DARPA CASE TA6 UAV Architectural Models

# Introduction

The Collins Aerospace DARPA CASE TA6 team has created architectural models in Architectural and Analysis Design Language (AADL) reflecting a notional unmanned aerial system (UAS) consisting of unmanned air vehicles (UAVs) hosting instances of the Air Force Research Laboratory (AFRL) UxAS software.

The models have been developed as an aid to understand the UxAS architecture, which is the experimental platform for CASE, and implementation in the context of a representative system.

This document is intended to provide an overview to aid in understanding the models; augmenting the AADL models themselves.

## Experimental Platform

To support the development and maturation of the CASE tools, Collins Aerospace is supporting the use of UxAS as an experimental platform. UxAS is collection of software services implementing mission-level automation for a team of UAVs. It is a distributed, embedded system, has well-developed existing functionality, has an established development and simulation environment, and is open and available to CASE researchers without restriction. The baseline system has been developed without concern for cybersecurity, so provides many opportunities for improvement to make it cyber resilient.

User context (i.e. use cases, activity diagrams), requirements and architectural models are being provided by the TA6 team assist the other CASE performers in their use of the experimental platform.

## UxAS Background and Overview

UxAS is a software platform that is used to coordinate a number of unmanned vehicles in order to collaboratively complete a mission. The architecture contains approximately 50 state of the art cooperative control, path planning, and surveillance software services.

UxAS consists of a collection of modular services that interact via a common message passing architecture. Similar in design to the Robot Operating System (ROS), each service subscribes to messages in the system and responds to queries. UxAS uses the open-source library ZeroMQ to connect all services to each other. The content of each message conforms to the Light-weight Message Control Protocol (LMCP) format. Software classes providing LMCP message creation, access, and serialization/deserialization are automatically generated from simple eXtensible Markup Language (XML) description documents. These same XML descriptions detail the exact data fields, units, and default values for each message. Since all UxAS services communicate with LMCP formatted messages, a developer can quickly determine the input/output data for each service. In a real sense, the message traffic in the system exposes the interaction of the services that are required to achieve autonomous behavior.

These services typically, but not necessarily, run on a mission computer on a vehicle. In most instances of UxAS a dedicated service running a mission computer will translate LMCP messages into commands to be sent to a vehicle’s autopilot. Likewise, a dedicated service on the mission computer translates messages sent from the autopilot into UxAS messages to be sent to other relevant services.

The majority of UxAS services are currently implemented by extending a library of existing C++ classes that contain common methods that most services may need to invoke. These methods are called whenever a service is initialized, receives an LMCP message, is terminated, or some other system related event occurs. A particular configuration of UxAS is specified using an XML document. This document determines which services will execute and what their configuration parameters are.

UxAS has a rich attack surface very similar to the CH-47F demonstration platform. Potential attack vectors include radio communication data links. These data links can carry remote pilot command, situational awareness data captures such as video feeds, as well as various other control data. Likewise, UxAS features similar navigation data aids such as remote GPS. Finally, UxAS features similar maintenance attack vectors, including the ability to collect data logs, corrupt data configurations, and update software applications.

# Model Organization

This section provides and overview of the model organization and folder structure. The use of projects and folders are for organizational purposes only and do not affect the contents of the models. The models are organized into two projects: TA6\_UAV\_Exemplar and TA6\_UxAS. The TA6\_UAV\_Exemplar model represents the physical instantiation of a notion system. The TA6\_UxAS represents the UxAS software architecture. The high-level project organization for is shown in Figure 1.

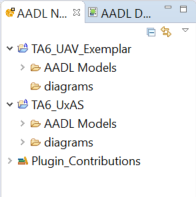


Figure 1, TA6 Exemplar Project Organization

## TA6\_UAV\_Exemplar Project Structure

The TA6\_UAV\_Exemplar project has been organized into a set of folders as shown in Figure 2. The AADL models are organized into files as follows:

* Hardware – Contains various, simplified, representations of different low level hardware components (boards, buses, memory, and processors). The hardware components exist primarily to give lower level context to the mission computing and UAV models.
* Property Sets – User defined AADL property sets used to describe in more details system or component characteristics. Typically, property sets are used to enable some sort system and component analysis (e.g. bandwidth).
* UAS – Representations of an unmanned aerial system (UAS). A UAS will consist of some sort of controller (e.g. ground station) and one or more UAVs. The UAS is the highest-level system representation.
* UAV – This folder contains the definitions of notional subsystems that comprise a UAV and specific instances of UAV themselves. The subsystems (flight controls, propulsion, mission computing, radios and sensors) are simplified and are used to provide overall context. The UAV.aadl package contains the different representations of UAVs where the subsystems and interfaces between subsystems differentiate between UAV types.

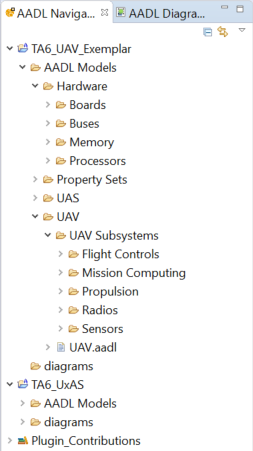


Figure 2, TA6\_UAV\_Exemplar Folder Structure

## TA6\_UxAS Project Structure

The TA6\_UxAS project has been organized into a set of folders as shown in Figure 3. The folder structure and the models within them are based on (a) AFRL’s baseline software, and (b) AADL models built by Collins Aerospace as part of the AFRL Summer of Innovation (SOI) project. The AADL models are organized into files as follows:

* Communications – Communication protocols within UxAS (between tasks and services). Currently there are no models defined here.
* Logging and Data Services – General services used to monitor activity and send messages within the system.
* Message Definitions – Message definitions used for communication between tasks and services.
* Services – UxAS primary services. Essentially, each instance of the UxAS includes all of the primary services.
* Tasks – A subset of the existing UxAS tasks. Tasks are generated/created by services in response to automation requests.
* UxAS – Models of the high-level processes that are instantiations of UxAS are defined in this folder. Each instance of UxAS is a process; each process contains as set of services and tasks.

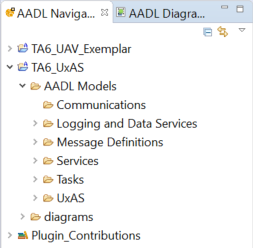


Figure 3, TA6\_UxAS Folder Structure

# AADL Modeling Approach

The table below shows general AADL modeling guidelines used to represent the UAS system/software architecture.

Table 1, AADL Modeling Approach Summary

|  |  |
| --- | --- |
| Element | Description |
| Device | A unit in the operational environment accessed or controlled by the application software. A device is considered a black box (i.e. no internal hardware/software details defined). |
| Feature Group | Represent generic collections of related data that allow one to add detailed data definitions in future model refinements. |
| Bus | Represent physical internal and external communication mechanisms. |
| Virtual Bus | May be used to represent protocols or virtual communication mechanisms within the software. |
| Data/Event Ports | Used to reflect data (or events) interfaces that can be physical or logical within the system. |
| Bindings | Used to show physical connections between system elements or allocation of software (process or thread) to specific physical processor. |

# AADL Models

## UAS System

The notional UAS defined to provide system context consists of a ground station communicating to dissimilar UAVs over a wireless (Wi-Fi) datalink. Each UAV hosts an instance of the UxAS software on a vehicle management computer (VMC). Figure 4 is the structure diagram for this UAS.

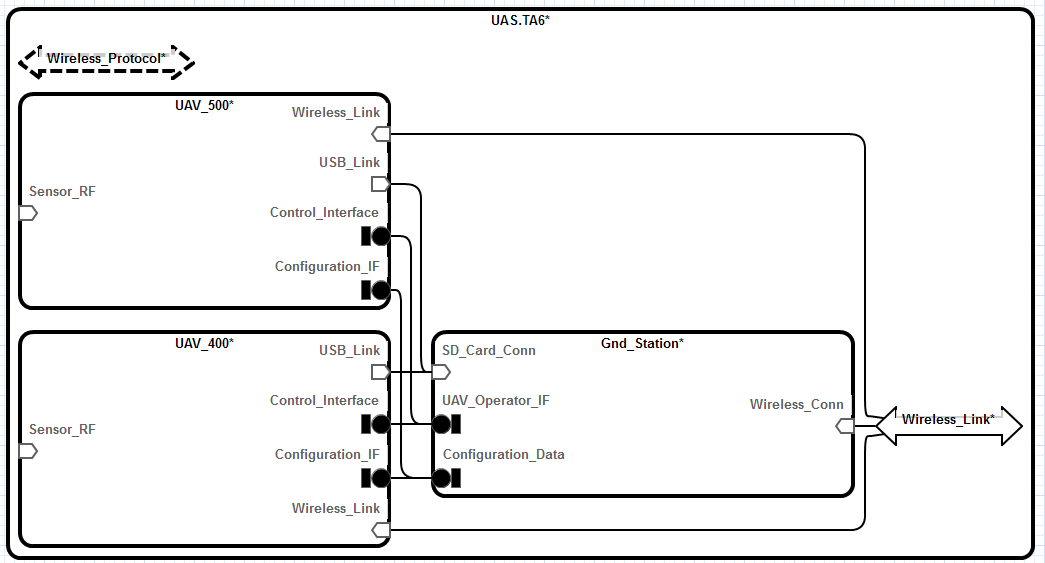


Figure 4, UAS Structure Diagram

Figure 5 below depicts a binding diagram for the UAS. The diagram shows the implementation view of the UAV subcomponents. The UxAS software instances are bound to the VMC processor. UAV\_500 differs from UAV\_400 in that it includes a notional EO\_IR\_Sensor.

Figure 6 and Figure 7 are the structure diagrams for UAV\_400 and UAV\_500 respectively. The structure diagram shows the interface connections with the UAV.

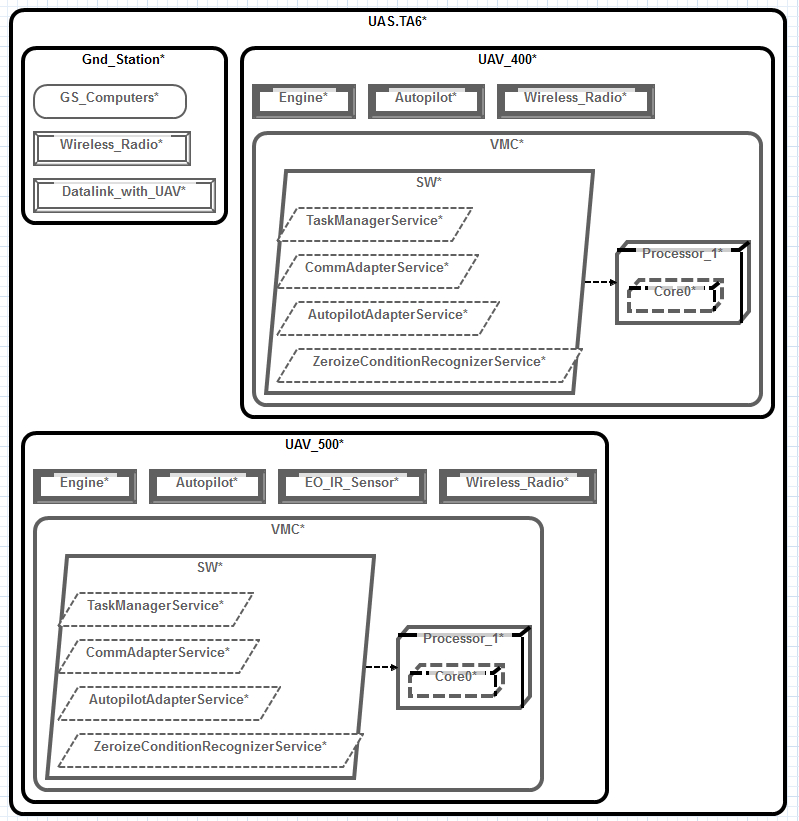


Figure 5, UAS Binding Diagram

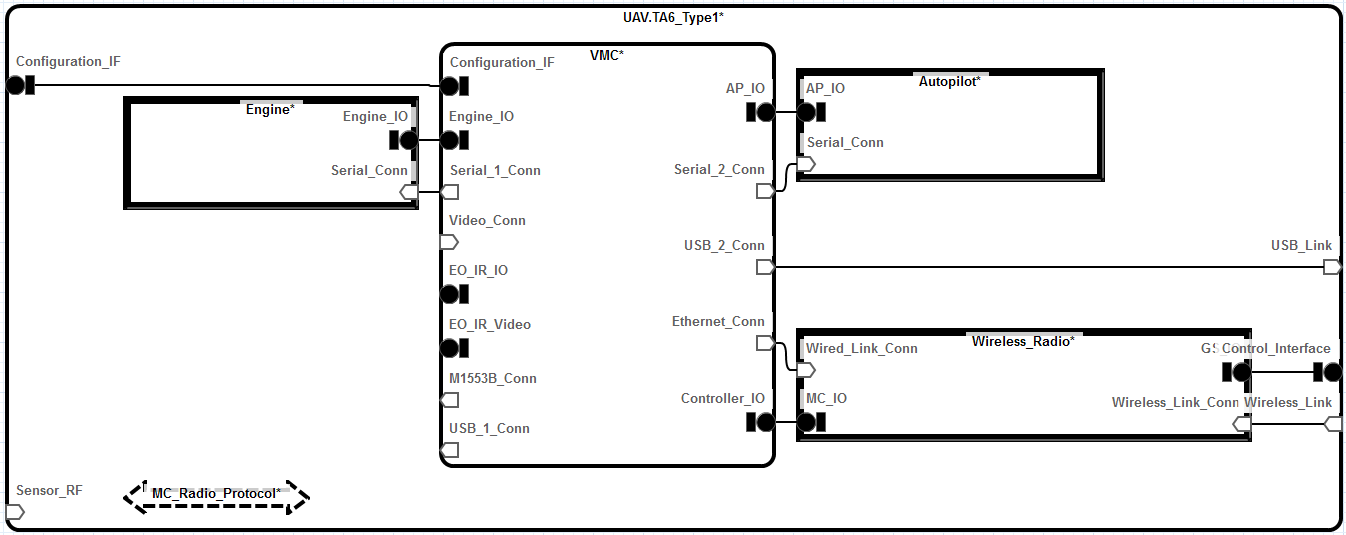


Figure 6, UAV Type 1 Structure Diagram

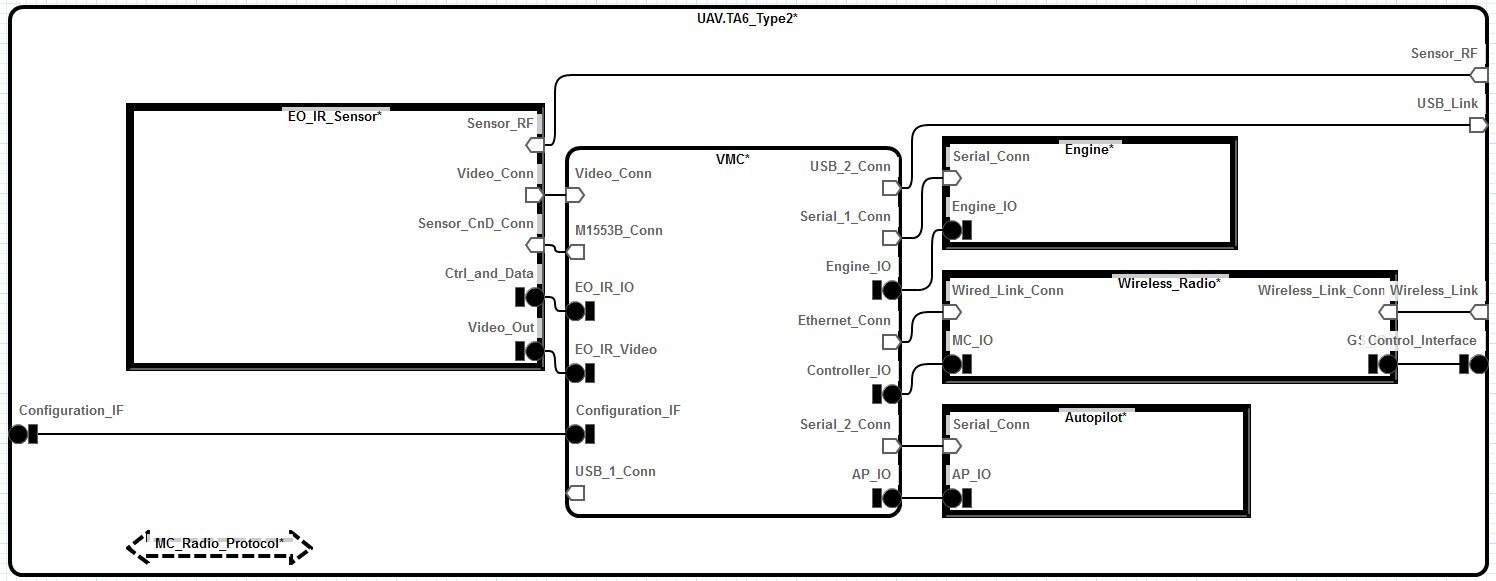


Figure 7, UAV Type 2 Structure Diagram

As a reference, the AADL model (text) defining the Type 1 UAV is shown below in Figure 8. In this example, the UAV implementation communicates to a controller (ground station) via a Wi-Fi datalink. In the AADL model each type of UAV is represented as a different implementation of the basic UAV type. The implementations define the subcomponents of the UAV, connections between subcomponents, and the binding of the connections to specific interfaces.

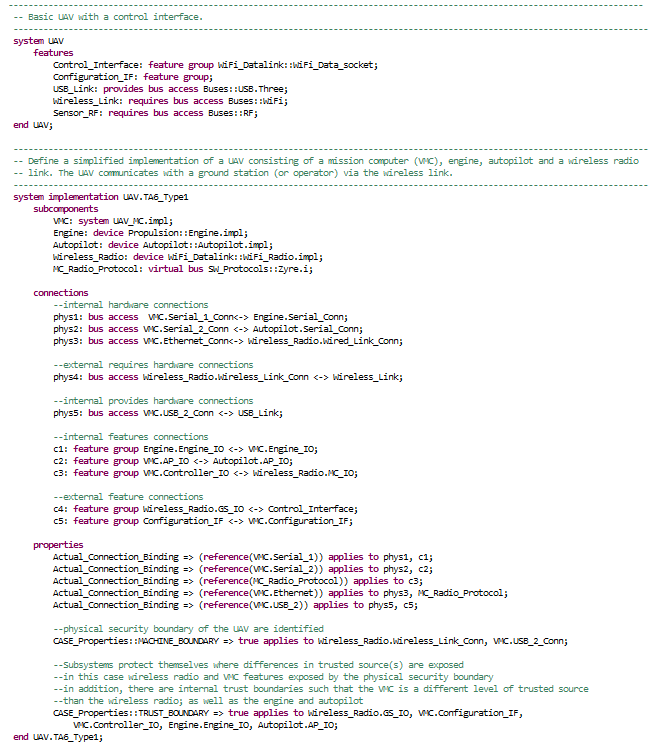


Figure 8, UAV AADL Model

## UxAS Software

UxAS software is modeled as a single process with multiple threads. Each thread is either a UxAS service or tasks. The services and tasks communicate with one other using the UxAS defined messages.

There is a unique instantiation of UxAS for each mission or vehicle configuration. Figure 9 shows an instantiation for the UxAS software. Figure 10 is the message sequence diagram that corresponds to the Zeroize service implementation in UxAS.

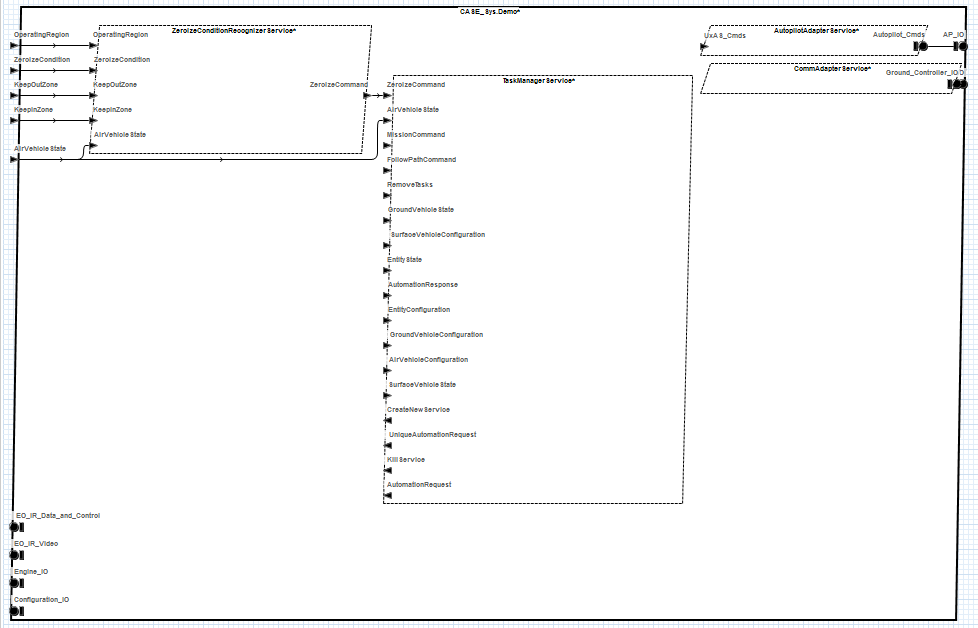


Figure 9, UxAS Zeroize Structure Diagram

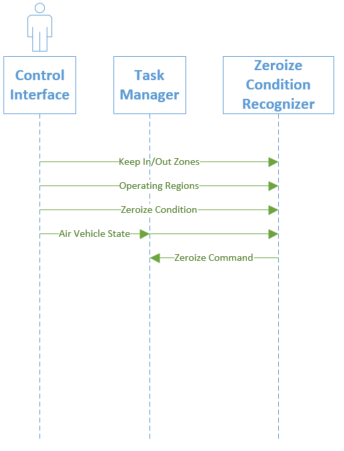


Figure , Zeroize Message Sequence Diagram

The TA6\_UxAS AADL model defines four separate representations of the UxAS software as shown in Figure 11.

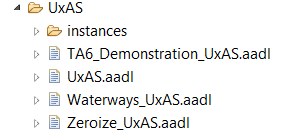


Figure 11, UxAS Software Configurations

* TA6\_Demonstration\_UxAS.aadl - Software configuration used in the UAV model. Includes the zeroize service, comm adapter, and autopilot adapter.
* UxAS.aadl - Simple declaration of a UxAS process
* Waterways\_UxAS.aadl – model of the original UxAS software without CASE modifications
* Zeroize\_UxAS.aadl – model focused on the zeroize service only

The TA6\_Demonstration\_UxAS is currently used in the UAV AADL model. It is bound in the mission computer extension as shown in Figure 12. The UxAS models can be set up to allow any needed modifications to support CASE tool development.

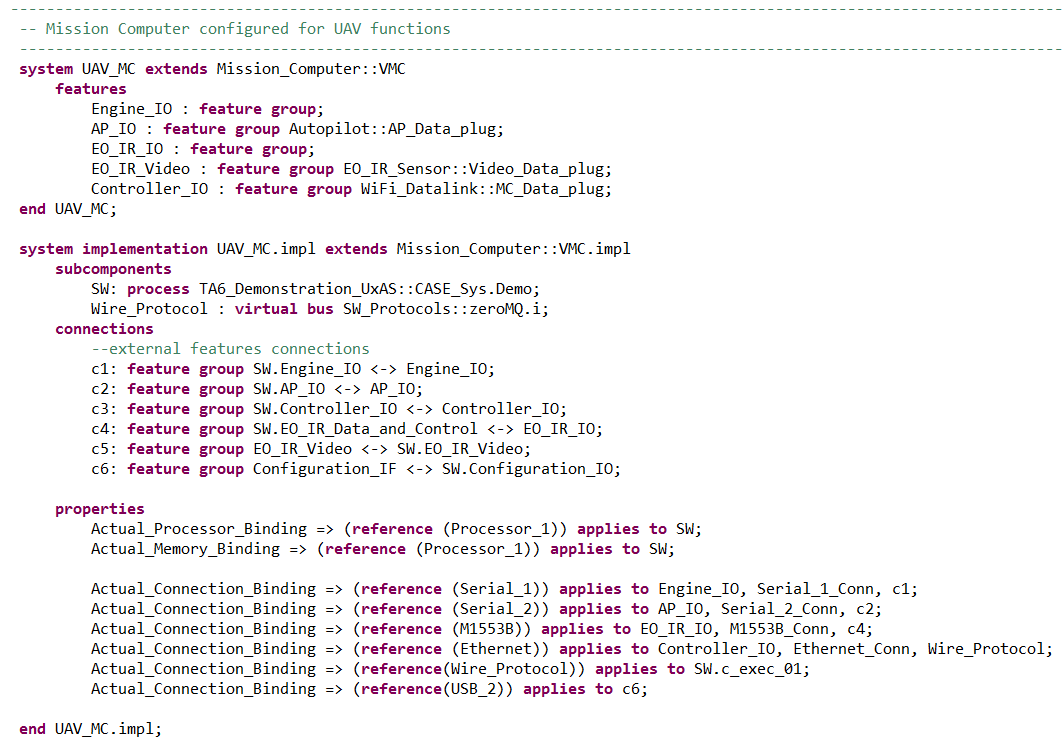


Figure 12, Mission Computer configured with UxAS Software