Software System Architecture Document

# Project ADSEN86 – Automated Drone Sentry

## List of Changes to Document:

**Draft Version: 1.0**

Initial version: 11/23/2024

Minimum Viable Product version: 1.0

### Project Name: Automated Drone Sentry

### Project ID: ADSEN86

## Introduction:

This document provides the overall Software system architecture of the ADSEN86 project. ADSEN is Automatic Drone Sentry that has AI capabilities to follow its main user’s commands and perform sentry function. This architecture document provides a high level overview of the entire system and how the drone is architected to provide its service.

## Abbreviations:

**CFI:** Core Function Intelligence

**DRIM:** Display Renderer and indication Module

**CMI:** Connectivity Module Interface

**MFM:** Motor Function Module

**VIM:** Vision Interpreter Module

**SM:** Sensor Module

# Design Diagram

### Basic Framework

CORE

FUNCTION

INTELLIGENCE

ADSEN86

Motor Function Module

ADSEN86-001

Vision Interpreter Module

ADSEN86-002

Connectivity Module Interface ADSEN86-005

Display Render and Indication Module ADSEN86-003/004

Sensor Module

ADSEN86-001

Control Signals

Data Signals

Figure 1

The diagram represented by Figure 1 provides a basic framework that the Sentry drone is designed around.

**Core Function Intelligence:**

The core functionality of the drone is operated from the Core Function Intelligence Module. This module controls the operations of the drone and the decision making of the drone based on inputs from different modules. The 5 main modules are implemented as per the Basic Requirements Documentation. These modules feed into the Core Intelligence and also operate based on information provided by the core intelligence module.

**Motor Function Module:**

This module provides the functionality of movement to the Drone sentry. This module mainly takes the input from the Core intelligence module.

**Vision Interpreter Module:**

This module provides the functionality of gathering the camera based input data. Sending the input for interpretation to the AI sub functionality and determining result which can be used to make decisions.

**Connectivity Module Interface:**

This module provides the ability to connect to the drone sentry over multiple different interfaces. The type of interface to connect over can be selected using the settings

**Display Renderer and Indication:**

This module provides the ability to indicate to the User the intentions of the Drone and the results of what it is seeing and interpreting. This can be through different LED based signage or Display system.

**Sensor Module:**

This module provides the inputs from different sensors to generate information of the drone’s surroundings. This input can be used by the CIM to figure out which way to move and obstacles to avoid.

# Connectivity Module Interface

***Introduction:***

The connectivity module interface provides a basic framework that exposes outward communicating hardware modules and technologies from the drone subsystem. The Connectivity Module has 3 different layers. Each layer abstracts a portion of the underlying interface and exposes less details to the higher up services.

The basic framework of the Connectivity module is provide an interface to the top level service. See the image below for more details

***Service Layer***

Service 4

Service 3

Service 2

Service 1

Protocol 1

Protocol 2

***Protocol Layer***

Protocol 2

Protocol 6

Protocol 3

Protocol 4

Protocol 5

Protocol 1

***Interface Layer***

The connectivity layer will consist of 3 layers. The Protocol Layer, Interface layer and the Service layer.

**The Service layers** provide a specific service that needs to be enabled.

For example let us say that we need the ability to configure the drone using Bluetooth. We need to provide some form of a command line interface that can set certain Wi-Fi settings or other settings in the drone. A Bluetooth command line service can run that acts as a shell. It will respond to commands and provide specific feedback that a certain setting was configured.

Another example is let us say we need the ability to output certain specific strings or video data over a Wi-Fi interface that is encoded and then sent over an encrypted link. The process of figuring out what video needs to be sent and where to fetch that data is done by the service layer.

This layer is generally designed as a factory.

**The Protocol layer** is a series of decorators that provide cascading levels of functionality that satisfy a specific protocol. The Standard interface protocol is the base protocol that will have the specific interface that will communicate over the LOB. All other protocols will use a lower level protocol as the lower layer of communication and are instantiated/called from higher level protocols or services.

In our example of the Bluetooth command line, the protocol layer may consist of the standard protocol that has the Bluetooth interface. Then it may be wrapped by a HID or SPP profile protocol layer that talks in terms of the Attribute table and instantiates the Bluetooth to talk in that format.

Similarly the protocol layer in the Wi-Fi video stream example may be the standard Wi-Fi protocol Interface wrapped by the TLS1.3 protocol wrapped by the HTTPS protocol interface.

This layer is designed as a Decorator.

**The Interface layer** is the final layer that uniquely specifies a hardware technology. It has all the functionalities that setup the hardware technology.

In the case of the Bluetooth command line the Bluetooth Standard protocol may communicate with the Bluetooth Low Energy Interface that sets up the Bluetooth capability. It provides APIs to setup the GAP and GATT table for the same.

In the case of the Wi-Fi video stream the Wi-Fi standard protocol talks to the Wi-Fi interface class and configures the Wi-Fi to be able associate and provide a socket connection as server or a client.