of services subsection, services annotated

with "~" prior to the service name, represent usage of the interface as a consumer. Otherwise, the listing represents usage of the interface as a provider of the service.

Quality of Service (QoS) for the Components (domain participants), service interfaces (DDS endpoints), and Topics are to be configurable without changes to or recompiling of the implementation source code. Specific requirements for QoS are to be described in the Component Specification.

Domain ID and partitions for Components and their services are to be configurable without changes to or recompiling of the implementation source code. Source and destination GUIDs are to be configurable using any valid GUID other than NIL without changes to or recompiling of the implementation source code.

Additional details on the services are contained within the UMAA 6.0 ICDs.

3.1 Autopilot

3.1.1 Component Name

Autopilot

3.1.2 Component Description

The Autopilot provides for vehicle control. The Autopilot may be commanded using GlobalVector, GlobalHover, or PrimitiveDriver services. The component specification must specify how to handle commands sent over each interface and how de-confliction of commands across multiple interfaces is handled. This includes commands over the same service but from two or more different sources.

The Autopilot provides UVPlatformSpecs in part to convey overall platform maneuvering limitations. Autopilot may also add conditionals and contraints to the mission plan using MissionPlanConstraintControl given more dynamic vehicle control constraints.

3.1.3 Component Services

- PrimitiveDriverControl
- GlobalVectorControl
- ~GlobalPoseStatus
- ~VelocityStatus
- ~SpeedStatus
- UVPlatformSpecs
- LogReport
- GlobalHoverControl
- HealthReport
- ~WindStatus
- ~WaterCurrentStatus
- ~MissionPlanConstraintControl
- ActiveConstraintsControl
- ~ConditionalReport

3.2 COLREGS And Hazard Avoidance

3.2.1 Component Name

COLREGS And Hazard Avoidance

3.2.2 Component Description

This component is the low level component responsible for ensuring safe maneuvering operation of the platform. By consolidating this functionality outside of every individual maneuvering TA, it centralizes the certification testing that needs to be done to ensure safe operation of the vessel.

Since UMAA does not define an interface for nautical charts, these are to be loaded into this component external to the UMAA interfaces.

The specification for this component must specify desired behaviour of the component when commands are not received over GlobalVectorControl, GlobalHoverControl, or GlobalWaypointControl.

This component must be capable of handling the following conditional types: DepthConditionalType, DepthRateConditionalType, HeadingSectorConditionalType, LogicalANDConditionalType, LogicalNOTConditionalType, LogicalORConditionalType, TimeConditionalType, PitchRateConditionalType, RelativeSpeedConditionalType, RollRateConditionalType, SpeedConditionalType, TimeConditionalType, WaterZoneConditionalType, and YawRateConditionalType. The component only gets constraints that are applicable/active in the current state.

3.2.3 Component Services

- GlobalWaypointControl
- GlobalVectorControl
- ~ContactReport
- ~GlobalPoseStatus
- ~ContactCOLREGSClassificationStatus
- ~ContactVisualClassificationStatus
- ~GlobalVectorControl
- LogReport
- ~GlobalHoverControl
- GlobalHoverControl
- ~UVPlatformSpecs
- ~SpeedStatus
- ~VelocityStatus
- HealthReport
- ContactManeuverInfluenceStatus
- ActiveConstraintsControl
- ~ConditionalReport

3.3 Logging

3.3.1 Component Name

Logging

3.3.2 Component Description

The Logging component consumes all LogReports from other autonomy components to store locally for later extraction. The component specification should include additional performance requirements related to storage size, bandwidth, etc.

3.3.3 Component Services

- ~LogReport

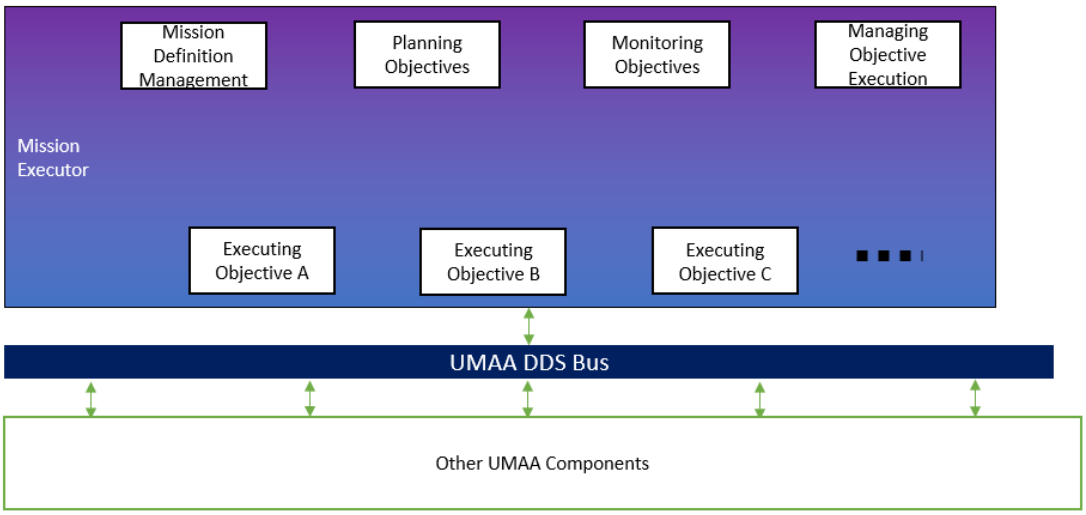
3.4 Mission Executor

3.4.1 Component Name

Mission Executor

3.4.2 Component Description

The Mission Executor component performs functions to: maintain the Mission Plan, decompose the Mission Plan into actionable autonomous objectives, manage execution of objectives, execute specific objectives, and monitor objective execution. The figure below illustrates Mission Executor Component and its interaction with other autonomy Components within the architecture.



The Mission Executor maintains the Mission Plan by tracking of the master list of conditionals and active constraints, as well as the MissionPlanReport associated with the overall set of missions loaded on the platform. This set of missions may contain more than just what is running at any given time.

The Mission Executor fully decomposes and allocates a given set of objectives without violating any constraints. The Mission Executor does not decompose missions or tasks.

The Mission Executor dispatches a given plan by managing one or more objective executions. The Mission Executor will evaluate when an objective needs to be started, stopped, or restarted based on the associated StateTriggers and availability of the associated resources. Functions to achieve active objectives execute on this component, using UMAA service interfaces to control other system components or consume data from other system components as needed.

The Mission Executor evaluates the current running plan to determine if it ever becomes invalid. Additionally, it provides execution status of all objectives being run as part of the Mission Plan.

The UMAA interfaces needed to evaluate each active constraint will be defined for each conditional statement.

3.4.3 Component Services

- ConditionalControl
- MissionPlanConstraintControl
- MissionPlanMissionControl
- MissionPlanTaskControl
- MissionPlanObjectiveControl

- MissionPlanExecutionStatus
- TaskPlanExecutionStatus
- MissionPlanExecutionControl
- TaskPlanExecutionControl
- LogReport
- ~ActiveConstraintsControl
- ConditionalReport
- HealthReport
- ObjectiveExecutionStatus
- ~GlobalPoseStatus
- ~SpeedStatus
- ~GlobalVectorControl
- ~GlobalWaypointControl
- ~GlobalHoverControl
- ~VelocityStatus
- ~ContactReport
- ~UVPlatformSpecs
- MissionPlanReport
- ObjectiveExecutionControl
- ~ContactCOLREGSClassificationStatus
- ~ContactVisualClassificationStatus
- ~WaterCurrentStatus
- ~WindStatus

3.5 USV Navigation

3.5.1 Component Name

USV Navigation

3.5.2 Component Description

The USV Navigation component provides own-ship navigational information including position and speed. This component is tailored for use on surface vessel platforms. The component specification should include additional performance attributes including accuracy and update rates for provided messages.

3.5.3 Component Services

- GlobalPoseStatus
- VelocityStatus
- SpeedStatus
- LogReport
- HealthReport

3.6 UUV Hazard Avoidance

3.6.1 Component Name

UUV Hazard Avoidance

3.6.2 Component Description

This component is the low level component responsible for ensuring safe maneuvering operation of the platform. It is tailored for underwater platforms: notably this component is not expected to comply with COLREGS.

Since UMAA does not define an interface for nautical charts, these are to be loaded into this component external to the UMAA interfaces.

The specification for this component must specify desired behavior of the component when commands are not received over GlobalVectorControl, GlobalHoverControl, or GlobalWaypointControl.

This component must be capable of handling the following conditional types: DepthConditionalType, DepthRateConditionalType, HeadingSectorConditionalType, LogicalANDConditionalType, LogicalNOTConditionalType, LogicalORConditionalType, TimeConditionalType, PitchRateConditionalType, RelativeSpeedConditionalType, RollRateConditionalType, SpeedConditionalType, TimeConditionalType, WaterZoneConditionalType, and YawRateConditionalType. The component only gets constraints that are applicable/active in the current state.

3.6.3 Component Services

- ActiveConstraintsControl
- GlobalWaypointControl
- GlobalVectorControl
- ~ContactReport
- ~GlobalPoseStatus
- ~GlobalVectorControl
- LogReport
- ~UVPlatformSpecs
- ~ConditionalReport
- GlobalHoverControl
- ~GlobalHoverControl
- HealthReport
- ~SpeedStatus
- ~VelocityStatus

3.7 UUV Navigation

3.7.1 Component Name

UUV Navigation

3.7.2 Component Description

The UUV Navigation component provides platform navigational data position and speed. This component is tailored for use on underwater. The component specification should include additional performance attributes including accuracy and update rates for provided messages.

The GlobalPoseStatus service interface is intended to be used to consume time-stamped position data from external sources for the purposes of navigation synchronization.

3.7.3 Component Services

- GlobalPoseStatus
- VelocityStatus
- SpeedStatus
- LogReport
- HealthReport
- ~GlobalPoseStatus

3.8 Water Environment

3.8.1 Component Name

Water Environment

3.8.2 Component Description

The Water Environment component provides information regarding external water environment characteristics via the WaterCurrentStatus service interface.

3.8.3 Component Services

- WaterCurrentStatus
- LogReport
- HealthReport

3.9 Weather

3.9.1 Component Name

Weather

3.9.2 Component Description

The Weather component provides information related to the weather using WindStatus service.

3.9.3 Component Services

- WindStatus
- LogReport
- HealthReport

END OF DOCUMENT